

PERTEMUAN 4

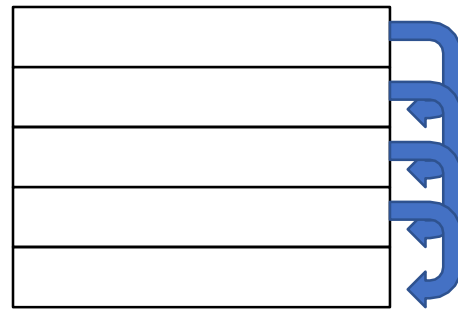
Runtunan (Sequential) dan Pemilihan (Selection)

Hermanto, S.Kom. M.Kom.



Runtunan (Sequential)

Algoritma Sekuensial merupakan algoritma yang langkah-langkahnya secara urut dari awal hingga akhir.



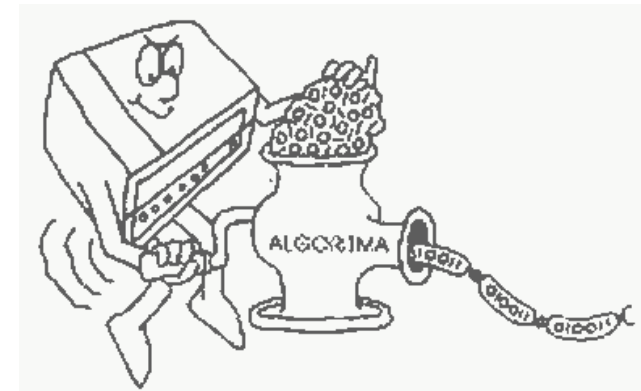
Runtutan

Sebuah runtutan terdiri dari satu atau lebih intruksi. Setiap intruksi dapat ditulis dalam satu baris atau beberapa intruksi ditulis dalam satu baris dan masing-masingnya dipisahkan dengan tanda koma atau tanda baca lainnya.

Runtunan (Sequential)

Pada dasarnya algoritma merupakan runtutan satu atau lebih intruksi, yang berarti bahwa :

1. Tiap intruksi dikerjakan satu persatu.
2. Tiap intruksi dilaksanakan tepat sekali, tidak ada intruksi yang diulang
3. Urutan intruksi yang dilaksanakan pemroses sama dengan urutan sebagaimana yang tertulis di dalam teks algoritmanya
4. Akhir dari intruksi terakhir merupakan akhir dari algoritma



Runtunan (Sequential)

Tinjau algoritma untuk menghitung harga barang di supermarket atau pasar swalayan setelah mendapat diskon sebesar p%.

ALGORITMA Menghitung harga barang setelah diskon :

1. input harga barang
 2. input p
 3. Hitung potongan harga = harga barang * p
 4. Hitung harga barang setelah diskon = harga barang – potongan harga
 5. Tulis harga barang setelah diskon
-

Menghitung Gaji Karyawan

Seorang karyawan menerima gaji berupa gaji pokok dan tunjangan. Penghasilan tersebut harus di kurangi lagi dengan besar pajak. Tulislah algoritma yang membaca nama karyawan dan gaji pokok bulanannya, lalu menghitung gaji bersih karyawan tersebut, dan menampilkan nama karyawan beserta gaji bersihnya. Gaji bersih yang diterima pegawai adalah :

Gaji bersih = gaji pokok + tunjangan – pajak

Tunjangan karyawan dihitung 20% dari gaji pokok, sedangkan pajak adalah 15% dari gaji pokok ditambah tunjangan. Gaji bersih karyawan dicetak kelayar.

Menghitung Gaji Karyawan

PROGRAM *GajiBersihKaryawan*

{Menghitung gaji bersih karyawan, data masukan adalah nama karyawan dan gaji pokok bulanannya. Gaji bersih = gaji pokok + tunjangan - pajak. Tunjangan adalah 20% dari gaji pokok, sedangkan pajak adalah 15% dari gaji pokok. Luarannya adalah nama karyawan dan gaji bersihnya}

DEKLARASI :

const PersenTunjangan = 0.2 {persentasi tunjangan gaji}

const PersenPajak = 0.15 {persentasi potongan pajak}

NamaKaryawan : **string**

GajiPokok, tunjangan, pajak, GajiBersih : **float**

ALGORITMA :

cin (NamaKaryawan, GajiPokok)

tunjangan ← PersenTunjangan * GajiPokok

pajak ← PersenPajak * (GajiPokok + tunjangan)

