UNIVERSITAS AMIKOM YOGYAKARTA

2018

Algoritma Struktur Data

Program Studi D3 Manajemen Informatika
Fakultas Ilmu Komputer
Universitas AMIKOM Yogyakarta

KATA PENGANTAR

Algoritma Struktur Data (ASD) merupakan mata kuliah wajib untuk mahasiswa semester 1 (ganjil) program studi D3 Manajemen Informatika. Tujuan pembelajaran mata kuliah ini adalah mahasiswa memahami bagaimana pengertian, konsep dan penerapan dari ilmu logika yang kemudian diterapkan ke dalam bahasa pemrograman sehingga dapat diaplikasikan dengan benar, bersifat logis dan sistematis.

Pokok pembahasan mata kuliah ini adalah konsep algoritma, dasar pemrograman dan struktur bahasa pemrograman C++, perintah dasar seperti input output, *preprosesor directive*, *header file*, variabel, konstanta, tipe data, pengaturan desimal, tipe data bentukan, percabangan, perulangan, modular atau fungsi, pointer, array 1 dan 2 dimensi, pengurutan data (*sorting*), pencarian data (*searching*), struktur (*struct*), tumpukan data (*stack*), dan antrian data (*queue*).

Secara garis besar capaian pembelajaran dari mata kuliah ini adalah mahasiswa diharapkan :

- Mampu membuat algoritma secara logis dan sistematis berdasarkan kasus.
- Mampu membuat program sederhana dengan menggunakan *prepreprosesor directive*, *header file* dan perintah input output, menerapkan variabel, konstanta, pengaturan desimal dan tipe data bentukan, menerapkan metode percabangan dan perulangan, program dengan teknik modular.
- Mampu membuat program dengan menggunakan variabel array 1 dimensi dan 2 dimensi.
- Mampu menerapkan pengelolaan data dengan sistem *sorting*, *searching* dan *struct* dalam program.
- Mampu menerapkan pengelolaan data dengan algoritma *stack* dan *queue*.

PENGESAHAN

Disusun Oleh	**Diperiksa & Dikendalikan Oleh**	**Disetujui oleh**
Ninik Tri Hartanti, M.Kom Yuli Astuti, M.Kom	Sri Ngudi Wahyuni, S.T., M.Kom D3 Manajemen Informatika	Hanif Al Fatta, M.Kom D3 Manajemen Informatika
Tgl.	Tgl.	Tgl.

Modul ini syah dan diberlakukan mulai: Tgl 17 September 2018

Krisnawati, S.Si., M.T.

Daftar Isi

BAB I. LOGIKA PROPOSISION (PROPOSITION LOGIC)

1.1 Tujuan Pembelajaran

Setelah mengikuti modul ini, mahasiswa diharapkan mampu:

1. Menjelaskan pengertian proposisi
2. Menjelaskan pengertian proposisi atom, proposisi majemuk, dan operator logika
3. Menjelaskan pernyataan ingkaran/negasi
4. Menjelaskan pernyataan majemuk yang meliputi konjungsi, disjungsi, disjungsi exclusive (XOR), implikasi dan biimplikasi
5. Menjelaskan kontrapositif, konvers, dan invers
6. Menjelaskan formula logika proposisi

1.2 Dasar Teori

1.2.1. Proposision (*Proposition)*

Proposisi adalah merupakan kalimat deklaratif atau pernyataan yang memiliki nilai kebenaran benar atau salah, tetapi tidak keduanya Logika proposisi merupakan dasar ilmu yang dipergunakan dalam *computer sciences* dan *software engineering*.. Logika Proposisi merupakan suatu sistem logika yang didasarkan atas proposisi. Logika proposisi juga diistilahkan dengan kalkulus proposisi (*propositional calculus*). Proposisi sederhana biasanya ditulis dengan huruf p, q, r, s, p_1, p_2. Nilai kebenaran yang mungkin untuk suatu proposisi :

- Benar, dapat pula ditulis: B, T, true, >, 1
- Salah, dapat pula ditulis: S, F, false, 0.

Berikut adalah beberapa contohnya :

1. **$2^3 < 3^2$**
    - Apakah merupakan pernyataan? < ya >
    - Apakah merupakan proposisi? < ya >
    - Apa nilai kebenarannya? < benar >
2. **3^4-$4^3 < 10$**
    - Apakah merupakan pernyataan? < ya >
    - Apakah merupakan proposisi? < ya >

- Apa nilai kebenarannya? < salah >

3. **x+3 >= 2015**
    - Apakah merupakan pernyataan? < ya >
    - Apakah merupakan proposisi? < bukan, karena nilai kebenaran tergantung pada nilai x, pernyataan benar apabila x>=2012 >
    - Merupakan kalimat terbuka
4. **x+2x-3x=0**
    - Apakah merupakan pernyataan? < ya >
    - Apakah merupakan proposisi? < ya, karena berapapun nilai x, pernyataan x+2x-3x=0 selalu benar >
    - Apa nilai kebenarannya? < benar >
5. **"pelajari matakuliah logika matematika dengan baik"**
    - Apakah merupakan pernyataan?
      < bukan, ini merupakan permintaan >
    - Apakah merupakan proposisi?
      < bukan, karena bukan pernyataan >
    - Hanya pernyataan yang bisa menjadi preposisi.

1.2.2. Proposisi Atom, Proposisi Majemuk dan Operator Logika

Beberapa contoh-contoh di atas adalah merupakan proposisi sederhana, yang juga disebut sebagai proposisi **atom**, yang dapat dibentuk menjadi proposisi baru menggunakan operator (penghubung) logika. Proposisi yang dihasilkan selanjutnya disebut sebagai proposisi majemuk (*compound proposition*). Berdasarkan banyaknya proposisi atom yang dioperasikan, ada dua jenis operator logika dasar, yaitu :

1. Operator *uner (unary)*→ hanya memerlukan satu operand → negasi
2. Operator *biner (binary)*→ memerlukan dua operand, diantaranya : konjungsi ($\vee$), disjungsi ($\wedge$), implikasi ($\rightarrow$), dan biimplikasi ($\leftrightarrow$).

Pernyataan adalah kalimat yang hanya benar saja atau salah saja, akan tetapi tidak sekaligus benar dan salah, dan diberi lambang dengan huruf kecil, seperti : a, b, p, q. Kalimat Terbuka adalah kalimat yang masih mengandung variabel, sehingga belum dapat ditentukan nilai kebenarannya (benar atau salah), sehingga, untuk menentukan

benar atau salahnya, kita perlu pengamatan lebih lanjut. Berikut adalah beberapa contohnya.

- Indonesia Raya adalah lagu kebangsaan Indonesia (pernyataan benar)
- Bika ambon berasal dari Ambon. (pernyataan salah)
- Jakarta adalah ibukota negara Indonesia. (pernyataan benar)
- 12x + 6 = 91 (pernyataan ini dinamakan kalimat terbuka karena masih harus dibuktikan kebenarannya. Apakah benar 12x jika dijumlahkan dengan 6 akan menghasilkan 91?).
- Terimakasih.., saya sudah kenyang. (Pernyataan ini dinamakan kalimat terbuka karena masih harus dibuktikan kebenarannya. Apakah benar dia tidak mau makan karena kenyang? Atau memang lagi malas makan?)

1.2.3. Ingkaran/Negasi/Penyangkalan

Membuat pernyataan baru berupa "ingkaran/negasi/penyangkalan" atas beberapa pernyataan. Berikut adalah tabel kebenaran ingkaran :

Tabel 1.1. Tabel Negasi

p	**– p**
T	F
F	T

T pernyataan bernilai benar, F pernyataan bernilai salah. Artinya, jika suatu pertanyaan (p) benar, maka ingkaran (– p) akan bernilai salah, begitu pula sebaliknya. Berikut adalah beberapa contoh:

p : Besi memuai jika dipanaskan (pernyataan bernilai benar).

– p : Besi **tidak** memuai jika dipanaskan (pernyataan bernilai salah).

p : **Semua** unggas **adalah** burung.

– p : **Ada** unggas yang **bukan** burung.

1.2.4. Pernyataan Majemuk

Majemuk adalah gabungan dari beberapa pernyataan tunggal yang dihubungkan dengan kata hubung. Dalam ilmu matematika, terdapat 4 macam pernyataan majemuk:

1.2.4.1. Konjungsi ($\wedge$)

Konjungsi adalah pernyataan majemuk dengan kata hubung "dan". Sehingga, notasi "p ^ q" dibaca "p dan q". Tabel kebenaran untuk konjungsi adalah sebagai berikut:

Tabel 1.2. Tabel Kebenaran Konjungsi

p	q	p $\wedge$ q
T	T	T
T	F	F
F	T	F
F	F	F

p = pernyataan 1

q = pernyataan 2

p $\wedge$ q = pernyataan 1 dan pernyataan 2

Dari tabel di atas, kita dapat melihat bahwa **konjungsi hanya akan benar jika kedua pernyataan (p dan q) benar.**

Contoh:

- p: 3 adalah bilangan prima (pernyataan bernilai benar)
- q: 3 adalah bilangan ganjil (pernyataan bernilai benar)
- p $\wedge$ q: 3 adalah bilangan prima **dan** ganjil (pernyataan bernilai benar)

1.2.4.2. Disjungsi ($\vee$)

Disjungsi adalah pernyataan majemuk dengan kata hubung "atau". Sehingga notasi "p $\vee$ q" dibaca "p atau q". Tabel nilai kebenaran disjungsi:

Tabel 1.3. Tabel Kebenaran Disjungsi

p	q	p $\vee$ q
T	T	T
T	F	T
F	T	T
F	F	F

p = pernyataan 1

q = pernyataan 2

p $\vee$ q = pernyataan 1 atau pernyataan 2

Dari tabel di atas, kita dapat melihat bahwa **disjungsi hanya akan benar jika salah satu dari kedua pernyataan (p atau q) benar,** atau disjungsi **hanya salah jika kedua pernyataan (p dan q) salah**.

Contoh:

p : Paus adalah mamalia (pernyataan bernilai benar)

q : Paus adalah herbivora (pernyataan bernilai salah)

$p \vee q$: Paus adalah mamalia atau herbivora (pernyataan bernilai benar)

1.2.4.3. Disjungsi Exclusive (XOR)

Apabila p dan q merupakan proposisi, maka $p \oplus q$ juga merupakan proposisi yang dinamakan sebagai disjungsi eksklusif/ exclusive or (xor) dari p dan q. $p \oplus q$ dibaca p xor q, $p \oplus q$ bernilai benar (T) tepat ketika p dan q memiliki nilai kebenaran yang berbeda. Berikut adalah tabel kebenaran XOR.

Tabel 1.4. Tabel Kebenaran XOR

p	**q**	**$p \oplus q$**
T	T	F
T	F	T
F	T	T
F	F	F

1.2.4.4. Implikasi (→)

Implikasi adalah pernyataan majemuk dengan kata hubung "jika… maka…" Sehingga notasi dari "p → q" dibaca "Jika p, maka q". Adapun tabel nilai kebenaran dari implikasi seperti berikut :

Tabel 1.5. Tabel Kebenaran Implikasi

p	**q**	**p → q**
T	T	T
T	F	F
F	T	T
F	F	T

p = pernyataan 1

q = pernyataan 2

p → q = jika pernyataan 1 maka pernyataan 2

Dari tabel terlihat bahwa implikasi hanya **bernilai salah jika anteseden (p) benar, dan konsekuen (q) salah.**

Contoh:

p : Andi belajar dengan aplikasi ruangguru. (pernyataan bernilai benar)

q : Andi dapat belajar di mana saja. (pernyataan bernilai benar)

p → q: Jika Andi belajar dengan aplikasi ruangguru, maka Andi dapat belajar di mana saja (pernyataan bernilai benar).

1.2.4.5. Biimplikasi (↔)

Biimplikasi adalah pernyataan majemuk dengan kata hubung "… jika dan hanya jika". Sehingga, notasi dari "p ↔ q" akan dibaca "p jika dan hanya jika q". Tabel nilai kebenaran **Biimplikasi** sebagai berikut**:**

Tabel 1.5. Tabel Kebenaran Biimplikasi

p	**q**	**p ↔ q**
T	T	T
T	F	F
F	T	F
F	F	T

p = pernyataan 1

q = pernyataan 2

p ↔ q = pernyataan 1 jika dan hanya jika pernyataan 2

Dari tabel kebenaran tersebut, dapat kita amati bahwa **biimplikasi akan bernilai benar jika sebab dan akibatnya (pernyataan p dan q) bernilai sama.** Baik itu sama-sama benar, atau sama-sama salah.

Contoh:

- p : 30 x 2 = 60 (pernyataan bernilai benar)
- q : 60 adalah bilangan ganjil (pernyataan bernilai salah)
- p ↔ q: 30 x 2 = 60 **jika dan hanya jika** 60 adalah bilangan ganjil (pernyataan bernilai salah).

1.2.4.6. Kontrapositif, Konvers, dan Invers

Diberikan suatu implikasi p → q, maka

- kontrapositif (atau kontraposisi) dari p → q adalah – q → – p
- konvers dari p → q adalah q → p
- invers dari p → q adalah – p → – q

Tabel kebenaran untuk kontrapositif, konvers, dan invers adalah sebagai berikut :

Tabel 1.6. Tabel Kebenaran Kontrapositif, Konvers dan Invers

p	q	– p	– q	p → q	– q → – p	q → p	– p → – q
T	T	F	F	T	T	T	T
T	F	F	T	F	T	T	T
F	T	T	F	T	F	F	F
F	F	T	T	T	T	T	T

1.2.5. Urutan Pengerjaan (Presedens) Operator Logika

Presedens operator logika memberikan suatu aturan operator mana yang harus lebih dulu dioperasikan (dikenakan pada suatu operand). Berikut adalah tabel presedens operator logika.

Tabel 1.7. Tabel Presedens Operator Logika

Operator	Urutan
–	1
$\wedge$	2
$\vee$	3
$\oplus$	4
$\rightarrow$	5
$\leftrightarrow$	6

Contoh *presedens* seperti di bawah ini:

- $p \vee q \wedge r$ berarti $p \vee (q \wedge r)$
- $- p \vee q$ berarti $(- p) \vee q$
- $p \wedge q \rightarrow r$ berarti $(p \wedge q) \rightarrow r$
- $p \rightarrow - q \wedge r$ berarti $p \rightarrow ((- q) \wedge r)$
- $- p \vee q \rightarrow r \wedge - s$ berarti $((- p) \vee q) \rightarrow (r \wedge (- s))$

1.2.6. Sifat Kalimat (*Properties of Sentences*)

Properties of Sentences adalah sifat-sifat yang dimiliki oleh kalimat logika. Terdapat tiga sifat yaitu:

1. *Valid*
2. *Contradictory*
3. *Satisfiable*

1.2.6.1.Valid

Suatu *sentence f* disebut *valid*, jika untuk setiap interpretation I for *f*, maka *f true.*

Contoh:

1. (f and g) if and only if (g and f)
2. f or not f
3. (p and (if r then s)) if only if ((if r then s) and p)
4. (p or q) or not (p or q)
5. (if p then not q) if and only if not (p and q)

1.2.6.2.Contradictory

Suatu *sentence* f disebut *contradictory*, jika untuk setiap interpretation I for *f*, maka f *false*.

Contoh:

1. p and not p
2. ((p or q) and not r) if and only if ((if p then r) and (if q then r)

1.2.6.3.Satisfiable

*Suatu sentence **f** disebut **satisfiable**, jika untuk suatu interpretation **I** for **f**, maka **f true***

Contoh:

1. *if (if p then q) then q*
2. *(if p then q) and (not r and s)*
3. *(if r then q) or p*

1.2.7. Metode Inferensi (Inference Method)

Metode inferensi merupakan suatu teknik/metode untuk menurunkan kesimpulan berdasarkan hipotesa yang diberikan, tanpa harus menggunakan tabel kebenaran. Metode yang sering digunakan adalah:

1. Modus Ponens
2. Modus Tolens
3. Prinsip Sylogisme

1.2.7.1. Modus Ponens

Modus Ponens adalah argumen yang menyatakan bahwa jika pernyataan 1 berimplikasi dengan pernyataan 2 bernilai benar, dan pernyataan 2 bernlai benar maka pernyataan 2 dianggap menjadi kesimpulan. Bisa disimbolkan seperti berikut:

1. p → q (premis 1)
2. p (premis 2)
---
3. q (kesimpulan)

Contoh:

1. Premis 1 : Jika suatu bilangan habis dibagi 2 maka bilangan tersebut bilangan genap.
   Premis 2 : Suatu bilangan habis dibagi 2.
   Kesimpulan : Bilangan tersebut adalah bilangan genap.
2. Premis 1 : Jika seorang anak rajin belajar, maka ia lulus ujian.
   Premis 2 : Ahmad adalah anak yang rajin belajar.
   Kesimpulan : Ahmad lulus ujian.

1.2.7.2. Modus Tollens

Modus Tollens adalah argumen yang menyatakan bahwa jika pernyataan 1 berimplikasi dengan pernyataan 2 bernilai benar, dan negasi pernyataan 2 bernilai benar maka negasi pernyataan 1 juga benar dan dianggap sebagai kesimpulan. Bisa disimbolkan seperti berikut :

1. p → q (premis 1)
2. ~q (premis 2)
3. ~p (kesimpulan)

Contoh:

1. Premis 1 : Jika suatu bilangan habis dibagi 2 maka bilangan tersebut bilangan genap
   Premis 2 : Ada suatu bilangan ganjil
   Kesimpulan : Bilangan tersebut tidak habis dibagi 2
2. Premis 1 : Jika hari minggu, maka Aisha bertamasya
   Premis 2 : Aisha tidak bertamasya
   Kesimpulan : Bukan hari minggu

1.2.7.3. Prinsip Sylogisme

Sylogisme adalah argumen yang menyatakan bahwa jika pernyataan 1 berimplikasi dengan pernyataan 2 bernilai benar, dan pernyataan 2 berimplikasi dengan pernyataan 3 juga bernilai benar maka pernyataan 1 berimplikasi dengan pernyataan 3 pun benar dan dinyatakan sebagai kesimpulan. Bisa disimbolkan seperti berikut :

1. p → q (premis 1)

2. q → r (premis 2)

3. p → r (kesimpulan)

Contoh:

1. Premis 1 : Jika ia belajar dengan baik maka ia akan pandai
   Premis 2 : Jika ia pandai maka ia akan lulus ujian
   Kesimpulan : Jika ia belajar dengan baik maka ia akan lulus ujian
2. Premis 1 : Jika Aisha rajin belajar, maka ia naik kelas
   Premis 2 : Jika Aisha naik kelas, maka akan dibelikan sepeda
   Kesimpulan : Jika Aisha rajin belajar, maka akan dibelikan sepeda.

1.2.8. Kalimat Pengukur (Quantifier Sentences)

Kalimat yang memuat ekspresi kuantitas objek yang terlibat, misalnya: semua, ada, beberapa, tidak semua, dan lain-lain. Ada dua macam, kalimat berkuantor:

1. *Universal Quantifier*
2. *Existential Quantifier*

1.2.8.1. Universal Quantifier (for all…)

Terdapat kata-kata yang mempunyai makna umum dan menyeluruh. Notasi yang digunakan adalah : $\forall$, dibaca semua, seluruh, setiap, sedangkan untuk penulisannya adalah : $\forall x \in S \rightarrow p(x)$. Semua *x* dalam semesta *S* mempunyai sifat *p*.
Contoh:

1. Semua orang yang hidup pasti mati
2. Setiap mahasiswa pasti pandai

1.2.8.2. Existential Quantifier

Terdapat kata-kata yg mempunyai makna khusus/sebagian. Notasi yang digunakan adalah : $\exists$, dibaca terdapat, ada, beberapa, sedangkan untuk penulisan adalah : $\exists y \in S \rightarrow q(y)$. Terdapat *y* dalam semesta *S* mempunyai sifat *q*.
Contoh:

1. Ada siswa di kelas ini yang ngantuk
2. Beberapa mahasiswa ada yang mendapat nilai A untuk mata kuliah Pemrograman.

1.2.9. Ingkaran Kalimat Berkuantor

Ingkaran dari suatu pernyataan menghasilkan kondisi yang berlawanan dari pernyataan awalnya. Sebuah ingkaran atau negasi akan menghasilkan nilai kebenaran yang berbeda dari pernyataannya. Jika pernyataan awal bernilai benar, maka ingkarannya bernilai salah. Begitu sebaliknya. Notasi yang digunakan adalah sebagai berikut:

(∀x) p(x) **=** ***(∃y) p(y)***

(∃ y) q(y) **=** ***(∀x) q(x)***

Contoh:

p : Semua mahasiswa di kelas ini enjoy belajar logika Informatika

~p : Ada mahasiswa di kelas ini yang tidak enjoy belajar logika Informatika

q : Ada pejabat yang korupsi

~q : Semua pejabat tidak korupsi

1.3. Soal Latihan

Kerjakan soal-soal di bawah ini!

1. Tentukan nilai kebenarannya dengan menggunakan tabel kebenaran!
   a) ***not (p and (not p)) or q***
   b) ***(if p then q) or (r and (not p))***
2. Diberikan kalimat logika:

   If (if q then not p) then (not q and p) else not ((p or s) if and only if (if r then q))

   Maka tentukan *truth value*-nya, jika ;
   a) Interpretasi p, q, r, dan s ***true***
   b) Interpretasi p, q, r, dan s ***false***
   c) Interpretasi p dan q ***true***, r dan s ***false***
   d) Interpretasi p dan q ***false***, r dan s ***true***
3. Dengan menggunakan tabel kebenaran (*truth value*), tentukan nilai kebenaran dari kalimat logika berikut:
   a) ***(p and (if r then s)) if and only if ((if r then s) and p)***
   b) ***(if not p then not s) or ((if q then s) and p)***

4. Dengan mengasumsikan ***p*** dan ***r*** benar, serta ***q*** dan ***s*** salah, tentukan nilai kebenaran dari setiap kalimat logika (*sentences*), berikut
   a) ***(p and (if r then s)) if and only if ((if r then s) and p)***
   b) ***(if not p then not s) or ((if q then s) and p)***
   c) ***((p or q) and not r) if and only if ((if p then r) and (if q then r)***
   d) ***if ((if not q then p) or not q) then (p if and only if q) else not (r and q)***

BAB II. KONSEP DASAR PEMROGRAMAN

2.1 Tujuan Pembelajaran

Setelah mengikuti modul ini, mahasiswa diharapkan mampu:

1. Menjelaskan tentang dasar-dasar bahasa pemrograman
2. Menjelaskan tentang elemen dasar bahasa pemrograman dari variabel dan tipe data yang digunakan
3. Menjelaskan dan mencontohkan operasi input dan output pada bahasa pemrogramanMemahami konsep dasar dari pemrograman
4. Menjelaskan konsep pemrograman secara umum

2.2 Dasar Teori

2.2.1. Konsep Dasar Pemrograman

Program adalah deretan instruksi yang digunakan untuk mengendalikan komputer, sehingga komputer dapat melakukan tindakan sesuai dengan yang dikehendaki pembuatnya. Sebuah program bisa dikatakan baik jika algoritmanya jelas terstruktur dan mudah dibaca oleh orang lain. Sedangkan Algoritma yaitu langkah-langkah untuk menyelesaikan sesuatu masalah.

Adapun pengertian dari Pemrograman Terstruktur yaitu metode untuk mengorganisasikan dan membuat kode-kode program supaya mudah untuk dimengerti, mudah di test dan di modifikasi. Dan pengertian dari struktur data yaitu cara penyimpanan dan pengorganisasian data-data pada memori komputer maupun file secara efektif sehingga dapat digunakan secara efisien, termasuk operasi-operasi di dalamnya. Dimana programnya dapat dipergunakan oleh pengguna secara mudah dan dapat dimengerti tentang proses yang sedang dilakukan oleh program tersebut. Serta dapat mengatur kebutuhan akan piranti masukan dan keluaran.

Setiap Bahasa pemrograman mempunyai kelebihan dan kekurangan masing-masing. Kriteria yang digunakan untuk menilai kelebihan dan kekurangan suatu bahasa Pemrograman antara lain :

1. Ekspresif

   Bahasa pemrograman yang baik harus jelas dalam menggambarkan algoritmanya yang dibuat.

2. Definitas (dapat didefinisikan dengan baik)

   Bahasa Pemrograman dapat didefinisikan dari adanya sintak dan semantik baik. Sintak dan semantik ini haruslah konsisten dan tidak bermakna ganda.
3. Tipe data dan Strukturnya

   Bahasa pemrograman yang baik harus berkemampuan dalam mendukung berbagai tipe data (integer, string, real). Serta struktur data (array, record, file).
4. Modularitas

   Bahasa pemrograman yang baik harus memiliki fasilitas sub program. Program yang besar dapat dikerjakan oleh beberapa pemrogram secara bersama-sama yang nantinya dengan mudah dapat digabungkan menjadi sebuah modul saja.
5. Adanya Input Output

   Bahasa pemrograman yang baik harus dapat mendukung berbagai jenis model file seperti sequential, random, index dsb dalam proses masukan dan keluaran.
6. Portabilitas

   Bahasa pemrograman yang dapat digunakan pada berbagai tipe mesin computer yang berbeda-beda.
7. Efisiensi

   Bahasa pemrograman yang dapat mengatur banyaknya instruksi program dalam membatasi waktu tempuh pemrosesan, mengatur jumlah memori yang digunakan program.
8. Interaktif

   Bahasa pemrograman yang baik harus mudah dipelajari dan diajarkan pada user. Serta dimengerti tentang proses yang sedang dilakukannya.
9. Umum

   Bahasa pemrograman yangn baik harus memiliki jangkauan yang luas untuk berbagai aplikasi pemrograman sehingga dapat bersifat bahasa serbaguna.

Berdasarkan beberapa kriteria diatas, seorang programmer dapat menentukan bahasa pemrograman manakah yang harus digunakan. Adanya pemahaman tentang berbagai bahasa pemrograman, programmer dapat membandingkan mana yang baik dan mana yang sekiranya kurang tepat untuk digunakan.

2.2.2. Tahapan Membuat Program

Pemrograman adalah sebuah rangkaian instruksi-instruksi dalam bahasa komputer yang disusun secara logis dan sistematis. Proses pemrograman komputer bertujuan untuk memecahkan suatu masalah dan membuat mudah pekerjaan dari user atau pengguna komputer. Adapun tahapan pemmrograman yang kompleks sebagai berikut:

1. Definisi Masalah

   Menentukan model/rancangan apa yang akan dibuat untuk penyelesaian masalah.
2. Analisa Kebutuhan

   Menentukan data untuk masukan dan keluaran yang diminta, bahasa pemrograman yang digunakan serta tipe komputer apa yang dibutuhkan.
3. Pembuatan Algoritma/Desain algoritma

   Membuat susunan langkah-langkah/instruksi penyelesaian masalah. Hal ini dapat dilakukan dengan 2 cara:
   a. Menggunakan Flowchart
   b. Menggunakan bahasa semu (*pseudocode*)
4. Pemrograman (dengan bahasa pemrograman)

   Pembuatan program dengan menggunakan bahasa pemrograman.
5. Pengujian Program

   Dapat dilakukan melalui 2 tahap:
   a. Pengujian Tahap *Debuging*

      Untuk mengecek kesalahan program, baik sintaksis maupun logika.
   b. Pengujian tahap *profiling*.
      - Untuk menentukan waktu tempuh dan banyak nya memori program yang digunakan.
      - Setelah program bebas dari kesalahan sehingga dapat dilakukan proses *excute* program.
6. Dokumentasi yang digunakan untuk *file backup*.
7. Pemeliharaan

   Upaya yang dilakukan dengan menghindari kerusakan atau hilangnya suatu program baik *hardware* maupun *Human Error*.

2.3. Latihan Soal

Kerjakan soal di bawah ini!

1. A dan B adalah dua orang bersaudara. A lebih tua daripada B, dengan selisih usia mereka adalah X tahun. Usia A dan usia B adalah Y. Berapakah usia A dan B?
2. Suatu segitiga mempunyai sisi dengan panjang berupa SA, SB dan SC. Berapakah luas segitiga tersebut?

BAB III. PENGENALAN BAHASA PEMROGRAMAN C++

3.1 Tujuan Pembelajaran

Setelah mengikuti modul ini, mahasiswa diharapkan mampu :

1. Menjelaskan tentang dasar-dasar bahasa pemrograman
2. Menjelaskan tentang elemen dasar bahasa pemrograman dari variabel dan tipe data yang digunakan
3. Menjelaskan dan mencontohkan operasi input dan output pada bahasa pemrograman
4. Memahami konsep dasar dari pemrograman
5. Menjelaskan konsep pemrograman secara umum

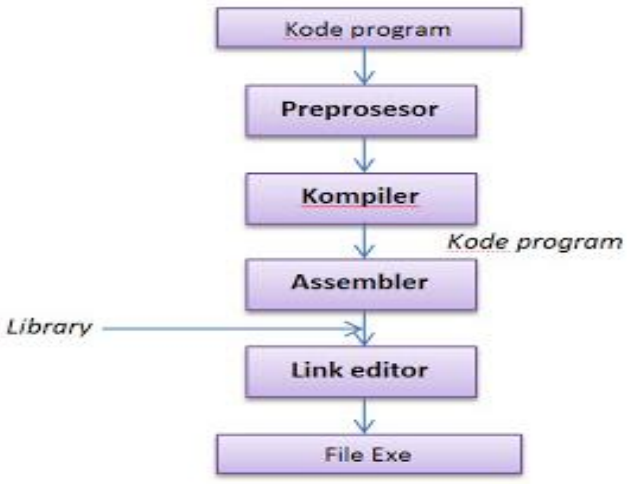
3.2 Dasar Teori

3.2.1. Pengenalan C++

C++ merupakan bahasa pemrograman yang mendukung pemrograman berorientasi objek atau *Object Oriented Programming (OOP).* C++ bersifat *case sensitive* yang artinya membedakan antara huruf besar dan huruf kecil, sehingga:

1. Instruksi harus ditulis dalam huruf kecil
2. Variabel yang ditulis dengan huruf kecil dan huruf besar berbeda.

Sebelum melangkah lebih jauh sebaiknya kita harus tahu terlebih dahulu tentang konsep kompilasi dan eksekusi program di dalam C++.



Gambar 3.1 Konsep Kompilasi dan Eksekusi Program C++

(Sumber: tutscpluplus.blogspot.com/2012/12/konsep-kompilasi-dan-eksekusi-program.html)

Keterangan:

Preprosesor

Perintah yang diawali tanda # (*pound*) dan menghasilkan file yang akan dilewatkan ke dalam *compiler*.

Contoh: # include
define
ifdef

Kompiler C++

Kompiler akan menerjemahkan kode program yang telah dilewatkan oleh *preprosesor* ke dalam bahasa *assembly*.

Assembler

Menerima keluaran dari compiler C++ dan akan membuat sebuah kode objek. Jika dalam kode program kita tidak menggunakan fungsi-fungsi yang terdapat pada *library* lain, maka kode objek ini akan langsung dieksekusi menjadi EXE.

Link editor

Bagian ini dikerjakan jika kode program yang kita buat menggunakan fungsi-fungsi luar yang disimpan dalam suatu *library* lain. *Link editor* akan mengkombinasikan kode objek dan *library* yang ada untuk menjadikan sebuah file EXE.

3.2.2. Struktur C++

Secara umum struktur program dalam C++ adalah sebagai berikut:

```
deklarasi header file/preprocessor
deklarasi konstanta
deklarasi var global
deklarasi fungsi
deklarasi class

program utama (fungsi utama)

main() {
  ............
  ............ }
```

Keterangan:

1. Deklarasi header file diawali dengan **#include <.....>** Titik-titik diisi dengan nama header file yang ingin digunakan. Header file merupakan suatu fungsi yang mendukung eksekusi instruksi tertentu dalam C++.
2. Deklarasi konstanta (bisa ada bisa tidak).
3. Deklarasi variabel global (bisa ada bisa tidak).
4. Deklarasi fungsi bisa ada bisa tidak. Fungsi juga dapat diletakkan setelah fungsi main( ).
5. Deklarasi class (bisa ada bisa tidak).
6. Fungsi utama

   Fungsi utama harus ada, diawali dengan ”{” dan diakhiri dengan ”}”, dan didalamnya berisi instruksi-instruksi yang nantinya akan dieksekusi berikut deklarasi variabel (variabel lokal) yang diperlukan.

Contoh Struktur Dasar Pemrograman C++ dapat terlihat pada gambar 3.2.

```
#include <iostream>

using namespace std;

int main()
{
   cout << "Hello World" << endl;

   return 0;
}
```

Gambar 3.2 Struktur Dasar Pemrograman C++

Keterangan:

`#include <iostream>`

Merupakan singkatan dari *input output stream header* yang digunakan sebagai standar input output operasi yang digunakan oleh bahasa C++.

`using namespace std;`

Merupakan perintah yang digunakan untuk mendeklarasikan atau memberitahukan kepada *compiler* bahwa kita akan menggunakan semua fungsi/class/file yang terdapat dalam *namespace std*.

```
int main()
{
 cout<<"Hello Word"<<endl;
 return 0;
}
```

Merupakan program utama yang diawali dengan kurung kurawal buka "{" dan diakhiri dengan kurung kurawal tutup "}", sedangkan isi dari program tersebut berada dalam kurung kurawal.

3.2.3. Header File

File header (file dengan ekstensi .h) adalah file yang berisi fungsi-fungsi yang mendukung eksekusi instruksi tertentu dalam C++ dan telah dikompilasi sebelumnya. Jika kita ingin menggunakan *file header* kita harus mendaftarkan terlebih dahulu melalui *preprosesor directive* #include

Contoh file header yang sering digunakan :

1. `#include <iostream>`
   Untuk fungsi: `cout` dan `cin`
2. `#include <conio.h>`
   Untuk fungsi: `getch()` dan `clrscr()`
3. `#include <iomani.h>`
   Untuk fungsi: `setiosflags (ios::fixed)` dan `setprecision(n)`
4. `#include <math.h>`
   Untuk fungsi: `sqrt(x)` dan `pow(x,y)`

3.2.4. Perintah Dasar C++

Untuk bisa membuat program maka harus paham minimal perintah-perintah dasar dari pemrograman C++. Beberapa contoh perintah dasar tersebut yaitu:

3.2.4.1. Input dan Output

Untuk membuat program perintah input dan output sangat dibutuhkan karena untuk komunikasi antara user dengan *interface* atau program yang telah dibuat. Beberapa contoh perintah input dan output yang sering digunakan seperti berikut:

cin>>

cin adalah kepanjangan dari c input yaitu perintah yang digunakan untuk menginputkan, perintah ini cukup dengan header file `iostream.`

cout<<

cout adalah kepanjangan dari c output yaitu perintah yang digunakan untuk menampilkan, perintah ini cukup dengan header file `iostream.`

printf()

printf adalah perintah yang digunakan untuk menginputkan, jika menggunakan perintah ini maka harus menambahkan header file `stdio.h`

scanf()

scanf adalah perintah yang digunakan untuk menampilkan, jika menggunakan perintah ini maka harus menambahkan header file `stdio.h`

3.2.4.2. Pindah Baris dan Tabulasi

Untuk membuat tampilan program yang baik pasti akan terdiri dari beberapa baris, dalam bahasa pemrograman C++ untuk pindah baris ada dua perintah yaitu:

Tanda \n

Tanda \n digunakan untuk pindah baris berikutnya, untuk penulisannya harus di dalam tanda petik bersama dengan kalimat yang akan ditampilkan dan bisa diletakkan awal kalimat atau di akhir kalimat. Contoh penggunaannya:

```
cout<<"\nHallo...selamat datang...";
```

Setelah program dieksekusi akan pindah baris terlebih dahulu sebelum menampilkan kalimat Hallo...selamat datang.

```
cout<<"Hallo...selamat datang\n";
```

Setelah program dieksekusi akan menampilkan kalimat Hallo...selamat datang kemudian kursor pindah ke bawah.

```
cout<<"\nHallo...selamat datang\n";
```

Setelah program dieksekusi akan pindah baris terlebih dahulu kemudian menampilkan kalimat Hallo...selamat datang kemudian kursor pindah ke bawah.

Tanda <<endl

Tanda <<endl berfungsi sama dengan tanda \n yaitu digunakan untuk pindah baris berikutnya, untuk penulisannya harus di luar tanda petik dan bisa diletakkan sebelum menuliskan perintah atau kalimat yang akan ditampilkan.

Contoh penggunaannya:

```
cout<<endl<<"Hallo...selamat datang...";
```

Setelah program dieksekusi akan pindah baris terlebih dahulu sebelum menampilkan kalimat Hallo...selamat datang.

```
cout<<"Hallo...selamat datang"<<endl;
```

Setelah program dieksekusi akan menampilkan kalimat Hallo...selamat datang kemudian kursor pindah ke bawah.

```
cout<<endl<<"Hallo...selamat datang"<<endl;
```

Setelah program dieksekusi akan pindah baris terlebih dahulu kemudian menampilkan kalimat Hallo...selamat datang kemudian kursor pindah ke bawah.

Tanda \t

Tanda \t adalah simbol tab yang digunakan untuk mengatur paragrap dengan posisi menjorok ke dalam.

3.2.5. Komentar

Proses pembuatan program yang kompleks akan banyak sekali listing program yang kita ketikkan sehingga cukup rumit dan sulit untuk dipahami. Agar program kita bisa dipahami oleh orang lain akan lebih baik jika setiap pokok bahasan kita berikan komentar. Komentar-komentar ini tidak akan ikut dieksekusi. Dalam C++ ada dua cara untuk memberikan komentar yaitu:

Menggunakan tanda //

Tanda ini digunakan untuk komentar yang terdiri dari satu baris, jika komentar lebih dari satu baris maka baris kedua dan seterusnya tidak dianggap sebagai komentar. Contoh penggunaanya:

```
// Mari kita belajar C++
```

Menggunakan tanda /* ... */

Tanda ini digunakan untuk komentar yang terdiri lebih dari satu baris, komentar dimulai dari `/*` dan berakhir sampai dengan tanda `*/`. Contoh penggunaanya:

```
/*
Belajar mambuat program sederhana Manggunakan bahasa
C++
*/
```

3.2.6. Menggunakan Dev C++

Berikut adalah contoh program dengan menggunakan Dev C++.

Program1.cpp

```
1  //ini contoh 1
2  #include <iostream>
3
4  using namespace std;
5
6  int main() {
7    cout<<"\nMari belajar bersama";
8    cout<<"Membuat program sederhana";
9    cout<<"\nDengan bahasa C++\n";
10 return 0;}
```

Output program1.cpp

```
Mari belajar bersamaMembuat program sederhana
Dengan bahasa C++
```

Program2.cpp

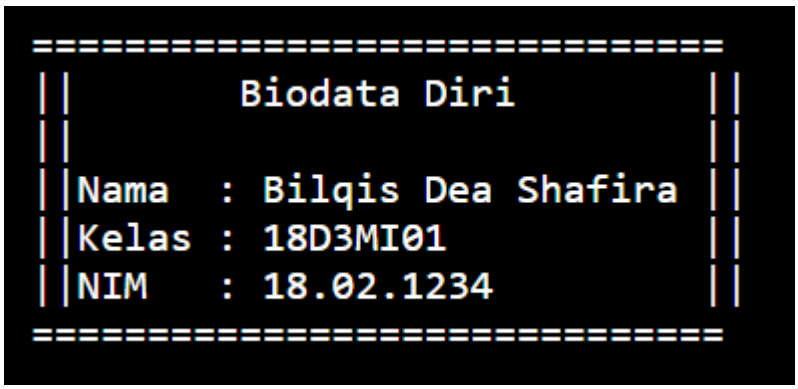
```
1  //ini contoh 2
2  #include <iostream>
3
4  using namespace std;
5
6  int main() {
7    cout<<"\nMari belajar bersama";
8    cout<<endl<<"Membuat program sederhana";
9    cout<<"\nDengan bahasa C++\n";
10 return 0;}
11
```

Output program2.cpp

```
Mari belajar bersama
Membuat program sederhana
Dengan bahasa C++
```

3.3. Latihan Soal

Buatlah program untuk menuliskan biodata masing-masing seperti pada contoh berikut:



BAB IV. TIPE DATA, VARIABEL DAN KONSTANTA

4.1. Tujuan Pembelajaran

Setelah mempelajari materi ini, mahasiswa diharapkan mampu :

1. Memahami definisi tipedata, variabel dan konstanta
2. Mendeklarasikan variabel dan konstanta dengan tipedata yang sesuai
3. Memanggil variabel dan konstanta
4. Membuat program sederhana menggunakan perintah input, output, variabel dan konstanta

4.2. Dasar Teori

4.2.1. Tipe Data

Tipe data digunakan untuk membatasi nilai dari suatu variabel. Nilai dari suatu variabel dibatasi karena untuk efisiensi pemakaian memori. Tipe data berdasarkan jenisnya dapat dibagi menjadi empat yaitu:

1. Tipe integer

   Variabel yang bertipe integer digunakan untuk menyimpan data-data bernilai bilangan bulat.
2. Tipe float

   Variabel yang bertipe integer digunakan untuk menyimpan data-data bernilai bilangan pecahan.
3. Tipe char

   Variabel yang bertipe integer digunakan untuk menyimpan data-data berupa karkter yaitu huruf, angka dan simbol.
4. Tipe tanpa bertipe (void)

   Variabel ini digunakan jika suatu fungsi tidak menghasilkan nilai.

