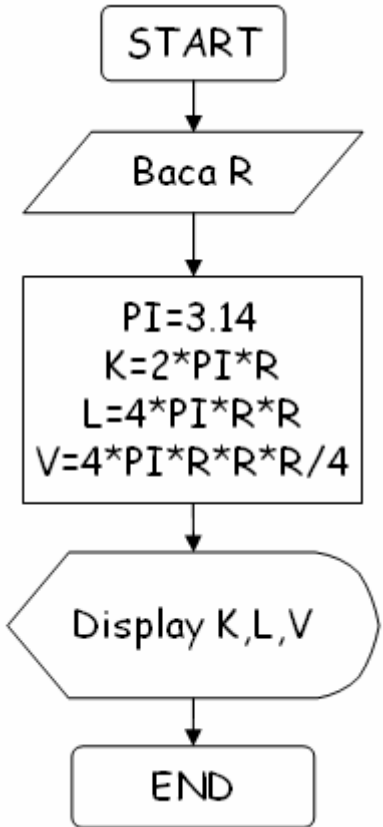


MODUL 5 - 6

I. RUNTUNAN(SEQUENCE)

Algoritma runtunan (sequence) :

- ❖ Tiap instruksi dikerjakan satu per satu
- ❖ Tiap instruksi dilaksanakan tepat sekali, tidak ada instruksi yang diulang
- ❖ Urutan instruksi yang dilaksanakan pemroses sama dengan urutan instruksi sebagaimana yang tertulis didalam teks algoritmanya
- ❖ Akhir dari instruksi terakhir merupakan akhir algoritma

 <pre> graph TD START([START]) --> BacaR[/Baca R/] BacaR --> Proses[PI=3.14 K=2*PI*R L=4*PI*R*R V=4*PI*R*R*R/4] Proses --> Display[/Display K,L,V/] Display --> END([END]) </pre>	<p><u>ALGORITMA</u> Bola {Menghitung Luas, Keliling, Volume Bola}</p> <p><u>DEKLARASI</u> R,K,L,V : <u>float</u></p> <p><u>DESKRIPSI</u> <u>read</u>(R) PI←3.14 K←2*PI*R L←4*PI*R*R V←4*PI*R*R*R/4 <u>write</u>(K,L,V)</p>	<pre> #include<iostream.h> void main(void) { float PI=3.14,R,L,K,V; cout<<"Masukan harga R : "<<endl; cin>>R; K=2*PI*R; L=4*PI*R*R; V=4*PI*R*R*R/4; cout<<"Keliling : "<<K<<endl; cout<<"Luas : "<<L<<endl; cout<<"Volume : "<<V<<endl; } </pre>
---	--	--

Buat Algoritma dan program untuk mempertukarkan dua buah nilai variabel A dan B.

II. PEMILIHAN/PENGAMBILAN KEPUTUSAN

Ekspresi Boolean adalah ekspresi yang menghasilkan nilai yang dapat berupa :

- ❖ Benar (true) atau salah (false)
- ❖ Bukan nol atau nol

Kondisi : suatu pernyataan atau ungkapan atau ekspresi yang mengandung nilai benar (TRUE) atau salah (FALSE).

1. Operator Pembandingan/relasional dalam C/C++

Operator	Makna	Contoh
==	Kesamaan	a==b
!=	Ketidaksamaan	a!=b
>	Lebih dari	a>b
<	Kurang dari	a=	Lebih dari atau sama dengan	a>=b
<=	Kurang dari atau sama dengan	a<=b

9 > 5 contoh ungkapan yg bernilai TRUE
 0 contoh ungkapan yg bernilai FALSE
 5 contoh ungkapan yg bernilai TRUE
 N > 60 dapat bernilai TRUE atau FALSE

2. Operator Logika dalam C/C++

Operator	Makna	Contoh
&&	dan	a==b&&c!=d
	atau	a==b c!=d
!	bukan	!(a==b)

opr_1 && opr_2		
opr_1	opr_2	Hasil
salah	salah	salah
salah	benar	salah
benar	salah	salah
benar	benar	benar

opr_1 opr_2		
opr_1	opr_2	Hasil
salah	salah	salah
salah	benar	benar
benar	salah	benar
benar	benar	benar