GajiBersih ← GajiPokok + tunjangan - pajak

cout(GajiBersih)

Konversi Waktu ke Jam-Menit-Detik

Jika diberikan total waktu dalam satuan detik, maka berapa jam, berapa menit dan berapa detikkah waktu tersebut? Sebagai contoh, misalkan lama percakapan seseorang di ponsel adalah 4000 detik, maka 4000 detik = 1 jam + 6 menit + 40 detik, ini diperoleh dengan perhitungan berikut :

$$\begin{array}{ll} 4000 \text{ div } 3600 & = 1 \text{ (jam)} \\ 4000 \text{ mod } 3600 & = 400 \text{ (sisanya detik)} \\ 400 \text{ div } 60 & = 6 \text{ (menit)} \\ 400 \text{ mod } 60 & = 40 \text{ (detik)} \end{array}$$

Tuliskan algoritma yang membaca waktu dalam satuan detik, lalu mengubahnya ke dalam jam-menit-detik dan menampilkan hasilnya di layar.

Konversi Waktu ke Jam- Menit-Detik

PROGRAM Konversi_detik_ke_JamMenitDetik

{Membaca lama percakapan telepon dalam detik, lalu mengkonversinya ke dalam jam-menit-detik. Hasil konversi ditampilkan ke layar}

DEKLARASI :

```
type Jam : record <hh : int, {jam}  
                mm : int, {menit}  
                ss : int {detik}  
                >
```

J : **Jam**

TotalDetik : int

sisa : int {peubah/variable bantu untuk mencatat sisa detik}

ALGORITMA :

cin (TotalDetik)