   Jenis-jenis tipe data yang umum digunakan terlihat pada tabel 4.1.

Tabel 4.1 Jenis-jenis Tipe Data

Name	Description	Size*	Range*
char	Character or small integer.	1byte	signed: -128 to 127 unsigned: 0 to 255
short int (short)	Short Integer.	2bytes	signed: -32768 to 32767 unsigned: 0 to 65535
int	Integer.	4bytes	signed: -2147483648 to 2147483647 unsigned: 0 to 4294967295
long int (long)	Long integer.	4bytes	signed: -2147483648 to 2147483647 unsigned: 0 to 4294967295
bool	Boolean value. It can take one of two values: true or false.	1byte	true or false
float	Floating point number.	4bytes	3.4e +/- 38 (7 digits)
double	Double precision floating point number.	8bytes	1.7e +/- 308 (15 digits)
long double	Long double precision floating point number.	8bytes	1.7e +/- 308 (15 digits)
wchar_t	Wide character.	2bytes	1 wide character

4.2.2. Variabel

Variabel merupakan sarana yang digunakan untuk menyimpan data secara sementara atau bersifat dinamis. Nama variabel (identifier) bebas, tetapi harus memperhatikan hal-hal sebagai berikut:

1. Tidak boleh ada spasi.
2. Tidak boleh mengandung operator aritmatik.

3. Tidak boleh diawali dengan angka.
4. Tidak boleh merupakan *reserved word* dalam bahasa pemrograman.
5. Mencerminkan data yang akan disimpan.

Berikut ini adalah bentuk umum pendeklarasian variabel:

tipe_data nama_variabel;

Jika kita ingin mendeklarasikan beberapa variabel yang bertipe sama, maka pendeklarasiannya bisa disingkat sebagai berikut :

tipe_data nama_variabel1,nama_variabel2;

Contoh :

```
float b
int c,d,e;
char nama[20];
```

Variabel juga dapat diberikan nilai dengan menggunakan tanda sama dengan "=" dengan contoh sebagai berikut:

```
int a=4;
char huruf='A';
char judul[20]="Pemrograman Terstruktur"
```

Dalam bahasa C++ tanda sama dengan "=" juga dapat digunakan untuk memberikan *multiple assigment* seperti contoh :

```
a=10;
b=10;
c=10;
```

bisa ditulis dengan a=b=c=10;

4.2.2.1. Variabel Lokal

Variabel lokal adalah variabel yang hanya dikenali oleh suatu fungsi saja, artinya tidak dikenal oleh lingkungan luar di dalam program yang kita buat. Contoh 1 penulisan variabel lokal:

```
3  #include <iostream>
4  using namespace std;
5
6  int main(){
7    int a=4; //variabel lokal
8
9    cout<<"Nilai a adalah : "<<a;
10
11 return 0;}
```

Output contoh 1:

```
Nilai a adalah : 4
```

4.2.2.2. Variabel Global

Variabel global yaitu variabel yang dapat dikenali oleh semua lingkungan dalam program kita. Letak pendeklarasian variabel global berada di luar fungsi utama. Contoh 2 penulisan variabel global:

```
3  #include <iostream>
4  using namespace std;
5
6  int a=4; //variabel global
7
8  int main(){
9    cout<<"Nilai a adalah : "<<a;
10 return 0;}
```

Output contoh 2:

```
Nilai a adalah : 4
```

4.2.3. Konstanta

Konstanta adalah jenis *identifier* yang bersifat konstan atau tetap artinya nilai dari konstanta di dalam program tidak dapat diubah. Dalam bahasa C++ ada dua cara untuk mendeklarasikan konstanta, yaitu :

1. **Menggunakan Preprosesor Directive #define**

   Letak pendeklarasian konstanta menggunakan #define adalah berada di bawah preprosesor directive, bentuk umumnya adalah :

   #define nama_konstanta nilai_tetap

   Contoh 3 penulisan konstanta menggunakan #define :

```
#include <iostream>
#define phi 3.14
#define teks "Contoh deklarasi konstanta"

using namespace std;

int main(){

cout<<phi<<"\n";
cout<<teks<<"\n";

return 0;}
```

Output contoh 3:

```
3.14
Contoh deklarasi konstanta
```

2. Menggunakan Kata Kunci const

Selain dengan kata kunci #define konstanta dapat dideklarasikan menggunakan kata kunci const, bentuk umumnya adalah :

const tipe_data nama_konstanta=nilai_tetap;

Contoh 4 penulisan konstanta menggunakan const :

```
#include <iostream>
using namespace std;

int main() {

//deklarasi konstanta
const float phi=3.14;
const int r1=3;
const float r2=10.7;
const char kar='A';
const char teks[30]="Contoh deklarasi konstanta";
const bool x=true;

//pemanggilan konstanta
cout<<teks<<"\n";
cout<<kar<<"\n";
cout<<x<<"\n";
cout<<phi<<"\n";
cout<<r1<<"\n";
cout<<r2<<"\n";

return 0;}
```

Output contoh 4:

```
Contoh deklarasi konstanta
A
1
3.14
3
10.7
```

4.3. Latihan Soal

Buatlah program dengan menerapkan menggunakan konstanta untuk menentukan nilai akar dan pangkat sudtu bilangan, seperti hasil di bawah ini!

```
Akar dari angka 4 = 2
4 pangkat 2 = 16
--------------------------------
Process exited after 17.98 seconds with return value 0
Press any key to continue . . .
```

BAB V. OPERATOR, TIPE CASTING DAN PENGATURAN DESIMAL

5.1 Tujuan Pembelajaran

Setelah mempelajari materi ini, mahasiswa diharapkan mampu :

1. Memahami operator gabungan dan *type casting*
2. Membuat program sederhana menggunakan perintah input, output dan konstanta
3. Menggunakan operator gabungan dan *type casting* pada program.
4. Menerapkan struktur proses *increment* dan *decrement*.

5.2 Dasar Teori

5.1.1. Operator

Dalam pemrograman operator sering digunakan untuk mendefinisikan operasi-operasi di dalamnya, baik itu operasi perhitungan, perbandingan maupun operasi-operasi yang lainnya. Operator-operator yang sering digunakan yaitu:

5.1.1.1. Operator Aritmetik

Operator aritmatik sering digunakan untuk operasi-operasi matematika, beberapa operator tersebut dapat dilihat pada tabel 5.1.

Tabel 5.2 Operator Aritmatik

Operator	Jenis Operasi	Contoh
+	Penjumlahan	2 + 3 = 5
-	Pengurangan	6 – 3 = 3
*****	Perkalian	2 * 4 = 8
/	Pembagian	5 / 2 = 2.5
%	Modulo	5 % 2 = 1

Contoh 1 penggunaan operator aritmatik:

```
2  #include <iostream>
3  using namespace std;
4
5  int main() {
6
7   int a,b,plus,min,kali,mod;
8   float bagi;
9
10  cout<<"Contoh penggunaan operator aritmatik\n";
11  cout<<"Masukkan angka pertama (a) = "; cin>>a;
12  cout<<"Masukkan angka kedua   (b)   = "; cin>>b;
13  plus=a+b;
14  min=a-b;
15  kali=a*b;
16  bagi=a/b;
17  mod=a%b;
18  cout<<"\nJumlah "<<a<<"+"<<b<<" = "<<plus;
19  cout<<"\nKurang "<<a<<"-"<<b<<" = "<<min;
20  cout<<"\nKali   "<<a<<"*"<<b<<" = "<<kali;
21  cout<<"\nBagi   "<<a<<"/"<<b<<" = "<<bagi;
22  cout<<"\nModulo "<<a<<"%"<<b<<" = "<<mod;
23
24 return 0;}
```

Output contoh 1 :

```
Contoh penggunaan operator aritmatik
Masukkan angka pertama (a) = 10
Masukkan angka kedua   (b)   = 3

Jumlah 10+3 = 13
Kurang 10-3 = 7
Kali   10*3 = 30
Bagi   10/3 = 3
Modulo 10%3 = 1
```

5.1.1.2. Operator Gabungan

Pada Bahasa pemrograman C++ operator gabungan digunakan untuk menyingkat penulisan perintah dengan syarat salah satu dari variabel yang akan dioperasikan sama dengan variabel hasil atau variabel penampung. Operator gabungan tersebut dapat dilihat pada tabel 5.2 berikut ini.

Tabel 5.2 Operator Gabungan

Operator	Contoh	Ekivalen
+=	bonus += 500;	bonus = bonus + 500;
-=	budget -= 50;	budget = budget - 50;
*=	gaji *= 1.2;	gaji=gaji * 1.2;
/=	faktor/= 50;	faktor= faktor/.50;
%=	jml_hari %=7;	jml_hari =jml_hari % 7;

Contoh 2 penggunaan operator gabungan:

```
#include <iostream>
using namespace std;

int main(){
 int i = 4;
 int j = 8;
 int k = 12,jwb;

 jwb = i + j;
 cout << jwb << "\n";
 jwb += k;
 cout << jwb << "\n";
 jwb /= 3;
 cout << jwb << "\n";
 jwb -= 5;
 cout << jwb << "\n";
 jwb *= 2;
 cout << jwb << "\n";
 jwb %= 4;
 cout << jwb << "\n";
 jwb *= 5+3;
 cout << jwb << "\n";
 jwb += 4-2;
 cout << jwb << "\n";
return 0;}
```

Output contoh 2:

```
12
24
8
3
6
2
16
18
```

5.1.1.3. Operator Logika

Operator logika adalah operator yang digunakan untuk melakukan operasi dimana nilai yang dihasilkan dari operasi tersebut hanya berupa nilai benar (*true*) dan salah (*false*), atau sering disebut dengan nilai boolean. Adapun operator logika tersebut dapat dilihat pada tabel 5.3.

Tabel 5.3 Operator Logika

Operator	Jenis Operator	Contoh
&&	AND (dan)	1 && 1 = 1
||	OR (atau)	1 || 0 = 0
!	NOT (negasi)	!0 = 1

Contoh 3 penggunaan operator logika:

```
#include <iostream>
using namespace std;

int main(){

 cout<<"1 && 1 = "<<(1 && 1)<<endl;
 cout<<"1 && 0 = "<<(1 && 0)<<endl;
 cout<<"0 && 0 = "<<(0 && 0)<<endl;
 cout<<"0 && 1 = "<<(0 && 1)<<endl;

 cout<<"1 || 1 = "<<(1 || 1)<<endl;
 cout<<"1 || 0 = "<<(1 || 0)<<endl;
 cout<<"0 || 0 = "<<(0 || 0)<<endl;
 cout<<"0 || 1 = "<<(0 || 1)<<endl;

 cout<<"!1 = "<<!1<<endl;
 cout<<"!0 = "<<!0<<endl;

return 0;}
```

Output contoh 3:

```
1 && 1 = 1
1 && 0 = 0
0 && 0 = 0
0 && 1 = 0
1 || 1 = 1
1 || 0 = 1
0 || 0 = 0
0 || 1 = 1
!1 = 0
!0 = 1
```

5.2.1.4. Operator Increment dan Descrement

Increase atau *increment* adalah proses penambahan satu (menaikkan satu), dalam bahasa C++ ada dua jenis increase yaitu *pre-increase* (prefix) dan *post-increase* (posfix). *Pre-increase* yaitu akan melakukan penambahan nilai sebelum suatu variabel tersebut diproses, sedangkan *post-increase* merupakan kebalikannya yaitu melakukan proses terlebih dahulu sebelum dilakukan penambahan nilai. Sedangkan *decrease* atau *descrement* adalah proses pengurangan satu (menurunkan satu), untuk *decrease* juga mempunyai dua jenis yaitu *pre-decrease* (prefix) dan *post-decrease* (posfix). *Pre-decrease* yaitu akan melakukan pengurangan nilai sebelum suatu variabel tersebut diproses, sedangkan *post-decrease* merupakan kebalikannya yaitu melakukan proses terlebih dahulu sebelum dilakukan pengurangan nilai. Operator increment dan descrement dapat dilihat pada tabel 5.4.

Tabel 5.4 Operator Increase dan Decrease

Operator	Contoh	Deskripsi	Pernyataan yang ekivalen
++	J++;	posfix	j = j + 1; → j += 1;
++	++j;	prefix	j = j + 1; → j += 1;
--	j--;	postfix	j = j - 1; → j -= 1;
--	--j;	prefix	j = j - 1; → j -= 1;

Contoh 4 Penggunaan operator *Increment* dan *Decrement*

```
#include <iostream>
using namespace std;

int main(){

 int i1=4,i2=4;
 int j1=8,j2=8;
 int a,b,c,d;
 a=++i1;
 b=i2++;
 c=--j1;
 d=j2--;

 cout<<"\nNilai a = "<<a;
 cout<<"\nNilai i1 = "<<i1;
 cout<<"\n\nNilai b = "<<b;
 cout<<"\nNilai i2 = "<<i2;
 cout<<"\n\nNilai c = "<<c;
 cout<<"\nNilai j1 = "<<j1;
 cout<<"\n\nNilai d = "<<d;
 cout<<"\nNilai j2 = "<<j2;
return 0;}
```

Output contoh 4:

```
Nilai a = 5
Nilai i1 = 5

Nilai b = 4
Nilai i2 = 5

Nilai c = 7
Nilai j1 = 7

Nilai d = 8
Nilai j2 = 7
```

5.2.1.5. Operator Rasional Untuk Proses Pengecekan

Operator Rasional ini biasanya digunakan untuk menuliskan kondisi/syarat selalu digunakan operator relasional sebagai sarana untuk melakukan proses pengecekan, operator-operator tersebut terdapat pada tabel 5.5.

Tabel 5.5 Operator-Operator Rasional Untuk Proses Pengecekan

Operator	Arti
>	Lebih besar
<	Lebih kecil
= =	Sama dengan
>=	Lebih atau sama
<=	Kurang atau sama
!=	Tidak sama dengan

5.2.2. Type casting

Tipe *casting* yaitu pembuatan tipe data sementara, artinya mengubah tipe data suatu variabel hanya dalam operasi aritmatika. Setelah operasi aritmatika, tipe data variabel tersebut tetap sesuai deklarasi awalnya. Contoh 5 program sebelum menggunakan tipe *casting*:

```
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5
6   int a,b,mod;
7   float bagi;
8
9   cout<<"Contoh penggunaan / dan %\n\n";
10  cout<<"Masukkan angka pertama (a) = "; cin>>a;
11  cout<<"Masukkan angka kedua   (b) = "; cin>>b;
12  bagi=a/b;
13  mod=a%b;
14  cout<<"\nHasil pembagian = "<<bagi;
15  cout<<"\nSisa hasil bagi = "<<mod;
16
17 return 0;}
```

Output contoh 5:

```
Contoh penggunaan / dan %

Masukkan angka pertama (a) = 10
Masukkan angka kedua   (b) = 3

Hasil pembagian = 3
Sisa hasil bagi = 1
```

Untuk hasil pembagian seharusnya 3,33333 karena dari 10 di bagi 3 adalah 3,33333. Mengapa demikian? Karena variabel a dan b bertipe integer. Tetapi jika variabel a dan b dirubah menjadi tipe data float maka akan error sehingga muncul pesan :

```
Compilation failed due to following error(s).

main.cpp: In function 'int main()':
main.cpp:20:8: error: invalid operands of types 'float' and 'float' to binary 'operator%'
  mod=a%b;
       ^
```

Apa arti dari pesan error tersebut? Arti pesan error tersebut adalah, dua variabel a dan b yang bertipe integer tersebut tidak bisa digunakan untuk operasi aritmatik modulo (mod atau %), sedangkan untuk operasi aritmatik pembagian (/), syaratnya untuk variabel hasil (variabel yang digunakan sebagai penampung hasil pembagian) dan salah satu variabel yang akan dibagi harus bertipe float. Dua operasi aritmatik dengan tipe variabel yang sama dalam satu program ini menjadi tidak bisa dikerjakan. Agar

kedua operasi tersebut bisa berjalan bersama dalam satu program maka dibutuhkan type casting (tipe data sementara), sehingga menjadi sebagai berikut:

Contoh 6 program menggunakan tipe *casting*:

```
#include <iostream>
using namespace std;

int main(){

int a,b,mod;
float bagi;

 cout<<"Contoh penggunaan / dan %\n\n";
 cout<<"Masukkan angka pertama (a) = "; cin>>a;
 cout<<"Masukkan angka kedua   (b) = "; cin>>b;
 bagi=float(a)/b; //atau bagi=a/float(b);
 mod=a%b;
 cout<<"\nHasil pembagian = "<<bagi;
 cout<<"\nSisa hasil bagi = "<<mod;

return 0;}
```

Untuk rumus **bagi=float(a)/b atau bagi=a/float(b)** ini yang disebut dengan tipe casting, pada rumus hasil variabel a sudah berubah tipe data menjadi float dan ini hanya sementara atau hanya berlaku pada rumus hasil atau pada operasi pembagian saja. Ketika pada operasi modulo atau pada rumus mod variabel a sudah kembali lagi ke tipe data integer yang sudah dideklarasikan di awal program, sehingga outputnya menjadi seperti output contoh 6.

Output contoh 6:

```
Contoh penggunaan / dan %

Masukkan angka pertama (a) = 10
Masukkan angka kedua   (b) = 3

Hasil pembagian = 3.33333
Sisa hasil bagi = 1
```

5.2.3. Pengaturan Desimal

Ada dua metode pengaturan desimal dan non desimal yaitu menggunakan instruksi **setprecision(n)** dan menggunakan instruksi **setiosflags(ios::fixed)** kedua perintah tersebut memerlukan header file **iomanip.**

Setprecision(n) adalah perintah yang digunakan untuk menampilkan jumlah digit baik nilai bulat maupun desimalnya, jumlah digit yang ditampilkan menyesuaikan nilai n yang ada pada kurung. Sedangkan perintah setiosflags(ios::fixed)<<setprecision(n) artinya n adalah ketelitian/presisi untuk desimalnya saja, atau bisa dikatakan pesan *space* (tempat) untuk angka di belakang koma. Jadi n yang pada awalnya untuk pengaturan bilangan bulat dan desimal, sekarang dengan tambahan perintah <<setiosflags(ios::fixed) sebelum perintah <<setprecision(3) sudah berubah fungi menjadi pengaturan bilangan desimal saja. Pada contoh 8 terdapat perintah :

cout<<"\n\t Hasil pembagian ="

<<setiosflags(ios::fixed)<<setprecision(3)<<bagi;

artinya pada variabel `bagi` hanya akan menampilkan 3 digit saja untuk bilangan desimalnya.

Perhatikan output dari contoh progam 6 yang mana nilai a = 10 nilai b = 3 dan hasil pembagiannya 3,33333. Yang dimaksud pengaturan decimal ini yaitu pengaturan hasil outputnya. Contoh program pengaturan desimal menggunakan setprecision (n) dapat dilihat pada contoh 7.

Contoh 7 Program pengaturan desimal menggunakan setprecision.

```
#include <iostream>
#include <iomanip>
using namespace std;

int main(){
 int a,b,mod;
 float bagi;

 cout<<"\n\tContoh penggunaan / dan %\n\n";
 cout<<"\tMasukkan angka pertama (a) = "; cin>>a;
 cout<<"\tMasukkan angka kedua   (b) = "; cin>>b;
 bagi=float(a)/b;
 mod=a%b;
 cout<<"\n\tHasil pembagian = "<<setprecision(3)<<bagi;
 cout<<"\n\tSisa hasil bagi = "<<mod;

return 0;}
```

Output contoh 7:

```
Contoh penggunaan / dan %

Masukkan angka pertama (a) = 10
Masukkan angka kedua   (b) = 3

Hasil pembagian = 3.33
Sisa hasil bagi = 1
```

Contoh 8 Program pengaturan desimal menggunakan setiosflags(ios::fixed)

```
#include <iostream>
#include <iomanip>
using namespace std;

int main(){
 int a,b,mod;
 float bagi;

 cout<<"\n\tContoh penggunaan / dan %\n\n";
 cout<<"\tMasukkan angka pertama (a) = "; cin>>a;
 cout<<"\tMasukkan angka kedua   (b) = "; cin>>b;
 bagi=float(a)/b;
 mod=a%b;
 cout<<"\n\tHasil pembagian = "<<setiosflags(ios::fixed)<<setprecision(3)<<bagi;
 cout<<"\n\tSisa hasil bagi = "<<mod;

return 0;}
```

Output contoh 8:

```
Contoh penggunaan / dan %

Masukkan angka pertama (a) = 10
Masukkan angka kedua   (b) = 3

Hasil pembagian = 3.333
Sisa hasil bagi = 1
```

5.3. Latihan Soal

1. Buatlah program untuk menghitung sisi miring dan keliling segitiga siku-siku dengan sisi tegak mendatar merupakan input dari keyboard (diinputkan user), dengan rumus:

   sisimiring = sqrt (alas * alas + tinggi * tinggi) keliling = alas + tinggi + sisi miring

2. Buatlah program untuk menghitung nilai rata-rata dari mata kuliah Algoritma Struktur Data, dengan ketentuan sebagai berikut:

   Nilai praktikum 40 %

   Nilai teori 40 %

   Nilai tugas dan final project 20 %

3. Buatlah program untuk mengetahui cicilan, total harga cicilan dan keuntungan dealer dari pembelian sepeda motor anda, dengan ketentuan sebagai berikut:

   Harga pokok = harga motor / lama kredit (dalam bln)

   Bunga = harga pokok * 0.1

   Cicilan = harga pokok + bunga

   Total harga motor = cicilan * lama kredit (dalam bulan)

   Keuntungan dealer = total harga motor - harga motor

BAB VI. PERCABANGAN

6.1. Tujuan Pembelajaran

Setelah mempelajari materi ini, mahasiswa diharapkan mampu :

1. Mampu menjelaskan pengertian percabangan
2. Menerapkan struktur percabangan (if, if else, dan switch) untuk menyelesaikan kasus.
3. Membedakan bagaimana alur atau jalannya program berdasarkan struktur if, if else, dan switch.

6.2. Dasar Teori

Dalam kehidupan sehari-hari pasti kita sering mengalamai berbagai masalah yang diharuskan untuk memilih suatu kondisi tertentu. Begitu juga dalam pembuatan program sering ditemui berbagai masalah salah satu masalah tersebut yaitu proses seleksi seleksi atau pemilihan statemen atau biasa disebut dengan percabangan.

Percabangan adalah suatu pemilihan statemen yang akan dieksekusi dimana pilihan tersebut didasarkan atas kondisi tertentu untuk mengarahkan perjalanan suatu proses. Artinya statemen yang terdapat pada suatu blok percabangan akan dieksekusi jika kondisi yang didefinisikan terpenuhi (bernilai benar) tetapi jika kondisi tersebut tidak terpenuhi (bernilai salah) maka statemen tersebut tidak akan dieksekusi atau diabaikan oleh *compiler*.

Penyeleksian kondisi dapat diibaratkan sebagai katup atau kran yang mengatur jalannya air. Bila katup terbuka maka air akan mengalir dan sebaliknya bila katup tertutup air tidak akan mengalir atau akan mengalir melalui tempat lain. Dalam percabangan ada dua jenis struktur percabangan yang digunakan untuk mengimplementasikan suatu percabangan, yaitu `if` dan `switch case`.

6.2.1. Struktur if

Struktur percabangan IF adalah struktur percabangan yang paling sering digunakan dan bisa digunakan untuk menangani semua masalah karena dapat digunakan untuk semua kondisi baik dengan nilai syarat yang pasti atau dengan nilai range atau nilai syarat tidak pasti. Pada percabangan `if` terbagi dalam tiga kondisi sebagai berikut:

6.2.1.1. Satu Kondisi

Struktur ini merupakan struktur yang paling sederhana karena hanya melibatkan satu buah ekspresi yang akan diperiksa. Jika kondisi bernilai benar maka pernyataan akan dikerjakan tetapi jika kondisi salah, tidak akan mengerjakan apapun didalam instruksi `if` (langsung menuju ke instruksi berikutnya). Adapun struktur percabangan `if` satu kondisi adalah sebagai berikut:

```
//jika hanya terdiri satu pernyataan
if (kondisi)
   pernyataan;

/*
jika terdapat lebih dari satu pernyataan,maka penulisan
pernyataan harus berada dalam kurung kurawal {}
*/

if (kondisi)
 { pernyataan1;
   pernyataan2;
   ......
   pernyataan_n;
 }
```

Contoh 1 program `if` satu kondisi

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main(){
6  int nilai;
7
8  cout<<"Masukkan Nilai = "; cin>>nilai;
9
10 if (nilai>0)
11    cout<<"Nilai yang anda inputkan bilangan positif";
12
13 return 0;}
```

Output contoh 1:

```
Masukkan Nilai = 8
Nilai yang anda inputkan bilangan positif
```

Keterangan:

`cout<<"Masukkan Nilai = "; cin>>nilai;`

Perintah di atas adalah perintah untuk menginputkan angka/bilangan ke dalam variabel nilai. Dan variabel nilai tersebut yang akan digunakan sebagai kondisi untuk percabangan.

```
if (nilai>0)
```

Perintah di atas merupakan kondisi dimana kondisi tersebut bernilai benar jika angka/bilangan yang diinpukan ke dalam variabel nilai lebih besar dari nol (0).

```
cout<<”Nilai yang anda inputkan bilangan positif”;
```

Perintah di atas adalah pernyataan yang akan dikerjakan jika kondisi tersebut benilai benar atau terpenuhi. Tetapi jika kondisi tersebut bernilai salah atau tidak terpenuhi maka tidak akan melakukan pernyataan apapun.

6.2.1.2. Dua Kondisi

Struktur percabangan dua kondisi sedikit lebih kompleks bila dibandingkan dengan struktur yang hanya memiliki satu kondisi. Karena dalam struktur ini memiliki dua pernyataan yaitu pernyataan yang akan dikerjakan jika kondisi bernilai benar atau terpenuhi dan pernyataan yang akan dikerjakan jika kondisi tersebut bernilai salah atau tidak terpenuhi. Adapun strukturnya yaitu:

```
if (kondisi)
 {pernyataan jika kondisi terpenuhi; }
else
      {pernyataan jika kondisi tidak terpenuhi;}
```

Contoh 2 program IF dua kondisi

```
1  #include <iostream>
2   using namespace std;
3
4  int main(){
5  int nilai;
6
7  cout<<"Masukkan Nilai = ";cin>>nilai;
8
9  if (nilai % 2 == 0)
10  cout<<"Nilai yang anda inputkan adalah bilangan genap";
11 else
12  cout<<"Nilai yang anda inputkan adalah bilangan ganjil";
13
14 return 0;}
```

Output 2 program IF dua kondisi

```
Masukkan Nilai = 4
Nilai yang anda inputkan adalah bilangan genap
```

```
Masukkan Nilai = 5
Nilai yang anda inputkan adalah bilangan ganjil
```

Keterangan:

```
cout<<"Masukkan Nilai = ";cin>>nilai;
```

Perintah di atas adalah perintah untuk menginputkan angka/bilangan ke dalam variabel nilai. Dan variabel nilai tersebut yang akan digunakan sebagai kondisi untuk percabangan.

```
if (nilai % 2 == 0)
```

Perintah di atas merupakan kondisi dimana kondisi tersebut bernilai benar jika angka/bilangan yang diinpukan ke dalam variabel nilai ketika dibagi 2 (mod 2) mempunyai sisa nol(0).

```
cout<<"Nilai yang anda inputkan adalah bilangan genap";
```

Perintah di atas adalah statemen yang akan dikerjakan jika kondisi tersebut benilai benar atau terpenuhi.

```
else
```

Perintah di atas merupakan perintah alernatif, artinya jika sudah tidak ada kondisi lagi dalam percabangan tersebut maka tuliskan perintah else sebelum statemen untuk menuju ke statemen alternatif.

```
cout<<"Nilai yang anda inputkan adalah bilangan ganjil";
```

Perintah di atas adalah statemen alternatif yang akan dikerjakan jika kondisi bernilai salah atau tidak terpenuhi.

6.2.1.3. Multi Dimensi

Percabangan jenis ini merupakan perluasan dari struktur percabangan dengan satu dan dua kondisi. Karena dalam struktur ini memiliki lebih dari dua kondisi dan lebih dari 2 pernyataan. Strukturnya yaitu:

```
if (kondisi_1)
 pernyataan jika kondisi1 terpenuhi;
else if (kondisi_2)
 pernyataan jika kondisi2 terpenuhi;
```

```
else if (kondisi_3)
 pernyataan jika kondisi3 terpenuhi;
else
 pernyataan jika semua kondisi diatas tidak terpenuhi;
```

Contoh 3 program IF multi kondisi

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main(){
6  int bil;
7
8  cout<<"Masukkan Bilangan yang akan dicek = ";
9  cin>>bil;
10 if (bil > 0)
11  cout<<bil<<" adalah bilangan Positif";
12 else if (bil < 0)
13  cout<<bil<<" adalah bilangan Negatif";
14 else
15  cout<<"Anda menginputkan bilangan Nol (0)";
16
17 return 0;}
```

Output 3 program IF dua kondisi

```
Masukkan Bilangan yang akan dicek = 2
2 adalah bilangan Positif
```

```
Masukkan Bilangan yang akan dicek = -2
-2 adalah bilangan Negatif
```

```
Masukkan Bilangan yang akan dicek = 0
Anda menginputkan bilangan Nol (0)
```

Keterangan:

`cout<<"Masukkan Bilangan yang akan dicek = "; cin>>bil;`

Perintah di atas adalah perintah untuk menginputkan angka/bilangan ke dalam variabel bil. Dan variabel bil tersebut yang akan digunakan sebagai kondisi untuk percabangan.

`if (bil > 0)`

Perintah di atas merupakan kondisi pertama dimana kondisi tersebut bernilai benar jika angka/bilangan yang diinpukan ke dalam variabel bil lebih besar dari nol(0).

```
cout<<bil<<" adalah bilangan Positif";
```

Perintah di atas adalah pernyataan yang akan dikerjakan jika kondisi tersebut benilai benar atau terpenuhi. Jika pada kondisi satu sudah terpenuhi maka percabangan tidak akan dilanjutkan lagi ke kondisi kedua.

```
else if (bil < 0)
```

Perintah di atas adalah kondisi kedua, kondisi ini akan dikerjakan jika kondisi pertama bernailai salah atau tidak terpenuhi, kondisi kedua ini benar jika angka dalam variabel bil lebih kecil dari nol (0).

```
cout<<bil<<" adalah bilangan Negatif";
```

Perintah di atas adalah pernyataan yang akan dikerjakan jika kondisi kedua terpenuhi, dan percabangan akan berhentikan atau tidak akan dilanjutkan. Tetapi jika kondisi kedua tidak terpenuhi maka percabangan akan terus dilakukan sampai proses selesai yaitu sampai statemen alternatif.

```
else
```

Perintah di atas merupakan perintah alernatif, artinya jika sudah tidak ada kondisi lagi dalam percabangan tersebut maka tuliskan perintah else sebelum pernyataan untuk menuju ke statemen alternatif.

```
cout<<"Anda mengInputkan bilangan Nol (0)";
```

Perintah di atas merupakan pernyataan alternatif yang akan dikerjkan jika semua kondisi sudah tidak ada yang terpenuhi.

6.2.2. Struktur case

Perintah ini digunakan sebagai alternatif pengganti dari percabangan `if` secara sederhana dimana alternatif pilihan bisa lebih dari satu. Pada dasarnya percabangan menggunakan struktur `if` dan `switch` sama hanya saja `switch-case` digunakan untuk pilihan berjumlah banyak dan perintah `switch-case` tidak bisa digunakan untuk pilihan yang melibatkan jangkauan nilai atau range tetapi hanya bisa digunakan untuk pilihan berupa konstanta atau dengan nilai yang sudah pasti.

Karakteristik `switch-case` adalah :

1. Perintah switch akan menyeleksi kondisi yang diberikan dan kemudian membandingkan hasilnya dengan konstanta-konstanta yang berada di `case`.

2. Pembandingan akan dimulai dari konstanta 1 sampai konstanta terakhir. Jika hasil dari kondisi sama dengan nilai konstanta tertentu, misalnya konstanta 1, maka pernyataan 1 akan dijalankan sampai ditemukan `break`.
3. Pernyataan `break` akan membawa proses keluar dari perintah `switch`. Jika hasil dari kondisi tidak ada yang sama dengan konstanta-konstanta yang diberikan, maka pernyataan pada default akan dijalankan.

Bentuk umum struktur percabangan `switch` yaitu:

```
switch (ekspresi){
 case nilai_konstanta1 :
      pernyataan;
      break;
 case nilai_konstanta2 :
      pernyataan;
      break;
 …….
 default :
      pernyataan_alternatif;
}
```

Contoh 4 Program Switch Case

```
#include <iostream>
using namespace std;

int main(){
int bil,a;

 cout<<"Masukkan bilangan : "; cin >>a;
 switch (a){
 case 1 :
        cout<<"Hari "<<a<<" : Minggu";
        break;
 case 2 :
        cout<<"Hari "<<a<<" : Senin";
        break;
 case 3 :
        cout<<"Hari "<<a<<" : Selasa";
        break;
 case 4 :
        cout<<"Hari "<<a<<" : Rabu";
        break;
 case 5 :
        cout<<"Hari "<<a<<" : Kamis";
        break;
 case 6 :
        cout<<"Hari "<<a<<" : Jum'at";
        break;
 case 7 :
        cout<<"Hari "<<a<<" : Sabtu";
        break;
 default :
        cout<<"Tidak ada hari ke "<<a;}
return 0;}
```

Output 4 program IF dua kondisi

```
Masukkan bilangan : 2
Hari 2 : Senin
```

Keterangan:

`cout<<"Masukkan bilangan : "; cin >>a;`

Perintah di atas adalah perintah untuk inputkan angka ke dalam variabel a.

`switch (a){`

variabel a sebagai ekspresi yang nanti nilainya akan di cek dalam perintah case, dan nilai yang dikenali hanya dari 1 sampai 7.

`case 1 :`

menunjukkan nilai konstanta yang pertama,

`cout<<"Hari ke "<<a<<" : Minggu"; break;`

perintah ini merupakan pernyataan yang akan dikerjakan jika `case 1` terpenuhi, kemudian berhenti karena ada perintah `break`, jika tidak ada perintah `break` maka pernyataan - pernyataan pada `case 2` dan `case` selanjutnya akan ikut dieksekusi.

`case 2 :`

menunjukkan nilai pernyataan yang kedua, jika angka yang diinputkan pada variabel a adalah 2 dan ini akan dikerjakan jika konstanta pada `case 1` bernilai salah.

`cout<<"Hari ke "<<a<<" : Senin"; break;`

perintah ini merupakan pernyataan yang akan dikerjakan jika `case 2` terpenuhi, kemudian berhenti karena ada perintah `break`, jika tidak ada perintah break maka pernyataan - pernyataan pada `case 3` dan selanjutnya akan ikut dieksekusi.

Perintah ini akan terus dikerjakan sampai pada `case 7,` jika ternyata variabel a bukan berisi angka dari 1 sampai 7 maka pernyataan alternatif dibawah `default` yang akan dikerjakan.

`default :cout<<"Tidak ada hari ke "<<a;}`

Artinya pernyataan setelah `default` akan dikerjakan jika `case 1` sampai `case 7` bernilai salah atau tidak terpenuhi.

6.2.3. Percabangan Bersarang

Percabangan bersarang adalah percabangan yang digunakan apabila terdapat pernyataan percabangan dibawah sebuah pernyataan percabangan lainnya. Percabangan bersarang ini bisa berisi if atau case semua atau kombinasi dari keduanya. Adapun struktur kondisi dari percabangan bersarang sebagai berikut:

```
if(kondisi_1)
{
   if(sub_kondisi_1)
     { sub_pernyataan_1
       ... }
   else
     { sub_pernyataan_lain
```

```
        ... }
else
 { pernyataan_lain
   ... }
```

Contoh 5 Percabangan Bersarang

```
#include <iostream>
#include <string>
using namespace std;

int main(){
   int nilai;
   string index, ket;
   cout << "Masukkan nilai = "; cin  >> nilai;

   if(nilai >= 60)
     {
       ket = "Selamat anda lulus.";
       if(nilai >= 80)
         {
           index = "A";
         }
       else if(nilai >= 70)
         {
           index = "B";
         }
       else
       {
         index = "C";
       }
     }
    else
      {
        ket = "Maaf, anda belum lulus.";
        if(nilai >= 40)
          {
            index = "D";
          }
        else
          {
            index = "E";
          }
      }
    cout << "Status = " << ket << endl;
    cout << "Index Nilai = " << index << endl;
 return 0;}
```

Output Contoh 5 Percabangan Bersarang

```
Masukkan nilai = 98
Status = Selamat anda lulus.
Index Nilai = A
```

6.3. Latihan Soal

1. Buatlah program untuk menghitung gaji karyawan, dengan ketentuan sbb:

   Input :

   a) Nama karyawan → bisa memuat spasi →contoh: rendra rena
   b) NIK
   c) Jenis kelamin [0=perempuan, 1=laki-laki]
   d) Status pernikahan [0=single, 1=menikah]
   e) Kendaraan [0=motor, 1=mobil]
   f) Gaji pokok
   g) Uang makan

   Output:

   a) Tunjangan → dengan syarat:
   b) Jika jenis kelamin laki-laki dan status menikah maka mendapat 500 selain itu tunjangan 0
   c) Transport → dengan syarat :
      - jika kendaraan mobil maka mendapat 1000
      - jika kendarran motor maka mendapat 500
      - jika selain itu maka 0
   d) Gaji kotor
   e) Gaji kotor = gaji pokok + tunjangan + uang makan + transport
   f) Pajak
      - 5% dari gaji kotor → 0.05*gaji kotor
   g) Gaji bersih
   h) Gaji bersih = gaji kotor – pajak

2. Buatlah program konversi nilai huruf dari nilai yang diinputkan user!
    - A → 80-100
    - B → 65-79
    - C → 50-64
    - D → 35-49
    - E → 0-34
3. Buatlah program untuk menentukan tahun kabisat dari tahun yang diinputkan user.
4. Buatlah program kalkulator menggunakan untuk operasi-operasi sbb :
    - Tambah
    - Kurang
    - Bagi
    - Modulo
    - Kali

BAB VII. PERULANGAN

7.1. Tujuan Pembelajaran

Setelah mempelajari materi ini, mahasiswa diharapkan mampu :

1. Menjelaskan struktur perulangan
2. Menjelaskan alasan kenapa harus menggunakan perulangan
3. Menyebutkan jenis perulangan dan menjelaskan perbedaannya (*for, while* dan *do while*)
4. Menyelesaikan masalah dengan teknik perulangan

7.2. Dasar Teori

Perulangan merupakan proses mengulang-ulang perintah tanpa henti, selama kondisi yang dijadikan acuan terpenuhi atau bernilai benar. Pada perulangan terbagi menjadi dua yaitu perulangan naik dan menurun. Perulangan naik adalah perulangan yang dimulai dari nilai terkecil meuju nilai terbesar sebaliknya dengan menurun yaitu perulangan dari nilai terbesar menuju nilai terkecil. Jenis perulangan pada pemrograman C++ ada tiga yaitu for, while dan do while.