3. Pemilihan 1 kasus (IF-THEN)

Pseudo-code	Flow chart	C/C++
<pre> if <u>kondisi</u> <u>then</u> pernyataan 1 pernyataan 2 endif </pre>	<pre> graph TD Entry(()) --> Kondisi{kondisi} Kondisi -- benar --> BlokPernyataan[blok pernyataan] Kondisi -- salah --> Connector((X)) BlokPernyataan --> Connector Connector --> Exit(()) </pre>	<pre> if(kondisi) { //blok pernyataan yang //dijalankan jika kondisi benar } </pre>

Contoh :

Buatlah algoritma dan program C++ yang membaca sebuah bilangan bulat dari keyboard, lalu mencetak pesan "genap" jika bilangan tersebut merupakan bilangan genap

Jawab :

Untuk mengetahui suatu bilangan itu genap atau bukan dapat dilakukan dengan cara membagi bilangan tersebut dengan 2. Jika sisa pembagiannya sama dengan 0 maka bilangan tersebut adalah genap. Algoritma dan code programnya sbb:

Pseudo-code	Flow chart	Kode C++
<p><u>ALGORITMA</u> bilGenap {mencetak pesan genap jika bilangan yang dimasukan dari keyboard merupakan bilangan genap}</p> <p><u>DEKLARASI</u> x : <u>integer</u></p> <p><u>DESKRIPSI</u> <u>read</u>(x) <u>if</u> x <u>mod</u> 2 = 0 <u>then</u> <u>write</u>('genap') <u>endif</u></p>	<pre> graph TD START([START]) --> Baca_x[/Baca x/] Baca_x --> s_x_mod_2[s = x mod 2] s_x_mod_2 --> s_eq_0{s == 0} s_eq_0 -- salah --> Join((X)) s_eq_0 -- benar --> genap{{'genap'}} genap --> Join Join --> END([END]) </pre>	<pre> //Penggunaan if #include<iostream.h> void main(void) { int x,s; cout<<"Masukan bilangan bulat : "; cin>>x; s=x%2; if(s==0) cout<<"Bilangan Genap"<<endl; } </pre>

4. Pemilihan 2 kasus(IF-THEN-ELSE)

Pseudo-code	Flow chart	C/C++
<pre> if <u>kondisi</u> <u>then</u> pernyataan 1 else pernyataan 2 endif </pre>	<pre> graph TD Entry(()) --> Kondisi{kondisi} Kondisi -- benar --> BlokIf[blok pernyataan if] Kondisi -- salah --> BlokElse[blok pernyataan else] BlokIf --> Connector((X)) BlokElse --> Connector Connector --> Exit(()) </pre>	<pre> if(kondisi) { //blok pernyataan yang //dijalankan jika kondisi benar } else { //blok pernyataan yang //dijalankan jika kondisi salah } </pre>

Contoh :

Buatlah algoritma dan code C++ untuk menentukan kelulusan suatu mata ujian. Dinyatakan Lulus jika nilai ujian lebih besar atau sama dengan 60.

Jawab :

Pseudo-code	Flow chart	Kode C++
<p>ALGORITMA Lulus {mencetak pesan Lulus jika nilai ujian ≥ 60}</p> <p>DEKLARASI x : integer</p> <p>DESKRIPSI <u>read</u>(x) <u>if</u> $x \geq 60$ <u>then</u> <u>write</u>('Lulus') <u>else</u> <u>write</u>('Tidak Lulus') <u>endif</u></p>	<pre> graph TD START([START]) --> Baca_x[/Baca x/] Baca_x --> Decision{x >= 60} Decision -- salah --> Tidak_Lulus[/Tidak Lulus/] Decision -- benar --> Lulus[/Lulus/] Tidak_Lulus --> Junction((X)) Lulus --> Junction Junction --> END([END]) </pre>	<pre> //Penggunaan if-else #include<iostream.h> void main(void) { int NilaiUjian; cout<<"Masukan Nilai : "; cin>>NilaiUjian; if(NilaiUjian >= 60) cout<<"Lulus"<<endl; else cout<<"Tidak Lulus"<<endl; } </pre>

5. Pemilihan tiga kasus atau lebih/Pernyataan IF bersarang (nested IF)

Merupakan pernyataan IF dalam IF.