J.hh ← TotalDetik **div** 3600 {mendapatkan jam}

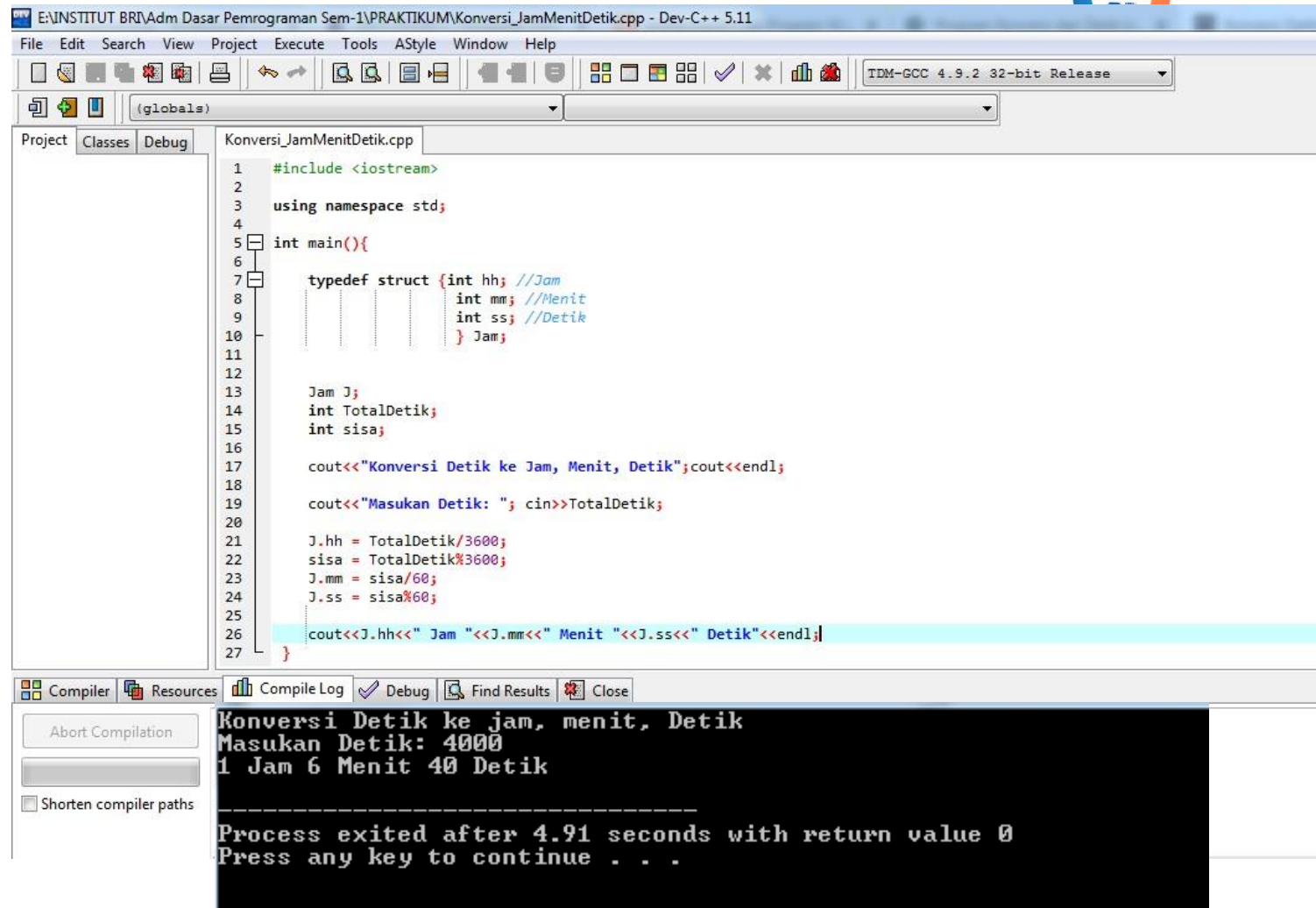
sisa ← TotalDetik **mod** 3600

J.mm ← sisa **div** 60 {mendapatkan menit}

J.ss ← sisa **mod** 60 {mendapatkan detik}

cout(J.hh, J.mm, J.ss)

Konversi Waktu ke Jam-Menit- Detik



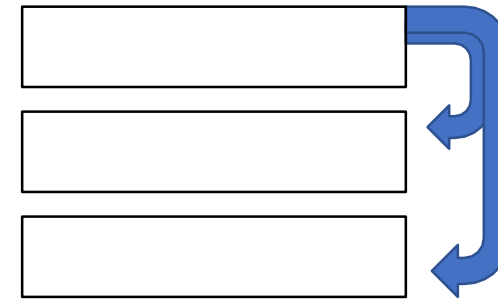
```
E:\INSTITUT BRI\Adm Dasar Pemrograman Sem-1\PRAKTIKUM\Konversi_JamMenitDetik.cpp - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
(globals)
Project Classes Debug Konversi_JamMenitDetik.cpp
1 #include <iostream>
2
3 using namespace std;
4
5 int main(){
6
7     typedef struct {int hh; //Jam
8                     int mm; //Menit
9                     int ss; //Detik
10                    } Jam;
11
12
13     Jam J;
14     int TotalDetik;
15     int sisa;
16
17     cout<<"Konversi Detik ke Jam, Menit, Detik";cout<<endl;
18
19     cout<<"Masukan Detik: "; cin>>TotalDetik;
20
21     J.hh = TotalDetik/3600;
22     sisa = TotalDetik%3600;
23     J.mm = sisa/60;
24     J.ss = sisa%60;
25
26     cout<<J.hh<<" Jam "<<J.mm<<" Menit "<<J.ss<<" Detik"<<endl;
27 }
Compiler Resources Compile Log Debug Find Results Close
Abort Compilation
Shorten compiler paths
Konversi Detik ke jam, menit, Detik
Masukan Detik: 4000
1 Jam 6 Menit 40 Detik
-----
Process exited after 4.91 seconds with return value 0
Press any key to continue . . .
```

Pemilihan (Selection)

Algoritma Selection dikerjakan jika memenuhi persyaratan (kondisi) tertentu. Ditulis dalam notasi pseudo-code sebagai berikut :

```
if kondisi then  
    aksi
```

Konstruksi pemilihan if-then hanya memberikan satu pilihan aksi bila kondisi dipenuhi (bernilai benar), dan tidak memberikan pilihan aksi alternatif bila kondisi bernilai salah. Bentuk pemilihan yang lebih umum ialah memilih satu dari dua buah aksi bergantung pada nilai kondisinya




```
if kondisi then  
    aksi 1  
else  
    aksi 2
```

Pemilihan (Selection)

Kelebihan konstruksi pemilihan terletak pada kemampuannya yang memungkinkan pemroses mengikuti jalur aksi yang berbeda berdasarkan kondisi yang ada. Tanpa konstruksi pemilihan, kita tidak mungkin menulis algoritma untuk permasalahan yang demikian kompleks.