7.2.1. Struktur for

Struktur perulangan for biasa digunakan untuk mengulang suatu proses yang telah diketahui jumlah perulangannya, statement perulangan ini yang paling sering digunakan. Penulisan struktur perulangan for sebagai berikut:

```
for(inisialisasi;syarat;penambahan/pengurangan)
    pernyataan;
```

Keterangan:

Inisialisasi

Inisialisasi menyatakan keadaan awal dari variabel kontrol. Inisialisasi bisa berisi pendeklarasian variabel dan pemberian nilai awal atau bisa juga berisi pemberian nilai awal saja.

Syarat

Syarat menyatakan kondisi untuk keluar dari perulangan atau batas akhir dari perulangan.

Penambahan/pengurangan

Pengatur perubahan nilai variabel control, jika pada inisialisasi diawali dengan nilai kecil maka menggunakan penambahan berlaku sebaliknya.

Contoh 1 Program Perulangan For Naik

```
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5
6  cout<<"\nPerulangan Naik \n";
7  for(int i=1;i<=5;i++)
8   cout<<i<<endl;
9
10 return 0;}
```

Output contoh 1 Program Perulangan For Naik

```
Perulangan Naik
1
2
3
4
5
```

Keterangan:

`for(int i=1;i<=5;i++)`

Perintah diatas adalah perintah untuk memberi nilai awal sebuah perulangan yang dimulai dari `i=1` yang sebelumnya `i` dideklarasikan terlebih dahulu dengan tipe `int` kemudian perulangan kan berhenti jika nilai `i=5,` dan perulangan ini bersifat *increment* atau akan ditambah dengan angka 1 setiap kali proses sampai batas berhenti.

`cout<<i<<endl;`

Perintah ini adalah pernyataan yang akan diulang sampai 5 kali dan outputnya adalah menampilkan nilai i.

Contoh 2 Program Perulangan For Turun

```
#include <iostream>
using namespace std;

int main(){

cout<<"\nPerulangan Naik \n";
for(int i=5;i>=1;i--)
 cout<<i<<endl;

return 0;}
```

Output contoh 2 Program Perulangan For Turun

```
Perulangan Naik
5
4
3
2
1
```

Pada perulangan juga bisa digunakan untuk pernyataan dengan nilai awal yang dapat dituliskan lebih dari satu, dengan sifat menurun / naik, seperti pada contoh program 3.

Contoh 3 Program Perulangan For dengan inisialisasi 2 variabel

```
#include <iostream>
using namespace std;

int main()
{
    int a;
    for(int i=1,a=5;i<=10;i++)
       { cout<<"\nNilai ke "<<i<<" = "<<a;
         a+=5;
       }
return 0;
}
```

Output contoh 3 Program Perulangan For dengan inisialisasi 2 variabel

```
Nilai ke 1 = 5
Nilai ke 2 = 10
Nilai ke 3 = 15
Nilai ke 4 = 20
Nilai ke 5 = 25
Nilai ke 6 = 30
Nilai ke 7 = 35
Nilai ke 8 = 40
Nilai ke 9 = 45
Nilai ke 10 = 50
```

7.2.2. Struktur while

Statemen perulangan `while` berbeda dengan statemen perulangan `for`, kaarena perulangan ini digunakan bila jumlah perulangannya belum diketahui. Proses perulangan akan terus berlanjut selama kondisinya bernilai benar (true) dan akan berhenti bila kondisinya bernilai salah. Sintaks pernyataan while sebagai berikut:

```
while (syarat)
{
  instruksi;
}
```

Keterangan:

`while (syarat)`

Syarat disini adalah berisi batas dari perulangan.

`instruksi;`

Perintah atau pernyataan yang akan diulang selama syaratnya masih terpenuhi atau bernilai benar (true), perintah atau pernyataan ini berada di dalam tanda kurung kurawal `({})`.

Contoh 4 Program Perulangan While

```
2  #include <iostream>
3  using namespace std;
4
5  int main()
6  {
7   int i=1;
8   cout<<"Nilai i :";
9   while (i<=5)
10  {
11      cout<<i<<endl;
12      i++;
13  }
14 return 0;
15 }
```

Output Contoh 4 Program Perulangan While

```
Nilai i :1
2
3
4
5
```

Keterangan:

`int i=1;`

perintah diatas adalah pendeklarasian dan pemberian nilai awal untuk variabel `i`,variabel ini yang akan dijadikan sebagai awal dari perulangan.

`cout<<"Nilai i : ";`

perintah ini untuk menampilkan judul nilai i

`while (i<=5)`

perintah diatas adalah batas berhenti dari perulangan atau syarat dai perulangan, perulangan ini akan terus berjalan selama variabel i bernilai lebih dari 1 sampai dengan 5.

`cout<<i<<endl;`

perintah diatas adalah pernyataan yang akan diulang selama 5 kali, yaitu menampilkan kalimat nilai i sebanyak 5 kali.

`i++;`

perintah ini menunjukkan sifat naik atau increment artinya dalam setiap perulangan akan ditambah 1 sampai batas maksimal dari syarat yaitu 5.

Contoh 5 Program Perulangan While

```
2  #include <iostream>
3  using namespace std;
4
5  int main(){
6
7   int n,i=1;
8   float nilai,rata,jml=0;
9
10  cout<<"Input banyak nilai = ";cin>>n;
11  while(i<=n)
12    {  cout<<"Input Nilai "<<i<<" = " ;cin>>nilai;
13       jml=jml+nilai;
14       rata=jml/n;
15       i++;
16    }
17  cout<<"\nTotal nilai = "<<jml;
18  cout<<"\nRata nilai = "<<rata;
19
20 return 0;}
```

Output Contoh 5 Program Perulangan While

```
Input banyak nilai = 5
Input Nilai 1 = 1
Input Nilai 2 = 4
Input Nilai 3 = 2
Input Nilai 4 = 6
Input Nilai 5 = 7

Total nilai = 20
Rata nilai = 4
```

Keterangan:

`int n,i=1;`

`float nilai,rata,jml=0;`

perintah diatas adalah pendeklarasian variabel dan pemberian nilai awal untuk variabel i=1 dan untuk variabel jml=0.

`cout<<"Input banyak nilai = ";cin>>n;`

perintah diatas artinya user disuruh menginputkan angka ke dalam variabel n, yang nantiya angka tersebut akan dijadikan sebagai batas dari perulangan.

`while(i<=n)`

artinya perulangan akan berhenti jika nilai dari variabel i sudah lebih besar 1 dan sama dengan nilai yang ada di variabel n.

pada proses perulangan yang pertama atau i=1 sampai dengan perulangan yang terakhir atau i=n, maka statemen yang akan dikerjakan adalah menginputkan nilai ke variabel nilai.dengan perintah : `cout<<"Input Nilai "<<i<<" = " ;cin>>nilai;`

dan akan menjumlahkan nilai dari masing-masing proses perulangan tersebut, dengan perintah : `jml=jml+nilai;`

setelah menjumlahkan nilai pada setiap proses perulangan maka akan dicari rata-rata dari nilai tersebut dengan cara nilai yang sudah dijumlahkan dan disimpan dalam variabel jml kemudian di bagi dengan variabel n sebagai batas berhenti dari perulangan tersebut, perintahnya yaitu: `rata=jml/n;`

`i++;`

perintah ini untuk proses incremen dari perulangan pertama sampai perulangan terakhir yaitu sampai `i=n`.

`cout<<"\nTotal nilai = "<<jml;`

```
cout<<"\nRata nilai = "<<rata;
```

kedua perintah diatas untuk menampilkan jumlah dan rata-rata yang tersimpan pada variabel jml dan rata.

7.2.3. Struktur do while

Pada dasarnya struktur perulangan `do..while` sama saja dengan struktur `while`, hanya saja pada proses perulangan dengan `while`, seleksi `while` berada di atas sementara pada perulangan `do....while`, seleksi `while` berada di bawah batas perulangan. Jadi dengan menggunakan struktur `do…while` sekurang-kurangnya akan terjadi satu kali perulangan. Perbedaan perulangan while dengan do while dapat dilihat pada tabel 7.1.

Tabel 7.1 Perbedaan While dan Do While

While	Do while
Bisa jadi tidak akan pernah dikerjakan jika syarat tidak dipenuhi.	Minimal dikerjakan satu kali walaupun syarat tidak dipenuhi.
Ini dikarenakan sebelum instruksi dikerjakan, syarat dicek terlebih dahulu.	Ini dikarenakan instruksi dikerjakan dahulu, baru syarat dicek untuk melanjutkan perulangan.

Sintaks pernyataan do while:

```
do
 {
   instruksi;
 }
 while (syarat);
```

Keterangan:

`do` adalah awal dari statemen perulangan `do while`

`instruksi` adalah statemen yang akan dikerjakan atau yang akan diulang, baik syarat dari perulangan tersebut terpenuhi atau tidak, karena akan dijalankan instruksinya terlebih dahulu setelah mengerjakan instruksi baru dicek syaratnya, apakah syaratnya masih terpenuhi atau tidak.

`While (syarat)` adalah syarat atau batas dari perulangan, dimana perulangan tersebut akan dikerjakan jika syaratnya masih terpenuhi atau bernilai true.

Contoh 6 Program Perulangan Do While

```
#include <iostream>
using namespace std;

int main(){

int i=1;
cout<<"\nNilai i :";
do {
    cout<<i<<endl;
    i++;
   }
while (i<=5);
return 0;}
```

Output Contoh 6 Program Perulangan Do While

```
Nilai i :1
2
3
4
5
```

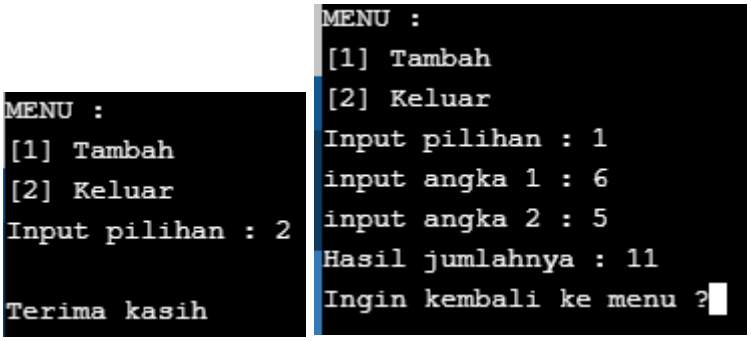
7.2.4. Penggunaan Kata Kunci goto

Kata kunci `goto` sebenarnya adalah perintah untuk mengulang yang cenderung dengan melompat atau lompatan. Contoh penggunaan goto dapat dilihat pada contoh program 7.

Contoh 7 Program Penggunaan goto

```
#include <iostream>
using namespace std;

int main(){
int pil,a,b,c,jawab;

menu:
cout<<"MENU :";
cout<<"\n[1] Tambah";
cout<<"\n[2] Keluar";

cout<<"\nInput pilihan : ";
cin>>pil;

if(pil==1)
 {
    cout<<"input angka 1 : ";cin>>a;
    cout<<"input angka 2 : ";cin>>b;
    c=a+b;
    cout<<"Hasil jumlahnya : "<<c;
    cout<<"\nIngin kembali ke menu ?";cin>>jawab;
    if(jawab==1)
      {
          goto menu;
      }
   else if(jawab==0)
      cout<<"Makasih";
 }
else
 cout<<"\nTerima kasih\n";
return 0;}
```

Output Contoh 7 Program Penggunaan goto



7.2.5. Penggunaan Kata Kunci goto

Tidak hanya percabangan saja yang memiliki pernyataan percabangan bersarang tetapi perulangan juga ada perulangan bersarang. Perulangan bersarang adalah perulangan yang perulangan dalam perulangan atau pada perulangan pertama yang pada pernyataan instruksi berisi perulangan kedua, dan pada pernyataan instruksi perulangan kedua berisi perulangan ketiga dan seterusnya. Proses kerja dari perulangan bersarang adalah pada saat penyelesaikan perulangan pertama maka akan dikerjakan terlebih dahulu perulangan kedua kemudian pada saat penyelesaian perulangan kedua maka akan dikerjakan terlebih dahulu perulangan ke tiga, sehingga perulangan yang paling dalam akan selesai lebih dulu prosesnya kemudian menyelesaikan perulangan kedua dan terakhir perulangan pertama. Perulangan bersarang bisa dengan for, while, do while atau kombinasi dari ketiganya. Adapun struktur perulangan bersarang secara umum sebagai berikut:

```
for(inisialisasi;syarat;penambahan/pengurangan)
   { for(inisialisasi;syarat;penambahan/pengurangan)
     { for(inisialisasi;syarat;penambahan/pengurangan)
       {  Pernyataan yang akan diulang
       }
     }
   }
```

Contoh 8 Program Perulangan Bersarang

```
#include <iostream>
using namespace std;

int main(){

for(int i=1;i<5;i++)
{
    for(int n=5;n>i;n--)
    {
      cout<<"*";
    }
    cout<<endl;
}
 return 0;}
```

Output Contoh 8 Program Perulangan Bersarang

```
****
***
**
*
```

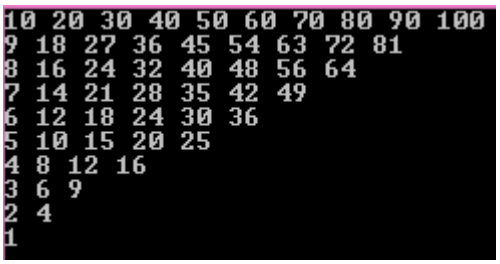
7.3. Latihan Soal

Kerjakan soal di bawah ini!

1. Buatlah program untuk menghitung IPK mahasiswa.
2. Buat program untuk mencari total penjualan dari n jenis barang, dan pemberian diskon penjualan, dengan ketentuan, pembelian di atas Rp 500.000 mendapat diskon 10%, pembelian diatas Rp 200.000 sampai dengan Rp 500.000 mendapat diskon 5%, pembelian Rp 200.000 ke bawah tidak mendapat diskon.
3. Dari kedua program di atas buatlah ke dalam menu pilihan dengan perintah goto sebagai loncatannya atau perintah untuk kembali ke menu pilihan.
4. Buatlah program dengan tampilan sebagai berikut :

```
1
1 2
1 2 3
1 2 3 4
4 4 4 4 4
3 3 3 3
2 2 2
1 1
0
```

5. Buatlah program dengan tampilan sebagai berikut :



BAB VIII. POINTER (PENUNJUK)

8.1. Tujuan Pembelajaran

Setelah mempelajari materi ini, mahasiswa diharapkan mampu :

1. Menjelaskan pengertian dan manfaat *pointer*
2. Menjelaskan kapan harus menggunakan *pointer*
3. Menyelesaikan masalah dengan *pointer*
4. Menyelesaikan masalah menggunakan *pointer* dalam *array* dan *struct*

8.2. Dasar Teori

Pointer (variabel penunjuk) adalah suatu variabel yang berisi alamat memori dari suatu variabel lain. Alamat ini merupakan lokasi dari obyek lain (biasanya variabel lain) di dalam memori. Contoh: jika variabel A berisi alamat dari variabel B, maka variabel A dikatakan menunjuk ke variabel B. Operator pointer ada 2 yaitu:

1. Operator `&`

   Operator `&` menghasilkan alamat dari operand atau alamat memori yang ditempati oleh variabel.

2. Operator `*`

   Operator `*` menghasilkan nilai yang berada pada sebuah alamat.

Perbedaan antara variabel biasa dengan variabel pointer yaitu:

Tabel 9.1 Perbedaan Pointer dengan Variabel Biasa

	Pointer	**Variabel biasa**
Deklarasi variabel	int *a;	int b;
Alamat memori	Tidak otomatis	Otomatis
Mengetahui alamat memori	A	&b
Mengetahui datanya	*a	b

Seperti halnya variabel yang lain, variabel pointer juga harus dideklarasikan terlebih dahulu sebelum digunakan. Bentuk Umum pendeklarasian variabel pointer:

```
Tipe_data *nama_pointer;
Contoh : int *nilai;
         char *huruf;
```

Ada dua cara yang dapat dilakukan untuk alokasi space di memori bagi pointer, yaitu:

1. **Menempati *space variable* lain yang sudah punya *space*.**

   Variabel lain tersebut dapat berupa variabel biasa (bukan pointer) atau pointer yang tentunya sudah punya alokasi space di memori.

   Sintaksnya : **`var_pointer = &var_biasa;`**

   Contoh 1 Program Penggunaan Pointer

```
#include <iostream>
using namespace std;

int main() {
 int x,*y;
 char *judul="PENGGUNAAN POINTER";

 y=&x;
 x=5;

 cout<<judul<<endl;
 cout<<"Nilai x = "<<x;
 cout<<"\nAlamat memori x = "<<&x;
 cout<<"\n\nNilai y = "<<*y;
 cout<<"\nAlamat memori y = "<<*y;

return 0;}
```

   Output Contoh 1 Program Penggunaan Pointer

```
PENGGUNAAN POINTER
Nilai x = 5
Alamat memori x = 0x7ffcc6cb509c

Nilai y = 5
Alamat memori y = 5
```

   Keterangan:

   Karena `*y` menempati alokasi memori yang sama dengan `x`, akibatnya data yang tersimpan di dalamnya pun akan sama. Jika ada instruksi yang menginisialisasi `x`, otomatis juga akan menginisialisasi `*y`.

Contoh 2 Program Penggunaan Pointer

```
#include <iostream>
using namespace std;

int main() {

 int x,*y;
 y=&x;
 x=5;
 *y=10;

 cout<<"Nilai x = "<<x;
 cout<<"\nAlamat memori x = "<<&x;
 cout<<"\n\nNilai y = "<<*y;
 cout<<"\nAlamat memori y = "<<y;

return 0;}
```

Output Contoh 2 Program Penggunaan Pointer

```
Nilai x = 10
Alamat memori x = 0x7ffea7bf8cdc

Nilai y = 10
Alamat memori y = 0x7ffea7bf8cdc
```

Contoh 3 Program Penggunaan Pointer

```
#include <iostream>
using namespace std;

int main() {

 int n1,n2;
 int *p;
 p=&n1;
 *p=10;
 p=&n2;
 *p=20;
 cout<<"Nilai n1 = "<<n1;
 cout<<"\nNilai n2 = "<<n2;
 cout<<"\nNilai p = "<<*p;

return 0;}
```

Output Contoh 3 Program Penggunaan Pointer

```
Nilai n1 = 10
Nilai n2 = 20
Nilai p = 20
```

2. **Dialokasikan tersendiri di memori (Memori dinamis).**

   Pointer tidak menempati space variabel lain, tetapi dialokasikan space tersendiri di memori dengan instruksi `new`**.**

   Sintaksnya :

   `var_pointer = new tipe_data-pointernya;`

   Contoh : **`int *a;`**

   `float *x;`

   `a=new int;`

   `x=new float;`

Contoh 4 Program Penggunaan Pointer

```
1  #include <iostream>
2  using namespace std;
3
4  int main() {
5
6  int x,*y;
7   y=new int;
8   x=5;
9   *y=10;
10  cout<<"Nilai x = "<<x;
11  cout<<"\nAlamat memori x = "<<&x;
12  cout<<"\n\nNilai y = "<<*y;
13  cout<<"\nAlamat memori y = "<<y;
14
15 return 0;}
```

Output Contoh 4 Program Penggunaan Pointer

```
Nilai x = 5
Alamat memori x = 0x7ffec84320dc

Nilai y = 10
Alamat memori y = 0x250fc20
```

`x` dan `y` mempunyai alokasi memori yang berbeda, sehingga data yang tersimpan di dalamnya pun akan berbeda pula. Dengan alokasi memori dinamis ini kita dapat menghemat alokasi memori dengan cara membebaskan memori dari variabel dinamis, jika memang variabel tersebut tidak digunakan lagi. Cara membebaskan memori dari variabel dinamis yaitu:

Sintaks : `delete var_pointer;`

Contoh 5 Program Penggunaan Pointer

```
1  #include <iostream>
2  using namespace std;
3
4  int main() {
5
6  int x,*y; y=new int;
7   x=5; *y=10;
8   cout<<"Nilai x = "<<x;
9   cout<<"\nAlamat memori x = "<<&x;
10  cout<<"\n\nNilai y = "<<*y;
11  cout<<"\nAlamat memori y = "<<y;
12  delete y;
13  cout<<"\n\nSetelah di delete";
14  cout<<"\nNilai y = "<<*y;
15  cout<<"\nAlamat memori y = "<<y;
16
17 return 0;}
```

Output Contoh 4 Program Penggunaan Pointer

```
Nilai x = 5
Alamat memori x = 0x7fff500a56ac

Nilai y = 10
Alamat memori y = 0xf1ac20

Setelah di delete
Nilai y = 0
Alamat memori y = 0xf1ac20
```

Keterangan:

Setelah dikenai instruksi `delete y;` maka nilai yang tersimpan dalam `*y` akan hilang. Itulah sebabnya `*y` dikatakan sebagai variable dinamis, sedangkan `x` merupakan variable statis, sehingga tidak bisa dibebaskan dari memori.

8.2. Latihan Soal

1. Buatlah program array dan pointer yang diterapkan di array tanggal lahir (tanggal, bulan, tahun) dengan hasil / tampilan sebagai berikut:

```
Diakses dengan pointer
Tanggal = 13
Bulan = 9
Tahun = 1982
Diakses dengan array biasa
Tanggal = 13
Bulan   = 9
Tahun   = 1982
```

2. Buatlah program dengan menggunakan pointer untuk mengetikkan huruf dalam nama anda maksimal 5 huruf, sehingga tampilannya seperti berikut:

```
Input elemen [0]=n

Input elemen [1]=i

Input elemen [2]=n

Input elemen [3]=i

Input elemen [4]=k


Elemen yang telah dimasukkan akan terbaca sebagai berikut:
[n i n i k ]
```

BAB IX. MODULAR (FUNGSI)

9.1. Tujuan Pembelajaran

Setelah mempelajari materi ini, mahasiswa diharapkan mampu :

1. Menjelaskan pengertian fungsi
2. Menjelaskan manfaat penggunaan fungsi
3. Menjelaskan kapan harus menggunakan modular/fungsi
4. Menjelaskan jenis-jenis fungsi
5. Menjelaskan perbedaan dari jenis-jenis fungsi (void dan non void)
6. Menulis deklarasi, definisi dan pemanggilan fungsi
7. Menyelesaikan masalah dengan teknik fungsi

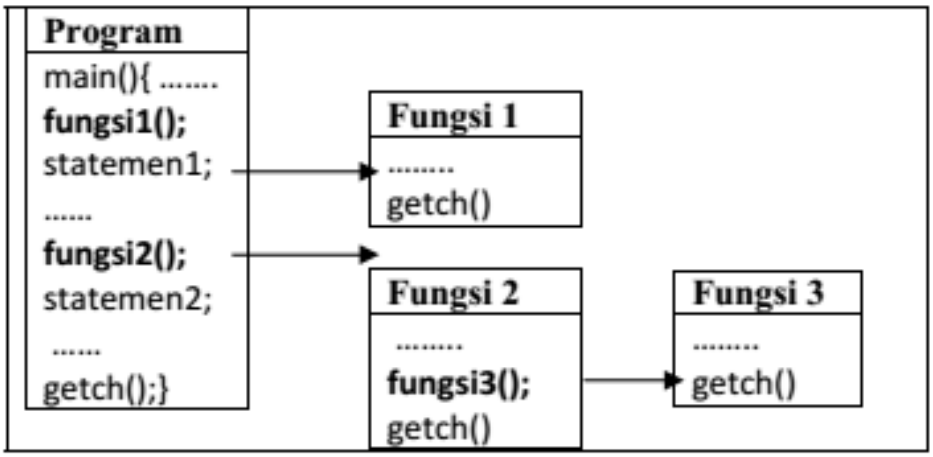
9.2. Dasar Teori

Modular adalah suatu teknik pemrograman dimana program yang biasanya cukup besar dibagi-bagi menjadi beberapa bagian program yang lebih kecil sehingga akan mudah dipahami dan dapat digunakan kembali, baik untuk program itu sendiri maupun program lain yang memiliki proses yang sama.

Modul pada bahasa C++ dikenal dengan nama fungsi (*function*), sedangkan bahas C++ terdiri dari fungsi-fungsi, baik yang langsung dideklarasikan dalam program ataupun dipisah di dalam *header file*. Fungsi yang selalu ada pada program C++ adalah fungsi **main.**

Fungsi/function adalah suatu kumpulan instruksi/perintah/ program yang dikelompokkan menjadi satu, letaknya terpisah dari program yang menggunakan fungsi tersebut, memiliki nama tertentu yang unik, dan digunakan untuk mengerjakan suatu tujuan tertentu. Dalam bahasa pemrograman lain fungsi dapat disebut sebagai *subrutin* (basic, VB) atau *procedure* (pascal, Delphi).

Fungsi digunakan jika kita dihadapkan oleh suatu permasalahan yang besar, untuk mempermudah pemecahannya, maka akan lebih baik jika dibagi-bagi menjadi sub permasalahan yang lebih kecil. Berikut ini gambar yang mengilustrasikan keberadaan fungsi dalam sebuah program.



Gambar 9.1 Ilustrasi Fungsi

Pemrograman modular mempunyai kelebihan yaitu:

1. Program lebih pendek.
2. Mudah dibaca dan dimengerti.
3. Mudah didokumentasi.
4. Mengurangi kesalahan dan mudah mencari kesalahan.
5. Kesalahan yang terjadi bersifat "lokal".

Pada pemrograman C++ modular atau fungsi terbagi menjadi 2 yaitu fungsi bawaan langsung dari C++ atau dengan nama *standard library function* dan fungsi yang dibuat oleh programmer atau dengan nama *programmer defined fungtion*.

9.2.1. Standard Library Function

Standard Library Function yaitu fungsi-fungsi yang telah disediakan oleh C dalam file-file header atau *library*nya. Untuk fungsi ini kita harus mendeklarasikan terlebih dahulu *library* yang akan digunakan, yaitu dengan menggunakan preprosesor direktif. Misalnya untuk penggunaan fungsi clrscr(), getch() maka kita harus mendeklarasikan terlebih dahulu librarynya yaitu dengan menuliskan #include <conio.h> pada *preprosesor directive* atau file headernya.

9.2.2 Programmer-Defined Function

Programmer-Defined Function yaitu fungsi yang dibuat oleh programmer sendiri. Fungsi ini memiliki nama tertentu yang unik dalam program, letaknya terpisah dari program utama, dan bisa dijadikan satu ke dalam suatu *library* buatan programmer itu

sendiri yang kemudian juga di-*include*-kan jika ingin menggunakannya. Dalam bahasa C++ ada dua jenis fungsi yang dibuat oleh programmer (Programmer-Defined Function) yaitu:

9.2.2.1. Void

Fungsi tanpa return value atau fungsi yang tidak mengembalikan nilai. Fungsi yang `void` sering disebut juga prosedur. Disebut `void` karena fungsi tersebut tidak mengembalikan suatu nilai keluaran yang didapat dari hasil proses fungsi tersebut.

Ciri-ciri fungsi tipe `void` antara lain :

1. Tidak adanya keyword return.
2. Tidak adanya tipe data di dalam deklarasi fungsi.
3. Menggunakan keyword `void`.
4. Tidak dapat langsung ditampilkan hasilnya.
5. Tidak memiliki nilai kembalian fungsi.
6. Keyword `void` juga digunakan jika suatu function tidak mengandung suatu parameter apapun.
7. Parameter formal fungsi ada dua macam :
8. Parameter value, adalah variable inputnya.
9. Parameter reference/address, adalah variable outputnya.
10. Instruksi fungsi adalah instruksi program dengan menghilangkan instruksi untuk menginputkan data pada variable input, dan instruksi untuk menampilkan hasil dari variable output.

9.2.2.2. Non-void (int, float, char dll …..)

Fungsi dengan return value atau fungsi yang mengembalikan nilai. Fungsi `non-void` disebut juga function, disebut `non-void` karena mengembalikan nilai kembalian yang berasal dari keluaran hasil proses function tersebut.

Ciri-ciri fungsi tipe `non-void` antara lain :

1. Ada keyword return.
2. Ada tipe data yang mengawali deklarasi fungsi dan tipe fungsi sama dengan tipe outputnya.
3. Tidak ada keyword `void`.
4. Memiliki nilai kembalian.

5. Dapat dianalogikan sebagai suatu variabel yang memiliki tipe data tertentu sehingga dapat langsung ditampilkan hasilnya.
6. Parameter formal (parameter value) fungsi adalah variabel inputnya.
7. Instruksi fungsi adalah instruksi program dengan menghilangkan instruksi untuk menginputkan data pada variable input, dan instruksi untuk menampilkan hasil dari variable output.

Dalam membuat fungsi, perlu diperhatikan:

1. Data yang diperlukan sebagai inputan.
2. Informasi apa yang harus diberikan oleh fungsi yang dibuat ke pemanggilnya.
3. Algoritma apa yang harus digunakan untuk mengolah data menjadi informasi.

Dimana kita harus meletakkan fungsi yang kita buat? Ada dua cara untuk meletakkan fungsi yaitu:

1. Dapat diletakkan sebelum `main()` atau langsung didefinisikan.
2. Dapat juga diletakkan setelah `main()`,namun jika fungsi diletakkan setelah `main`, pada posisi sebelum `main` harus dituliskan prototype fungsinya atau dideklarasikan terlebih dahulu.

Kapan kita menggunakan deklarasi dan definisi fungsi? Karena prinsip kerja program C sekuensial, maka

1. Jika bagian dari program yang menggunakan fungsi diletakkan sebelum definisi dari fungsi, maka deklarasi dari fungsi diperlukan.
2. Akan tetapi jika bagian dari program yang menggunakan fungsi terletak setelah definisi dari fungsi, maka deklarasi dari fungsi dapat tidak dituliskan.

Bentuk umum deklarasi fungsi :

```
tipe_fungsi
nama_fungsi(parameter_formal);
```

Bentuk umum pendefinisian fungsi :

```
tipe_fungsi
nama_fungsi(parameter_formal)
        {
         instruksi
         dan deklarasi var lokal }
```

Contoh 1 Program dengan Fungsi Void dengan deklarasi dan definisi

```
1  #include <iostream>
2  using namespace std;
3
4  void cetak(); // deklarasi fungsi cetak
5
6  int main()
7  {
8   cetak();    // pemanggilan fungsi cetak
9   return 0;
10 }
11
12 void cetak() // definisi fungsi cetak
13 {
14   cout<<"Membuat Fungsi Void"; // isi fungsi
15 }
```

Output Contoh 1 Program dengan Fungsi Void

```
Membuat Fungsi Void
```

Contoh 2 Program dengan Fungsi Void dengan definisi

```
1  #include <iostream>
2  using namespace std;
3
4  void luas(int &ls, int p, int l) //definisi fungsi luas
5  { ls = p*l; }                    //isi fungsi
6
7  int main(){
8
9   int pj,lb, hsl;
10  cout<<"Panjang = ";cin>>pj;
11  cout<<"Lebar = ";cin>>lb;
12  luas(hsl,pj,lb);          //pemanggilan fungsi
13  cout<<"\nLuasnya = "<<hsl;
14  return 0;}
```

Output Contoh 2 Program dengan Fungsi Void dengan definisi

```
Panjang = 2
Lebar = 4

Luasnya = 8
```

Keterangan:

`void luas(int &ls, int p, int l)`

`{ ls = p*l; }`

Fungsi dengan nama `luas` yang bertipe void dengan parameter `ls, p, l` dengan tipe data `integer` parameter dengan nama `&ls` digunakan untuk menampung hasil luas atau sebagai output, parameter `p, l` digunakan untuk input panjang dan lebar atau sebagai input dan proses. `ls = p*l` merupakan rumus luas yang diperoleh dari panjang dikali lebar, dalam penulisan rumus parameter output tidak perlu ditambahkan tanda `&`.

`int pj,lb, hsl;` merupakan pendeklarasian variabel aktual, `pj` dan `lb` adalah variabel input yang digunakan untuk menginputkan nilai yang akan dihitung, sedangkan `hsl` adalah variabel output yang digunakan untuk menampung hasil luas.

`cout<<"Panjang = ";cin>>pj;` perintah untuk menginputkan nilai ke dalam variabel `pj`.

`cout<<"Lebar = ";cin>>lb;` perintah untuk menginputkan nilai ke dalam variabel `lb`.

`luas(hsl,pj,lb);` merupakan perintah untuk pemanggilan fungsi `luas` dengan parameter aktual `hsl, pj dan lb`.

`cout<<"\nLuasnya = "<<hsl;` perintah yang digunakan untuk memanggil atau melihat nilai dari variabel `hsl`.

Contoh 3 Program dengan Fungsi Non Void dengan deklarasi dan definisi

```
1  #include <iostream>
2  using namespace std;
3
4  int luas(int p, int l); //deklarasi fungsi non void
5
6  int main(){
7
8  int pj,lb;
9
10 cout<<"Panjang = ";cin>>pj;
11 cout<<"Lebar = ";cin>>lb;
12 cout<<"\nLuasnya = "<<luas(pj,lb); //pemanggilan fungsi
13
14 return 0;}
15
16 int luas(int p, int l) //definisi fungsi
17 {return (p*l); }  // isi fungsi dg return value
```

Output Contoh 3 Program dengan Fungsi Non Void dengan deklarasi dan definisi

```
Panjang = 7
Lebar = 6

Luasnya = 42
```

Contoh 4 Program dengan Fungsi Non Void dengan definisi

```
2   #include <iostream>
3   using namespace std;
4
5   int luas(int p, int l) //definisi fungsi
6   {return (p*l); }  // isi fungsi dg return value
7
8   int main(){
9
10  int pj,lb;
11
12  cout<<"Panjang = ";cin>>pj;
13  cout<<"Lebar = ";cin>>lb;
14  cout<<"\nLuasnya = "<<luas(pj,lb); //pemanggilan fungsi
```

Output Contoh 4 Program dengan Fungsi Non Void dengan definisi

```
Panjang = 6
Lebar = 7

Luasnya = 42
```

Keterangan:

`int luas(int p, int l)`

`{return (p*l); }`

Fungsi dengan nama `luas` yang bertipe `integer` dengan parameter `p, l` digunakan untuk input panjang dan lebar atau sebagai input dan proses. `return( p*l)` pemanggilan untuk luas yang diperoleh dari panjang dikali lebar.

`int pj,lb;` merupakan pendeklarasian variabel aktual, `pj` dan `lb` adalah variabel input yang digunakan untuk menginputkan nilai yang akan dihitung.

`cout<<"Panjang = ";cin>>pj;`

perintah untuk menginputkan nilai ke dalam variabel `pj`.

`cout<<"Lebar = ";cin>>lb;` perintah untuk menginputkan nilai ke dalam variabel `lb`.

`cout<<"\nLuasnya = "<<luas(pj,lb);` merupakan perintah untuk pemanggilan fungsi `luas` dengan parameter aktual `pj` dan `lb`.

Catatan :

1. Parameter formal yaitu variabel yang dideklarasikan pada fungsi baik yang digunakan untuk variabel input, output maupun proses.
2. Parameter aktual yaitu variabel yang dideklarasikan pada program utama (diantara `main` dan `return 0`) yang digunakan untuk memanggil fungsi.

Untuk pemanggilan fungsi urutan parameter aktualnya harus urut dan tidak boleh terbalik sesuai pada saat pendeklarasian pada parameter formal.

9.2.3 Fungsi dengan Parameter Pointer

Fungsi dengan parameter pointer digunakan untuk parameter keluaran yaitu parameter yang berfungsi untuk menampung nilai yang dihasilkan dari proses di dalam fungsi. Parameter keluaran ini digunakan dalam fungsi tipe void (tanpa return value),dengan kata lain parameter tersebut digunakan sebagai nilai keluaran dari sebuah fungsi. Dengan demikian parameter keluaran ini harus dilewatkan berdasarkar alamat atau referensinya, yaitu menggunakan pointer atau reference.

Contoh 5 Program dengan Fungsi Parameter Pointer

```
2   #include <iostream>
3   using namespace std;
4
5   void luas(int *ls, int p, int l)
6   { *ls = p*l; }
7
8   int main(){
9       int pj,lb,ls;
10      cout<<"Panjang = ";cin>>pj;
11      cout<<"Lebar   = ";cin>>lb;
12      luas(&ls,pj,lb);
13      cout<<"\nLuasnya = "<<ls;
14  return 0;}
```

Output Contoh 5 Program dengan Fungsi Parameter Pointer

```
Panjang = 2
Lebar   = 9

Luasnya = 18
```

Contoh 6 Program dengan Fungsi Parameter Pointer

```
1   #include <iostream>
2   using namespace std;
3
4   void luas(int *ls,int *kel, int p, int l);
5
6   int main(){
7       int pj,lb,ls,kl;
8       cout<<"Panjang = ";cin>>pj;
9       cout<<"Lebar   = ";cin>>lb;
10      luas(&ls,&kl,pj,lb);
11      cout<<"\nLuasnya     = "<<ls;
12      cout<<"\nKelilingnya = "<<kl;
13  return 0;}
14
15  void luas(int *ls,int *kel, int p, int l)
16  { *ls = p*l;  *kel = 2*(p+l);}
17
```

Output Contoh 6 Program dengan Fungsi Parameter Pointer

```
Panjang = 3
Lebar   = 5

Luasnya     = 15
Kelilingnya = 16
```

9.3. Latihan Soal

Kerjakan soal-soal di bawah ini!

1. Buatlah fungsi kalkulator untuk menghitung dua buah bilangan (+,-,*,/,%)
2. Buatlah program untuk menentukan IPK mahasiswa
3. Buatlah program untuk menghitung gaji seorang karyawan dengan ketentuan sbb:
   a. Gapok
      - Jika pendidikan d3 maka gapok 500.000

- Jika pendidikan s1 maka gapok 750.000
- Jika pendidikan s2 maka gapok 1.000.000

b. Tunjangan istri 500.000

Tunjangan istri didapat jika karyawan laki-laki dan sudah menikah

c. Tunjangan anak 200.000/anak

Tunjangan anak didapat jika karyawan laki-laki, sudah menikah dan memiliki anak max 2 orang anak

d. Hitung gaji bersihnya setelah dipotong pajak 5%

4. Buatlah program untuk mencari nilai maksimal dari n bilangan menggunakan fungsi dengan parameter pointer.
5. Buatlah program untuk menentukan keterangan dari nilai mata kuliah menggunakan fungsi dengan parameter pointer. Dengan ketentuan jika :

   Nilai >= 90 - 100 maka A

   Nilai >= 80 - < 90 maka B

   Nilai >= 70 - < 80 maka C

   Nilai >= 60 - < 70 maka D

   Nilai >= 50 - < 60 maka E

   Nilai < 50 maka tidak lulus

BAB X. ARRAY (LARIK)

10.1. Tujuan Pembelajaran

Setelah mempelajari materi ini, mahasiswa diharapkan mampu :

1. Memahami konsep array dan penyimpanannya dalam memori
2. Mempelajari pengunaan variabel array berdimensi satu
3. Memahami penggunaan variabel array berdimensi dua
4. Menerapkan penggunaan array berdimensi satu dan dua pada program sederhana

10.2. Dasar teori

Array adalah suatu tipe data terstuktur yang berupa sejumlah data sejenis (bertipe data sama) yang jumlahnya tetap dan diberi suatu nama tertentu, elemen-elemen array tersusun secara berderet dan dapat diakses secara random di dalam memori. Array memiliki alamat yang besebelahan/berdampingan tergantung lebar tipe datanya dan dapat berupa array 1 dimensi, 2 dimensi, bahkan n-dimensi. Elemen-elemen array bertipe data sama dan bisa berisi nilai yang sama atau berbeda-beda.

Elemen-elemen array dapat diakses oleh program menggunakan suatu indeks tertentu secara random ataupun berurutan. Pengisian dan pengambilan nilai pada indeks tertentu dapat dilakukan dengan mengeset nilai atau menampilkan nilai pada indeks yang dimaksud. Dalam C++, tidak terdapat error handling terhadap batasan nilai indeks, apakah indeks tersebut berada di dalam indeks array yang sudah didefinisikan atau belum. Hal ini merupakan tanggung jawab programmer, sehingga jika programmer mengakses indeks yang salah, maka nilai yang dihasilkan akan berbeda atau rusak karena mengakses alamat memori yang tidak sesuai. Array terbagi dalam dua jenis yaitu array satu dimensi dan dua dimensi atau multi dimensi.