Contoh penggunaan IF bersarang yaitu untuk menentukan nilai suatu ujian tergolong sebagai A, B, C, D, E dengan kriteria sebagai berikut :

Nilai	Skor
$\text{Nilai} \geq 90$	A
$80 \leq \text{Nilai} < 90$	B
$60 \leq \text{Nilai} < 80$	C
$50 \leq \text{Nilai} < 60$	D
$\text{Nilai} < 50$	E

```
#include<iostream.h>
void main(void)
{
    double NilaiUjian;
    char skor;
    cout<<"Masukan Nilai Ujian : ";
    cin>>NilaiUjian;
    if(NilaiUjian >= 90)
        skor='A';
    else
        if(NilaiUjian >= 70)
            skor='B';
        else
            if(NilaiUjian >= 60)
                skor='C';
            else
                if(NilaiUjian >= 50)
                    skor='D';
                else
                    skor='E';
    cout<<"Skor : "<<skor<<endl;
}
```

Jika salah satu *if* sudah terpenuhi maka program akan langsung mengeksekusi *cout*.

} if pertama

} if kedua

} if ketiga

} if keempat

Flow chart	Struktur C/C++
<pre> graph TD Start(()) --> Kond1{kond1} Kond1 -- T --> S1[s1] Kond1 -- F --> Kond3{kond3} S1 --> Kond2{kond2} Kond2 -- T --> S2[s2] Kond2 -- F --> S3[s3] S2 --> Merge1((X)) S3 --> Merge1 Merge1 --> Merge2((X)) Kond3 -- T --> S4[s4] Kond3 -- F --> S5[s5] S4 --> Merge2 S5 --> Merge2 Merge2 --> End(()) </pre>	<pre> if(kond1) { //awal kond1 TRUE s1; if(kond2) { //awal kond2 TRUE s2; } //akhir kond2 TRUE else { //awal kond2 FALSE s3; } // akhir kond2 FALSE //akhir kond1 TRUE } else { //awal kond1 FALSE if(kond3) { //awal kond3 TRUE s4; } //akhir kond3 TRUE else { //awal kond3 FALSE s5; } //akhir kond3 FALSE s6; } //akhir kond1 FALSE </pre>

Flow chart	Struktur C/C++
<pre> graph TD Start(()) --> Kond1{kond1} Kond1 -- T --> Kond2{kond2} Kond1 -- F --> Conn1(()) Kond2 -- T --> Kond3{kond3} Kond2 -- F --> Conn2(()) Kond3 -- T --> Kond4{kond4} Kond3 -- F --> Conn3(()) Kond4 -- T --> S1[s1] Kond4 -- F --> Conn4(()) S1 --> Conn4 Conn1 --> Exit(()) Conn2 --> Conn1 Conn3 --> Conn2 Conn4 --> Conn3 Exit --> End(()) </pre>	<pre> if(kond1) { //awal kond1 TRUE if(kond2) { //awal kond2 TRUE if(kond3) { //awal kond3 TRUE if(kond4) { //awal kond4 TRUE s1; } //akhir kond4 TRUE //akhir kond3 TRUE } //akhir kond2 TRUE } //akhir kond1 TRUE } </pre>

Flow chart	Struktur C/C++
<pre> graph TD Start(()) --> Kond1{kond1} Kond1 -- T --> S1[s1] Kond1 -- F --> Kond2{kond2} S1 --> Merge1((X)) Kond2 -- T --> S2[s2] Kond2 -- F --> Kond3{kond3} S2 --> Merge2((X)) Kond3 -- T --> S3[s3] Kond3 -- F --> Kond4{kond4} S3 --> Merge3((X)) Kond4 -- T --> S4[s4] Kond4 -- F --> S5[s5] S4 --> Merge4((X)) S5 --> Merge4 Merge4 --> Merge3 Merge3 --> Merge1 Merge1 --> End(()) </pre>	<pre> if(kond1) { s1; } else { if(kond2) { s2; } else { if(kond3) { s3; } else { if(kond4) { s4; } else { s5; } } } } </pre> <p> //awal kond1 TRUE //akhir kond1 TRUE //awal kond1 FALSE //awal kond2 TRUE //akhir kond2 TRUE //awal kond2 FALSE //awal kond3 TRUE //akhir kond3 TRUE //awal kond3 FALSE //awal kond4 TRUE //akhir kond4 TRUE //awal kond4 FALSE //akhir kond4 FALSE //akhir kond3 FALSE //akhir kond2 FALSE //akhir kond1 FALSE </p>

6. Operator berkondisi(?:)

Disebut juga operator *ternary* karena melibatkan 3 buah argumen.



```
//Terbesar.cpp
#include<iostream.h>
void main(void)
{
    int x,y,maks;
    cout<<"Masukan sebuah bilangan : ";
    cin>>x;
    cout<<"Masukan lagi sebuah bilangan : ";
    cin>>y;
    maks = x > y ? x : y;
    cout<<"Terbesar adalah : "<<maks;endl;
}
```