Dalam menganalisis kasus, semua kasus harus dijabarkan dengan lengkap. Bergantung pada persoalannya, ada persoalan yang terdiri :

1. Pemilihan (Selection) 1 kasus
 2. Pemilihan (Selection) 2 kasus
 3. Pemilihan (Selection) 3 kasus atau lebih.
- 

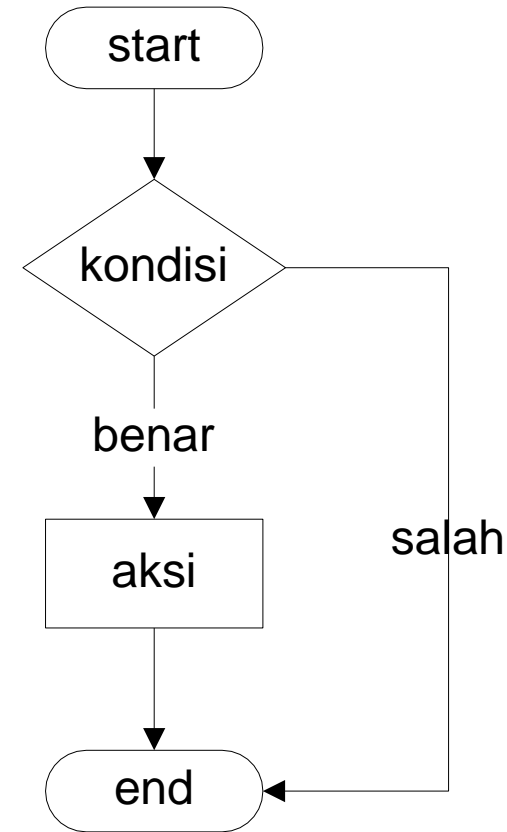
Pemilihan (Selection) 1

Kasus

Notasi algoritma untuk analisis dengan satu kasus adalah dengan menggunakan konstruksi *IF-THEN* (jika-maka) berbentuk :

if kondisi **then**
aksi
end if

Intruksi di atas berarti bahwa *aksi* hanya dilaksanakan bila *kondisi* benar(*true*). Bila *kondisi* salah (*false*), maka tidak ada aksi apa pun yang dikerjakan. Kata **end if** sengaja kita tambahkan untuk mempertegas awal dan akhir struktur IF-THEN.



Pemilihan (Selection) 1

Kasus

Contoh : Tulislah algoritma yang menerima input sebuah karakter, lalu menuliskan pesan “huruf vokal” jika karakter tersebut merupakan salah satu dari huruf vocal.

Penyelesaian : Huruf vocal ada lima, yaitu a, i, u, e, dan o. Baca sebuah karakter, lalu bandingkan karakter yang dibaca dengan kelima huruf tersebut. Jika karakter masukan sama dengan salah satu huruf vocal tersebut, maka tulislah pesan bahwa karakter tersebut adalah “huruf vokal”.

Pemilihan (Selection) 1 Kasus

PROGRAM HurufVokal

{Mencetak pesan "Huruf Vokal" bila sebuah karakter yang dibaca merupakan huruf hidup.
Asumsikan karakter yang dibaca adalah huruf kecil saja}

DEKLARASI :

character : **char**

ALGORITMA :

cin (character)

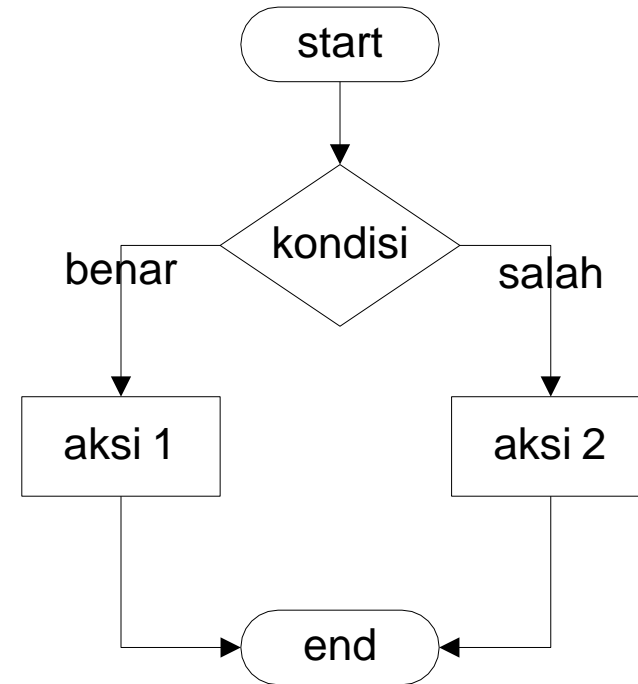
if (character='a') or (character='i') or (character='u') or (character='e') or (character='o') **then**