10.2.1. Array Satu Dimensi

Pada array satu dimensi hanya terdapat satu baris dan bayak kolom. Deklarasi Array satu dimensi secara umum : `tipe_data nama_var_array [ukuran];`
Keterangan:

tipe_data : menyatakan jenis tipe data elemen larik (int, char, float, dll)
nama_var_array : menyatakan nama variabel yang dipakai.

ukuran : menunjukkan jumlah maksimal elemen larik.

Contoh : `int nilai[6];`

Untuk menginisialisasi array sama dengan memberikan nilai awal array pada saat didefinisikan, dengan cara sebagai berikut:

```
int nilai[6] = {8,7,5,6,4,-3};
```

bisa disederhanakan sehingga menjadi :

```
int nilai = {8,7,5,6,4,3};
```

Keterangan:

Contoh diatas berarti anda memesan tempat di memori komputer sebanyak 6 tempat dengan indeks dari 0-5, dimana indeks ke-0 bernilai 8, ke-1 bernilai 7, dst, dan semua elemennya bertipe data integer.

Catatan

Untuk memberikan nilai 0 terhadap seluruh elemen array pada saat didefinisikan, Anda dapat memberikan nilai awal 0 pada elemen pertama.

Sebagai contoh: `int temp[100] = {0};`

Akan memberikan hasil pemberian nilai nol dari subscript bernilai 0 hingga 99. Struktur diatas merupakan pendeklarasian dan inisialisasi array satu dimensi, karena untuk array dua dimensi digunakan untuk matrik.

Tiga hal penting dalam array yang harus dipahami yaitu:

1. Index

   Index bisa disebut dengan urutan, index pada array secara default dimulaia dari 0 (nol).

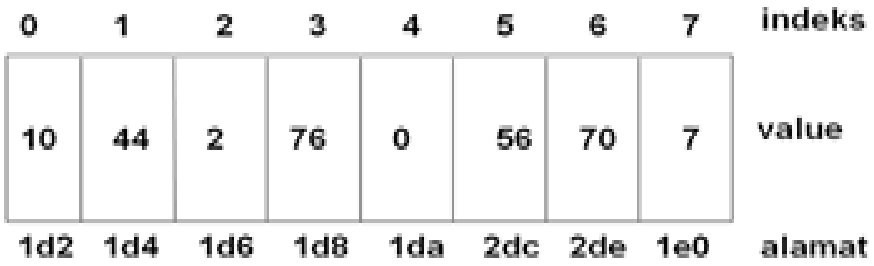
2. Value

   Value adalah nilai yang mengisi di setiap elemen array, dan elemen-elemen ini dapat diakses oleh program menggunakan suatu indeks tertentu secara random ataupun berurutan.

3. Reference

   Reference adalah alamat memori dari masing-masing elemen array.

Ilustrasi array satu dimensi dapat dilihat pada gambar 10.1



Gambar 10.1 Contoh Ilustrasi Array Satu Dimensi

Kelebihan dari array :

1. Array sangat cocok untuk pengaksesan acak. Sembarang elemen di array dapat diacu secara langsung tanpa melalui elemen-elemen lain.
2. Jika berada di suatu lokasi elemen, maka sangat mudah menelusuri ke elemen-elemen tetangga, baik elemen pendahulu atau elemen penerus.
3. Jika elemen-elemen array adalah nilai-nilai independen dan seluruhnya harus terjaga, maka penggunaan penyimpanannya sangat efisien.

Kekurangan dari array:

1. Array harus bertipe homogen. Kita tidak dapat mempunyai array dimana satu elemen adalah karakter, elemen lain bilangan, dan elemen lain adalah tipe-tipe lain.
2. Tidak efisien dalam penggunaan memori.
3. Menyiakan banyak waktu komputasi.
4. Pada suatu aplikasi, representasi statis tidak dimungkinkan.

Cara menginputkan nilai ke array:

1. Pada saat deklarasi/inisialisai

```
Contoh:
int nilai[3] = {1,3,2};
char nama[3][10] = {"Yuni","Yani","Yono"};
```

2. Input secara statis

```
Contoh:
Nilai[0]=1;
Nilai[1]=3;
Nilai[2]=2;

Nama[0]="Yuni";
Nama[1]="Yani";
```

```
Nama[2]="Yono";
```

3. Input secara dinamis

   Contoh:

```
for(i=0;i<3;i++)
  {cout<<"Nilai : "; cin>>nilai[i];}

for(i=0;i<3;i++)
 {cout<<"Nama : "; cin>>nama[i];}
```

Contoh 1 Program Array 1 Dimensi

```
#include <iostream>
using namespace std;

int main(){

 int y [] ={1, 2, 7, 4, 5};
 int n, r=0;

 for ( n=0 ; n<5 ; n++ )
  {
     r += y[n];
  }
 cout<<"Hasilnya : "<<r;
return 0;}
```

Output Contoh 1 Program Array 1 dimensi

```
Hasilnya : 19
```

Keterangan:

`int y [] = {1, 2, 7, 4, 5};`

y adalah variabel array yang memiliki 5 elemen dengan tipe integer

`int n, r=0;`

pendeklarasian n dan r sebagai variabel biasa.

`for ( n=0 ; n<5 ; n++ )`

perulangan yang dimulai dari 0 sampai n=4

`{ r += y[n]; }`

Instruksi yang akan diulang setiap proses perulangan dikerjakan/setiap iterasi, yaitu penjumlahan nilai dari elemen array.

`cout<<"Hasilnya : "<<r;`

pemanggilan variabel r, yang berisi jumlah akhir dari proses perulangan atau iterasi, nilai yang ditampilkan yaitu 19.

Contoh 2 Program Array 1 dimensi

```
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5
6   char hari[3][10]={"senin","selasa","rabu"};
7   int n, r=0;
8
9   for ( n=0 ; n<5 ; n++ )
10    {
11       cout<<endl<<hari[n];
12    }
13 return 0;}
```

Output Contoh 2 Program Array 1 dimensi

```
senin
selasa
rabu
```

Keterangan:

`char hari[3][10]={"senin","selasa",”rabu"};`

array dengan nama hari dengan maksimal 3 elemen dan memiliki maksimal space 10 digit, jadi untuk masing-masing nama hari maksimal 10 digit.

Contoh 3 Program Input dan Output Array

```
1  #include <iostream>
2  using namespace std;
3
4  int main(){
5
6   int A[5];
7   cout<<"\nInput Array  \n";
8   for (int i=0;i<5;i++)
9    {
10     cout<<"A["<<i<<"] = ";
11     cin>>A[i];
12   }
13
14 for(int c=0;c<5;c++)
15  { cout<<"Nilai pada elemen ke "<<c+1
16  <<" adalah = "<<A[c]<<endl; }
17
18 return 0;}
```

Output Contoh 3 Program Input dan Output Array

```
Input Array
A[0] = 1
A[1] = 4
A[2] = 2
A[3] = 6
A[4] = 0
Nilai pada elemen ke 1 adalah = 1
Nilai pada elemen ke 2 adalah = 4
Nilai pada elemen ke 3 adalah = 2
Nilai pada elemen ke 4 adalah = 6
Nilai pada elemen ke 5 adalah = 0
```

Keterangan:

`int A[5];`

deklarasi array A yang bertipedata integer dengan maksimal 5 elemen.

`for (int i=0;i<5;i++)`

proses input dikerjakan mulai dari indeks 0 dampai indeks ke 4

`{ cout<<"A["<<i<<"] = ";`

Menampilkan A[]=, yang berada pada [] adalah I dimana I nanti berubah menjadi 0 sampai 4 sesuai proses dari perulangan di atas.

`cin>>A[i]; }`

perintah input nilai yang akan disimpan pada array A pada masing-masing indeks.

`for(int c=0;c<5;c++)`

proses menampilkan dikerjakan mulai dari indeks 0 dampai indeks ke 4

```
{ cout<<"Nilai pada elemen ke "<<c+1
 <<" adalah = "<<A[c]<<endl; }
```

Menampilkan kalimat "`Nilai pada elemen ke`" dan `c+1` menunjukkan elemen ke 1 sampai 5 dan `<<A[c]` menampilkan nilai yang tersimpan pada array A pada masing-masing indek.

Contoh 4 Program Edit Array

```
#include <iostream>
#include <string>
using namespace std;

int main(){

int A [5]={20,9,1986,200,13};
int n;

 system("cls");
 cout<<"Data yang lama\n";
 for (n=0;n<5;n++)
   {
      cout<<" "<<A[n];
   }

 cout<<"\nData yang baru : \n";
 A[0]=4; A[1]=2; A[2]=1; A[3]=3; A[4]=5;

 for (n=0;n<5;n++)
   {
      cout<<" "<<A[n];
   }
return 0;}
```

Output Contoh 4 Program Edit Array

```
Data yang lama
 20 9 1986 200 13
Data yang baru :
 4 2 1 3 5
```

Keterangan:

Pada dasarnya perintah edit disini yaitu mengiputkan ulang nilai kedalam array yang sama.

Contoh 5 Program Hapus Elemen Array

```
#include <iostream>
using namespace std;

int main(){

int n,hapus,A [5]={20,9,1986,200,13};

cout<<"Data yang lama\n";
for (n=0;n<5;n++)
  {
    cout<<" "<<A[n];
  }

cout<<" \ndata yang ingin dihapus : ";
cin>>hapus;

cout<<"\nData yang baru : \n";
for (n=hapus-1;n<5-1;n++)
 {
   A[n]=A[n+1];
 }

for (n=0;n<4;n++)
 {
    cout<<" "<<A[n];
 }
return 0;}
```

Output Contoh 5 Program Hapus Elemen Array

```
Data yang lama
 20 9 1986 200 13
data yang ingin dihapus : 3

Data yang baru :
 20 9 200 13
```

Keterangan:

Untuk proses hapus terlebih dahulu data/nilai diinputkan, perintah input sama dengan contoh sebelumnya dan sudah dijelaskan.

```
cout<<" data yang ingin dihapus : ";
cin>>hapus;
```

user diminta menginputkan elemen ke berapa yang nilainya akan dihapus.

```
for (n=hapus-1;n<5-1;n++)
```

proses perulangan dimulai dari nilai hapus(elemen yang akan dihapus)-1, kenapa harus di kurangi 1? Karena proses array dimulai dari indek 0 sedangkan hapus mulai elemen 1 jadi biar nilainya sama maka dari hapus(elemen)-1 sehingga nilainya sama dengan indek. Dan perulangan berhenti sampai batas-1. Dimana batas kurang dari 5.

```
{ A[n]=A[n+1];}
```

Array A indek ke n posisinya sama dengan array indek ke n+1. Nilai yang sudah terhapus pada indek ke n posisinya sama dengan elemen ke n+1.

```
for (n=0;n<4;n++)
```

perulangan untuk menampilkan nilai yang sudah dihapus dimulai dari indek ke 0 sampai indek ke 3. Kenapa tidak sampai indek ke 4? Karena 1 indek sudah terhapus jadi indek yang tersisa hanya sampai indek ke 3.

```
{ cout<<" "<<A[n]; }
```

Perintah untuk menampilkan nilai yang sudah terhapus, tetapi nilai yang terhapus tidak ditampilkan.

Contoh 6 Program Cari Nilai Array

```
2  #include <iostream>
3  using namespace std;
4
5  int main(){
6
7  //deklarasi array
8  int A[10]={12,24,13,25,10,11,21,20,15,18};
9  int bil;
10
11 //menampilkan elemen array
12 for (int i=0;i<10;i++)
13  {
14     cout<<A[i]<<" ";
15  }
16 cout<<endl;
17
18 //memasukkan nilai yang akan dicari
19 cout<<"Masukkan nilai yang akan dicari : ";
20 cin>>bil;
21
22 //pencarian data
23 for (int c=0;c<10;c++)
24  {
25     if (A[c]==bil)
26      {
27        cout<<"Nilai yang anda cari terdapat pada indek ke- "<<c;
28        break;
29      }
30  }
31 return 0;}
```

Output Contoh 6 Program Cari Nilai Array

```
12 24 13 25 10 11 21 20 15 18
Masukkan nilai yang akan dicari : 11
Nilai yang anda cari terdapat pada indek ke- 5
```

Keterangan:

Untuk proses cari terlebih dahulu data/nilai diinputkan, perintah input sama dengan contoh sebelumnya dan sudah dijelaskan.

```
cout<<"Masukkan nilai yang akan dicari : ";
cin>>bil;
```

user diminta untuk menginputkan nilai yang akan dicari, dan nilai tersebut disimpan pada variabel bil.

```
for (int c=0;c<10;c++)
```

perulangan untuk pencarian nilai dimulai dari indek ke 0 sampai indek ke 9

```
{ if (A[c]==bil)
{cout<<"Nilai yang dicari ada pada indek ke- "<<c;
```

Jika nilai yang ada pada array A di masing-masing indek nilainya sama dengan nilai yang ada di variabel bil maka akan muncul pesan `Nilai yang dicari ada pada indek ke-` sesuai dengan indek dimana nilai tersebut ditemukan.

10.2.2. Array Dua Dimensi (Multi Dimensi)

Array dua dimensi yang sering digambarkan sebagai sebuah matrik dimana array dua dimensi memiliki banyak baris dan banyak kolom. Deklarasi array dua dimensi secara umum:

```
tipe_data nama_var_array [batas_baris] [batas_kolom];
```

Keterangan:

`tipe_data` : menyatakan jenis tipe data elemen larik (int, char, float, dll)

`nama_var_array` : menyatakan nama variabel yang dipakai.

`batas_baris` : menyatakan jumlah/ukuran baris pada array

`batas_kolom` : menyatakan jumlah/ukuran komlom pada array.

Contoh : `int nilai[3][2];`

Keterangan contoh: array dua dimensi dengan nama nilai bertipe data integer dengan jumlah baris 3 dan jumlah kolom 2.

Untuk menginisialisasi array sama dengan memberikan nilai awal array pada saat didefinisikan.

```
int nilai[3][2] = {{8,-3},{9,0},{5,2}};
```

contoh array 2 dimensi dalam bentuk tabel dapat terlihat pada tabel 10.1

Tabel 10.1 Contoh Array Dua Dimensi dalam Bentuk Tabel

Jurusan	1992	1993	1994	1995
1. Teknik Informatika	35	45	80	120
2. Manajemen Informatika	100	110	70	101
3. Teknik Komputer	10	15	20	17

Bentuk seperti pada tabel diatas dapat dituangkan pada array berdimensi dua sebagai berikut: `int data_lulus [3] [4];`

Pada pendefinisian diatas :

3 menyatakan jumlah baris (mewakili jurusan) dan 4 menyatakan jumlah kolom (mewakili tahun kelulusan).

Array hasil pendefinisian diatas dapat dinyatakan seperti pada tabel 10.2

Tabel 10.1 Hasil Pendefinisian Array

TI → 0	35	45	80	120
MI→ 1	100	110	70	101
TK → 2	10	15	20	17
	0	1	2	3
	1992	1993	1994	1995

1. Pada saat deklarasi/inisialisai

   Contoh:

```
int matriks[3][4];
int matriks2[3][4] = {{5,2,1,18},
                      {4,7,6,-9},
                      {9,0,4,43}
                     };
```

2. Input secara statis

   Contoh:

```
Nilai[0][0]=1;
Nilai[0][1]=2;
Nama[0][0]=”Yuni”;
Nama[0][1]=”Yani”;
```

3. Inputan secara dinamis

   Contoh:

```
int A[3][2];
for(int j=0;j<3;j++)
{ for(int k=0;k<2;k++)
  {cout<<"A["<<j<<"]["<<k<<"] = ";
   cin>>A[j][k];}}
```

Contoh 7 Program Array 2D

```
#include <iostream>
using namespace std;

int main(){

int A[3][2];
cout<<"Input Array \n";
for(int j=0;j<3;j++)
  {
    for(int k=0;k<2;k++)
      {
        cout<<"A["<<j<<"]["<<k<<"] = ";
        cin>>A[j][k];
      }
  }
cout<<"\nOutput Array \n";
for(int j=0;j<3;j++)
  {
    for(int k=0;k<2;k++)
      {
        cout<<"A["<<j<<"]["<<k<<"]= "<<A[j][k]<<endl;
      }
  }
    return 0;
}
```

Output Contoh 7 Program Array 2D

```
Input Array
A[0][0] = 1
A[0][1] = 2
A[1][0] = 3
A[1][1] = 4
A[2][0] = 5
A[2][1] = 6

Output Array
A[0][0]= 1
A[0][1]= 2
A[1][0]= 3
A[1][1]= 4
A[2][0]= 5
A[2][1]= 6
```

Contoh 8 Program Array 2D

```
#include <iostream>
#include <conio.h>
#include <windows.h>

using namespace std;

//declarasi variabel
int biaya[4][4], i, j, enter=0;

//deklarasi fungsi gotoxy (header conio.h) + header windows.h
void gotoxy(short x, short y) {
    COORD pos = {x, y};
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), pos);
}

main(){
    cout<<endl;
    cout<<"\t\t   BIAYA SPP TETAP DI AMIKOM YOGYAKARTA"<<endl;
    cout<<endl;
    cout<<"=================================================================="<<endl;
    cout<<"|            |     2011    |     2012    |     2013    |     2014    |"<<endl;
    cout<<"=================================================================="<<endl;
    cout<<"|    D3MI    |             |             |             |             |"<<endl;
    cout<<"=================================================================="<<endl;
    cout<<"|    D3TI    |             |             |             |             |"<<endl;
    cout<<"=================================================================="<<endl;
    cout<<"|    S1SI    |             |             |             |             |"<<endl;
    cout<<"=================================================================="<<endl;
    cout<<"|    S1T1    |             |             |             |             |"<<endl;
    cout<<"=================================================================="<<endl;

    //perulangan untuk mengisikan data pada tabel
    for( i=0 ; i<4 ; i++)
        {
            for ( j=0 ; j<4 ; j++)
                {
                gotoxy(18*(i+1)-(i*4),6+j+enter);
                cin>>biaya[i][j];
                enter++;
                }
        }
    getch();
}
```

Output Contoh 8 Program Array 2D

```
                BIAYA SPP TETAP DI AMIKOM YOGYAKARTA

==================================================================
|            |     2011    |     2012    |     2013    |     2014    |
==================================================================
|    D3MI    |             |             |             |             |
==================================================================
|    D3TI    |             |             |             |             |
==================================================================
|    S1SI    |             |             |             |             |
==================================================================
|    S1T1    |             |             |             |             |
==================================================================
```

Pada array 2 dimensi sering digunakan untuk pembuatan tabel dan matrik, adapun operasi-operasi matrik yaitu:

1. **Input Matrik**

   Perintah input matrik sama seperti input pada array 2 dimensi, adapun perintahnya sebagai berikut:

```
int A[3][2];//matrik 3x2
for(int j=0;j<3;j++)
  {
    for(int k=0;k<2;k++)
      {
        cout<<"A["<<j<<"]["<<k<<"] = ";
        cin>>A[j][k];
      }
  }
```

2. **Output Matrik**

   Perintah output matrik sama seperti output pada array 2 dimensi, adapun perintahnya sebagai berikut:

```
int C[3][2]
for(int j=0;j<3;j++)
  {
    for(int k=0;k<2;k++)
     {
       cout<<"C["<<j<<"]["<<k<<"]="<<C[j][k]<<endl;;
     }
  }
```

3. **Penjumlahan Matrik**

   1. Agar kedua matrik dapat dijumlahkan harus memiliki jumlah baris dan kolom yang sama.
   2. Ada 3 matrik yang dibutuhkan yaitu matrik A,B dan C.
   3. Inputkan matrik A dan matriks B.
   4. Matrik C untuk menampung hasil penjumlahan matriks A dan B sesuai dengan elemen-elemennya.
   5. Elemen matrik A [0][0] dijumlahkan dengan elemen matrik B [0][0] juga dan disimpan di elemen matriks C [0][0] dan seterusnya, dengan rumus C[0][0] = A[0][0] + B[0][0].

C[2][3]

11	4	3
6	13	9

=

A[2][3]

2	3	1
6	9	8

+

B[2][3]

9	1	2
0	4	1

Adapun perintah untun penjumlahan matrik sebagai berikut:

```
typedef int matrik[3][2];
matrik A,B,C;
for(int j=0;j<3;j++)
 {
   for(int k=0;k<2;k++)
    {
      C[j][k]=A[j][k] + B[j][k];
    }
 }
```

4. Pengurangan Matrik

1. Agar kedua matrik dapat dikurangkan harus memiliki jumlah baris dan kolom yang sama.
2. Ada 3 matrik yang dibutuhkan yaitu matrik A,B dan C.
3. Inputkan matrik A dan matriks B.
4. Matrik C untuk menampung hasil penjumlahan matriks A dan B sesuai dengan elemen-elemennya.
5. Elemen matrik A [0][0] dijumlahkan dengan elemen matrik B [0][0] juga dan disimpan di elemen matriks C [0][0] dan seterusnya, dengan rumus C[0][0] = A[0][0] - B[0][0].

C[2][3]

-7	2	-1
6	5	7

=

A[2][3]

2	3	1
6	9	8

-

B[2][3]

9	1	2
0	4	1

Adapun perintah pengurangan matrik sebagai berikut:

```
typedef int matrik[3][2];
matrik A,B,C;
for(int j=0;j<3;j++)
  {
    for(int k=0;k<2;k++)
      {
        C[j][k]=A[j][k] - B[j][k];
      }
  }
```

5. Transpose Matrik

1. Transpose adalah elemen baris matriks akan menjadi kolom matriks dan sebaliknya kolom matriks akan menjadi baris matriks.
2. Siapkan matriks hasil untuk menampung hasil transpose

Matriks Awal :

1	2	3
4	5	6
7	8	9

Hasil Transpose:

1	4	7
2	5	8
3	6	9

Adapun perintah transpose sebagai berikut:

```
for(int j=0;j<3;j++)
 {
   for(int k=0;k<3;k++)
    {
      transpose[j][k] = A[k][j];
    }
 }
```

6. Mengambil Diagonal Matrik

Mengambil diagonal matrik yaitu mengambil nilai dari baris dan kolom yang sama.

Matrik awal :

1	2	3
4	5	6
7	8	9

Hasil Diagonal : 1,5,9

Adapun perintah diagonal matrik sebagai berikut:

```
for(int i=0;i<3;i++)
  {
    for(int j=0;j<3;j++)
     {
       if (i==j)
        {
          cout<<diagonal[i][j];
        }
     }
  }
```

7. Perkalian Matrik Dengan Konstanta Tertentu

Perkalian matrik (Syarat !!! Jumlah kolom matrik A harus sama dgn jumlah baris matrik B).

A[2][3] B[3][5] C[2][5]

1	2	3
4	5	6

x

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15

=

46	52	58	64	70
100	115	130	145	160

8. Perkalian Matrik

Syarat !!!

Jumlah kolom matrik A harus sama dgn jumlah baris matrik B. Adapun rumus perkalian matriknya sebagai berikut:

(1*1)+(2*6)+(3*11)= 1+12+33 = 46

(1*2)+(2*7)+(3*12)= 2+14+36 = 52

(1*3)+(2*8)+(3*13)= 3+16+39 = 58

(1*4)+(2*9)+(3*14)= 4+18+42 = 64

(1*5)+(2*10)+(3*15)= 5+20+45 = 70

(4*1)+(5*6)+(6*11)= 4+30+66 = 100

(4*2)+(5*7)+(6*12)= 8+35+72 = 115

(4*3)+(5*8)+(6*13)= 12+40+78 = 130

(4*4)+(5*9)+(6*14)= 16+45+84 = 145

(4*5)+(5*10)+(6*15)= 20+50+90 = 160

Adapun perintah perkalian antar matrik sebagai berikut:

```
for(i=0;i<2;i++)
 {
   for(j=0;j<5;j++)
    {
      C[i][j] = 0;
      for(k=0;k<3;k++)
       {
         C[i][j] = C[i][j] + A[i][k] * B[k][j];
       }
    }
 }
```

10.3. Latihan Soal

Gabungkanlah contoh progrm 1 sampai contoh program 6 di atas (pada array satu dimensi) sehingga menjadi satu program yang sempurna mulai dari input nilai ke dalam array, menampilkan nilai yang telah diinputkan, mengedit nilai dan tampilkan kembali nilai yang sudah diedit, menghapus nilai dan menampilkan nilai yang belum terhapus dan pencarian nilai.

BAB XI. SORTING (PENGURUTAN)

11.1. Tujuan Pembelajaran

Setelah mempelajari materi ini, mahasiswa diharapkan mampu :

1. Memahami pengertian dan perbedaan algoritma pada sorting data.
2. Menerapkan algoritma sorting disetiap metodenya (*Bubble, Selection / max-min* dan *Insertion*)

11.2. Dasar Teori

Pengurutan data (sorting) didefinisikan sebagai suatu proses untuk menyusun kembali himpunan obyek menggunakan aturan tertentu. Menurut Microsoft Book-shelf, definisi algoritma pengurutan adalah algoritma untuk meletakkan kumpulan elemen data ke dalam urutan tertentu berdasarkan satu atau beberapa kunci dalam tiap-tiap elemen.

Ada dua macam pengurutan data yaitu:

1. Urut naik (*ascending*) yaitu dari data yang mempunyai nilai paling kecil sampai paling besar.
2. Urut turun (*descending*) yaitu data yang mempunyai nilai paling besar sampai paling kecil.

Keuntungan data yang sudah terurutkan yaitu data mudah dicari (misalnya dalam buku telepon atau kamus bahasa), mudah untuk dibetulkan, dihapus, disisipi atau digabungkan. Dalam keadaan terurutkan, kita mudah melakukan pengecekan apakah ada data yang hilang dan mempercepat proses pencarian data yang harus dilakukan berulang kali. Ada 3 metode sorting yang sering digunakan yaitu *bubble sort* (gelembung), *selection sort* (maksimum/minimun) dan *insertion sort* (sisip).

11.2.1. Bubble Sort

Metode pengurutan gelembung (*bubble sort*) diinspirasi oleh gelembung sabun yang ada di permukaan air. Karena berat jenis gelembung sabun lebih ringan daripada berat jenis air maka gelembung sabun akan selalu mengapung.

Prinsip pengapungan ini juga dipakai pada pengurutan gelembung. Elemen yang berharga paling kecil “diapungkan”, artinya diangkat ke atas (atau ke ujung paling kiri)

melalui pertukaran. Proses pengapungan ini dilakukan N kali langkah. Pada langkah ke-I, Larik[1..N] akan terdiri dari 2 bagian yaitu:

- Bagian yang sudah terurut yaitu L[1]..L[i].
- Bagian yang belum terurut L[i+1]..L[n].

Algoritma bubble sort (gelembung):

Langkah 1:

Mulai dari elemen K=N,N-1,N-2,..2 bandingkan L[K] jika L[K] < L[K-1], pertukarkan L[K] dengan L[K-1].

Pada akhir langkah 1, elemen L[1] berisi harga minimum pertama.

Langkah 2:

Mulai dari elemen K=N,N-1,N-2,..3 bandingkan L[K] jika L[K] < L[K-1], pertukarkan L[K] dengan L[K-1].

Pada akhir langkah 2, elemen L[2] berisi harga minimum kedua dan L[1]..L[2] terurut..

Langkah 3:

Mulai dari elemen K=N,N-1,N-2,..4 bandingkan L[K] jika L[K] < L[K-1], pertukarkan L[K] dengan L[K-1].

Pada akhir langkah 3, elemen L[3] berisi harga minimum ketiga dan L[1]..L[3] terurut ...

Langkah N-1:

Mulai dari elemen K=N,N-1,N-2,..4 bandingkan L[K] jika L[K] < L[K-1], pertukarkan L[K] dengan L[K-1].

Contoh:

Ada array/larik dengan N=6 buah elemen dibawah ini, dan akan diurutkan secara ascending/menaik.

25	**27**	**10**	**8**	**76**	**21**
1	2	3	4	5	6

Langkah 1:

K=N=6					21	76
K=5				8	21	76
K=4			8	10	21	76
K=3		8	27	10	21	76

K=2 8 25 27 10 21 76

Hasil akhir langkah 1 **8** 25 27 10 21 76

Karena pada hasil langkah 1 data belum terurut secara ascending maka harus ada proses ke 2.

8 25 27 10 21 76

Langkah 2:

K=N=6 21 76

K=5 10 21 76

K=4 10 27 21 76

K=3 10 25 27 21 76

Hasil akhir langkah 2 **8** **10** 25 27 21 76

Karena pada hasil langkah 2 data belum terurut secara ascending maka harus ada proses ke 3.

8 **10** 25 27 21 76

Langkah 3:

K=N=6 21 76

K=5 21 27 76

K=4 21 25 27 76

Hasil akhir langkah 3 **8** **10** **21** 25 27 76

Hasil langkah 3 data sudah terurut secara ascending tetapi proses pengurutan masih tetap dikerjakan seperti pada langkah ke 4.

8 **10** **21** 25 27 76

Langkah 4:

K=N=6 27 76

K=5 25 27 76

Hasil akhir langkah 4 **8** **10** **21** **25** 27 76

Hasil langkah 4 data sudah terurut secara ascending tetapi proses pengurutan masih tetap dikerjakan seperti pada langkah ke 5.

8 **10** **21** **25** 27 76

Langkah 5:

K=N=6 27 76

Hasil akhir langkah 5 **8** **10** **21** **25** **27** 76

Hasil dari langkah ke 5 sudah terurutkan dan proses pengurutan sudah selesai.

Contoh 1 Program Bubble Sort Ascending

```
#include <iostream>
using namespace std;

int main(){
int L[5],i,k,temp;

L[0]=1;
L[1]=50;
L[2]=10;
L[3]=3;
L[4]=2;

//Proses secara Ascending(naik)
 for(i=0;i<=4;i++)
   {
     for(k=4;k>=0;k--)
       {
          if (L[k]<L[k-1])
           {
             temp=L[k];
             L[k]=L[k-1];
             L[k-1]=temp;
           }
       }
   }
cout<<"Hasil Urut Ascending :\n";
for(i=0;i<=4;i++)
  cout<<L[i]<<endl;

return 0;}
```

Output Contoh 1 Program Bubble Sort Ascending

```
Hasil Urut Ascending :
1
2
3
10
50
```

Keterangan:

Pada array L sudah diinputkan nilai secara statis sebanyak 5 elemen yaitu 1, 50, 10, 3, 2 dari ke lima nilai tersebut masih acak atau belum terurutkan. Perintah untuk mengurutkan nilai tersebut dengan metode bubble sort secara ascending yaitu :

```
for(i=0;i<=4;i++)
```

perulangan untuk larik/array dari indek 0 sampai indek 4

```
for(k=4;k>=0;k--)
```

perulangan untuk proses pengurutan dari posisi 4 sampai posisi 0. Pada setiap proses iterasi akan menjalankan perintah perbandingan nilai seperti dibawah ini:

```
if (L[k]<L[k-1])
```

Proses perbandingan, jika nilai yang ada di posisi k array L nilainya lebih kecil dari nilai yang ada di posisi k-1 array L, maka:

```
{temp=L[k];
```

Nilai yang ada di posisi k array L dipindah kedalam variabel `temp`. Variabel `temp` disini digunakan sebagai variabel temporari atau menyimpan data sementara. karena nilai yang ada di posisi k array L sudah dipindah ke `temp` sehingga posisi k array L sekarang tidak ada nilainya.

```
L[k]=L[k-1];
```

Kemudian nilai yang ada di posisi k-1 array L dipindah ke posisi k array L, sehingga posisi k array L yang tadinya kosong sekarang sudah terisi nilai dari posisi k+1 array L dan searang yang kosong adalah posisi k array L.

```
 L[k-1]=temp; }
```

Kemudian nilai yang ada di variabel `temp` dipindah ke posisi k-1 array L. sehingga sekarang yang kosong adalah variabel `temp` dan nilai sudah tertukar.

```
for(i=0;i<=4;i++)
  cout<<L[i]<<endl;
```

Perintah untuk menampilkan nilai pada larik/array L yang sudah terurutkan.

Contoh 2 Program Bubble Sort Descending

```
#include <iostream>
using namespace std;

int main(){
int L[5],i,k,temp;

L[0]=1;
L[1]=50;
L[2]=10;
L[3]=3;
L[4]=2;

//Proses secara Ascending(naik)
 for(i=0;i<=4;i++)
   {
     for(k=5;k>=1;k--)
       {
          if (L[k]>L[k-1])
           {
             temp=L[k];
             L[k]=L[k-1];
             L[k-1]=temp;
           }
       }
   }
cout<<"Hasil Urut Descending :\n";
for(i=0;i<5;i++)
  cout<<L[i]<<endl;

return 0;}
```

Output Contoh 2 Program Bubble Sort Descending

```
Hasil Urut Descending :
50
10
3
2
1
```

Keterangan:

Pada array L sudah diinputkan nilai secara statis sebanyak 5 elemen yaitu 1, 50, 10, 3, 2 dari ke lima nilai tersebut masih acak atau belum terurutkan. Perintah untuk mengurutkan nilai tersebut dengan metode bubble sort secara descending yaitu :

`for(i=0;i<=4;i++)`

perulangan untuk larik/array dari indek 0 sampai indek 4

`for(k=5;k>=1;k--)`

perulangan untuk proses pengurutan dari posisi 5 sampai posisi 1. Pada setiap proses iterasi akan menjalankan perintah perbandingan nilai seperti dibawah ini:

```
if (L[k]>L[k-1])
```

Proses perbandingan, jika nilai yang ada di posisi k array L nilainya lebih besar dari nilai yang ada di posisi k-1 array L, maka:

```
{temp=L[k];
```

Nilai yang ada di posisi k array L dipindah kedalam variabel `temp`. Variabel `temp` disini digunakan sebagai variabel temporari atau menyimpan data sementara. karena nilai yang ada di posisi k array L sudah dipindah ke `temp` sehingga posisi k array L sekarang tidak ada nilainya.

```
L[k]=L[k-1];
```

Kemudian nilai yang ada di posisi k-1 array L dipindah ke posisi k array L, sehingga posisi k array L yang tadinya kosong sekarang sudah terisi nilai dari posisi k-1 array L dan sekarang yang kosong adalah posisi k array L.

```
L[k-1]=temp; }
```

Kemudian nilai yang ada di variabel `temp` dipindah ke posisi k-1 array L. sehingga sekarang yang kosong adalah variabel `temp` dan nilai sudah tertukar.

```
for(i=0;i<5;i++)
 cout<<L[i]<<endl;
```

Perintah untuk menampilkan nilai pada larik/array L yang sudah terurutkan.

Kesimpulan:

Pengurutan dengan metode bubble sort ini kurang efisien karena terlalu banyak penukaran yang dilakukan pada setiap langkah dan membutuhkan banyak waktu serta proses lebih lama, sehingga tidak direkomendasikan untuk dipakai. Namun metode ini mudah dipahami dan sederhana.

11.2.2. Selection sort (Maksimal atau Minimal)

Metode pengurutan ini disebut pengurutan maksimum/minimum karena didasarkan pada pemilihan elemen maksimum atau minimum kemudian mempertukarkan elemen maksimum/minimum tersebut dengan elemen terujung larik (elemen ujung kiri atau elemen ujung kanan). Selanjutnya elemen terujung itu kita "isolasi" dan tidak diikut sertakan pada proses selanjutnya. Karena proses utama dalam

pengurutan adalah pemilihan elemen maksimum / minimum, maka metode ini disebut metode pemilihan (*selection sort*). Berikut algoritma pengurutan maksimum/minimum:

Algoritma Selection Sort (maksimum/minimun):

Langkah 1:

Tentukan Harga Maksimum didalam L1[1..N]

Pertukarkan harga maksimum dng L[N]

Langkah 2:

Tentukan Harga Maksimum didalam L1[1..N-1]

Pertukarkan harga maksimum dng L[N-1]

Langkah 3:

Tentukan Harga Maksimum didalam L1[1..N-2]

Pertukarkan harga maksimum dng L[N-2]

Langkah N-1:

Tentukan Harga Maksimum didalam L1[1..2]

Pertukarkan harga maksimum dng L[2].

Contoh:

Ada array/larik dengan N=6 buah elemen dibawah ini, dan akan diurutkan secara ascending/menaik.

29	**27**	**10**	**8**	**76**	**21**
1	2	3	4	5	6

Langkah 1:

Cari elemen maksimum di dalam larik L[1..6]→ maks = L[5] = 76

Tukar maks dengan L[N],hasil akhir langkah 1:

29	27	10	8	21	**76**
1	2	3	4	5	6

Langkah 2:

(berdasarkan susunan larik hasil langkah 1)

Cari elemen maksimum di dalam larik L[1..5]→ maks = L[1] = 29

Tukar maks dengan L[5],hasil akhir langkah 2:

21	27	10	8	**29**	**76**
1	2	3	4	5	6

Langkah 3:

(berdasarkan susunan larik hasil langkah 2)

Cari elemen maksimum di dalam larik L[1..4]→ maks = L[2] = 27

Tukar maks dengan L[4],hasil akhir langkah 3:

21	8	10	**27**	**29**	**76**
1	2	3	4	5	6

Langkah 4:

(berdasarkan susunan larik hasil langkah 3)

Cari elemen maksimum di dalam larik L[1..3]→ maks = L[1] = 21

Tukar maks dengan L[3],hasil akhir langkah 4:

10	8	**21**	**27**	**29**	**76**
1	2	3	4	5	6

Langkah 5:

(berdasarkan susunan larik hasil langkah 4)

Cari elemen maksimum di dalam larik L[1..2]→ maks = L[1] = 10

Tukar maks dengan L[2],hasil akhir langkah 5:

8	**10**	**21**	**27**	**29**	**76**
1	2	3	4	5	6

Jika nilai pada array/larik sudah terurutkan maka proses sorting sudah selesai.

Contoh 3 Program Selection Sort Ascending

```
#include <iostream>
using namespace std;

int main(){

int A[7];
int j,k,i,temp;
int jmax,u=6;

cout<<"Masukkan nilai pada elemen array :"<<endl;
for(i=0;i<7;i++)
  {
    cout<<"A["<<i<<"]=";
    cin>>A[i];
  }

for(j=0;j<7;j++)
  {
    jmax=0;
    for(k=1;k<=u;k++)
      {
        if (A[k] > A[jmax])
          jmax=k;
      }
    temp=A[u];
    A[u]=A[jmax];
    A[jmax]=temp;
    u--;
  }

//menampilkan nilai setelah diurutkan
cout<<"\nNilai setelah diurutkan ="<<endl;
for(i=0;i<7;i++)
   cout<<A[i]<<" ";
return 0;}
```

Output Contoh 3 Program Selection Sort Ascending

```
Masukkan nilai pada elemen array :
A[0]=1
A[1]=5
A[2]=2
A[3]=4
A[4]=0
A[5]=8
A[6]=3

Nilai setelah diurutkan =
0 1 2 3 4 5 8
```

11.2.3. Insertion Sort (Sisip)

Metode pengurutan data ini disebut pengurutan sisip (insertion sort) yaitu metode pengurutan dengan cara menyisipkan elemen larik pada posisi yang tepat. Pencarian posisi yang tepat dilakukan dengan pencarian beruntun. Selama pencarian

posisi yang tepat dilakukan pergeseran elemen larik. Cara pengurutan datanya berbeda dengan cara pengurutan dua metode sebelumnya karena pengurutan ini dimulai dari elemen ke dua. Berikut algoritma pengurutan sisip:

Algoritma Insertion Sort (Sisip)

Andaikan:

L[1] dianggap sudah pada tempatnya.

Langkah 2:

L[2] harus dicari tempatnya yang tepat pada L[1..2] dengan cara menggeser elemen L[1] ke kanan bila L[1] lebih besar dari L[2].
Misalkan posisi elemen yang tepat adalah K sisipkan L[2] pada K.

Langkah 3:

L[3] harus dicari tempatnya yang tepat pada L[1..3] dengan cara menggeser elemen L[1..2] ke kanan bila L[1..2] lebih besar dari L[3]. Misalkan posisi elemen yang tepat adalah K sisipkan L[3] pada K.

Langkah 4:

L[4] harus dicari tempatnya yang tepat pada L[1..4] dengan cara menggeser elemen L[1..4] ke kanan bila L[1..4] lebih besar dari L[4]. Misalkan posisi elemen yang tepat adalah K sisipkan L[4] pada K.

Langkah N:

L[N] harus dicari tempatnya yang tepat pada L[1..N] dengan cara menggeser elemen L[1..N ke kanan bila L[1..N] lebihbesar dari L[N].
Misalkan posisi elemen yang tepat adalah K sisipkan L[N] pada K.