7. Struktur Case

Untuk masalah dengan dua kasus atau lebih, penggunaan CASE dapat lebih menyederhanakan penulisan.

```
case ekspresi
    nilai_1 : pernyataan_1
    nilai_2 : pernyataan_2
    .....
    nilai_n : pernyataan_n
    otherwise : pernyataan_x
endcase
```

Flow chart	Kode C++
<pre> graph LR Ekspresi([ekspresi]) --> J1(()) J1 -- Nilai_1 --> P1[Pernyataan 1 break] J1 -- Nilai_2 --> P2[Pernyataan 2 break] J1 -- Nilai_3 --> P3[Pernyataan 3 break] J1 -- Tdk ada yg cocok --> P4[Bagian default dijalankan] P1 --> J2(()) P2 --> J2 P3 --> J2 P4 --> J2 J2 --> End((X)) </pre>	<pre> switch(ekspresi) { case nilai_1 : pernyataan_1 ; break; case nilai_2 : pernyataan_2 ; break; case nilai_3 : pernyataan_3 ; break; default : pernyataan_n ; } </pre>

Pada pernyataan switch :

- ❖ masing-masing pernyataan (1 sd n) dapat berupa satu atau beberapa perintah dan tidak perlu berupa blok pernyataan
- ❖ pernyataan_1 dijalankan kalau nilai ekspresi dengan nilai_1. Setelah dijalankan eksekusi dilanjutkan ke akhir pernyataan switch
- ❖ begitu juga dengan pernyataan lainnya dijalankan jika nilai ekspresi sesuai dengan nilai bs.
- ❖ bagian default bersifat opsional. Jika ada, bagian

ini dijalankan jika nilai ekspresi tidak ada yang cocok dengan nilai_1, nilai_2,...,nilai_n.

- ❖ nilai kondisi hanya dapat berupa tipe primitif
- ❖ pernyataan break digunakan untuk mengendalikan eksekusi ke akhir pernyataan switch.

Contoh : mencetak nama hari berdasarkan nomornya

Algoritma	Kode C++
<p><u>ALGORITMA</u> hari</p> <p>{Mencetak nama hari berdasarkan nomor (1 ..7)}</p> <p><u>DEKLARASI</u></p> <p>NomorHari : <u>integer</u></p> <p><u>DESKRIPSI</u></p> <p><u>read</u>(NomorHari)</p> <p><u>case</u> NomorHari</p> <p>1 : <u>write</u>('Minggu')</p> <p>2 : <u>write</u>('Senin')</p> <p>3 : <u>write</u>('Selasa')</p> <p>4 : <u>write</u>('Rabu')</p> <p>5 : <u>write</u>('Kamis')</p> <p>6 : <u>write</u>('Jumat')</p> <p>7 : <u>write</u>('Sabtu')</p> <p>otherwisw : <u>write</u>('Salah Nomor')</p> <p><u>endcase</u></p>	<pre>#include<iostream.h> void main(void) { int NomorHari; cout<<"Masukan Nomor Hari : "<<endl; cin>>NomorHari; switch(NomorHari) { case 1: cout<<"Minggu"<<endl; break; case 2: cout<<"Senin"<<endl; break; case 3: cout<<"Selasa"<<endl; break; case 4: cout<<"Rabu"<<endl; break; case 5: cout<<"Kamis"<<endl;</pre>

```

        break;
    case 6:
        cout<<"Jumat"<<endl;
        break;
    case 7:
        cout<<"Sabtu"<<endl;
        break;
    default :
        cout<<"Salah Nomor"<<endl;
    }
}

```

Tugas:

1. Buatlah algoritma dan program C++ untuk menghitung luas, keliling, panjang diagonal persegi panjang, dengan tampilan sbb (masukan : panjang dan lebar) :

```

=====
MENU EMPAT PERSEGI PANJANG
1. Hitung Luas
2. Hitung Keliling
3. Hitung Panjang Diagonal
4. Keluar Program
=====
Pilih Nomor :

```

2. Buatlah algoritma dan program C++ untuk menghitung upah mingguan karyawan. Masukan yang dibaca adalah nama karyawan, golongan, dan jumlah jam kerja. Keluaran program adalah nama karyawan dan upahnya.

Ketentuan :

jam kerja normal = 48 jam

upah per jam :

Golongan A : Rp. 4000

Golongan B : Rp. 5000

Golongan C : Rp. 6000

Golongan D : Rp. 7000

upah lembur : Rp.3000/jam