cout('huruf vokal')

end if

Pemilihan (Selection) 2 Kasus

Konstruksi *IF-THEN* hanya menyediakan satu alternative aksi jika suatu kondisi dipenuhi. Kadang-kadang kita perlu memilih melakukan aksi alternatif jika kondisi tidak memenuhi. Notasi algoritma untuk masalah dengan dua buah kasus adalah dengan menggunakan konstruksi *IF-THEN-ELSE* (jika-maka-kalau tidak)



Pemilihan (Selection) 2 Kasus

Contoh : Karyawan honorer di PT ABC digaji berdasarkan jumlah jam kerjanya selama satu minggu. Upah per jam misalkan Rp20000. Bila jumlah jam kerja lebih besar dari 48 jam, maka sisanya dianggap sebagai jam lembur. Upah lembur misalkan Rp30000/jam. Tulislah algoritma yang membaca jumlah jam kerja seorang karyawan selama satu minggu, lalu menentukan upah mingguannya

Penyelesaian :

Misalkan jumlah jam kerja karyawan adalah JJK.

Analisis kasus :

Kasus 1 : Jika $JJK \leq 48$, maka upah = $JJK * 20000$

Kasus 2 : Jika $JJK > 48$, maka :

lembur = $JJK - 48$

upah = $(48 * 20000) + (lembur * 30000)$

Pemilihan (Selection) 2 Kasus

PROGRAM UpahKaryawan

{Menentukan upah mingguan seorang karyawan. Data yang di inputkan adalah nama karyawan, golongan, dan jumlah jam kerja. Luaran program adalah upah karyawan tersebut}

DEKLARASI :

nama : **string** {Nama Karyawan}
JJK : **int** {Jumlah Jam Kerja}
lembur : **int** {Jumlah Jam Lembur}
upah : **float** {Upah Karyawan}

ALGORITMA:

cin (nama, JJK)
if JJK \leq 48 **then**
 upah \leftarrow JJK * 20000
else {berarti JJK > 48}
 lembur \leftarrow JJK - 48
 upah \leftarrow (48 * 20000) + (lembur * 30000)
end if
cout (upah)

Pemilihan (Selection) 2 Kasus

PROGRAM *UpahKaryawan*

{Menentukan upah mingguan seorang karyawan. Data yang di inputkan adalah nama karyawan, golongan, dan jumlah jam kerja. Luaran program adalah upah karyawan tersebut}

DEKLARASI :

const JamNormal = 48 {jumlah jam kerja normal per minggu}

const UpahPerJam = 20000 {upah per jam, Rp20000}

const UpahLembur = 30000 {upah lembur per jam, Rp30000}

nama : **string** {Nama Karyawan}

JJK : **int** {Jumlah Jam Kerja}

lembur : **int** {Jumlah Jam Lembur}

upah : **float** {Upah Karyawan}

ALGORITMA :

cin (nama, JJK)

if JJK \leq JamNormal **then**

 upah \leftarrow JJK * UpahPerJam

else {berarti JJK > 48}

 lembur \leftarrow JJK - JamNormal

 upah \leftarrow (JamNormal * UpahPerJam) + (lembur * UpahLembur)