Contoh:

Elemen array/larik dengan N=6 buah elemen dibawahini yang akan diurutkan secara ascending.

29	**27**	**10**	**8**	**76**	**21**
1	2	3	4	5	6

Langkah 1:

Elemen L[1] dianggap sudah terurut

29	27	10	8	76	21
1	2	3	4	5	6

Langkah 2:

(berdasarkan susunan larik pada langkah 1)

Cari posisi yang tepat untuk L[2] pada L[1..2],diperoleh :

27	**29**	10	8	76	21
1	2	3	4	5	6

Langkah 3:

(berdasarkan susunan larik pada langkah 2)

Cari posisi yang tepat untuk L[3] pada L[1..3],diperoleh :

10	**27**	**29**	8	76	21
1	2	3	4	5	6

Langkah 4:

(berdasarkan susunan larik pada langkah 3)

Cari posisi yang tepat untuk L[4] pada L[1..4],diperoleh :

8	**10**	**27**	**29**	76	21
1	2	3	4	5	6

Langkah 5:

(berdasarkan susunan larik pada langkah 4)

Cari posisi yang tepat untuk L[5] pada L[1..5],diperoleh :

8	**10**	**27**	**29**	**76**	21
1	2	3	4	5	6

Langkah 6:

(berdasarkan susunan larik pada langkah 5)

Cari posisi yang tepat untuk L[6] pada L[1..6],diperoleh :

8	**10**	**21**	**27**	**29**	**76**
1	2	3	4	5	6

Jika nilai pada array/larik sudah terurutkan maka proses sorting sudah selesai.

Contoh 4 Program Insertion Sort Ascending

```
2   using namespace std;
3
4   main(){
5   int L[5],k,temp,j;
6   L[0]=1;
7   L[1]=25;
8   L[2]=10;
9   L[3]=30;
10  L[4]=2;
11  cout<<"\nData sebelum diurutkan :"<<endl;
12  for(k=0;k<=4;k++)
13   {
14      cout<< L[k]<<" ";
15   }
16  for(k=2;k<=5;k++)
17   {
18    temp=L[k];/* ambil elemen L[k] supaya tidak tertimpa penggeseran*/
19              /* Cari Posisi Yang tepat dalam L[1..k-1] sambil menggeser*/
20    j=k-1;
21    while(temp<=L[j])
22      {
23         L[j+1]=L[j];
24         j--;
25      }
26    if((temp >= L[j])|| ( j=1))
27      L[j+1]=temp; /*posisi yg tepat untuk L[k] ditemukan*/
28    else
29      {
30        L[j+1]=L[j];
31        L[j]=temp;
32      }
33   }
34  cout<<"\n\nData setelah diurutkan :"<<endl;
35  for(k=1;k<=5;k++)
36   {cout<< L[k]<<" ";}
37  return 0;}
```

Output Contoh 4 Program Insertion Sort Ascending

```
Data setelah diurutkan :
1 2 10 25 30
```

11.3. Latihan Soal

Kerjakan soal di bawah ini!

1. Tulis algoritma pengurutan no *handphone* anda masing-masing menggunakan 3 metode secara *ascending* dan *descending*.
2. Dari soal no 1 maka tuliskan proses 3 metode dari masing-masing programnya.
3. Buatlah program untuk mengurutkan nama anda masing-masing secara *ascending* dan *descending* menggunakan salah satu metode sorting.

BAB XII. SEARCHING (PENCARIAN)

12.1. Tujuan Pembelajaran

Setelah mempelajari materi ini, mahasiswa diharapkan mampu :

1. Memahami pengertian dan perbedaan algoritma pada *searching* data
2. Menulis dan menjelaskan algoritma sorting disetiap metodenya (*Binary, Sentinel* dan *Sequential)*

12.2. Dasar Teori

Pada bab ini akan menjelaskan tentang searching yaitu pencarian data dengan membandingkan setiap elemen larik satu per satu secara urut (beruntun), mulai dari elemen pertama sampai dengan elemen yang terakhir. Ada 2 macam pencarian beruntun,yaitu pencarian pada array yang **sudah** terurut, dan pencarian pada array yang **belum** terurut. Dalam materi ini akan dijelaskan 2 metode pencarian yaitu pencarian dengan metode Sekuensial Search (beruntun) dan pencarian dengan metode Binary Search (bagi dua).

12.2.1. Binary Search (Bagi Dua)

Pada pencarian metode binary ini ada syaratnya yaitu data harus urut terlebih dahulu kemudian baru bisa diurutkan. Salah satu keuntungan data yang terurut adalah memudahkan pencarian, yang dalam hal ini adalah pencarian bagi dua. Sebenarnya dalam kehidupan sehari-hari kita sering menerapkan algoritma ini. Untuk mencari kata tertentu dalam kamus (misalnya kamus bahasa Inggris), kita tidak membuka kamus tersebut dari halaman awal sampai halaman akhir satu persatu, namun kita mencarinya dengan cara membelah atau membagi halaman-halaman buku tersebut. Begitu seterusnya sampai kita menemukan kata yang dicari.

Algoritma Binary Search

Contoh:

Elemen array/larik dengan N=6 buah elemen dibawahini.

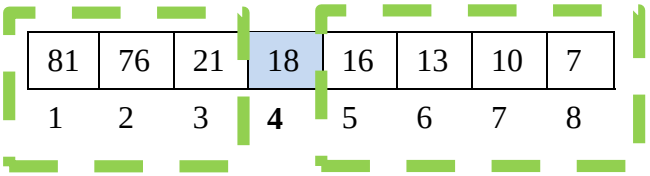
81	76	21	18	16	13	10	7
ia=1	2	3	4	5	6	7	8=ib

a. Misal elemen yang dicari adalah X = 18

   langkah 1:

ia=1 dan ib=8

elemen tengah K=(1+8)/2 = 4 (diarsir)



Langkah 2:

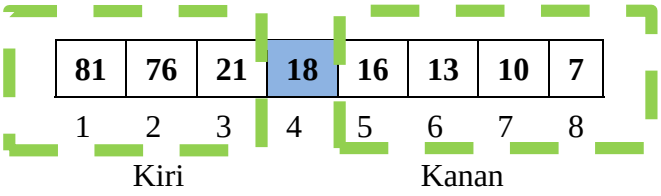
L[4]=18? Ya! (X ditemukan, pencarian dihentikan)

b. Misal elemen yang dicari adalah X = 16

langkah 1:

ia=1 dan ib=8

elemen tengah K=(1+8)/2 = 4 (diarsir)

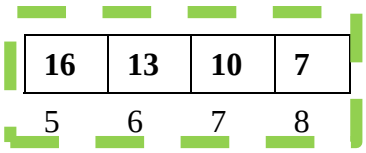


Langkah 2:

L[4]=18? Tidak !

harus diputuskan apakan pencarian akan dilakukan di bagian kiri atau kanan dengan pemeriksaan sbb:

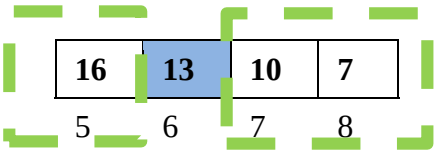
L[4] > 16 ? Ya ! Lakukan pencarian pada larik bagian kanan dengan ia=k+1 = 5 dan ib = 8 (tetap).



Langkah 2.1:

ia = 5 dan ib = 8

elemen tengah K = (5+8)/2 = 6 (diarsir)

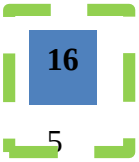


Langkah 2.2:

L[6] = 16? Tidak !

harus diputuskan apakan pencarian akan dilakukan di bagian kiri atau kanan dengan pemeriksaan sbb:

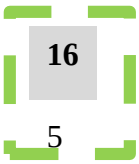
L[6] > 16 ? Tidak ! Lakukan pencarian pada larik bagian kiri dengan ia= 5 dan ib = k-1 = 5



Langkah 2.1.1:

ia = 5 dan ib = 5

elemen tengah K = (5+5)/2 = 5 (diarsir)



Langkah 2.1.2:

L[5] = 16? Ya ! (X ditemukan,pencarian dihentikan)

Contoh 3 Program Binary Search

```
4  int main(){
5  const int arraySize = 5;
6  int target;
7  // Array size and values already known.
8  int array[arraySize] = {1, 2, 3, 4, 5};
9  int first, mid, last;
10
11 cout<<"Enter a target to be found: ";cin>>target;
12 // Initialize first and last variables.
13 first = 0;
14 last = 4;
15
16 while(first <= last)
17  {
18     mid = (first + last)/2;
19     if(target > array[mid])
20      {
21        first = mid + 1;
22      }
23     else if(target < array[mid])
24      {
25        last = mid - 1;
26      }
27     else
28      {
29        first = last + 1;
30      }
31  }
32 if(target == array[mid])
33  {
34     cout << "Target found" << endl;
35  }
36 else
37  {
38     cout << "Target not found" << endl;
39  }
40 return 0;}
```

Output Contoh 1 Program Binary Search

```
Enter a target to be found: 1
Target found
```

Keterangan:

```
const int arraySize = 5;
int target;
int array[arraySize] = {1, 2, 3, 4, 5};
int first, mid, last;
```

pendeklarasian variabel dan pemberian nilai pada array.

```
cout << "Enter a target to be found: ";
cin >> target;
```

input nilai yang akan dicari

```
first = 0;
last = 4;
```
inisialisasi variabel first dan variabel last

```
while(first <= last)
  { mid = (first + last)/2;
    if(target > array[mid])
     { first = mid + 1; }
    else if(target < array[mid])
     { last = mid - 1; }
    else
     { first = last + 1; }
  }
```

Perulangan akan terus diulang selama variabel first lebih kecil sama dengan variabel last. Variabel mid digunakan untuk mencari titik tengah yaitu dari posisi (first + last)/2.

Jika nilai yang dicari pada variabel target lebih besar dari titik tengah pada array maka proses pencarian akan dimulai dari first yaitu pada posisi kanan dari titik tengah atau mid+1, tetapi jika nilai yang dicari pada variabel target lebih kecil dari titik tengah pada array maka proses pencarian berakhir pada last yaitu dari titk tengah/mid-1. Jika selain keduanya maka pencarian sama dengan titik tengah.

```
if(target == array[mid])
  { cout << "Target found" << endl; }
else
  { cout << "Target not found" << endl; }
```

Jika nilai yang dicari pada variabel target sama dengan nilai yang ada pada titik tengah aray maka akan muncul pesan "`Target found`" Jika tidak maka akan muncul pesan "`Target not found`".

12.2.2. Sequential (Beruntun)

Konsep metode ini yaitu membandingkan setiap elemen larik satu per satu secara urut (beruntun), mulai dari elemen pertama sampai dengan elemen yang terakhir. Ada 2 macam pencarian beruntun, yaitu pencarian pada array yang **sudah** terurut dan pencarian pada array yang **belum** terurut.

Algoritma Sequential Search

Contoh: Elemen array/larik dengan N=6 buah elemen dibawahini.

13	**16**	**14**	**21**	**76**	**21**
1	2	3	4	5	6

Nilai yang akan dicari ada pada variabel X = 21

Maka, elemen yang di cek adalah 13, 16, 14, 21 (ditemukan), lanjutkan cek ke 76 dan 21 berarti data ditemukan pada indeks ke 3 elemen ke 4.

Contoh 2 Program Sequential Search

```
1   #include <iostream>
2   #include <iomanip>
3
4   using namespace std;
5
6   main(){
7
8   int A[]={12,24,13,25,10,13,21,15,15,18};
9   int bil;
10
11  for(int i=0;i<10;i++)
12   {
13      cout<<setw(4)<<A[i];
14   }
15  cout<<endl;
16  cout<<"Angka yang dicari = ";cin>>bil;
17  cout<<endl;
18  for(int i=0;i<10;i++)
19   {
20     if(A[i]==bil)
21       cout<<"angka yang dicari berada di indeks ke-"<<i<<endl;
22   }
23  return 0;}
```

Output Contoh 2 Program Sequential Search

```
  12   24   13   25   10   13   21   15   15   18
Angka yang dicari = 15

angka yang dicari berada di indeks ke-7
angka yang dicari berada di indeks ke-8
```

Keterangan:

```
int A[]={12,24,13,25,10,13,21,15,15,18};
int bil;
```

pendeklarasian variabel bil, array A dan array A langsung didefinisikan nilainya.

```
for(int i=0;i<10;i++){
cout<<setw(4)<<A[i];}
```

perulangan untuk menampilkan nilai ayang ada pada array A secara vertikal dengan jarak 4 spasi untuk masing-masing nilai.

```
cout<<endl;
cout<<"Angka yang dicari = ";cin>>bil;
cout<<endl;
```

perintah untuk menginputkan nilai yang akan dicari. Nilai yang dicari boleh nilai yang ada pada array A boleh juga nilai yang tidak ada di array A.

```
for(int i=0;i<10;i++){
  if(A[i]==bil)
     cout<<"angka yang dicari berada di indeks ke- "
         <<i<<endl;}
```

perulangan untuk pencarian dimulai dari indek 0 sampai indek 9, jika nilai pada masing-masing indek nilai pada array A elemen i sama dengan nilai yang ada pada variabel bil/nilai yang akan dicari maka akan muncul tampilan " ngka yang dicari berada di indeks ke- i, i disini nanti akan muncul angka tergantung nilai ditemukan pada posisi indek berapa.

12.3. Latihan Soal

Buatlah program untuk pencarian data beserta pengurutan data, dengan tampilan sebagai berikut :

```
      PROGRAM SEARCHING DAN SORTING

Input banyak data        :
Input data               :
Tampil data acak         :

Pilih etode sorting      :
   1. Bubble sort
   2. selection sort
   3. insertion sort
Tampilkan data terurut   :

Pilih metode searching   :
   1. Sequensial search
   2. Binary search
Input nilai yang dicari  :
Tampil nilai yg dicari   :

     Selamat mencoba, happy coding……
```

BAB XIII. STRUCTURE (STRUKTUR)

13.1. Tujuan Pembelajaran

Setelah mempelajari materi ini, mahasiswa diharapkan mampu :

1. Memahami cara mendeklarasikan struktur
2. Memahami cara menginisialisasi struktur
3. Memahami cara mengakses elemen struktur
4. Memahami pembentukan dan cara mengakses array dari struktur *(array of struct)*

13.2. Dasar Teori

Struktur merupakan kumpulan elemen data yang digabungkan menjadi satu kesatuan data. Masing-masing elemen data tersebut dinamakan field atau elemen struktur. Field tersebut bisa memiliki tipe data yang ataupun berbeda, meskipun field tersebut dalam satu kesatuan tetapi tetap bisa siakses secara individu. Pendeklarasian struct secara umum yaitu:

```
struct nama_struktur // nama struktur, kata struct
harus ada
 {
  type1 element1;
  type2 element2;  anggota / elemen dari struktur
  type3 element3;
  .
  .
} nama_object;   // identifier yang digunakan untuk
                 pemanggilan struktur

----------atau----------

struct nama_struktur
{
  type1 element1;
  type2 element2;
  type3 element3;
  .
```

Cara mendeklarasikan :

1. Struktur dengan tipe data berbeda dengan nama mahasiswa

```
struct Mahasiswa {
char nama [20];
float ip;
int semester;
} mhs;
```

ATAU

```
struct Mahasiswa{
char nama[20];
float ip;
int semester;
};
struct Mahasiswa mhs;
```

ATAU

```
struct Mahasiswa{
char nama[20];
float ip;
int semester;
};
Mahasiswa mhs;
```

2. Struktur dengan tipe data sama

```
struct  tanggal
{
  int tanggal;
  int bulan;
  int tahun;
}tgl;
```

---atau bisa ditulis ---

```
struct  tanggal
{
  int tanggal, bulan, tahun;
}tgl;
```

Contoh format pengaksesan struct:

```
nama_object.nama_member
```

Contoh pengaksesan struct mahasiswa

```
mhs.semester = 2;
mhs.ip = 3.67;
```

Membaca data dari keyboard

```
cin>>mhs.semester;
cin>>mhs.ip;
```

Contoh 1 Program Struct

```
#include <iostream>
using namespace std;


struct stok {
  char nama [50];
  int jml;
}stoks ;

int main (){
  cout<<"Masukkan nama barang   = ";cin>>stoks.nama;
  cout<<"Masukkan jumlah barang = ";cin>>stoks.jml;
  cout<<"";
  cout<<"------------------------\n";
  cout<<"";
  cout<<"Output "<<stoks.nama<<" = ";
  cout<<stoks.jml;
return 0;}
```

Output contoh 1 Program Struct

```
Masukkan nama barang   = Indomie
Masukkan jumlah barang = 2
------------------------
Output Indomie = 2
```

Contoh 2 Program Struct

```
#include <iostream>
using namespace std;


struct stok {
  char nama [50];
  int jml;
};
struct stok persedian;
int main (){
  cout<<"Masukkan nama barang   = ";cin>> persedian.nama;
  cout<<"Masukkan jumlah barang = ";cin>> persedian.jml;
  cout<<"";
  cout<<"------------------------\n";
  cout<<"";
  cout<<"Output "<< persedian.nama<<" = ";
  cout<< persedian.jml;

return 0;}
```

Output contoh 2 Program Struct

```
Masukkan nama barang   = Indomie
Masukkan jumlah barang = 2
------------------------
Output Indomie = 2
```

c

Contoh 3 Program Struct

```
#include <iostream>
using namespace std;

struct data_tgl
{ int tgl,bln,thn; };

struct teman
{ char nama[20];
  char j_kel[1];
  struct data_tgl tgl; };

struct teman info;
int main (){
  //input data
  cout<<"Masukkan nama anda = ";cin>>info.nama;
  cout<<"Jenis kelamin anda = ";cin>>info.j_kel;
  cout<<"Tanggal lahir anda = ";cin>>info.tgl.tgl;
  cout<<"Bulan lahir anda   = ";cin>>info.tgl.bln;
  cout<<"Tahun lahir anda   = ";cin>>info.tgl.thn;
  cout<<"";
  cout<<"------------------------\n";
  cout<<"";
  //output data
  cout<<"Nama       : "<<info.nama;
  cout<<"\nKelamin       : "<<info.j_kel;
  cout<<"\nTanggal lahir : "<<info.tgl.tgl<<"-"<<info.tgl.bln
    <<"-"<<info.tgl.thn;
return 0;}
```

Output contoh 3 Program Struct

```
Tanggal lahir anda = 28
Bulan lahir anda   = 10
Tahun lahir anda   = 2017
------------------------
Nama       : Hafizh
Kelamin       : L
Tanggal lahir : 28-10-2017
```

Suatu struktur dapat berisi dengan elemen berupa struktur yang lain.

Contoh 4 Program Struct dalam Struct

```
#include <iostream>
using namespace std;

struct tanggal{ int hari;
        int bulan;
        int tahun; };
struct alamat { char jalan[30];
        char kota[20]; };
struct { char nama[40];
     struct tanggal masuk;
     struct alamat tinggal;
     float gaji;
          }karyawan={"Arief Kurniawan", 17,11,87,"Jalan Raya 5","Yogyakarta", 750000.00};
int main(){
/*menampilkan data karyawan*/
cout<<"Nama Karyawan    :  "<< karyawan.nama;
cout<<"\nTangggal Lhr   :  "<< karyawan.masuk.hari<<"-"<<karyawan.masuk.bulan<<"-"<<karyawan.masuk.tahun ;
cout<<"\nalamat         :  "<< karyawan.tinggal.jalan;
cout<<"\n                 "<< karyawan.tinggal.kota;
cout<<"\nGaji Karyawan  Rp "<< karyawan.gaji;

return 0;}
```

Output Contoh 4 Program Struct dalam Struct

```
Nama Karyawan    :  Arief Kurniawan
Tangggal Lhr   :  17-11-87
alamat         :  Jalan Raya 5
                  Yogyakarta
Gaji Karyawan  Rp 750000
```

Program untuk menghitung spp mahasiswa menggunakan struktur, diketahui :

a. D3

- spp tetap Rp 500.000
- spp var Rp 25.000/sks

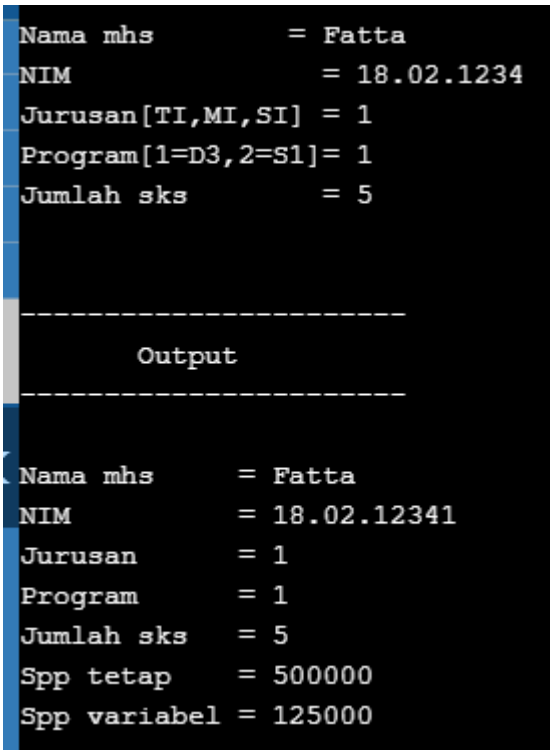
b. S1

- spp tetap Rp 750.000
- spp var Rp 50.000/sks

Contoh 5 Program Struct SPP

```
#include <iostream>
using namespace std;

struct mhs
{ char nama[20],nim[10],jurusan[2];
  int sks,program; };
struct mhs bayar;
main ()
{
  int bts,var,tetap;
  char prog[2];
    //input data
    cout<<"\nNama mhs         = ";cin>>bayar.nama;
    cout<<"NIM               = ";cin>>bayar.nim;
    cout<<"Jurusan[TI,MI,SI] = ";cin>>bayar.jurusan;
    input:
    cout<<"Program[1=D3,2=S1]= ";cin>>bayar.program;
    if (bayar.program > 2)
       {cout<<"Program tidak sesuai\n";
        goto input;}
    cout<<"Jumlah sks        = ";cin>>bayar.sks;
    if (bayar.program==1)
       {tetap=500000;
        var=bayar.sks*25000;}
    else if (bayar.program==2)
       {tetap=750000;
        var=bayar.sks*50000;}
    cout<<"";
  //output data
    cout<<"\n\n----------------------\n";
    cout<<"       Output  ";
    cout<<"\n----------------------\n";
    cout<<"\nNama mhs     = "<<bayar.nama;
    cout<<"\nNIM          = "<<bayar.nim;
    cout<<"\nJurusan      = "<<bayar.jurusan;
    cout<<"\nProgram      = "<<bayar.program;
    cout<<"\nJumlah sks   = "<<bayar.sks;
    cout<<"\nSpp tetap    = "<<tetap;
    cout<<"\nSpp variabel = "<<var;
    cout<<endl;
return 0;}
```

Output Contoh 5 Program Struct SPP



13.2.1. Struct of Array

Struct of array ini apabila hendak menggunakan 1 struct untuk beberapa kali, ada 2 cara :

Deklarasi manual :

```
typedef struct Mahasiswa {
     char NIM[8];
     char nama[50];
     float ipk;
};
int main()
   {
     Mahasiswa a,b,c;
       ……
       ……
       ……
   }
```

artinya struct mahasiswa digunakan untuk 3 variabel, yaitu a,b,c

Struct of array 1 :

```
typedef struct Mahasiswa {
     char NIM[8];
   char nama[50];
     float ipk;
}mhs;
   void main()
   {
     mhs biodata[3];
       ……
       ……
       ……
   }
```

artinya struct mahasiswa digunakan untuk mhs[0], mhs[1], dan mhs[2]

Struct of array 2 :

```
struct Mahasiswa {
     char NIM[8];
   char nama[50];
     float ipk;
};
   void main()
   {
     struct mhs biodata[3];
       ……
       ……
       ……
   }
```

artinya struct mahasiswa digunakan untuk mhs[0], mhs[1], dan mhs[2]

Contoh 6 Program Struct of Array

```
#include <iostream>
#include <stdio.h>
using namespace std;

struct mahasiswa
{ char nim[10],nama[20],jur[20],sem[10],th[10];}mhs;

struct kartuhasilstudi
{   char kode[10],mkul[20],nilai;
    int sks;
}khs[10];

int tsks,n;
int main()
{
    cout<<"\nINPUT BIODATA\n";
    cout<<"NIM          : ";gets(mhs.nim);
    cout<<"Nama         : ";gets(mhs.nama);
    cout<<"Jurusan      : ";gets(mhs.jur);
    cout<<"Semester     : ";cin>>mhs.sem;
    cout<<"Tahun        : ";cin>>mhs.th;
    cout<<"Jumlah Mkul  : ";cin>>n;

    for(int a=1;a<=n;a++)
    {
       cout<<"Kode         : " ;cin>>khs[a].kode;
       cout<<"Mkul         : " ;cin>>khs[a].mkul;
       cout<<"Nilai Hrf    : " ;cin>>khs[a].nilai;
       cout<<"SKS          : " ;cin>>khs[a].sks;
       tsks+=khs[a].sks;
    }
    cout<<"\nKARTU HASIL STUDI\n";
    cout<<"NIM          : "<<mhs.nim<<endl;
    cout<<"Nama         : "<<mhs.nama<<endl;
    cout<<"Jurusan      : "<<mhs.jur<<endl;
    cout<<"Semester     : "<<mhs.sem<<endl;
    cout<<"Tahun        : "<<mhs.th<<endl;
    cout<<"================================\n";
    cout<<"Kode"<<"\t"<<"Matakuliah"<<"\t"<<"SKS"<<"\t"<<"Nilai Huruf"<<endl;

    for(int a=1;a<=n;a++)
    {
       cout<<khs[a].kode<<"\t";
       cout<<khs[a].mkul<<"\t\t";
       cout<<khs[a].sks<<"\t";
       cout<<khs[a].nilai<<"\t";
       cout<<endl;
    }
    cout<<"\nTotal SKS     : "<<tsks;
    return 0;}
```

Output Contoh 6 Program Struct of Array

```
INPUT BIODATA
NIM          : 18.02.8986
Nama         : Yuli Astuti
Jurusan      : D3 MI
Semester     : 2
Tahun        : 2018-2019
Jumlah Mkul : 3
Kode         : 01
Mkul         : ASD
Nilai Hrf    : A
SKS          : 8
Kode         : 02
Mkul         : PTI
Nilai Hrf    : A
SKS          : 2
Kode         : 03
Mkul         : PT
Nilai Hrf    : B
SKS          : 4
```

```
KARTU HASIL STUDI
NIM          : 18.02.8986Yuli Astuti
Nama         : Yuli Astuti
Jurusan      : D3 MI
Semester     : 2
Tahun        : 2018-2019
================================
Kode    Matakuliah      SKS     Nilai Huruf
01      ASD             8       A
02      PTI             2       A
03      PT              4       B

Total SKS    : 14
```

13.3. Latihan Soal

1 Buatlah program untuk menghitung IPK mahasiswa mengunakan struct.

2 Buatlah program untuk menghitung tagihan biaya listrik rumah menggunakan struct.

3 Buatlah program untuk menghitung tagihan biaya telepon rumah menggunakan struct.

4 Buat program untuk menghitung jumlah nilai akhir mahasiswa menggunakan structure dengan ketentuan:

Nilai akhir = 10%*tugas + 20%*kuis + 30%*mid + 40%*uas

Nilai Huruf:

Nilai akhir >85 : A

85 >= nilai akhir > 70 : B

70 >= nilai akhir > 55 : C

55 >= nilai akhir > 40 : D

Nilai akhir <=40 : E

5 Buat sebuah program untuk menghitung gaji harian pegawai, bila diketahui ketentuannya sebagai berikut :

Gaji per jam = 500

Bila jumlah jam kerja hari itu > 7 jam, maka kelebihannya dihitung lembur yang besarnya 15 x gaji per jam.

Langkah pengerjaan :

1. Inputkan identitas pekerja, meliputi nama dan no.identitas kerja
2. Inputkan berapa jam ia bekerja setiap hari nya dari hari senin s/d jumat
3. Lalu jumlahkan dan tampilkan jam kerjanya
4. Output berupa total gaji

BAB XIV. STACK (TUMPUKAN)

14.1. Tujuan Pembelajaran

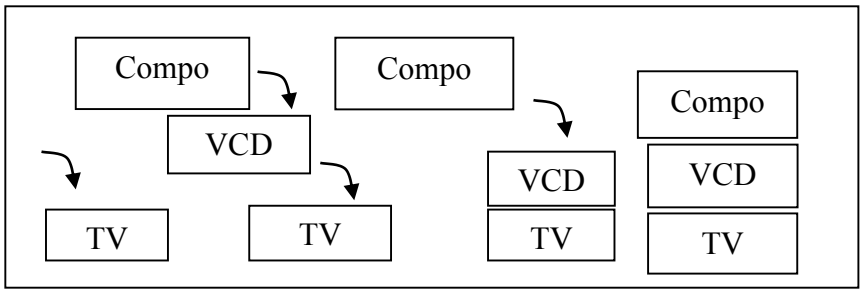
Setelah mempelajari materi ini, mahasiswa diharapkan mampu :

1. Memahami pengertian *stack* (tumpukan data)
2. Menerapkan algoritma *stack*

14.2. Dasar Teori

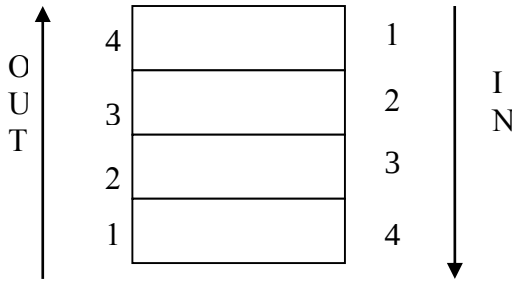
Stack adalah suatu susunan koleksi data dimana data dapat ditambahkan dan dihapus selalu dilakukan pada bagian akhir data, yang disebut dengan top of stack. Stack bersifat LIFO (*Last In First Out*) artinya "Benda yang **terakhir masuk** ke dalam stack akan menjadi yang **pertama keluar** dari *stack*.

Contoh gambaran *stack* sebagai berikut:



Gambar 14.1 Contoh Ilustrasi Stack

Operasi Stack



Gambar 14.2 Contoh Operasi Strack

Push

Digunakan untuk menambah item pada stack pada tumpukan paling atas.

Pop

Digunakan untuk mengambil item pada stack pada tumpukan paling atas.

Clear

Digunakan untuk mengosongkan stack.

IsEmpty

Fungsi yang digunakan untuk mengecek apakah stack sudah kosong.

IsFull

fungsi yang digunakan untuk mengecek apakah stack sudah penuh.

Stack with Array of Struct

- Definisikan **Stack** dengan menggunakan suatu struct.
- Definisikan konstanta **MAX_STACK** untuk menyimpan maksimum isi stack.
- Elemen struct Stack adalah **array data** dan **top** untuk menadakan posisi data teratas.
- Buatlah variabel **tumpuk** sebagai implementasi dari struct **Stack.**
- Deklarasikan operasi-operasi/function di atas dan buat implemetasinya.

```
#define MAX_STACK 10
```

Keterangan:

Mendeklarasikan konstanta `MAX_STACK` dengan nilai maksimal 10.

```
typedef struct STACK{
 int top;
 int data[10];
};
```

Keterangan:

Pendeklarasian struktur `stack` dengan dua member yaitu top dan data. `Typedef` digunakan untuk pendeklarasian tipe data baru, sehingga struktur `stack` bisa digunakan sebagai tipedata baru.

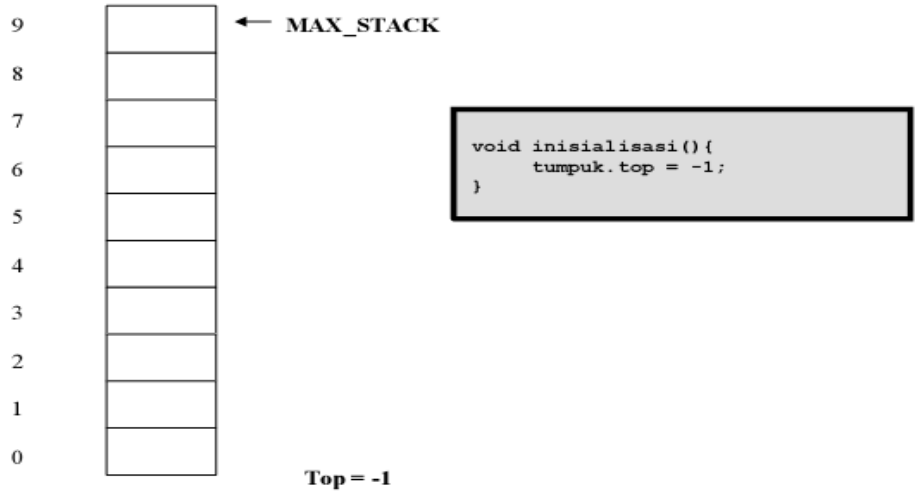
```
STACK tumpuk;
```

Keterangan:

Karena struktur `stack` dideklarasikan menggunakan `typedef` sehingga sekarang `stack` merupakan tipe data baru untuk mendeklarasikan variabel `tumpuk`.

Inisialisasi Stack

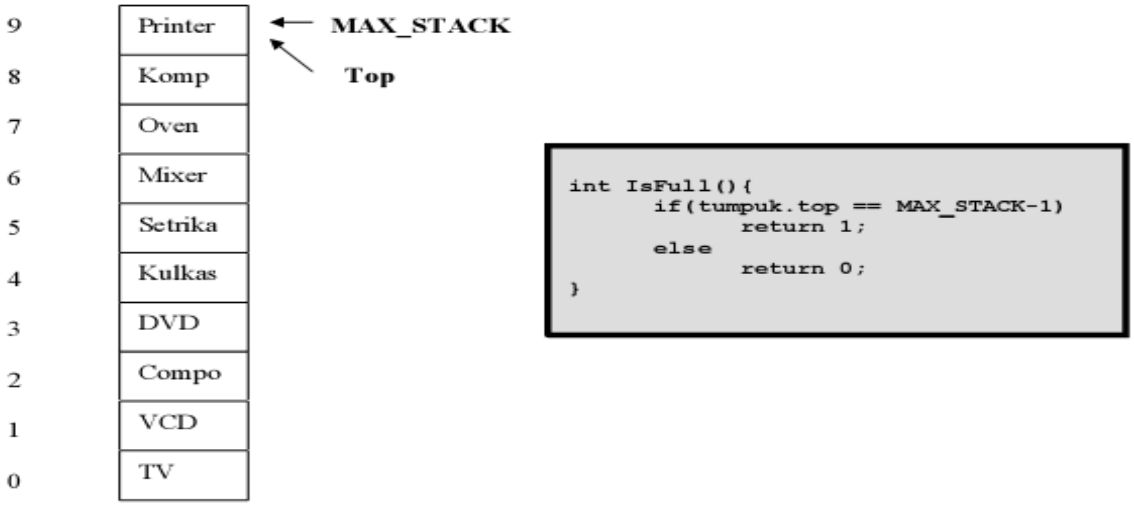
- Pada mulanya isi **top** dengan -1, karena array dalam bahasa C dimulai dari 0, yang berarti bahwa data stack adalah KOSONG!
- **Top** adalah suatu variabel penanda dalam Stack yang menunjukkan elemen teratas data Stack sekarang. **Top Of Stack** akan selalu bergerak hingga mencapai MAX of STACK yang menyebabkan stack PENUH!



Gambar 14.3 Ilustrasi Strack Pada Saat Inisialisasi

Fungsi IsFull

- Untuk memeriksa apakah stack sudah penuh?
- Dengan cara memeriksa **top of stack**, jika sudah sama dengan MAX_STACK-1 maka **full**, jika belum (masih **lebih kecil** dari MAX_STACK-1) maka belum full



Gambar 14.4 Ilustrasi Strack Kondisi Full

Fungsi IsEmpty

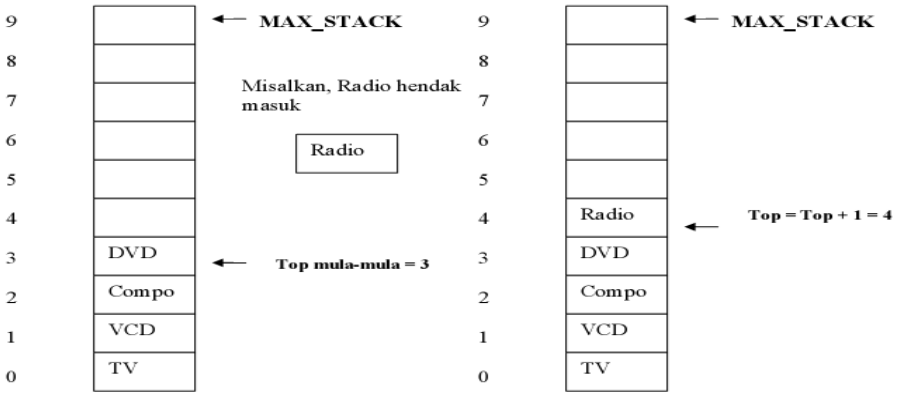
- Untuk memeriksa apakah data Stack masih kosong?
- Dengan cara memeriksa top of stack, jika masih -1 maka berarti data Stack masih kosong!

```
int IsEmpty(){
  if(tumpuk.top == -1)
      return 1;
  else
      return 0;
}
```

Gambar 14.5 Perintah Fungsi Isempty

Fungsi Push

- Untuk memasukkan elemen ke data Stack. Data yang diinputkan **selalu** menjadi elemen teratas Stack (yang ditunjuk oleh ToS).
- Jika **data belum penuh**,
- Tambah satu (increment) nilai **top of stack** lebih dahulu setiap kali ada penambahan ke dalam array data Stack.
- Isikan data baru ke stack berdasarkan indeks top of stack yang telah di-increment sebelumnya.
- Jika tidak, outputkan "Penuh".

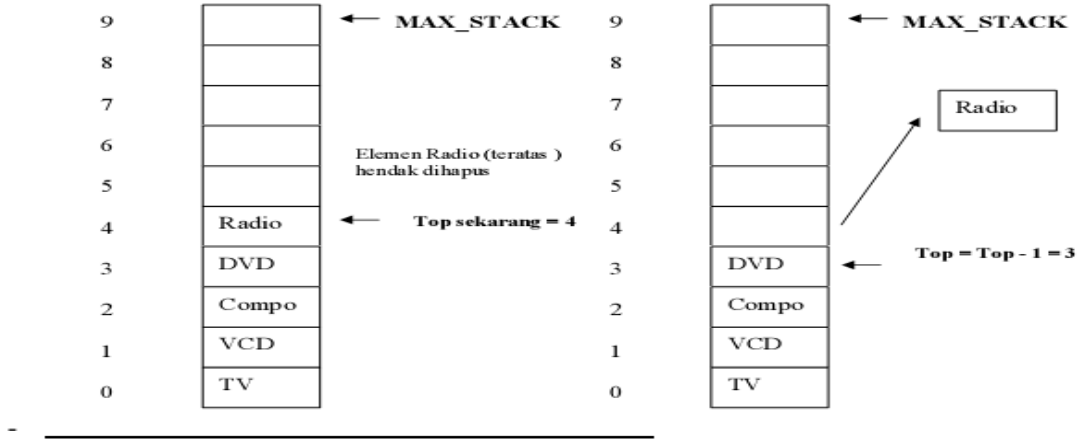



```
void Push(char d[10]){
  tumpuk.top++;
  strcpy(tumpuk.data[tumpuk.top],d);
}
```

Gambar 14.6 Ilustrasi Strack Input Data

Fungsi Pop

- Untuk mengambil data Stack yang terletak paling atas (data yang ditunjuk oleh TOS).
- **Tampilkan terlebih dahulu** nilai elemen teratas stack dengan mengakses indeksnya sesuai dengan top of stacknya, baru dilakukan di-decrement nilai top of stacknya sehingga jumlah elemen stack berkurang.



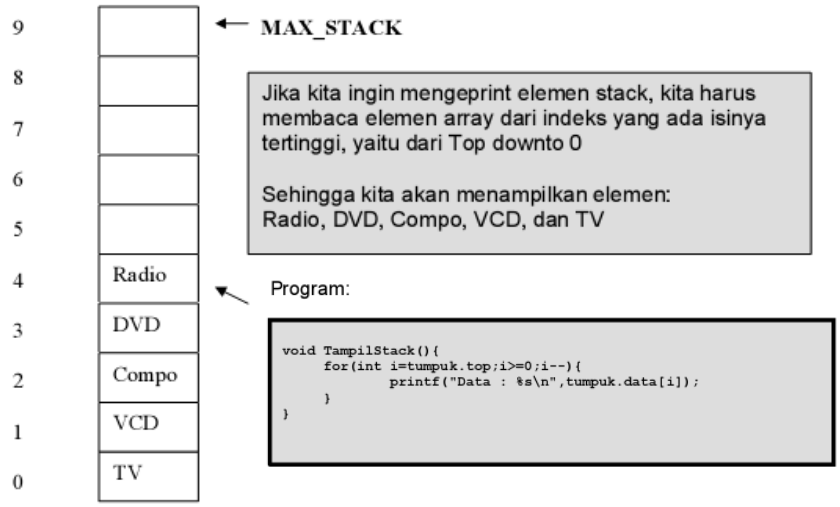
Programnya:

```
void Pop(){
   printf("Data yang terambil = %s\n",tumpuk.data[tumpuk.top]);
   tumpuk.top--;
}
```

Gambar 14.7 Ilustrasi Strack Pengambilan Data

Fungsi Print

- Untuk menampilkan semua elemen-elemen data Stack.
- Dengan cara me-loop semua nilai array secara **terbalik**, karena kita harus mengakses dari indeks array tertinggi terlebih dahulu baru ke indeks yang lebih kecil!



Gambar 14.8 Ilustrasi Strack Pencetakan Data

Fungsi peek

- Digunakan untuk melihat top of stack

```
int peek(){
    return tumpuk.data[tumpuk.top];
}
```

Contoh 1 Program Animasi Stack

```
#include <iostream>
#include <conio.h>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>

using namespace std;

int maks,top,pil,karakter,i;
int top2;
char elemen,aa,bb;
char stack[20],infiks[20], postfiks[20], stack2[10], hasil[20];
void delay();

int create(int x)
{
  top = 0;
  gotoxy(50,3);
  printf("                    ");
  for(i=0;i<100;i++)
    {
     gotoxy(59,4+i);
     printf("      ");
    }
   if(x<=20)
     {
       for(i=0;i<=x;i++)
        {
          if(i==x)
            { gotoxy(60,4+i);printf("â€”");}
          else
            { gotoxy(59,4+i);
              printf("|    |");
            }
        }
     }
}

int push(char aa)
{
  gotoxy(50,2);
  printf("                                    ");
  if(top == maks)
    {
      gotoxy(53,2);
      printf("â€”-OVERFLOWâ€”-");
    }
```

```
48      else
49        {
50          stack[top]=aa;
51          for(i=0;i<11;i++)
52            {
53              gotoxy(50+i,3);
54              cout<<" ";
55              gotoxy(51+i,3);
56              cout<<aa;
57              delay();
58            }
59          for(i=0;i<(maks-top);i++)
60            {
61              gotoxy(61,3+i);
62              cout<<" ";
63              gotoxy(61,4+i);
64              cout<<aa;
65              delay();
66            }
67         top = top + 1;
68        }
69  }
```

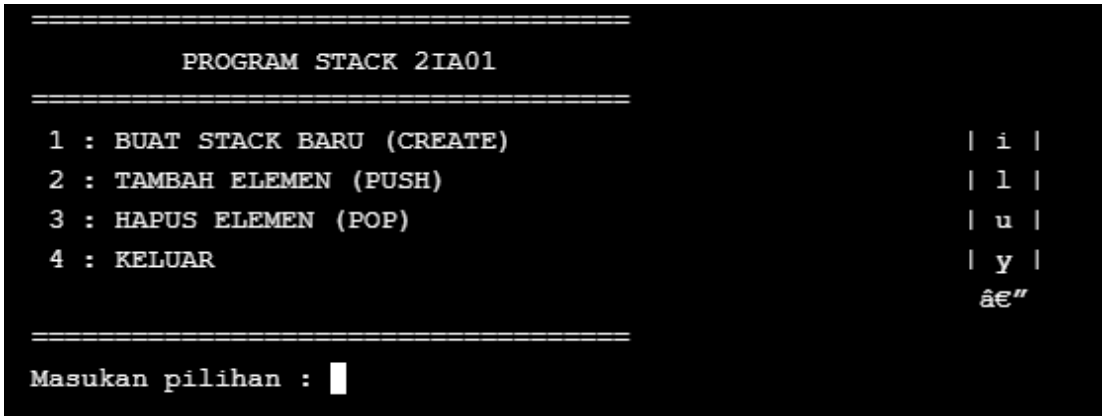
```
70  int pop(){
71    gotoxy(50,2);
72    printf("                                    ");
73    if(top == 0)
74     {
75       gotoxy(53,2);
76       printf("â€”-UNDERFLOWâ€”-");
77     }
78    else
79     {
80       bb = stack[top-1];
81       for(i=(maks-(top-1));i>0;i--)
82        {
83          gotoxy(61,3+i);
84          cout<<" ";
85          gotoxy(61,2+i);
86          cout<<bb;
87          delay();
88        }
89       for(i=10;i<21;i++)
90        {
91          gotoxy(50+i,3);
92          cout<<" ";
93          gotoxy(51+i,3);
94          cout<<bb;
95          delay();
```

```
96          }
97          gotoxy(71,3);
98          printf(" ");
99          top = top-1;
100       }
101   }
102
103   int main(){
104   maks = 0;
105   top = 0;
106   do {
107         gotoxy(3,1);
108         printf("====================================");
109         gotoxy(3,2);
110         printf("          PROGRAM STACK 2IA01        ");
111         gotoxy(3,3);
112         printf("====================================");
113         gotoxy(4,4);
114         printf("1 : BUAT STACK BARU (CREATE)");
115         gotoxy(4,5);
116         printf("2 : TAMBAH ELEMEN (PUSH)");
117         gotoxy(4,6);
118         printf("3 : HAPUS ELEMEN (POP)");
119         gotoxy(4,7);
120         printf("4 : KELUAR");
121         gotoxy(3,9);
```

```
122     printf("====================================");
123     for(i=10;i<20;i++)
124      {
125         gotoxy(3,i);
126         printf("                                                  ");
127      }
128     gotoxy(3,10);
129     printf("Masukan pilihan : ");
130     cin>>pil;
131     switch(pil){
132     case 1:{
133               gotoxy(3,12);
134               printf("Masukan kapasitas stack (maksimal 20) : ");
135               cin>>maks;
136               create(maks);
137               break;
138             }
139     case 2:{
140               if(maks==0){
141                 gotoxy(3,12);
142                 printf("Stack belum dibuat.Create stack terlebih dahulu");
143                 getch();
144               }
145              else
146                {
147                 gotoxy(3,12);
148                 printf("masukan satu karakter : ");
149                 cin>>elemen;
150                 push(elemen);
151               }
152              break;
153            }
154     case 3:{
155               if(maks==0){
156                 gotoxy(3,12);
157                 printf("Stack belum dibuat.Create stack terlebih dahulu");
158                 getch();
159               }
160              else
161                 {pop();}
162              break;
163            }
164    }
165  }
166  while(pil!=4);
167 }
```

```
169  void delay()
170  {
171    for(int y=1;y<100;y++)
172       for(int x=1;x<100;x++)
173          for(int p=1;p<30;p++)
174            cout<<"";
175  }
```

Output Contoh 1 Program Animasi Stack



14.3. Latihan soal

Kerjakan soal-soal di bawah ini!

1. Gabungkan penggalan-penggalan fungsi di atas sehingga menjadi sebuah program *Stack.*
2. Modifikasilah dari contoh program animasi *stack* tersebut menjadi input kata.

BAB XV. QUEUE (ANTRIAN)

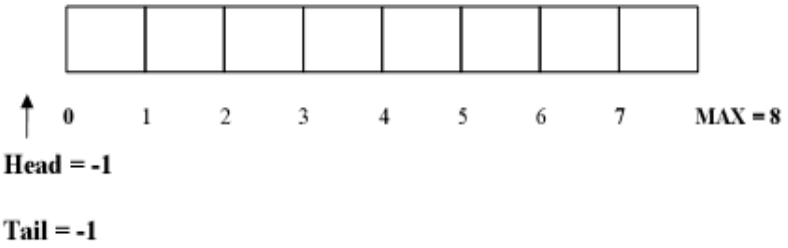
15.1. Tujuan Pembelajaran

Setelah mempelajari materi ini, mahasiswa diharapkan mampu :

1. Dapat menjelaskan pengertian *queue*.
2. Dapat menulis perintah dan menjelaskan algoritma *queue*

15.2. Dasar Teori

Queue adalah antrian data yang bersifat FIFO (*First In First Out*) dimana elemen yang pertama kali masuk ke antrian akan keluar pertama kalinya. *Queue* (antrian) dibuat menggunakan array dan dua buah variabel bertipe integer yang menunjukkan posisi awal (*head*) dan akhir antrian *(tail*). Jika belum ada data yang mengantri atau queue masih kosong, maka posisi head dan tail berada di index -1, seperti pada gambar berikut:



Gambar 15.1 Ilustrasi Queue Pada Kondisi Kosong

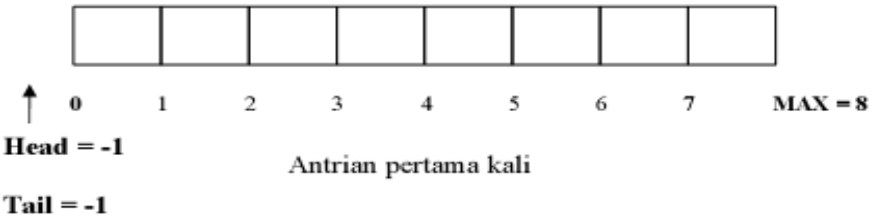
Queue memiliki beberapa operasi-opeasi sebagai berikut:

1. Create()

Fungsi create() digunakan untuk menciptakan dan menginisialisasi queue dengan cara membuat head dan tail pada posisi -1. Perintah deklarasi queue sebagai berikut:

```
#define MAX 8
typedef   struct{
                    int data[MAX];
                    int head;
                    int tail;
                } Queue;

Queue antrian;
```

Gambar 15.2 Perintah Deklarasi Queue

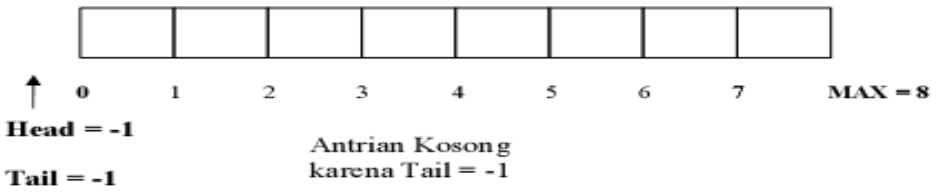



```
void Create(){
     antrian.head=antrian.tail=-1;
}
```

Gambar 15.3 Perintah Untuk Menentukan Posisi Antrian Pertama

2. IsEmpty

Fungsi isempty digunakan untuk memeriksa apakah antrian sudah penuh atau belum, dengan cara memeriksa nilai Tail, jika Tail = -1 maka empty. Head tidak akan diperiksa, karena Head adalah tanda untuk kepala antrian (elemen pertama dalam antrian) yang tidak akan berubah-ubah dan pergerakan pada antrian terjadi dengan penambahan elemen antrian kebelakang, yaitu menggunakan nilai Tail. Perintahnya sebagai berikut:

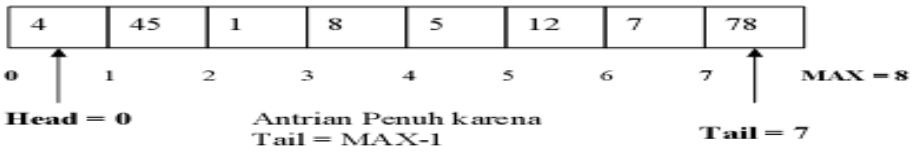



```
int IsEmpty(){
     if(antrian.tail==-1)
          return 1;
     else
          return 0;
}
```

Gambar 15.4 Perintah Untuk Memeriksa Posisi Kosong

3. IsFull

Fungsi isfull digunakan untuk mengecek apakah antrian sudah penuh atau belum, dengan cara mengecek nilai Tail, jika Tail >= MAX-1 (karena MAX-1 adalah batas elemen array pada C) berarti sudah penuh. Perintahnya sebagai berikut:

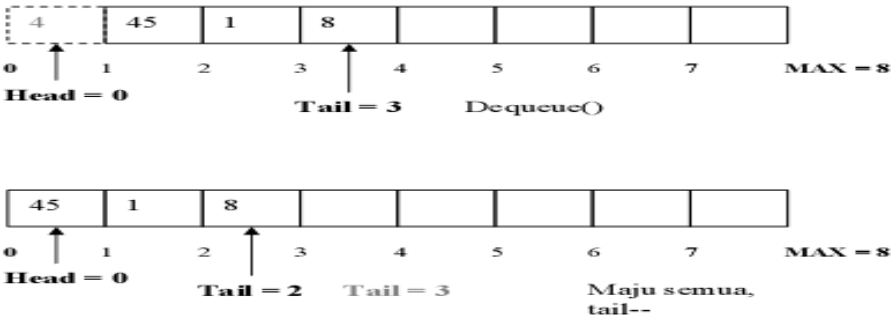



```
int IsFull(){
      if(antrian.tail==MAX-1) return 1;
      else return 0;
}
```

Gambar 15.5 Perintah Untuk Memeriksa Antrian Penuh atau Belum Penuh

4. Dequeue

Perintah dequeue digunakan untuk menghapus elemen terdepan/pertama (head) dari antrian dengan cara menggeser semua elemen antrian kedepan dan mengurangi Tail dengan satu penggeseran dilakukan dengan menggunakan looping, contoh ilustrasi dequeue dan perintah dequeue sebagai berikut:

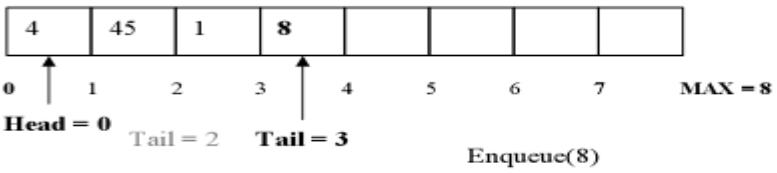



```
int Dequeue(){
      int i;
      int e = antrian.data[antrian.head];
      for(i=antrian.head;i<=antrian.tail-1;i++){
            antrian.data[i] = antrian.data[i+1];
      }
      antrian.tail--;
      return e;
}
```

Gambar 15.6 Ilustrasi dan Perintah Dequeue

5. Enqueue

Fungsi enqueue digunakan untuk menambahkan elemen ke dalam antrian, penambahan elemen selalu ditambahkan di elemen paling **belakang.** Penambahan elemen selalu menggerakan variabel Tail dengan cara increment counter Tail terlebih dahulu. Contoh ilustrasi enqueue dan perintah enqueue sebagai berikut:




```
void Enqueue(int data){
     if(IsEmpty()==1){
          antrian.head=antrian.tail=0;
          antrian.data[antrian.tail]=data;
          printf("%d masuk!",antrian.data[antrian.tail]);
     } else
     if(IsFull()==0){
          antrian.tail++;
          antrian.data[antrian.tail]=data;
          printf("%d masuk!",antrian.data[antrian.tail]);
     }
}
```

Gambar 15.7 Ilustrasi dan Perintah enqueue

6. Clear

Fungsi clear digunakan untuk menghapus elemen-elemen antrian dengan cara membuat Tail dan Head = -1. Penghapusan elemen-elemen Antrian sebenarnya tidak menghapus arraynya, namun hanya mengeset indeks pengaksesan-nya ke nilai -1 sehingga elemen-elemen antrian tidak terbaca lagi. Perintah Clear sebagai berikut:

```
void Clear(){
     antrian.head=antrian.tail=-1;
     printf("data clear");
}
```

Gambar 15.8 Perintah Clear

7. Tampil

Perintah tampil digunakan untuk menampilkan nilai-nilai elemen antrian menggunakan looping dari head sampai dengan tail, dengan perintah sebagai berikut:

```
void Tampil(){
     if(IsEmpty()==0){
          for(int i=antrian.head;i<=antrian.tail;i++){
               printf("%d ",antrian.data[i]);
          }
     }else printf("data kosong!\n");
}
```

Gambar 15.9 Perintah Tampil

15.3. Latihan soal

Gabungkan penggalan-penggalan fungsi diatas sehingga menjadi sebuah program *Queue*.

BAB XVI. CONTOH SOAL DAN PROGRAM

16.1. Program Menu Makanan Kantin Menggunakan Percabangan Bersarang

```
#include <iostream>
using namespace std;
int main(){
int angka,mkn,mnm;char jwb;

menu:
cout<<"MENU KANTIN AMIKOM"<<endl<<endl;
cout<<"1. Makanan\n";
cout<<"2. Minuman\n";
cout<<"\nPilih : ";cin>>angka;
switch (angka){
 case 1:
 cout<<"\n\n\t\tMAKANAN : "
 <<endl<<endl;
 cout<<"\t[1] Mie ayam"<<endl;
 cout<<"\t[2] Bakso"<<endl;
 cout<<"\t[3] Rames"<<endl;
 cout<<"\n\tPilih : ";cin>>mkn;
 if(mkn==1)
 {cout<<"\n\n\t\tMakan mie ayam seharga 5000";
 cout<<"\n\n\t\tIngin pesan lagi[y/t]? ";cin>>jwb;
 if (jwb=='y'||jwb=='Y')
 {system("cls");
 goto menu;
 }
 else
 cout<<"\n\n\t\tTerima kasih...";
 }
```

```
else if (mkn==2)
{cout<<"\n\n\t\tMakan bakso seharga 6000";
cout<<"\n\n\t\tIngin pesan lagi[y/t]? ";
cin>>jwb;
if (jwb=='y'||jwb=='Y')
{system("cls");
goto menu;
}
else
cout<<"\n\n\t\tTerima kasih";
}
else if (mkn==3)
{cout<<"\n\n\t\tMakan nasi rames seharga 10000";
cout<<"\n\n\t\tIngin pesan lagi[y/t]? ";cin>>jwb;
if (jwb=='y'||jwb=='Y')
{system("cls");
goto menu;
}
else
cout<<"\n\n\t\tTerima kasih";
}
else
{cout<<"\n\n\t\tMenu tidak tersedia";
goto menu;
}
break;
case 2:
     cout<<"\n\n\t\tMINUMAN : " <<endl<<endl;
cout<<"\t[1] Es jeruk"<<endl;
cout<<"\t[2] Jus jambu"<<endl;
cout<<"\t[3] Air putih"<<endl<<endl;
```

```
 cout<<"\n\tPilih : ";cin>>mnm;
 if(mnm==1)
 {cout<<"\n\n\t\tMinum es jeruk seharga 2000";
 cout<<"\n\n\t\tIngin pesan lagi[y/t]? ";cin>>jwb;
 if(jwb=='y'||jwb=='Y')
 {system("cls");
 goto menu;
 }
 else
 cout<<"\n\n\t\tTerima kasih";
 }
 else if(mnm==2)
 {cout<<"\n\n\t\tMinum jus jambu seharga 5000";
 cout<<"\n\n\t\tIngin pesan lagi[y/t]? ";cin>>jwb;
 if(jwb=='y'||jwb=='Y')
 {system("cls");
 goto menu;
 }
 else
 cout<<"\n\n\t\tTerima kasih";
 }
 else if(mnm==3)
 {cout<<"\n\n\t\tMinum air putih gratis";
 cout<<"\n\n\t\tIngin pesan lagi[y/t]? ";
cin>>jwb;
     if(jwb=='y'||jwb=='Y')
 {system("cls");
 goto menu;}
 else
 cout<<"\n\n\t\tTerima kasih";}
 break;
```

```
 default:
 cout<<"\n\nTidak ada menu pilihan selain 1 dan 2..\n";
 goto menu;}
return 0;}
```

16.2. Program Tahun Kabisat Menggunakan Percabangan

```
#include <iostream>
using namespace std;
main(){
bool jwb;
int thn,bil,pil;

menu:
cout<<"Pilihlah menu di bawah ini :\n\n";
cout<<"[1] Tahun Kabisat\n";
cout<<"[2] Bilangan genap ganjil\n";
cout<<"[3] Bilangan Positif negatif\n";
cout<<"\nMasukkan pilihan anda : ";
cin>>pil;


switch (pil) {
case 1:
     cout<<"\nMenentukan tahun kabisat\n";
     cout<<"Input tahun = ";cin>>thn;
 if(thn % 4==0)
 {cout<<"Tahun "<<thn<<" adalah tahun kabisat\n\n";
 cout<<"Apa anda ingin memilih lg?(1=ya,0=tdk)";
 cin>>jwb;
 if(jwb==1)
 goto menu;
 else
 cout<<"anda tdk memilih";}
 else
 {cout<<"Tahun "<<thn<<" bukan tahun kabisat\n\n";
 cout<<"Apa anda ingin memilih lg?(1=ya,0=tdk)";
 cin>>jwb;
 if(jwb==1)
 goto menu;
 else
 cout<<"anda tdk memilih";}
 break;
case 2:
     cout<<"\nMenentukan Bilangan genap ganjil\n";
```

```
    cout<<"Input bilangal = ";cin>>bil;
 if(bil % 2==0)
 {cout<<"Bilangan "<<bil<<" adalah genap\n\n";
 cout<<"Apa anda ingin memilih lg?(1=ya,0=tdk)";
 cin>>jwb;
 if(jwb==1)
 goto menu;
 else
 cout<<"anda tdk memilih";}
 else if(bil % 2 !=0)
 {cout<<"Bilangan "<<bil<<" adalah ganjil\n\n";
 cout<<"Apa anda ingin memilih lg?(1=ya,0=tdk)";
 cin>>jwb;
 if(jwb==1)
 goto menu;
 else
 cout<<"anda tdk memilih";}
 else
 {cout<<"Bilangan "<<bil<<" adalah nol\n\n";
 cout<<"Apa anda ingin memilih lg?(1=ya,0=tdk)";
 cin>>jwb;
 if(jwb==1)
 goto menu;
 else
 cout<<"anda tdk memilih";}
 break;
 case 3:
          cout<<"\nMenentukan   Bilangan   negatif   dan
positif\n";
          cout<<"Input bilangan= ";
 cin>>bil;
 if(bil < 0)
 {cout<<"Bilangan "<<bil<<" adalah negatif\n\n";
 cout<<"Apa anda ingin memilih lg?(1=ya,0=tdk)";
 cin>>jwb;
 if(jwb==1)
 goto menu;
 else
 cout<<"anda tdk memilih";}
 else if(bil > 0)
 {cout<<"Bilangan "<<bil<<" adalah positif\n\n";
 cout<<"Apa anda ingin memilih lg?(1=ya,0=tdk)";
 cin>>jwb;
 if(jwb==1)
 goto menu;
 else
 cout<<"anda tdk memilih";}
```

```
 else
 {cout<<"Bilangan "<<bil<<" adalah nol\n\n";
 cout<<"Apa anda ingin memilih lg?(1=ya,0=tdk)";
 cin>>jwb;
 if(jwb==1)
 goto menu;
 else
 cout<<"anda tdk memilih";}
 break;
 default:
 {cout<<"\nMaaf,menu belum terdaftar\n";
 cout<<"Apa anda ingin memilih lg?(1=ya,0=tdk)";
 cin>>jwb;
 if(jwb==1)
 goto menu;
 else
 cout<<"anda tdk memilih";}
 }
return 0;}
```

16.3. Program Sederhana Tarif Parkir Menggunakan Percabangan

```
#include<iostream>
#define standar 1
using namespace std;

int main(){
int msk,klr,lama,biaya,sisa;

cout<<"Jam masuk parkir : ";cin>>msk;
cout<<"Jam keluar parkir : ";cin>>klr;

lama=klr-msk;
sisa=lama-standar;
if (lama==1 && sisa<=0)
 biaya=2000;
else
 biaya=2000+(sisa*500);

cout<<"Lama parkir : "<<lama<<endl;
cout<<"Biaya standar 1 jam : "
 <<standar*2000<<endl;
cout<<"Biaya setelah 1 jam : "<<sisa*500
 << " selama : "<<sisa<<" jam x 500 "
 <<endl;
cout<<"Tarif parkir : "<<biaya;
```

```
return 0;}
```

16.4. Program Piramida Menggunakan Perulangan

```
#include <iostream>
using namespace std;

int main() {
int batas;

cout<<"input batas : ";cin>>batas;
 for(int i=batas;i>=1;i--){
 for(int a=i;a>=1;a--)
 {cout<<a*i<<" ";}
 cout<<endl;}
 return 0;}
```

16.5. Program Menghitung IPK Sederhana Menggunakan Percabangan

```
#include<iostream>
#include<stdio.h>
#include<iomanip.h>
using namespace std;

int main(){
 int i=1,n,nilai=0;
 float ipk,sks,tsks;
 char mkul[20],ket;
 cout<<"Input banyak mata kuliah = ";cin>>n;

 while(i<=n)
 {
  cout<<"\n\nData ke = "<<i;
  cout<<"\n-----------------\n";
  cout<<"Mata kuliah = " ;gets(mkul);
  cout<<"Jumlah SKS = ";cin>>sks;
  cout<<"Nilai huruf = ";cin>>ket;
  if(ket=='A'|| ket=='a')
   nilai+=(4*sks);
  else if(ket=='B' || ket=='b')
   nilai+=(3*sks);
  else if(ket=='C' || ket=='c')
   nilai+=(2*sks);
  else if(ket=='D' || ket=='d')
   nilai+=(1*sks);
```

```
  else if(ket=='E' || ket=='e')
   nilai+=(0*sks);
  else
   cout<<"Range nilai hanya dari A-E";
  tsks+=sks;
  ipk=nilai/tsks;
  i++;
 }

  cout<<"\nNilai total = "<<nilai;
  cout<<"\nTotal SKS = "<<tsks;
  cout<<"\nIPK = "
 <<setiosflags(ios::fixed)
 <<setprecision(2)
 <<ipk;
return 0;}
```

16.6. Program Penggajian Sederhana Menggunakan Percabangan

```
#include <iostream>
#include <stdio.h>
using namespace std;

int main(){
char nama[20],nip[5],gol,jkel;
int st,tgol,tistri,pend,ppn,gapok,gator;
int gaber=0,jkry,tgator,tppn,tgaji=0;

cout<<"PENGGAJIAN KARYAWAN RENTAL CD"<<endl<<endl;
cout<<"Jumlah karyawan : ";
cin>>jkry;
for(int i=1;i<=jkry;i++){
cout<<"NIP : ";
cin>>nip;
cout<<"Nama : ";
gets(nama);
cout<<"Golongan[A/B/C] : ";
cin>>gol;
if (gol=='A'||gol=='a')
 tgol=100000;
else if (gol=='B'||gol=='b')
 tgol=150000;
else if (gol=='C'||gol=='c')
 tgol=200000;
else
```

```
 tgol=0;
cout<<"Tunjangan Golongan : "
 <<tgol<<endl;
cout<<"Jenis kelamin[P/W] : ";
cin>>jkel;
cout<<"Status[1.M 2.S] : ";
cin>>st;
if(st==1 && jkel=='P')
 tistri=200000;
else
 tistri=0;
cout<<"Tunjangan Istri : "
 <<tistri<<endl;
cout<<"Pendidikan \n";
cout<<"[1.D3 2.S1 3.S2 4.S3] : ";
cin>>pend;
if(pend==1)
 gapok=500000;
else if(pend==2)
 gapok=750000;
else if(pend==3)
 gapok=1000000;
else if(pend==4)
 gapok=1500000;
else
 gapok= 0;
cout<<"Gaji Pokoknya : "
 <<gapok<<endl;
gator=tgol+tistri+gapok;
ppn=gator*0.05;
gaber=gator-ppn;

cout<<"Gaji Kotor : "
 <<gator<<endl;
cout<<"Potong pajak : "
 <<ppn<<endl;
cout<<"Gaji Bersih : "
 <<gaber<<endl<<endl<<endl;
tgaji+=gaber;}

cout<<"GAJI UNTUK SEMUA KARYAWAN "
 <<endl<<endl;
cout<<"Gaji Bersih semua karywan : "
 <<tgaji<<endl;
return 0;}
```

16.7. Program Database Mahasiswa Menggunakan Array

```
#include <stdio.h>
#include <iostream>
#include <iomanip.h>
using name space std;

judul(){
cout<<"\t PROGRAM DATABASE MAHASISWA"<<endl;
return 0;}

struct mahasiswa{
char nm[20][20];
char nim[20][20];
char kls[20][20];
char thang[20][20];
}mhs;

int main(){
int x=1000;
int data=0;
int i,plh,sisa,jml;
char a;

do{
clrscr();
judul();
cout<<"================================\n";
cout<<"\t\tMENU \n";
cout<<"================================\n";
cout<<"\t1.Masukan data\n";
cout<<"\t2.Lihat data\n";
cout<<"\t3.Hapus data terakhir \n";
cout<<"\t4.Hapus semua data\n";
cout<<"\t5.Cek kapasitas data\n";
cout<<"\t6.Keluar\n";
cout<<"================================\n";
cout<<"\tMenu Pilihan : ";
cin>>plh;
if (plh<=6){
switch(plh){
case 1:
 if(data==x){
      clrscr();
      judul();
      cout<<"Maaf,data penuh";}
     else {
```

```
 cout<<"Masukkan jumlah data yang
      ingin diinputkan : ";
 cin>>jml;
 if (jml>x){
 clrscr();
          judul();
 cout<<"Data yang ingin diinputkan
      melebihi kapasitas ";
 getch();}
 else{
     for (i=0;i<jml;i++){
     cout<<"\nNama : ";
          gets(mhs.nm[data+1]);
          cout<<"NIM : ";
          cin>>mhs.nim[data+1];
          cout<<"Kelas : "; cin>>mhs.kls[data+1];
     cout<<"Angkatan : ";
          cin>>mhs.thang[data+1];
     data++; }
 clrscr();
     judul();
 cout<<"Data Telah Diinputkan ";
     getch();
     } }
     break;
case 2:
     clrscr();
     cout<<"_____________________________________
_____________________________"<<endl;
     cout<<setiosflags(ios::left)<<setw(8)
 <<"NO";
     cout<<setiosflags(ios::left)<<setw(20)
      <<"NAMA";
     cout<<setiosflags(ios::left)<<setw(20)
      <<"NIM";
     cout<<setiosflags(ios::left)<<setw(20)
         <<"KELAS";
     cout<<setiosflags(ios::left)<<setw(20)
         <<"ANGKATAN";
     cout<<"\n_____________________________________________________
_____________________________"
<<endl<<endl;
     if (data==0)
      cout<<"Data Kosong";
     else
      for(i=1; i<=data; i++){
 cout<<setiosflags(ios::left)
```

```
 <<setw(8)<<i;
      cout<<setiosflags(ios::left)
 <<setw(20)<<mhs.nm[i];
      cout<<setiosflags(ios::left)
 <<setw(20)<<mhs.nim[i];
      cout<<setiosflags(ios::left)
 <<setw(20)<<mhs.kls[i];
 cout<<setiosflags(ios::left)
 <<setw(20)<<mhs.thang[i]<<endl;}
          getch();
          break;
case 3:
     if(data<1){
      clrscr();
      judul();
      cout<<"Data Kosong";}
     else{
      cout<<"\nHapus data :"<<data<<"\n\n";
      cout<<"Nama : "
 <<mhs.nm[data]<<endl;
      cout<<"NIM : "
 <<mhs.nim[data]<<endl;
      cout<<"Kelas : "
 <<mhs.kls[data]<<endl;
 cout<<"Angkatan : "
 <<mhs.thang[data]<<endl<<endl;
 cout<<"Apakah anda yakin ingin
 menghapus data ini [Y/N]? ";
 cin>>a;
 if (a=='Y' || a=='y'){
          data--;
          clrscr();
 judul();
          cout<<"Data Telah Dihapus";}
 else{
 clrscr();
     judul();
 cout<<"Data tidak terhapus";}
     getch();
     break;
case 4:
 cout<<"Apakah anda yakin ingin
          menghapus semua data [Y/N]? ";
 cin>>a;
 if (a=='Y' || a=='y'){
 clrscr();
 judul();
```

```
      data=0;
      cout<<"Semua Data Telah Terhapus";}
 else{
 clrscr();
 judul();
 cout<<"Data tidak terhapus";}
 getch();
      break;
case 5:
     if(data==x){
      clrscr();
      judul();
      cout<<"Kapasitas Data Penuh";}
     else if(data>=1){
      sisa=x-data;
      clrscr();
      judul();
      cout<<"Sisa Kapsitas Data "<<sisa;}
     else if(data==0){
      clrscr();
      judul();
      cout<<"Data Kosong";}}
      getch();
      break;

case 6:
     break;}}
else{
 cout<<"Inputan salah, Silahkan masukan
 pilihan 1-6 "<<endl;
getch();}}
while(plh!=6);}
```

16.8. Program Kalkulator Menggunakan Array

```
#include <iostream>
#include <stdio>
#include <stdlib>
using namespace std;

int answer[100];
int a=0;
int j;

void octal(){
long int num;
```

```
cout<<"\n\t masukkan nilai desimalnya :";
cin>>num;

while (num>0){
answer[a]=num%8;
num=num/8;
a++;}

cout<<"\n\t hasilnya dalam octal adalah : ";

j=a-1;
for (a=j;j>=0;j--)
 {cout<<answer[j];}
}

void binary(){
long int num;

cout<<"\n\t masukkan nilai desimalnya :";
cin>>num;

while (num>0){
 answer[a]=num%2;
 num=num/2;
 a++;}

cout<<"\n\thasilnya dalam binary adalah : ";

j=a-1;
for (a=j;j>=0;j--)
 {cout<<answer[j];}
}

void hexadesimal(){
long int num;

cout<<"\n\t masukkan nilai desimalnya :";
cin>>num;

while (num>0){
 answer[a]=num%16;
 num=num/16;
 a++;}

cout<<"\n\thasilnya dalam hexadesimal:";
j=a-1;
for (a=j;j>=0;j--)
```

```
 {if(answer[j]<10){
 cout<<answer[j];}
 else{
 switch(answer[j]){
          case 10:
               cout<<"A";
               break;
          case 11:
                cout<<"B";
                break;
           case 12:
                cout<<"C";
                break;
           case 13:
                cout<<"D";
                break;
           case 14:
                cout<<"E";
                break;
           case 15:
                cout<<"F";
                break;
          }}}}

void delay(int a)
 { for(int x=0;x<a*100;x++)
 {for(int y=0;y<a*100;y++)
 { }
 }
}

void main ()
{for(int i=0;i<=100;i++)
 {delay(10);
 gotoxy(30,12);cout<<"loading..."<<i<<"%";}

awal:
struct awal {
int pil,pil1;
char d;
}a;

clrscr();
cout<<"\n\t
=========================================================
======\n";
cout<<"\n\t PROGRAM KALKULATOR ";
```

```
cout<<"\n\t
============================================================
======\n";
cout<<"\n\n\n\t 1.calculator programer";
cout<<"\n\t (ini untuk menghitung konversi dari bilangan
desimal )";
cout<<"\n\n \t 2.calculator standar ";
cout<<"\n\t  (ini  untuk  menghitung  operasi  aritmatika
biasa)";
cout<<"\n\t pilih [1/2] :";cin>>a.pil1;

if (a.pil1==1)
 goto layar1;
else if (a.pil1==2)
 goto layar2;

layar1 :
clrscr();
cout<<"\n\n\n";
cout<<"\t
============================================================
===========\n";
cout<<"\t program untuk mengkonversi decimal ke binary
,hexadecimal dan octal\n";
cout<<"\t
============================================================
===========\n";
cout<<"\t 1. decimal to binary\n";
cout<<"\t 2. decimal to octall\n";
cout<<"\t 3. decimal to hexadecimal\n";
cout<<"\n\t **pilih salah satu program yang akan di hitung
[1-3] :";cin>>a.pil;
switch (a.pil){
 case 1:
 binary ();
 break;
 case 2:
 octal ();
 break;
 case 3:
      hexadesimal();
      break;
}

cout<<"\n\t apakah akan lanjut [y/t]";
cin>>a.d;
```

```
if(a.d=='y'||a.d=='Y')
 goto awal;
else
 exit(0);
getch();

layar2 :
{
 clrscr();
 struct layar{
 int pil,bil1,bil2;
 float jawab;
 char d;
}l;

cout<<"\n\n\n";
cout<<"\t
============================================================
===========\n";
cout<<"\t program untuk aritmatika biasa\n";
cout<<"\t
============================================================
===========\n";
cout<<"\t 1. tambah\n";
cout<<"\t 2. kurang\n";
cout<<"\t 3. kali\n";
cout<<"\t 4. bagi\n";
cout<<"\n\t **pilih salah satu program yang akan di hitung
[1-4] :";
cin>>l.pil;

if(l.pil==1) {
 cout<<"\n\tmasukkan bilangan pertama :";
 cin>>l.bil1;
 cout<<"\tmasukkan bilangan kedua :";
 cin>>l.bil2;
 l.jawab=l.bil1+l.bil2;
 cout<<"\tjawabannya adalah"<<l.jawab;}
else if (l.pil==2){
 cout<<"\n\tmasukkan bilangan pertama :";
 cin>>l.bil1;
 cout<<"\tmasukkan bilangan kedua :";
 cin>>l.bil2;
 l.jawab=l.bil1-l.bil2;
 cout<<"\tjawabannya adalah"<<l.jawab;}
else if (l.pil==3){
 cout<<"\n\tmasukkan bilangan pertama :";
```

```
 cin>>l.bil1;
 cout<<"\tmasukkan bilangan kedua :";
 cin>>l.bil2;
 l.jawab=l.bil1*l.bil2;
 cout<<"\t jawabannya adalah"<<l.jawab;}
else if (l.pil==4){
 cout<<"\n\tmasukkan bilangan pertama :";
 cin>>l.bil1;
 cout<<"\tmasukkan bilangan kedua :";
 cin>>l.bil2;
 l.jawab=l.bil1/l.bil2;
 cout<<"\t jawabannya adalah"<<l.jawab;}
else
 {cout<<"\t anda salah input angaka";}

cout<<"\n\t apakah akan lanjut [y/t]";
cin>>l.d;
if(l.d=='y')
 goto awal;
else
 {cout<<"\n\n\tterima ";}
}

cout<<"\n\t apakah akan lanjut [y/t]";
cin>>a.d;
if(a.d=='y')
 goto awal;
else
 exit(0);

getch();
exit(0);}
```

16.9. Program Peminjaman Pakaian Adat Menggunakan Struktur

```
#include <conio.h>
#include <stdio.h>
#include <iostream.h>

garis()
{cout<<"************************************
****************************************"<<endl;
}

main(){
char nama[30],pilihan,lagi,lama_sewa;
```

```
int i,n,total_bayar=0,bayar,kembali;

struct{
 int jumlah_sewa,harga,jumlah_harga;
 char kode[10],nama_baju[20],ukuran;
}baju[100];

atas:

clrscr();
cout<<"\t\t\tPENYEWAAN PAKAIAN ADAT NASIONAL"<<endl;
cout<<"\t\t\t\tCHONIO BOUTIQUE "<<endl;

garis();

cout<<"Selamat Datang Di Chonio Boutique"
 <<endl;
cout<<endl;
cout<<"Pilihan Menu : "<<endl;
cout<<"1. Input Data "<<endl;
cout<<"2. Log Out"<<endl;
cout<<"========================================"<<endl;
cout<<"Inputkan Pilihan Anda : ";
cin>>pilihan;

if (pilihan=='1')
 goto input;
else if (pilihan=='2')
 goto keluar;
else
 goto atas;

input:

clrscr();
cout<<"\t\t\tPENYEWAAN PAKAIAN ADAT NASIONAL"<<endl;
cout<<"\t\t\t\tCHONIO BOUTIQUE "<<endl;
cout<<endl;

garis();

cout<<endl;
cout<<"Masukkan Nama Penyewa : ";gets(nama);
cout<<"Input Lama Sewa : ";
cin>>lama_sewa;
cout<<"Input Jumlah Data : ";cin>>n;
cout<<endl;
```

```
for (i=1;i<=n;i++)
{cout<<"==================================="
 <<endl;
 cout<<"Pilihan Menu : "<<endl;
 cout<<"1. Jawa Barat [JB] "<<endl;
 cout<<"2. Jawa Tengah [JT]"<<endl;
 cout<<"3. Sumatra Barat [SB]"<<endl;
 cout<<"==================================="
 <<endl;
 cout<<"DATA KE - "<<i<<endl;
 cout<<"INPUT KODE PAKET BAJU [JB/JT/SB]: ";
 cin>>baju[i].kode;
 cout<<"INPUT KODE UKURAN BAJU [S/M/L]: ";
 cin>>baju[i].ukuran;
 if((strcmp(baju[i].kode,"JB")==0)||
 (strcmp(baju[i].kode,"jb")==0))
 {
 strcpy(baju[i].nama_baju,"JAWA BARAT");
 if(baju[i].ukuran=='S'||
 baju[i].ukuran=='s')
 baju[i].harga=40000;
 else if(baju[i].ukuran=='M'||
 baju[i].ukuran=='m')
 baju[i].harga=50000;
 else if(baju[i].ukuran=='L'||
 baju[i].ukuran=='l')
 baju[i].harga=55000;
 else
 baju[i].harga=0;}
 elseif((strcmp(baju[i].kode,"JT")==0)||
 (strcmp(baju[i].kode,"jt")==0))
 {strcpy(baju[i].nama_baju,
 "JAWA TENGAH");
 if(baju[i].ukuran=='S'||
 baju[i].ukuran=='s')
 baju[i].harga=45000;
           else if(baju[i].ukuran=='M'
               ||baju[i].ukuran=='m')
              baju[i].harga=55000;
           else if(baju[i].ukuran=='L'||
 baju[i].ukuran=='l')
 baju[i].harga=60000;
 else
 baju[i].harga=0;}
 elseif((strcmp(baju[i].kode,"SB")==0)||
 (strcmp(baju[i].kode,"sb")==0))
 {strcpy(baju[i].nama_baju,
```

```
                "SUMATRA BARAT");
            if (baju[i].ukuran=='S'||
            baju[i].ukuran=='s')
            baju[i].harga=50000;
       else if(baju[i].ukuran=='M'||
       baju[i].ukuran=='m')
 baju[i].harga=60000;
            else if(baju[i].ukuran=='L'||
            baju[i].ukuran=='l')
                 baju[i].harga=70000;
            else
                 baju[i].harga=0;}
     else
{strcpy(baju[i].nama_baju,"Tidak Milih");
if(baju[i].ukuran=='S'|| aju[i].ukuran=='s')
 baju[i].harga=0;
else if(baju[i].ukuran=='M'||baju[i].ukuran=='m')
 baju[i].harga=0;
else if(baju[i].ukuran=='L'||baju[i].ukuran=='l')
 baju[i].harga=0;
else
baju[i].harga=0;}

cout<<"JUMLAH SEWA : ";
cin>>baju[i].jumlah_sewa;
baju[i].jumlah_harga=baju[i].harga*baju[i].jumlah_sewa;}

clrscr();

cout<<"\t\t\tPENYEWAAN PAKAIAN ADAT NASIONAL"<<endl;
cout<<"\t\t\t\tCHONIO BOUTIQUE "<<endl;
garis();
cout<<"NAMA PENYEWA : "<<nama<<endl;
cout<<"LAMA SEWA BAJU : "<<lama_sewa<<endl;
cout<<"JUMLAH DATA : "<<n<<endl;
cout<<endl;
cout<<"DATA BAJU YANG DISEWA "<<endl;
cout<<"================================================
==================="<<endl;
cout<<"No.  Nama  Pake  Harga  ukuran  Jml_sewa  Subtotal
"<<endl;
cout<<"================================================
==================="<<endl;
for(i=1;i<=n;i++)
{
cout<<i<<" "<<baju[i].nama_baju<<"\t "
 <<baju[i].harga<<" "
```

```
 <<baju[i].ukuran<<" "
 <<baju[i].jumlah_sewa<<"\t "
 <<baju[i].jumlah_harga<<endl;
total_bayar=total_bayar+baju[i].jumlah_harga;}
cout<<"================================================
==================="<<endl;
cout<<endl;
cout<<"\t\t\t\t\t Total Bayar : "
 <<total_bayar<<endl;
cout<<"\t\t\t\t\t Uang Bayar : ";
cin>>bayar;
kembali=bayar-total_bayar;
cout<<"\t\t\t\t\t Uang Kembali : "<<kembali;
cout<<endl;
cout<<endl;
cout<<endl;
cout<<"INPUT DATA LAGI [Y/T] : ";
cin>>lagi;
if (lagi=='Y' || lagi=='y')
goto atas;
keluar:
getch();}
```

16.10. Program Penjualan Komputer PC menggunakan Struktur

```
#include <iostream.h>
#include <conio.h>
#include <stdlib.h>
#include <stdio.h>
#define max 200

void delay(int a)
     {for(int x=0;x<a*100;x++)
      { for(int y=0;y<a*100;y++)
 { }
 }
 }

struct komputer{
     char proc[20], vga[20], ram[10],
      per[20], harga[30], nama[30],
      alamat[50], kontak[15]; };

struct tumpukan{
     int atas, data[max];
```

```
      struct komputer pc[50];}t;

struct rakitan{
     int motherboard,casing,processor,vga,
      ram,total;}ra;

void awal(){
     t.atas=-1;}
int kosong(){
     if(t.atas==-1)
      return 1;
 else
      return 0;}

int penuh(){
     if(t.atas==max)
      return 1;
 else
      return 0;}

void jual(){
 char lanjut;
 if(t.atas == max-1){
     clrscr();
     gotoxy(33,12);cout<<"Maaf Penuh";
 getch();}
 else{t.atas++;
     clrscr();
 gotoxy(26,1);
 cout<<"============================";
     gotoxy(26,2);
     cout<<"||KOMPUTER MART||";
     gotoxy(26,3);
 cout<<"||Jual beli PC baru & bekas||";
 gotoxy(26,4);
 cout<<"============================";
     gotoxy(27,6);
 cout<<"============================";
     gotoxy(27,7);
     cout<<" PASANG IKLAN";
     gotoxy(27,8);
     cout<<"============================";
 gotoxy(3,10);cout<<"Keterangan :";
 gotoxy(3,11);cout<<"------------";
 gotoxy(3,12);
      cout<<"1. Identitas diri harus asli.";
 gotoxy(3,13);
```

```
     cout<<"2. Keterangan barang sesuai
     dengan barang aslinya.";
gotoxy(3,14);
     cout<<"3. Tidak ada unsur penipuan.";
gotoxy(3,15);
     cout<<"4. Biaya pasang iklan gratis";
gotoxy(3,17);
     cout<<"Lanjutkan (y/n) : ";
     cin>>lanjut;
if(lanjut=='y' || lanjut=='Y'){
     clrscr();
    gotoxy(26,1);
    cout<<"============================";
    gotoxy(26,2);
    cout<<"||KOMPUTER MART ||";
    gotoxy(26,3);
    cout<<"||Jual beli PC baru dan
     bekas||";
    gotoxy(26,4);
    cout<<"============================"
    gotoxy(27,7);
    cout<<"============================";
    gotoxy(27,8);cout<<" PASANG IKLAN";
    gotoxy(27,9);
    cout<<"============================ ";
gotoxy(3,11);cout<<"Nama lengkap\t: ";
    gets( t.pc[t.atas].nama);
gotoxy(3,12);cout<<"Alamat\t: ";
    gets( t.pc[t.atas].alamat);
gotoxy(3,13);cout<<"No.telp\t: ";
    gets( t.pc[t.atas].kontak);
gotoxy(3,15);cout<<"Keterangan barang";
    gotoxy(3,16);cout<<"-----------------";
gotoxy(3,17);cout<<"> Processor\t\t: ";
    gets( t.pc[t.atas].proc);
gotoxy(3,18);cout<<"> VGA\t\t\t: ";
    gets(t.pc[t.atas].vga);
gotoxy(3,19);cout<<"> RAM\t\t\t: ";
    gets(t.pc[t.atas].ram);
gotoxy(3,20);cout<<">Perlengkapan\t: ";
    gets(t.pc[t.atas].per);
gotoxy(3,21);cout<<"> Harga\t\t: ";
    gets(t.pc[t.atas].harga);
gotoxy(3,23);cout<<"Setuju?";
}
else{}
}
```

```
}
void lihat(){
     if(kosong()==0){
     for(int i=t.atas;i>=0;i--){
          cout<<endl;
          cout<<endl;
          cout<<"Nama\t\t: "<<t.pc[i].nama<<endl;
          cout<<"Alamat\t\t: "<<t.pc[i].alamat<<endl;
          cout<<"No.telp\t\t: "<<t.pc[i].kontak<<endl;
          cout<<endl;
          cout<<"Keterangan barang"<<endl;
          cout<<"-----------------"<<endl;
          cout<<" > Processor\t: "<<t.pc[i].proc<<endl;
          cout<<" > VGA\t\t: "<<t.pc[i].vga<<endl;
          cout<<" > RAM\t\t: "<<t.pc[i].ram<<endl;
          cout<<" > Perlengkapan\t: "<<t.pc[i].per<<endl;
          cout<<" > Harga\t: "<<t.pc[i].harga<<endl;

     cout<<"=============================================
==="<<endl;
          cout<<endl;
          cout<<endl;
 }
          cout<<"\t\t\tOKE..!!";
 }
 else{
          clrscr();
          gotoxy(33,12);cout<<"Kosong";
     }
}

void hapus(){
 if(t.atas==-1){
     cout<<"\nkosong";
 cout<<"\ntekan enter";
 getch();
     }
 else{
     cout<<endl;
     cout<<endl;
     cout<<"\tNama\t\t: "<<t.pc[t.atas].nama<<endl;
     cout<<"\tAlamat\t\t: "<<t.pc[t.atas].alamat<<endl;
     cout<<"\tNo.telp\t\t: "<<t.pc[t.atas].kontak<<endl;
     cout<<endl;
     cout<<"\tKeterangan barang"<<endl;
     cout<<"\t-----------------"<<endl;
     cout<<"\t > Processor\t: "<<t.pc[t.atas].proc<<endl;
```

```
     cout<<"\t > VGA\t\t: "<<t.pc[t.atas].vga<<endl;
     cout<<"\t > RAM\t\t: "<<t.pc[t.atas].ram<<endl;
     cout<<"\t > Perlengkapan\t:
"<<t.pc[t.atas].per<<endl;
     cout<<"\t > Harga\t: "<<t.pc[t.atas].harga<<endl;
     t.atas--;
     cout<<endl;
     cout<<"oke";
     getch();
     }
     }
void game(){
char p[10],oke;
float c;
int game,v,vs,r;
     clrscr();

gotoxy(26,1);cout<<"==================================";
 gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
 gotoxy(26,3);cout<<"|| Jual beli PC baru dan bekas ||";

gotoxy(26,4);cout<<"==================================";
     cout<<endl;
 cout<<endl;
 cout<<"List Game : \n";
 cout<<"============";
 cout<<endl;
 cout<<"1. Assassins Creed 4 Black Flag\n";
 cout<<"2. Battlefield 4\n";
 cout<<"3. Call Of Duty Ghosts\n";
 cout<<"4. Crysis 3\n";
 cout<<"5. Fifa 14\n";
 cout<<endl;
 cout<<"Pilih : ";cin>>game;
 if(game==1){
     clrscr();

gotoxy(26,1);cout<<"==================================";
     gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
     gotoxy(26,3);cout<<"|| Jual beli PC baru dan bekas
||";

     gotoxy(26,4);cout<<"================================
===";
 gotoxy(26,7);cout<<"Assassins Creed 4 Black Flag";
 cout<<endl;
 cout<<endl;
```

```
 cout<<"Spesifikasi Minimum :\n";
 cout<<"Processor\t: Core 2 Quad Q6400 2.13 GHz\n";
 cout<<"VGA\t\t: GeForce GT 250 1 GB\n";
 cout<<"RAM\t\t: 2 GB\n\n";
 cout<<"Recommended spec :\n";
 cout<<"Processor\t: Core i5-2400S 2.5 GHz\n";
 cout<<"VGA\t\t: GeForce GTX 470\n";
 cout<<"RAN\t\t: 4 GB\n\n";

cout<<"=========================================\n\n";
 cout<<"Spesifikasi PC anda :\n\n";
 cout<<"Processor\t: ";gets(p);
 cout<<"Clock Speed\t: ";cin>>c;
 cout<<endl;
 cout<<"1. Nvidia Geforce\n";
 cout<<"2. Nvidia GT\n";
 cout<<"3. Nvidia GTX\n\n";
 cout<<"pilih VGA\t: ";cin>>v;
     if(v==1){
 v=25;
          cout<<"Nvidia Gerofce\n";
 cout<<"Seri VGA\t: ";cin>>vs;
          cout<<"RAM\t\t: ";cin>>r;
     clrscr();
     for(int i=0;i<=100;i+=3){
     delay(80);
     gotoxy(33,12);cout<<"proses . . . ."<<i<<"%";
     }
     clrscr();

     gotoxy(26,1);cout<<"================================
===";
     gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
     gotoxy(26,3);cout<<"|| Jual beli PC baru dan bekas
||";

     gotoxy(26,4);cout<<"================================
===";
 cout<<endl;
 cout<<endl;
 cout<<"============================\n";
          cout<<"Spec PC anda: \n\n";
          cout<<"Processor\t: "<<p<<" "<<c<<" GHz"<<endl;
          cout<<"VGA\t\t: "<<"Nvidia Geforce"<<"
"<<vs<<endl;
          cout<<"RAM\t\t: "<<r<<" GB"<<endl;
 }
```

```
 if(v==2){
     v=50;
          cout<<"Nvidia GT\n";
 cout<<"Seri VGA\t: ";cin>>vs;
          cout<<"RAM\t\t: ";cin>>r;
     clrscr();
     for(int i=0;i<=100;i+=3){
     delay(80);
     gotoxy(33,12);cout<<"proses . . . ."<<i<<"%";
     }
     clrscr();

     gotoxy(26,1);cout<<"================================
===";
     gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
     gotoxy(26,3);cout<<"|| Jual beli PC baru dan bekas
||";

     gotoxy(26,4);cout<<"================================
===";
 cout<<endl;
 cout<<endl;
 cout<<"============================\n";
          cout<<"Spec PC anda: \n\n";
          cout<<"Processor\t: "<<p<<" "<<c<<" GHz"<<endl;
          cout<<"VGA\t\t: "<<"Nvidia GT"<<" "<<vs<<endl;
          cout<<"RAM\t\t: "<<r<<" GB"<<endl;
 }
 if(v==3){
     v=100;
          cout<<"Nvidia GTX\n";
 cout<<"Seri VGA\t: ";cin>>vs;
          cout<<"RAM\t\t: ";cin>>r;
     clrscr();
     for(int i=0;i<=100;i+=3){
     delay(80);
     gotoxy(33,12);cout<<"proses . . . ."<<i<<"%";
     }
     clrscr();

     gotoxy(26,1);cout<<"================================
===";
     gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
     gotoxy(26,3);cout<<"|| Jual beli PC baru dan bekas
||";
```

```
     gotoxy(26,4);cout<<"================================
===";
 cout<<endl;
 cout<<endl;
 cout<<"============================\n";
          cout<<"Spec PC anda: \n\n";
          cout<<"Processor\t: "<<p<<" "<<c<<" GHz"<<endl;
          cout<<"VGA\t\t: "<<"Nvidia GTX"<<" "<<vs<<endl;
          cout<<"RAM\t\t: "<<r<<" GB"<<endl;
 }
 if(c<=2.13 && v<25 && vs<=250 && r<2){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                         cout<<endl;
 cout<<endl;
 cout<<"Processor\t: LOW\n\n";
 cout<<"VGA\t\t: LOW\n\n";
 cout<<"RAM\t\t: LOW\n\n";
 cout<<endl;
 cout<<"> PC anda tidak mampu untuk menjalankan game ini
dengan lancar\n";
 }
     else{}
          }
 if(c>2.13 && v<25 && vs<=250 && r<2){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                         cout<<endl;
 cout<<endl;
 cout<<"Processor\t: OKE\n\n";
 cout<<"VGA\t\t: LOW\n\n";
 cout<<"RAM\t\t: LOW\n\n";
 cout<<endl;
 cout<<"> Processor bisa berjalan dengan baik\n";
 cout<<"> Permasalahan ada pada VGA dan RAM yang
kurang\n";
 cout<<"Saran :\n";
 cout<<"> Upgrade VGA dan RAM PC anda";
 }
     else{}
          }
 if(c>2.13 && v<25 && vs<=250 && r>=2){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
```

```
          if(oke=='y' || oke=='Y'){
                          cout<<endl;
 cout<<endl;
 cout<<"Processor\t: OKE\n\n";
 cout<<"VGA\t\t: LOW\n\n";
 cout<<"RAM\t\t: OKE\n\n";
 cout<<endl;
 cout<<"> Processor dan RAM bisa berjalan dengan baik\n";
 cout<<"> Permasalahan ada pada VGA yang kurang baik\n";
 cout<<"Saran :\n";
 cout<<"> Upgrade VGA PC anda";
 }
     else{}
          }
 if(c>2.13 && v>=25 && vs>=250 && r<2){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                          cout<<endl;
 cout<<endl;
 cout<<"Processor\t: OKE\n\n";
 cout<<"VGA\t\t: OKE\n\n";
 cout<<"RAM\t\t: LOW\n\n";
 cout<<endl;
 cout<<"> Processor dan VGA bisa berjalan dengan baik\n";
 cout<<"> RAM PC anda kurang\n";
 cout<<"Saran :\n";
 cout<<"> Walaupun bisa untuk menjalankan game ini dengan
setting low,\n";
 cout<<" tetapi disarankan untuk upgrade RAM PC anda";
 }
     else{}
          }
 if(c<=2.13 && v>=25 && vs>=250 && r>=2){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                          cout<<endl;
 cout<<endl;
 cout<<"Processor\t: LOW\n\n";
 cout<<"VGA\t\t: OKE\n\n";
 cout<<"RAM\t\t: OKE\n\n";
 cout<<endl;
 cout<<"> Processor kurang mendukung\n";
 cout<<"Saran :\n";
 cout<<"> Ganti Processor PC anda";
 }
```

```
    else{}
         }
 if(c<=2.13 && v<25 && vs<=550 && r>=2){
         cout<<endl;
              cout<<"Lanjut (y/n) : ";cin>>oke;
         if(oke=='y' || oke=='Y'){
                        cout<<endl;
 cout<<endl;
 cout<<"Processor\t: LOW\n\n";
 cout<<"VGA\t\t: LOW\n\n";
 cout<<"RAM\t\t: OKE\n\n";
 cout<<endl;
 cout<<"> Processor dan VGA tidak mendukung\n";
 cout<<"Saran :\n";
 cout<<"> Ganti Processor dan upgrade VGA PC anda";
 }
    else{}
         }
 if(c<=2.13 && v>=25 && vs>=250 && r<2){
         cout<<endl;
              cout<<"Lanjut (y/n) : ";cin>>oke;
         if(oke=='y' || oke=='Y'){
                        cout<<endl;
 cout<<endl;
 cout<<"Processor\t: LOW\n\n";
 cout<<"VGA\t\t: OKE\n\n";
 cout<<"RAM\t\t: LOW\n\n";
 cout<<endl;
 cout<<"> Processor dan RAM tidak mendukung\n";
 cout<<"Saran :\n";
 cout<<"> Ganti Processor dan upgrade RAM PC anda";
 }
    else{}
         }
 if(c>2.13 && v>=50 && r>=2 && v<100){
         cout<<endl;
              cout<<"Lanjut (y/n) : ";cin>>oke;
         if(oke=='y' || oke=='Y'){
                        cout<<endl;
 cout<<endl;
 cout<<"Processor\t: OKE\n\n";
 cout<<"VGA\t\t: OKE\n\n";
 cout<<"RAM\t\t: OKE\n\n";
 cout<<endl;
 cout<<"> PC anda bisa menjalankan game dengan graphic
disesuaikan\n";
```

```
 cout<<"> Akan lebih baik jika spec PC sesuai dengan
Recommended";
 }
     else{}
          }
 if(c>=2.5 && v==100 && r>=4){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                         cout<<endl;
 cout<<endl;
 cout<<"Processor\t: GOOD\n\n";
 cout<<"VGA\t\t: GOOD\n\n";
 cout<<"RAM\t\t: GOOD\n\n";
 cout<<endl;
 cout<<"> PC anda bisa menjalankan game dengan lancar\n";
 }
     else{}
          }
 }
 else if(game==2){
     clrscr();

gotoxy(26,1);cout<<"==================================";
     gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
     gotoxy(26,3);cout<<"|| Jual beli PC baru dan bekas
||";

     gotoxy(26,4);cout<<"================================
===";
 gotoxy(26,7);cout<<"Battlefield 4";
 cout<<endl;
 cout<<endl;
 cout<<"Spesifikasi Minimum :\n";
 cout<<"Processor\t: Core 2 Duo E6600 2.4 GHz\n";
 cout<<"VGA\t\t: GeForce 8800 GT\n";
 cout<<"RAM\t\t: 4 GB\n\n";
 cout<<"Recommended spec :\n";
 cout<<"Processor\t: Core i7-930 Quad 2.8 GHz\n";
 cout<<"VGA\t\t: GeForce GTX 660\n";
 cout<<"RAN\t\t: 8 GB\n\n";

cout<<"=========================================\n\n";
 cout<<"Spesifikasi PC anda :\n\n";
 cout<<"Processor\t: ";gets(p);
 cout<<"Clock Speed\t: ";cin>>c;
 cout<<endl;
```

```
 cout<<"1. Nvidia Geforce\n";
 cout<<"2. Nvidia GT\n";
 cout<<"3. Nvidia GTX\n\n";
 cout<<"pilih VGA\t: ";cin>>v;
     if(v==1){
 v=25;
          cout<<"Nvidia Gerofce\n";
 cout<<"Seri VGA\t: ";cin>>vs;
          cout<<"RAM\t\t: ";cin>>r;
     clrscr();
     for(int i=0;i<=100;i+=3){
     delay(80);
     gotoxy(33,12);cout<<"proses . . . ."<<i<<"%";
     }
     clrscr();

     gotoxy(26,1);cout<<"================================
===";
     gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
     gotoxy(26,3);cout<<"|| Jual beli PC baru dan bekas
||";

     gotoxy(26,4);cout<<"================================
===";
 cout<<endl;
 cout<<endl;
 cout<<"============================\n";
          cout<<"Spec PC anda: \n\n";
          cout<<"Processor\t: "<<p<<" "<<c<<" GHz"<<endl;
          cout<<"VGA\t\t: "<<"Nvidia Geforce"<<"
"<<vs<<endl;
          cout<<"RAM\t\t: "<<r<<" GB"<<endl;
 }
 if(v==2){
     v=50;
          cout<<"Nvidia GT\n";
 cout<<"Seri VGA\t: ";cin>>vs;
          cout<<"RAM\t\t: ";cin>>r;
     clrscr();
     for(int i=0;i<=100;i+=3){
     delay(80);
     gotoxy(33,12);cout<<"proses . . . ."<<i<<"%";
     }
     clrscr();

     gotoxy(26,1);cout<<"================================
===";
```

```
     gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
     gotoxy(26,3);cout<<"|| Jual beli PC baru dan bekas
||";

     gotoxy(26,4);cout<<"================================
===";
 cout<<endl;
 cout<<endl;
 cout<<"============================\n";
          cout<<"Spec PC anda: \n\n";
          cout<<"Processor\t: "<<p<<" "<<c<<" GHz"<<endl;
          cout<<"VGA\t\t: "<<"Nvidia GT"<<" "<<vs<<endl;
          cout<<"RAM\t\t: "<<r<<" GB"<<endl;
 }
 if(v==3){
     v=100;
          cout<<"Nvidia GTX\n";
 cout<<"Seri VGA\t: ";cin>>vs;
          cout<<"RAM\t\t: ";cin>>r;
     clrscr();
     for(int i=0;i<=100;i+=3){
     delay(80);
     gotoxy(33,12);cout<<"proses . . . ."<<i<<"%";
     }
     clrscr();

     gotoxy(26,1);cout<<"================================
===";
     gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
     gotoxy(26,3);cout<<"|| Jual beli PC baru dan bekas
||";

     gotoxy(26,4);cout<<"================================
===";
 cout<<endl;
 cout<<endl;
 cout<<"============================\n";
          cout<<"Spec PC anda: \n\n";
          cout<<"Processor\t: "<<p<<" "<<c<<" GHz"<<endl;
          cout<<"VGA\t\t: "<<"Nvidia GTX"<<" "<<vs<<endl;
          cout<<"RAM\t\t: "<<r<<" GB"<<endl;
 }
 if(c<=2.4 && v<=25 && vs<=610 && r<=4){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                         cout<<endl;
```

```
 cout<<endl;
 cout<<"Processor\t: BAD\n\n";
 cout<<"VGA\t\t: BAD\n\n";
 cout<<"RAM\t\t: BAD\n\n";
 cout<<endl;
 cout<<"> PC anda tidak mampu untuk menjalankan game
ini\n";
 }
     else{}
          }
 if(c>2.4 && v<=25 && vs<=610 && r<=4){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                         cout<<endl;
 cout<<endl;
 cout<<"Processor\t: OKE\n\n";
 cout<<"VGA\t\t: LOW\n\n";
 cout<<"RAM\t\t: LOW\n\n";
 cout<<endl;
 cout<<"> Processor bisa berjalan dengan baik\n";
 cout<<"> Permasalahan ada pada VGA dan RAM yang
kurang\n";
 cout<<"Saran :\n";
 cout<<"> Upgrade VGA dan RAM PC anda";
 }
     else{}
          }
 if(c>2.4 && v<=25 && vs<=610 && r>=4){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                         cout<<endl;
 cout<<endl;
 cout<<"Processor\t: OKE\n\n";
 cout<<"VGA\t\t: LOW\n\n";
 cout<<"RAM\t\t: OKE\n\n";
 cout<<endl;
 cout<<"> Processor dan RAM bisa berjalan dengan baik\n";
 cout<<"> Permasalahan ada pada VGA yang kurang baik\n";
 cout<<"Saran :\n";
 cout<<"> Upgrade VGA PC anda";
 }
     else{}
          }
 if(c>2.4 && v>25 && vs>610 && r<4){
          cout<<endl;
```

```
                cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                          cout<<endl;
 cout<<endl;
 cout<<"Processor\t: OKE\n\n";
 cout<<"VGA\t\t: OKE\n\n";
 cout<<"RAM\t\t: LOW\n\n";
 cout<<endl;
 cout<<"> Processor dan VGA bisa berjalan\n";
 cout<<"> RAM PC anda kurang\n";
 cout<<"Saran :\n";
 cout<<"> Walaupun bisa untuk menjalankan game ini dengan
setting low,\n";
 cout<<" tetapi disarankan untuk upgrade RAM PC anda";
 }
     else{}
          }
 if(c<=2.4 && v>25 && vs>610 && r>=4){
          cout<<endl;
                cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                          cout<<endl;
 cout<<endl;
 cout<<"Processor\t: LOW\n\n";
 cout<<"VGA\t\t: OKE\n\n";
 cout<<"RAM\t\t: OKE\n\n";
 cout<<endl;
 cout<<"> Processor kurang mendukung\n";
 cout<<"Saran :\n";
 cout<<"> Ganti Processor PC anda";
 }
     else{}
          }
 if(c<=2.4 && v<=25 && vs<=610 && r>=4){
          cout<<endl;
                cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                          cout<<endl;
 cout<<endl;
 cout<<"Processor\t: LOW\n\n";
 cout<<"VGA\t\t: LOW\n\n";
 cout<<"RAM\t\t: OKE\n\n";
 cout<<endl;
 cout<<"> Processor dan VGA tidak mendukung\n";
 cout<<"Saran :\n";
 cout<<"> Ganti Processor dan upgrade VGA PC anda";
 }
```

```
    else{}
         }
 if(c<=2.4 && v>25 && vs>610 && r<4){
         cout<<endl;
              cout<<"Lanjut (y/n) : ";cin>>oke;
         if(oke=='y' || oke=='Y'){
                        cout<<endl;
 cout<<endl;
 cout<<"Processor\t: LOW\n\n";
 cout<<"VGA\t\t: OKE\n\n";
 cout<<"RAM\t\t: LOW\n\n";
 cout<<endl;
 cout<<"> Processor dan RAM tidak mendukung\n";
 cout<<"Saran :\n";
 cout<<"> Ganti Processor dan upgrade RAM PC anda";
 }
    else{}
         }
 if(c>2.4 && v>25 && vs>610 && r>=4 && c<2.8 && v<=100 &&
vs<760 && r<8){
         cout<<endl;
              cout<<"Lanjut (y/n) : ";cin>>oke;
         if(oke=='y' || oke=='Y'){
                        cout<<endl;
 cout<<endl;
 cout<<"Processor\t: OKE\n\n";
 cout<<"VGA\t\t: OKE\n\n";
 cout<<"RAM\t\t: OKE\n\n";
 cout<<endl;
 cout<<"> PC anda bisa menjalankan game dengan graphic
disesuaikan\n";
 cout<<"> Akan lebih baik jika spec PC sesuai dengan
Recommended";
 }
    else{}
         }
 if(c>=2.8 && v>=100 && vs>=660 && r>=8){
         cout<<endl;
              cout<<"Lanjut (y/n) : ";cin>>oke;
         if(oke=='y' || oke=='Y'){
                        cout<<endl;
 cout<<endl;
 cout<<"Processor\t: GOOD\n\n";
 cout<<"VGA\t\t: GOOD\n\n";
 cout<<"RAM\t\t: GOOD\n\n";
 cout<<endl;
 cout<<"> PC anda bisa menjalankan game dengan lancar\n";
```

```
 }
     else{}
          }
 }
 else if(game==3){
     clrscr();

gotoxy(26,1);cout<<"==================================";
     gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
     gotoxy(26,3);cout<<"|| Jual beli PC baru dan bekas
||";

     gotoxy(26,4);cout<<"================================
===";
 gotoxy(26,7);cout<<"Call Of Duty Ghost";
 cout<<endl;
 cout<<endl;
 cout<<"Spesifikasi Minimum :\n";
 cout<<"Processor\t: Core 2 Duo E8200 2.5 GHz\n";
 cout<<"VGA\t\t: GeForce GT 450\n";
 cout<<"RAM\t\t: 4 GB\n\n";
 cout<<"Recommended spec :\n";
 cout<<"Processor\t: Core 2 Quad Q8400 2.66 GHz\n";
 cout<<"VGA\t\t: GeForce GTX 660\n";
 cout<<"RAN\t\t: 8 GB\n\n";

cout<<"=========================================\n\n";
 cout<<"Spesifikasi PC anda :\n\n";
 cout<<"Processor\t: ";gets(p);
 cout<<"Clock Speed\t: ";cin>>c;
 cout<<endl;
 cout<<"1. Nvidia Geforce\n";
 cout<<"2. Nvidia GT\n";
 cout<<"3. Nvidia GTX\n\n";
 cout<<"pilih VGA\t: ";cin>>v;
     if(v==1){
 v=25;
          cout<<"Nvidia Gerofce\n";
 cout<<"Seri VGA\t: ";cin>>vs;
          cout<<"RAM\t\t: ";cin>>r;
     clrscr();
     for(int i=0;i<=100;i+=3){
     delay(80);
     gotoxy(33,12);cout<<"proses . . . ."<<i<<"%";
     }
     clrscr();
```

```
     gotoxy(26,1);cout<<"================================
===";
     gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
     gotoxy(26,3);cout<<"|| Jual beli PC baru dan bekas
||";

     gotoxy(26,4);cout<<"================================
===";
 cout<<endl;
 cout<<endl;
 cout<<"============================\n";
          cout<<"Spec PC anda: \n\n";
          cout<<"Processor\t: "<<p<<" "<<c<<" GHz"<<endl;
          cout<<"VGA\t\t: "<<"Nvidia Geforce"<<"
"<<vs<<endl;
          cout<<"RAM\t\t: "<<r<<" GB"<<endl;
 }
 if(v==2){
     v=50;
          cout<<"Nvidia GT\n";
 cout<<"Seri VGA\t: ";cin>>vs;
          cout<<"RAM\t\t: ";cin>>r;
     clrscr();
     for(int i=0;i<=100;i+=3){
     delay(80);
     gotoxy(33,12);cout<<"proses . . . ."<<i<<"%";
     }
     clrscr();

     gotoxy(26,1);cout<<"================================
===";
     gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
     gotoxy(26,3);cout<<"|| Jual beli PC baru dan bekas
||";

     gotoxy(26,4);cout<<"================================
===";
 cout<<endl;
 cout<<endl;
 cout<<"============================\n";
          cout<<"Spec PC anda: \n\n";
          cout<<"Processor\t: "<<p<<" "<<c<<" GHz"<<endl;
          cout<<"VGA\t\t: "<<"Nvidia GT"<<" "<<vs<<endl;
          cout<<"RAM\t\t: "<<r<<" GB"<<endl;
 }
 if(v==3){
```

```
     v=100;
          cout<<"Nvidia GTX\n";
 cout<<"Seri VGA\t: ";cin>>vs;
          cout<<"RAM\t\t: ";cin>>r;
     clrscr();
     for(int i=0;i<=100;i+=3){
     delay(80);
     gotoxy(33,12);cout<<"proses . . . ."<<i<<"%";
     }
     clrscr();

     gotoxy(26,1);cout<<"================================
===";
     gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
     gotoxy(26,3);cout<<"|| Jual beli PC baru dan bekas
||";

     gotoxy(26,4);cout<<"================================
===";
 cout<<endl;
 cout<<endl;
 cout<<"============================\n";
          cout<<"Spec PC anda: \n\n";
          cout<<"Processor\t: "<<p<<" "<<c<<" GHz"<<endl;
          cout<<"VGA\t\t: "<<"Nvidia GTX"<<" "<<vs<<endl;
          cout<<"RAM\t\t: "<<r<<" GB"<<endl;
 }
 if(c<=2.5 && v<=25 && vs<=620 && r<=4){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                         cout<<endl;
 cout<<endl;
 cout<<"Processor\t: BAD\n\n";
 cout<<"VGA\t\t: BAD\n\n";
 cout<<"RAM\t\t: BAD\n\n";
 cout<<endl;
 cout<<"> PC anda tidak mampu untuk menjalankan game
ini\n";
 }
     else{}
          }
 if(c>2.5 && v<=25 && vs<=620 && r<=4){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                         cout<<endl;
```

```
 cout<<endl;
 cout<<"Processor\t: OKE\n\n";
 cout<<"VGA\t\t: LOW\n\n";
 cout<<"RAM\t\t: LOW\n\n";
 cout<<endl;
 cout<<"> Processor bisa berjalan dengan baik\n";
 cout<<"> Permasalahan ada pada VGA dan RAM yang
kurang\n";
 cout<<"Saran :\n";
 cout<<"> Upgrade VGA dan RAM PC anda";
 }
     else{}
          }
 if(c>2.5 && v<=25 && vs<=620 && r>=4){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                         cout<<endl;
 cout<<endl;
 cout<<"Processor\t: OKE\n\n";
 cout<<"VGA\t\t: LOW\n\n";
 cout<<"RAM\t\t: OKE\n\n";
 cout<<endl;
 cout<<"> Processor dan RAM bisa berjalan dengan baik\n";
 cout<<"> Permasalahan ada pada VGA yang kurang baik\n";
 cout<<"Saran :\n";
 cout<<"> Upgrade VGA PC anda";
 }
     else{}
          }
 if(c>2.5 && v>25 && vs>620 && r<4){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                         cout<<endl;
 cout<<endl;
 cout<<"Processor\t: OKE\n\n";
 cout<<"VGA\t\t: OKE\n\n";
 cout<<"RAM\t\t: LOW\n\n";
 cout<<endl;
 cout<<"> Processor dan VGA bisa berjalan\n";
 cout<<"> RAM PC anda kurang\n";
 cout<<"Saran :\n";
 cout<<"> Walaupun bisa untuk menjalankan game ini dengan
setting low,\n";
 cout<<" tetapi disarankan untuk upgrade RAM PC anda";
 }
```

```
    else{}
         }
if(c<=2.5 && v>25 && vs>620 && r>=4){
         cout<<endl;
              cout<<"Lanjut (y/n) : ";cin>>oke;
         if(oke=='y' || oke=='Y'){
                        cout<<endl;
cout<<endl;
cout<<"Processor\t: LOW\n\n";
cout<<"VGA\t\t: OKE\n\n";
cout<<"RAM\t\t: OKE\n\n";
cout<<endl;
cout<<"> Processor kurang mendukung\n";
cout<<"Saran :\n";
cout<<"> Ganti Processor PC anda";
}
    else{}
         }
if(c<=2.5 && v<=25 && vs<=620 && r>=4){
         cout<<endl;
              cout<<"Lanjut (y/n) : ";cin>>oke;
         if(oke=='y' || oke=='Y'){
                        cout<<endl;
cout<<endl;
cout<<"Processor\t: LOW\n\n";
cout<<"VGA\t\t: LOW\n\n";
cout<<"RAM\t\t: OKE\n\n";
cout<<endl;
cout<<"> Processor dan VGA tidak mendukung\n";
cout<<"Saran :\n";
cout<<"> Ganti Processor dan upgrade VGA PC anda";
}
    else{}
         }
if(c<=2.5 && v>25 && vs>620 && r<4){
         cout<<endl;
              cout<<"Lanjut (y/n) : ";cin>>oke;
         if(oke=='y' || oke=='Y'){
                        cout<<endl;
cout<<endl;
cout<<"Processor\t: LOW\n\n";
cout<<"VGA\t\t: OKE\n\n";
cout<<"RAM\t\t: LOW\n\n";
cout<<endl;
cout<<"> Processor dan RAM tidak mendukung\n";
cout<<"Saran :\n";
cout<<"> Ganti Processor dan upgrade RAM PC anda";
```

```
 }
     else{}
          }
 if(c>2.5 && v>25 && vs>620 && r>=4 && c<2.66 && v<=100
&& vs<760 && r<8){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                         cout<<endl;
 cout<<endl;
 cout<<"Processor\t: OKE\n\n";
 cout<<"VGA\t\t: OKE\n\n";
 cout<<"RAM\t\t: OKE\n\n";
 cout<<endl;
 cout<<"> PC anda bisa menjalankan game dengan graphic
disesuaikan\n";
 cout<<"> Akan lebih baik jika spec PC sesuai dengan
Recommended";
 }
     else{}
          }
 if(c>=2.66 && v>=100 && vs>=660 && r>=8){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                         cout<<endl;
 cout<<endl;
 cout<<"Processor\t: GOOD\n\n";
 cout<<"VGA\t\t: GOOD\n\n";
 cout<<"RAM\t\t: GOOD\n\n";
 cout<<endl;
 cout<<"> PC anda bisa menjalankan game dengan lancar\n";
 }
     else{}
          }
 }
 else if(game==4){
     clrscr();

gotoxy(26,1);cout<<"==================================";
     gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
     gotoxy(26,3);cout<<"|| Jual beli PC baru dan bekas
||";

     gotoxy(26,4);cout<<"==============================
===";
 gotoxy(26,7);cout<<"Crysis 3";
```

```
 cout<<endl;
 cout<<endl;
 cout<<"Spesifikasi Minimum :\n";
 cout<<"Processor\t: Dual core 2.4 GHz\n";
 cout<<"VGA\t\t: GeForce 610\n";
 cout<<"RAM\t\t: 2 GB\n\n";
 cout<<"Recommended spec :\n";
 cout<<"Processor\t: Core i7-2600K 4-Core 3.40 GHz\n";
 cout<<"VGA\t\t: GeForce GTX 680\n";
 cout<<"RAN\t\t: 8 GB\n\n";

cout<<"=========================================\n\n";
 cout<<"Spesifikasi PC anda :\n\n";
 cout<<"Processor\t: ";gets(p);
 cout<<"Clock Speed\t: ";cin>>c;
 cout<<endl;
 cout<<"1. Nvidia Geforce\n";
 cout<<"2. Nvidia GT\n";
 cout<<"3. Nvidia GTX\n\n";
 cout<<"pilih VGA\t: ";cin>>v;
     if(v==1){
 v=25;
          cout<<"Nvidia Gerofce\n";
 cout<<"Seri VGA\t: ";cin>>vs;
          cout<<"RAM\t\t: ";cin>>r;
     clrscr();
     for(int i=0;i<=100;i+=3){
     delay(80);
     gotoxy(33,12);cout<<"proses . . . ."<<i<<"%";
     }
     clrscr();

     gotoxy(26,1);cout<<"================================
===";
     gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
     gotoxy(26,3);cout<<"|| Jual beli PC baru dan bekas
||";

     gotoxy(26,4);cout<<"================================
===";
 cout<<endl;
 cout<<endl;
 cout<<"============================\n";
          cout<<"Spec PC anda: \n\n";
          cout<<"Processor\t: "<<p<<" "<<c<<" GHz"<<endl;
          cout<<"VGA\t\t: "<<"Nvidia Geforce"<<"
"<<vs<<endl;
```

```
          cout<<"RAM\t\t: "<<r<<" GB"<<endl;
 }
 if(v==2){
     v=50;
          cout<<"Nvidia GT\n";
 cout<<"Seri VGA\t: ";cin>>vs;
          cout<<"RAM\t\t: ";cin>>r;
     clrscr();
     for(int i=0;i<=100;i+=3){
     delay(80);
     gotoxy(33,12);cout<<"proses . . . ."<<i<<"%";
     }
     clrscr();

     gotoxy(26,1);cout<<"================================
===";
     gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
     gotoxy(26,3);cout<<"|| Jual beli PC baru dan bekas
||";

     gotoxy(26,4);cout<<"================================
===";
 cout<<endl;
 cout<<endl;
 cout<<"============================\n";
          cout<<"Spec PC anda: \n\n";
          cout<<"Processor\t: "<<p<<" "<<c<<" GHz"<<endl;
          cout<<"VGA\t\t: "<<"Nvidia GT"<<" "<<vs<<endl;
          cout<<"RAM\t\t: "<<r<<" GB"<<endl;
 }
 if(v==3){
     v=100;
          cout<<"Nvidia GTX\n";
 cout<<"Seri VGA\t: ";cin>>vs;
          cout<<"RAM\t\t: ";cin>>r;
     clrscr();
     for(int i=0;i<=100;i+=3){
     delay(80);
     gotoxy(33,12);cout<<"proses . . . ."<<i<<"%";
     }
     clrscr();

     gotoxy(26,1);cout<<"================================
===";
     gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
     gotoxy(26,3);cout<<"|| Jual beli PC baru dan bekas
||";
```

```
     gotoxy(26,4);cout<<"================================
===";
 cout<<endl;
 cout<<endl;
 cout<<"============================\n";
          cout<<"Spec PC anda: \n\n";
          cout<<"Processor\t: "<<p<<" "<<c<<" GHz"<<endl;
          cout<<"VGA\t\t: "<<"Nvidia GTX"<<" "<<vs<<endl;
          cout<<"RAM\t\t: "<<r<<" GB"<<endl;
 }
 if(c<=2.4 && v<=25 && vs<=610 && r<=2){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                         cout<<endl;
 cout<<endl;
 cout<<"Processor\t: BAD\n\n";
 cout<<"VGA\t\t: BAD\n\n";
 cout<<"RAM\t\t: BAD\n\n";
 cout<<endl;
 cout<<"> PC anda tidak mampu untuk menjalankan game
ini\n";
 }
     else{}
          }
 if(c>2.4 && v<=25 && vs<=610 && r<=2){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                         cout<<endl;
 cout<<endl;
 cout<<"Processor\t: OKE\n\n";
 cout<<"VGA\t\t: LOW\n\n";
 cout<<"RAM\t\t: LOW\n\n";
 cout<<endl;
 cout<<"> Processor bisa berjalan dengan baik\n";
 cout<<"> Permasalahan ada pada VGA dan RAM yang
kurang\n";
 cout<<"Saran :\n";
 cout<<"> Upgrade VGA dan RAM PC anda";
 }
     else{}
          }
 if(c>2.4 && v<=25 && vs<=610 && r>=2){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
```

```
          if(oke=='y' || oke=='Y'){
                          cout<<endl;
 cout<<endl;
 cout<<"Processor\t: OKE\n\n";
 cout<<"VGA\t\t: LOW\n\n";
 cout<<"RAM\t\t: OKE\n\n";
 cout<<endl;
 cout<<"> Processor dan RAM bisa berjalan dengan baik\n";
 cout<<"> Permasalahan ada pada VGA yang kurang baik\n";
 cout<<"Saran :\n";
 cout<<"> Upgrade VGA PC anda";
 }
     else{}
          }
 if(c>2.4 && v>=25 && vs>=610 && r<2){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                          cout<<endl;
 cout<<endl;
 cout<<"Processor\t: OKE\n\n";
 cout<<"VGA\t\t: OKE\n\n";
 cout<<"RAM\t\t: LOW\n\n";
 cout<<endl;
 cout<<"> Processor dan VGA bisa berjalan\n";
 cout<<"> RAM PC anda kurang\n";
 cout<<"Saran :\n";
 cout<<"> Walaupun bisa untuk menjalankan game ini dengan
setting low,\n";
 cout<<" tetapi disarankan untuk upgrade RAM PC anda";
 }
     else{}
          }
 if(c<=2.4 && v>=25 && vs>=610 && r>=2){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                          cout<<endl;
 cout<<endl;
 cout<<"Processor\t: LOW\n\n";
 cout<<"VGA\t\t: OKE\n\n";
 cout<<"RAM\t\t: OKE\n\n";
 cout<<endl;
 cout<<"> Processor kurang mendukung\n";
 cout<<"Saran :\n";
 cout<<"> Ganti Processor PC anda";
 }
```

```
    else{}
         }
 if(c<=2.4 && v<25 && vs<610 && r>=2){
         cout<<endl;
              cout<<"Lanjut (y/n) : ";cin>>oke;
         if(oke=='y' || oke=='Y'){
                        cout<<endl;
 cout<<endl;
 cout<<"Processor\t: LOW\n\n";
 cout<<"VGA\t\t: LOW\n\n";
 cout<<"RAM\t\t: OKE\n\n";
 cout<<endl;
 cout<<"> Processor dan VGA tidak mendukung\n";
 cout<<"Saran :\n";
 cout<<"> Ganti Processor dan upgrade VGA PC anda";
 }
    else{}
         }
 if(c<=2.4 && v>=25 && vs>=610 && r<2){
         cout<<endl;
              cout<<"Lanjut (y/n) : ";cin>>oke;
         if(oke=='y' || oke=='Y'){
                        cout<<endl;
 cout<<endl;
 cout<<"Processor\t: LOW\n\n";
 cout<<"VGA\t\t: OKE\n\n";
 cout<<"RAM\t\t: LOW\n\n";
 cout<<endl;
 cout<<"> Processor dan RAM tidak mendukung\n";
 cout<<"Saran :\n";
 cout<<"> Ganti Processor dan upgrade RAM PC anda";
 }
    else{}
         }
 if(c>2.4 && v>=25 && vs>=610 && r>=2 && c<3.4 && v<=100
&& vs<760 && r<8){
         cout<<endl;
              cout<<"Lanjut (y/n) : ";cin>>oke;
         if(oke=='y' || oke=='Y'){
                        cout<<endl;
 cout<<endl;
 cout<<"Processor\t: OKE\n\n";
 cout<<"VGA\t\t: OKE\n\n";
 cout<<"RAM\t\t: OKE\n\n";
 cout<<endl;
 cout<<"> PC anda bisa menjalankan game dengan graphic
disesuaikan\n";
```

```
 cout<<"> Akan lebih baik jika spec PC sesuai dengan
Recommended";
 }
     else{}
          }
 if(c>=3.4 && v>=100 && vs>=680 && r>=8){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                         cout<<endl;
 cout<<endl;
 cout<<"Processor\t: GOOD\n\n";
 cout<<"VGA\t\t: GOOD\n\n";
 cout<<"RAM\t\t: GOOD\n\n";
 cout<<endl;
 cout<<"> PC anda bisa menjalankan game dengan lancar\n";
 }
     else{}
          }
 }
 else if(game==5){
     clrscr();

gotoxy(26,1);cout<<"==================================";
     gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
     gotoxy(26,3);cout<<"|| Jual beli PC baru dan bekas
||";

     gotoxy(26,4);cout<<"================================
===";
 gotoxy(26,7);cout<<"FIFA 14";
 cout<<endl;
 cout<<endl;
 cout<<"Spesifikasi Minimum :\n";
 cout<<"Processor\t: Celeron E1200 Dual-Core 1.6 GHz\n";
 cout<<"VGA\t\t: GeForce 8400 Series\n";
 cout<<"RAM\t\t: 2 GB\n\n";
 cout<<"Recommended spec :\n";
 cout<<"Processor\t: Core 2 Duo E6600 2.4GHz\n";
 cout<<"VGA\t\t: GeForce 8800 GT 256MB\n";
 cout<<"RAN\t\t: 2 GB\n\n";

cout<<"=========================================\n\n";
 cout<<"Spesifikasi PC anda :\n\n";
 cout<<"Processor\t: ";gets(p);
 cout<<"Clock Speed\t: ";cin>>c;
 cout<<endl;
```

```
 cout<<"1. Nvidia Geforce\n";
 cout<<"2. Nvidia GT\n";
 cout<<"3. Nvidia GTX\n\n";
 cout<<"pilih VGA\t: ";cin>>v;
     if(v==1){
 v=25;
          cout<<"Nvidia Gerofce\n";
 cout<<"Seri VGA\t: ";cin>>vs;
          cout<<"RAM\t\t: ";cin>>r;
     clrscr();
     for(int i=0;i<=100;i+=3){
     delay(80);
     gotoxy(33,12);cout<<"proses . . . ."<<i<<"%";
     }
     clrscr();

     gotoxy(26,1);cout<<"================================
===";
     gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
     gotoxy(26,3);cout<<"|| Jual beli PC baru dan bekas
||";

     gotoxy(26,4);cout<<"================================
===";
 cout<<endl;
 cout<<endl;
 cout<<"============================\n";
          cout<<"Spec PC anda: \n\n";
          cout<<"Processor\t: "<<p<<" "<<c<<" GHz"<<endl;
          cout<<"VGA\t\t: "<<"Nvidia Geforce"<<"
"<<vs<<endl;
          cout<<"RAM\t\t: "<<r<<" GB"<<endl;
 }
 if(v==2){
     v=50;
          cout<<"Nvidia GT\n";
 cout<<"Seri VGA\t: ";cin>>vs;
          cout<<"RAM\t\t: ";cin>>r;
     clrscr();
     for(int i=0;i<=100;i+=3){
     delay(80);
     gotoxy(33,12);cout<<"proses . . . ."<<i<<"%";
     }
     clrscr();

     gotoxy(26,1);cout<<"================================
===";
```

```
     gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
     gotoxy(26,3);cout<<"|| Jual beli PC baru dan bekas
||";

     gotoxy(26,4);cout<<"================================
===";
 cout<<endl;
 cout<<endl;
 cout<<"============================\n";
          cout<<"Spec PC anda: \n\n";
          cout<<"Processor\t: "<<p<<" "<<c<<" GHz"<<endl;
          cout<<"VGA\t\t: "<<"Nvidia GT"<<" "<<vs<<endl;
          cout<<"RAM\t\t: "<<r<<" GB"<<endl;
 }
 if(v==3){
     v=100;
          cout<<"Nvidia GTX\n";
 cout<<"Seri VGA\t: ";cin>>vs;
          cout<<"RAM\t\t: ";cin>>r;
     clrscr();
     for(int i=0;i<=100;i+=3){
     delay(80);
     gotoxy(33,12);cout<<"proses . . . ."<<i<<"%";
     }
     clrscr();

     gotoxy(26,1);cout<<"================================
===";
     gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
     gotoxy(26,3);cout<<"|| Jual beli PC baru dan bekas
||";

     gotoxy(26,4);cout<<"================================
===";
 cout<<endl;
 cout<<endl;
 cout<<"============================\n";
          cout<<"Spec PC anda: \n\n";
          cout<<"Processor\t: "<<p<<" "<<c<<" GHz"<<endl;
          cout<<"VGA\t\t: "<<"Nvidia GTX"<<" "<<vs<<endl;
          cout<<"RAM\t\t: "<<r<<" GB"<<endl;
 }
 if(c<=1.6 && v<=25 && vs<=610 && r<=2){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                         cout<<endl;
```

```
 cout<<endl;
 cout<<"Processor\t: BAD\n\n";
 cout<<"VGA\t\t: BAD\n\n";
 cout<<"RAM\t\t: BAD\n\n";
 cout<<endl;
 cout<<"> PC anda tidak mampu untuk menjalankan game
ini\n";
 }
     else{}
          }
 if(c>1.6 && v<=25 && vs<=610 && r<=2){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                         cout<<endl;
 cout<<endl;
 cout<<"Processor\t: OKE\n\n";
 cout<<"VGA\t\t: LOW\n\n";
 cout<<"RAM\t\t: LOW\n\n";
 cout<<endl;
 cout<<"> Processor bisa berjalan dengan baik\n";
 cout<<"> Permasalahan ada pada VGA dan RAM yang
kurang\n";
 cout<<"Saran :\n";
 cout<<"> Upgrade VGA dan RAM PC anda";
 }
     else{}
          }
 if(c>1.6 && v<=25 && vs<=610 && r>=2){
          cout<<endl;
               cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                         cout<<endl;
 cout<<endl;
 cout<<"Processor\t: OKE\n\n";
 cout<<"VGA\t\t: LOW\n\n";
 cout<<"RAM\t\t: OKE\n\n";
 cout<<endl;
 cout<<"> Processor dan RAM bisa berjalan dengan baik\n";
 cout<<"> Permasalahan ada pada VGA yang kurang baik\n";
 cout<<"Saran :\n";
 cout<<"> Upgrade VGA PC anda";
 }
     else{}
          }
 if(c>1.6 && v>25 && vs>610 && r<2){
          cout<<endl;
```

```
                cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                          cout<<endl;
 cout<<endl;
 cout<<"Processor\t: OKE\n\n";
 cout<<"VGA\t\t: OKE\n\n";
 cout<<"RAM\t\t: LOW\n\n";
 cout<<endl;
 cout<<"> Processor dan VGA bisa berjalan\n";
 cout<<"> RAM PC anda kurang\n";
 cout<<"Saran :\n";
 cout<<"> Walaupun bisa untuk menjalankan game ini dengan
setting low,\n";
 cout<<" tetapi disarankan untuk upgrade RAM PC anda";
 }
     else{}
          }
 if(c<=1.6 && v>25 && vs>610 && r>=2){
          cout<<endl;
                cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                          cout<<endl;
 cout<<endl;
 cout<<"Processor\t: LOW\n\n";
 cout<<"VGA\t\t: OKE\n\n";
 cout<<"RAM\t\t: OKE\n\n";
 cout<<endl;
 cout<<"> Processor kurang mendukung\n";
 cout<<"Saran :\n";
 cout<<"> Ganti Processor PC anda";
 }
     else{}
          }
 if(c<=1.6 && v<=25 && vs<=610 && r>=2){
          cout<<endl;
                cout<<"Lanjut (y/n) : ";cin>>oke;
          if(oke=='y' || oke=='Y'){
                          cout<<endl;
 cout<<endl;
 cout<<"Processor\t: LOW\n\n";
 cout<<"VGA\t\t: LOW\n\n";
 cout<<"RAM\t\t: OKE\n\n";
 cout<<endl;
 cout<<"> Processor dan VGA tidak mendukung\n";
 cout<<"Saran :\n";
 cout<<"> Ganti Processor dan upgrade VGA PC anda";
 }
```

```
    else{}
         }
 if(c<=1.6 && v>25 && vs>610 && r<2){
         cout<<endl;
              cout<<"Lanjut (y/n) : ";cin>>oke;
         if(oke=='y' || oke=='Y'){
                        cout<<endl;
 cout<<endl;
 cout<<"Processor\t: LOW\n\n";
 cout<<"VGA\t\t: OKE\n\n";
 cout<<"RAM\t\t: LOW\n\n";
 cout<<endl;
 cout<<"> Processor dan RAM tidak mendukung\n";
 cout<<"Saran :\n";
 cout<<"> Ganti Processor dan upgrade RAM PC anda";
 }
    else{}
         }
 if(c>1.6 && v>25 && vs>610 && r>=2 && c<2.4 && v<=100 &&
vs<760 && r<3){
         cout<<endl;
              cout<<"Lanjut (y/n) : ";cin>>oke;
         if(oke=='y' || oke=='Y'){
                        cout<<endl;
 cout<<endl;
 cout<<"Processor\t: OKE\n\n";
 cout<<"VGA\t\t: OKE\n\n";
 cout<<"RAM\t\t: OKE\n\n";
 cout<<endl;
 cout<<"> PC anda bisa menjalankan game dengan graphic
disesuaikan\n";
 cout<<"> Akan lebih baik jika spec PC sesuai dengan
Recommended";
 }
    else{}
         }
 if(c>=2.4 && v>=50 && vs>=680 && r>=3){
         cout<<endl;
              cout<<"Lanjut (y/n) : ";cin>>oke;
         if(oke=='y' || oke=='Y'){
                        cout<<endl;
 cout<<endl;
 cout<<"Processor\t: GOOD\n\n";
 cout<<"VGA\t\t: GOOD\n\n";
 cout<<"RAM\t\t: GOOD\n\n";
 cout<<endl;
 cout<<"> PC anda bisa menjalankan game dengan lancar\n";
```

```
 }
     else{}
          }
 }
 else{}
}
void main(){
char p,a;
int lanjut,mot,cas,pro,v,r;
int pil,beli,admin,log;
awal();
do
{
     for(int i=0;i<=100;i+=3){
     delay(100);
     gotoxy(1,1);cout<<"loading . . . ."<<i<<"%";
 }
     clrscr();

gotoxy(26,1);cout<<"==================================";
 gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
 gotoxy(26,3);cout<<"|| Jual beli PC baru dan bekas ||";

gotoxy(26,4);cout<<"==================================";
 gotoxy(26,6);cout<<" Menu Utama";
 gotoxy(26,7);cout<<"==============";
 gotoxy(26,8);cout<<"[ 1 ] Beli";
 gotoxy(26,9);cout<<"[ 2 ] Jual";
 gotoxy(26,10);cout<<"[ 3 ] Login";
 gotoxy(26,11);cout<<"[ 4 ] Uji Spec PC";
 gotoxy(26,12);cout<<"[ 5 ] Keluar";
 gotoxy(26,13);cout<<"==============";
 gotoxy(26,14);cout<<"pilih : ";cin>>pil;
     switch(pil){
          case 1:
 clrscr();

     gotoxy(26,1);cout<<"==================================
===";
          gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
          gotoxy(26,3);cout<<"|| Jual beli PC baru dan
bekas ||";

     gotoxy(26,4);cout<<"==================================
===";
          gotoxy(26,6);cout<<"======================";
               gotoxy(26,7);cout<<"[ 1 ] Lihat iklan";
```

```
               gotoxy(26,8);cout<<"[ 2 ] Beli PC
rakitan";
 gotoxy(26,9);cout<<"[ 3 ] Menu utama";
          gotoxy(26,10);cout<<"======================";
          gotoxy(26,12);cout<<"pilih : ";cin>>beli;
     if(beli==1){
          lihat();
     }
          else if(beli==2){
                         clrscr();

     gotoxy(26,1);cout<<"================================
===";
                         gotoxy(26,2);cout<<"|| KOMPUTER
MART ||";
                         gotoxy(26,3);cout<<"|| Jual beli
PC baru dan bekas ||";

     gotoxy(26,4);cout<<"================================
===";
                         gotoxy(26,6);cout<<"lanjut untuk
membeli? (y/n) : ";cin>>p;
                         if(p=='y' || p=='Y'){
                         for(lanjut=1;lanjut<=5;lanjut++)
                         {
                         clrscr();

     gotoxy(26,1);cout<<"================================
===";
                         gotoxy(26,2);cout<<"|| KOMPUTER
MART ||";
                         gotoxy(26,3);cout<<"|| Jual beli
PC baru dan bekas ||";

     gotoxy(26,4);cout<<"================================
===";
                         if(lanjut==1){
                         gotoxy(3,6);cout<<"MOTHERBOARD";
                         gotoxy(3,7);cout<<"-----------";
                         gotoxy(3,9);cout<<" 1. ASUS
Motherboard Socket LGA1155 : Rp.920.000";
                         gotoxy(3,10);cout<<" 2. ASROCK
Motherboard Socket LGA1155 : Rp.1.550.000";
                         gotoxy(3,11);cout<<" 3. BIOSTAR
Motherboard Socket FM1 : Rp.860.000";
                         gotoxy(3,13);cout<<"Pilih :
";cin>>mot;
```

```
                        if(mot==1){
                             ra.motherboard=920000;
                             }
                        if(mot==2){
                             ra.motherboard=1550000;
                             }
                        if(mot==3){
                             ra.motherboard=860000;
                             }
                    }
                   else if(lanjut==2){
                        gotoxy(3,6);cout<<"CASING";
                        gotoxy(3,7);cout<<"------";
                        gotoxy(3,9);cout<<" 1. Dazumba
DE-320 : Rp.200.000";
                        gotoxy(3,10);cout<<" 2. Casing
Dazumba Dvito 911 : Rp.650.000";
                        gotoxy(3,11);cout<<" 3. Dazumba
Dvito 782 : Rp.430.000";
                        gotoxy(3,12);cout<<" 4. Casing
Dazumba DE-610 : Rp.240.000";
                        gotoxy(3,14);cout<<"Pilih :
";cin>>cas;
                        if(cas==1){
                             ra.casing=200000;
                             }
                        if(cas==2){
                             ra.casing=650000;
                             }
                        if(cas==3){
                             ra.casing=430000;
                             }
                         if(cas==4){
                                   ra.casing=240000;
                             }
                    }
                   else if(lanjut==3){
                        gotoxy(3,6);cout<<"PROCESSOR";
                        gotoxy(3,7);cout<<"---------";
                        gotoxy(3,9);cout<<" 1. AMD Llano
[A8-3850] : Rp.1.300.000";
                        gotoxy(3,10);cout<<" 2. AMD
Llano [A6-3500] : Rp.800.000";
                        gotoxy(3,11);cout<<" 3. AMD
Llano [A4-3300] : Rp.550.000";
                        gotoxy(3,12);cout<<" 4. AMD
Bulldozer[FX 4130] : Rp 1.440.000";
```

```
                        gotoxy(3,13);cout<<" 5. AMD
Phenom II [X2 555] : Rp 1.100.000";
                        gotoxy(3,14);cout<<" 6. AMD
opteron[2214] : Rp 1.700.000";
                        gotoxy(3,15);cout<<" 7. Intel
Core i7 3820 : Rp 4.200.000";
                        gotoxy(3,16);cout<<" 8. Intel
Core i7-3700K : Rp 4.600.000";
                        gotoxy(3,17);cout<<" 9. Intel
Core i5-4430 : Rp 2.600.000";
                        gotoxy(3,18);cout<<" 10. Intel
Core i5-3570 : Rp 2.900.000";
                        gotoxy(3,19);cout<<" 11. Intel
Core i3-3220 : Rp 1.600.000";
                        gotoxy(3,21);cout<<"Pilih :
";cin>>pro;
                        if(pro==1){
                             ra.processor=1300000;
                             }
                        if(pro==2){
                             ra.processor=800000;
                             }
                        if(pro==3){
                             ra.processor=550000;
                             }
                         if(pro==4){
                                  ra.processor=1440000;
                             }
                         if(pro==5){
                                  ra.processor=1100000;
                             }
                         if(pro==6){
                                  ra.processor=1700000;
                             }
                         if(pro==7){
                                  ra.processor=4200000;
                             }
                         if(pro==8){
                                  ra.processor=4600000;
                             }
                         if(pro==9){
                                  ra.processor=2600000;
                             }
                         if(pro==10){
                                  ra.processor=2900000;
                             }
                         if(pro==11){
```

```
                                        ra.processor=1600000;
                                   }
                              }
                              else if(lanjut==4){
                                   gotoxy(3,6);cout<<"VGA CARD";
                                   gotoxy(3,7);cout<<"--------";
                                   gotoxy(3,9);cout<<" 1. Geforce
GT 640 1GB : Rp.980.000";
                                   gotoxy(3,10);cout<<" 2. Geforce
GT 620 1GB : Rp.640.500";
                                   gotoxy(3,11);cout<<" 3. Geforce
GTX 690 4GB : Rp.12.100.000";
                                   gotoxy(3,12);cout<<" 4. Geforce
GTX 680 2GB : Rp.6.700.000";
                                   gotoxy(3,13);cout<<" 5. Geforce
GTX 670 4GB : Rp 5.500.000";
                                   gotoxy(3,14);cout<<" 6. Geforce
GTX 570 1280MB : Rp 3.400.000";
                                   gotoxy(3,15);cout<<" 7. Geforce
GTX 440 1GB : Rp 1.200.000";
                                   gotoxy(3,16);cout<<" 8. Radeon
HD 7970 3GB : Rp 6.300.000";
                                   gotoxy(3,17);cout<<" 9. Asus
Radeon HD 7970 3GB : Rp 5.500.000";
                                   gotoxy(3,18);cout<<" 10. Radeon
HD 7770 1GB : Rp 1.700.000";
                                   gotoxy(3,19);cout<<" 11. Radeon
HD 6670 2GB : Rp 990.000";
                                   gotoxy(3,20);cout<<" 12. Radeon
HD 6570 1GB : Rp 620.000";
                                   gotoxy(3,22);cout<<"Pilih :
";cin>>v;
                                   if(v==1){
                                        ra.vga=980000;
                                        }
                                   if(v==2){
                                        ra.vga=640500;
                                        }
                                   if(v==3){
                                        ra.vga=12100000;
                                        }
                                    if(v==4){
                                              ra.vga=6700000;
                                        }
                                    if(v==5){
                                              ra.vga=550000;
                                        }
```

```
                              if(v==6){
                                        ra.vga=3400000;
                                   }
                              if(v==7){
                                        ra.vga=1200000;
                                   }
                              if(v==8){
                                        ra.vga=6300000;
                                   }
                              if(v==9){
                                        ra.vga=5500000;
                                   }
                              if(v==10){
                                        ra.vga=1700000;
                                   }
                              if(v==11){
                                        ra.vga=990000;
                                   }
                              if(v==12){
                                        ra.vga=620000;
                                   }
                         }
                        else if(lanjut==5){
                              gotoxy(3,6);cout<<"RAM";
                              gotoxy(3,7);cout<<"---";
                              gotoxy(3,9);cout<<" 1. V-GEN
Memory 8GB DDR3 : Rp.990.000";
                              gotoxy(3,10);cout<<" 2. V-GEN
Memory 4GB DDR3 : Rp.560.000";
                              gotoxy(3,11);cout<<" 3. V-GEN
Memory 2GB DDR3 : Rp.340.000";
                              gotoxy(3,13);cout<<"Pilih :
";cin>>r;
                              if(r==1){
                                   ra.ram=990000;
                                   }
                              if(r==2){
                                   ra.ram=560000;
                                   }
                              if(r==3){
                                   ra.ram=340000;
                                   }
                         }
                         }
           clrscr();

gotoxy(26,1);cout<<"==================================";
```

```
           gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
           gotoxy(26,3);cout<<"|| Jual beli PC baru dan
bekas ||";

gotoxy(26,4);cout<<"==================================";
           gotoxy(26,7);cout<<"1.Motherboard :
Rp."<<ra.motherboard;
           gotoxy(26,8);cout<<"2.Casing :
Rp."<<ra.casing;
           gotoxy(26,9);cout<<"3.Processor :
Rp."<<ra.processor;
           gotoxy(26,10);cout<<"4.VGA Card :
Rp."<<ra.vga;
           gotoxy(26,11);cout<<"5.RAM : Rp."<<ra.ram;

ra.total=ra.motherboard+ra.casing+ra.processor+ra.vga+ra.
ram;
           gotoxy(26,13);cout<<"Total bayar
Rp."<<ra.total;
           gotoxy(26,15);cout<<" TERIMA HASIH";
           gotoxy(26,16);cout<<" ============";
               gotoxy(26,18);cout<<"Ingin keluar? (y/n) :
";cin>>a;
               if(a=='y' || a=='Y'){
                    exit(0);
                    }
               else{}
      }
          else{}
                         }
                    else{}
               break;
     case 2:jual();
          break;
     case 3:
          clrscr();
          gotoxy(32,12);cout<<"Selamat datang admin";
          gotoxy(31,14);cout<<"password : ";cin>>log;
          if(log==0){
               clrscr();

     gotoxy(26,1);cout<<"==============================
===";
               gotoxy(26,2);cout<<"|| KOMPUTER MART ||";
               gotoxy(26,3);cout<<"|| Jual beli PC baru
dan bekas ||";
```

```
     gotoxy(26,4);cout<<"================================
===";
                gotoxy(26,6);cout<<"Selamat datang Admin";
                gotoxy(26,7);cout<<"====================";
                gotoxy(26,8);cout<<"Penghasilan :
Rp."<<ra.total;
                gotoxy(26,9);cout<<"[ 1 ] Hapus data
iklan";
 gotoxy(26,10);cout<<"[ 2 ] Logout";

     gotoxy(26,11);cout<<"======================";
                gotoxy(26,13);cout<<"Pilih : ";cin>>admin;
                if(admin==1){
                     hapus();
                     }
                else{}
          }
          else {
                clrscr();
                gotoxy(32,12);cout<<"password salah";
          }
          break;
 case 4:game();
     break;
     case 5:exit(0);
          break;
     default:
     clrscr();
          gotoxy(33,12);cout<<"Inputan tidak tersedia";
     awal();
 }
getch();
}
while(pil!=4);
}
```

16.11. Program Random dengan Array

```
#include <iostream>
#include <stdlib.h>
using namespace std;

int main() {
     int Random;
     int Input;
```

```
     int A[100];

ulang:
     cout << "Input batas elemen array : ";
     cin >> Input;
     if(Input >= 1 && Input <= 100)
     {
          for(int i=0;i<Input;i++)
          {
            cout<<"A["<<i<<"]=";
            cin>>A[i];
          }
          for(int i = 0; i < Input; i++)
          {
            Random=(rand() % Input) ;
            cout << A[Random]<< '\n';
          }
    }
     else
     {
          cout<<"Batas array hanya sampai 100\n";
          goto ulang;
     }
return 0;
}
```

16.12. Soal-Soal Pilihan Ganda

1. Penulisan Preprocessor yang benar di awali dengan tanda pound atau tanda :
   a. # c. @
   b. & d. =

2. Contoh penulisan file header yang benar yaitu:
   a. &include <conio.h> c. =include <conio.h>
   b. #include <conio.h> d. *include <conio.h>

3. Cara lain untuk menuliskan file header #include <conio.h> adalah :
   a. #include {conio.h} c. #include "conio.h"
   b. #include (conio.h) d. #include [conio.h]

4. Fungsi **clrscr ()** merupakan fungsi miliknya file header :
   a. #include <iostream.h> c. #include <stdio.h>
   b. #include <stdlib.h> d. #include <conio.h>

5. Perintah **cout<< dan cin>>** merupakan perintah miliknya file header :

a. #include <iostream.h> c. #include <stdio.h>
b. #include <stdlib.h> d. c. #include <conio.h>

6. Dalam bahasa pemrograman C++ untuk membuat komentar satu baris menggunakan :
   a. // c. \\
   b. || d. {}

7. Dalam bahasa pemrograman C++ untuk membuat komentar lebih dari satu baris menggunakan :
   a. // c. \\
   b. || d. /* … */

8. Untuk mendeklarasikan variabel dalam bahasa pemrograman C++ yaitu:
   a. int a,b; c. int a;b;
   b. int 1; d. char a<3>

9. Tipe data **Boolean** yaitu tipe data yang :
   a. Tipe data untuk numeric
   b. Tipe data untuk string
   c. Tipe data yang hanya memiliki dua nilai true dan false
   d. Tipe data untuk bilangan ganjil

10. Bagaimana cara untuk mendeklarasikan konstanta **phi = 3.14** :
    a. #define phi 3.14 c. #define float phi = 3.14;
    b. #define phi = 3.14; d. #define phi 3.14;

11. Untuk mendeklarasikan konstanta dalam C++ ada dua cara yaitu menggunakan :
    a. #define atau const c. #difine atau const
    b. #const atau define d. const atau define

12. Untuk menampilkan text “Selamat Datang” menggunakan perintah :
    a. Cout<<” Selamat Datang”;
    b. COUT<< “Selamat Datang”;
    c. cout<<”Selamat Datang”;
    d. cout>>”Selamat Datang”;

13. Untuk menginputkan data ke variabel menggunakan perintah :
    a. cout<< c. <<endl;
    b. getch() d. cin>>
14. Untuk pindah ke baris berikutnya menggunakan perintah :
    a. <<end; c. <<endl;
    b. <<\n d. \\nn

15. Manakah yang salah dari 4 pilihan di bawah ini :
    a. cout<<”Selamat Pagi Semua”<<end

b. cout<<endl<<"Selamat Pagi Semua";
c. cout<<"Selamat Pagi Semua"<<endl;
d. cout<<"\nSelamat Pagi Semua";

16. Perintah setiosflags(ios::fixed) << setprecision(2) digunakan untuk :
    a. Untuk menampilkan angka sebanyak 2 digit
    b. Untuk menampilkan angka desimal sebanyak 2 digit
    c. Untuk menampilkan angka bulat
    d. Untuk menampilkan angka desimal

17. Dari program di bawah ini, berapa hasil output dari variabel jwb :
    a. 12 dan 20
    b. 24 dan 30
    c. 12 dan 24
    d. 24 dan 24

```
#include <iostream.h>
#include <conio.h>
main(){
   int i  = 4;
   int j  = 8;
   int k = 12;
   int jwb;

   jwb = i + j;
   cout << jwb ;
   jwb += k;
   cout <<endl<< jwb;
   getch();}
```

18. Dari program di bawah ini apa outputnya jika kita inputkan nilai -5 :
    a. bilangan genap
    b. bilangan ganjil
    c. bilangan nol
    d. semua salah

```
#include<iostream.h>
#include<conio.h>
main(){
   int nilai;

   cout<<"Masukkan Nilai
=";cin>>nilai;
   if (nilai % 2 == 0)
      cout<<"bilangan
genap";
   else
      cout<<"bilangan
ganjil";
getch();}
```

19. Ada 2 percabangan di dalam C++ yaitu:
    a. if then else dan case of
    b. if else dan switch case
    c. if then dan switch
    d. if dan case

20. Apa maksud dari pesan kesalahan Statement missing ;
    a. Kurang ; pada akhir baris
    b. Kurang " pada akhir baris
    c. Kurang {}
    d. Deklarasi variabel salah

21. Apa maksud dari pesan kesalahan Compound Statement missing }
    a. Kurang ; pada akhir baris
    b. Kurang " pada akhir baris
    c. Kurang }
    d. Deklarasi variabel salah

22. Yang bukan termasuk perulangan yaitu:
    a. repeat until c. for
    b. while d. do while

23. Perulangan yang sudah diketahui batas perulangannya, merupakan perulangan :
    a. while c. do while
    b. goto d. For

24. Perulangan yang melakukan pengecekan kondisi di awal blok struktur, merupakan perulangan :
    a. while c. do while
    b. goto d. For

25. Perulangan yang melakukan pengecekan kondisi di akhir blok struktur, merupakan perulangan :
    a. while c. do while
    b. goto d. For

26. Berapa hasil outputnya :
    a. i=1 a=1 dan i=2 a=5
    b. i=5 a=1 dan i=4 a=2
    c. i=1 a=5 dan i=2 a=10
    d. i=2 a=1 dan i=5 a=5

```
#include <iostream.h>
#include <conio.h>

main() {
  for(int i=1,a=5;i<=2;i++)
  { cout<<"\ni = "<<i<<" a = "<<a;
    a+=5;
  }  getch();}
```

27. Berapa hasil outputnya :
    a. i=1 a=1 dan i=2 a=5
    b. i=5 a=1 dan i=5 a=2
    c. i=1 a=5 dan i=3 a=2
    d. i=3 a=5 dan i=1 a=2

```
#include <iostream.h>
#include <conio.h>

main() {
  for(int i=3,a=5;i>=0;i-=2)
  { cout<<"\ni  = "<<i<<" a = "<<a;
    a-=3;
  }getch();}
```

28. Berapa hasil outputnya, jika kita menginputkan angka 10 :
    a. 10 9 6 3 1
    b. 10 8 6 4 2
    c. 8 6 4 2 0
    d. 10 9 8 7 6 5 4 3 2 1

```
#include <iostream.h>
#include <conio.h>

main() {
  int b;
  cout<<"Masukkan batas = ";cin>>b;
  for(int i=b;i>=1;i=i-2)
     cout<<i<<"\n";
  getch();}
```

29. Berapa hasil outputnya, jika kita menginputkan angka 5 :
    a. 5 3 1
    b. 5 3 1 0
    c. 5 4 3 2 1
    d. 1 2 3 4 5

```
#include <iostream.h>
#include <conio.h>

main() {
  int i=5;
  while (i>=1)
  { cout<<"\n"<<i;
    i-=2;
  }
  getch();}
```

30. Berapa hasil outputnya dari program di bawah ini:
    a. i=1 a=3 dan i=5 a=15
    b. i=1 a=5 dan i=3 a=15
    c. i=1 a=1 dan i=5 a=15

d. i=1 a=1 dan i=2 a=2

```
#include <iostream.h>
#include <conio.h>

main() {
  int a=5,i=1;
  do
  { cout<<"\ni = "<<i<<" a = "<<a;
    a+=10;
    i+=2;
  }
  while (i<=3);
  getch();}
```

31. Sub program yang berguna untuk menjadikan program dapat lebih besifat modular sehingga akan mudah dipahami dan dapat digunakan kembali, baik untuk program itu sendiri maupun program lain yang memiliki proses yang sama, pengertian dari
    a. fungsi c. array
    b. pointer d. Looping

32. Tanda Desimal dalam bahasa C++ adalah …
    a. Titik (.) c. Koma (,)
    b. Kutip (‘) d. Titik koma (;)

33. Tempat menampung data disebut …
    a. Konstanta c. Relasi
    b. Variabel d. Integer

34. Symbol operator logika “OR” yaitu:
    a. // c. &&
    b. || d. <<

35. Symbol operator logika “AND” yaitu:
    a. // c. &&
    b. || d. <<

36. Symbol operator tidak sama dengan dilmbangkan dengan :
    a. =! c. <>=
    b. != d. ==

37. Untuk memilih jalur proses, gunakan fungsi …
    a. If c. For
    b. Switch d. Array

38. Memilih satu dari sejumlah alternative, adalah fungsi …
    a. Array c. Switch

b. For d. If

39. Melakukan sejumlah iterasi yang telah ditetapkan sebelumnya adalah Fungsi dari …
    a. Array c. Fungsi
    b. Switch d. For
40. Variabel yang memiliki tipe data dan nama yang sama/homogen tetapi dengan index yang berbeda, adalah pengertian dari …
    a. Array c. For
    b. Fungsi d. Structure

41. Ada 3 komponen penting dalam array yaitu, kecuali :
    a. Index c. Reference
    b. Value d. Null

42. Index pada Array adalah …
    a. Jumlah masukan c. Memory Array
    b. Banyaknya Array d. Variabel

43. Berikut pendeklarasian array kecuali :
    a. int A[3]; c. char hari[7][10];
    b. int X[]; d. int Y[]={1,4,6};

44. Berikut yang bukan termasuk jenis fungsi dalam bahasa C++ yaitu:
    a. void cetak();
    b. int jumlah(int a,int b);
    c. void jumlah(int &jml, int a,int b);
    d. string cetak()

45. Berikut adalah fungsi yang tidak mengembalikan nilai atau return value, kecuali :
    a. int jumlah(int a,int b);
    b. void jumlah(int &jml,int a,int b);
    c. void jumlah(int *jml,int a,int b);
    d. void jumlah(int A[],int n);

DAFTAR PUSTAKA

Amborowati, Armadyah,. Pengantar Pemrograman Terstruktur , ANDI, 2007.

IlmuKomputer.Com

Raharjo, Budi., *Pemrograman C++ mudah dan cepat menjadi master C++*. Informatika, 2007

www.cplusplus.com

Yatini, Indra., *Pemrograman Terstruktur*, J&J Learning Yogyakarta, 2001