end if

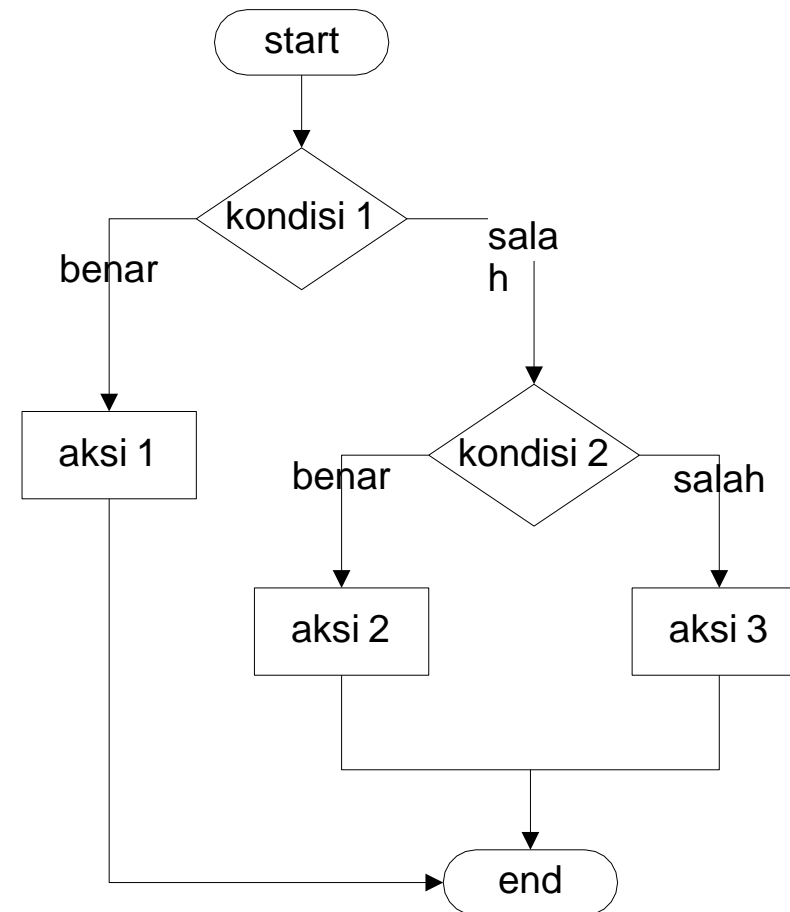
cout (upah)

Pemilihan (Selection) 3 Kasus atau Lebih

Masalah yang mempunyai tiga buah kasus atau lebih dapat dianalisis dengan konstruksi *IF-THEN-ELSE* bertingkat-tingkat.

Tiga Kasus :

```
if kondisi 1 then
  aksi 1
else
  if kondisi 2 then
    aksi 2
  else
    if kondisi 3 then
      aksi 3
    end if
  end if
end if
```



Pemilihan (Selection) 3 Kasus atau Lebih

Empat Kasus :

if kondisi 1 **then**

aksi 1

else

if kondisi 2 **then**

aksi 2

else

if kondisi 3 **then**

aksi 3

else

if kondisi 3 **then**

aksi 3

end if

end if

end if

end if

Dan seterusnya untuk lima kasus, enam kasus,

Contoh 4 Kasus

Contoh : Karyawan di PT ABC dikelompokkan berdasarkan golongannya. Upah per jam tiap karyawan bergantung pada golongannya (lihat tabel). Jumlah jam kerja yang normal selama satu minggu adalah 48 jam. Kelebihan jam kerja dianggap lembur dengan upah Rp30000/jam untuk semua golongan karyawan. Buatlah algoritma yang membaca nama karyawan dan jumlah jam kerjanya

selama satu minggu, lalu menghitung gaji mingguannya. Persoalan ini pengembangan dari contoh sebelumnya.

Golongan	Upah Per Jam
A	Rp 40000
B	Rp 50000
C	Rp 60000
D	Rp 75000

Contoh 4 Kasus

Penyelesaian :

Analisis kasus masalah ini lebih rumit. Mula-mula kita harus menentukan upah per jam berdasarkan golongannya.

Analisis kasus :


Kasus 1 : jika golongan = 'A' maka upah perjam = 40000

Kasus 2 : jika golongan = 'B' maka upah perjam = 50000

Kasus 3 : jika golongan = 'C' maka upah perjam = 60000

Kasus 4 : jika golongan = 'D' maka upah perjam = 75000

Selanjutnya, kita menghitung upah yang dihitung dari jumlah jam kerja. Upah per jam bergantung pada hasil analisis kasus. Jika ada jam lembur, maka upah total adalah upah kerja + upah lembur.



Contoh 4 Kasus

PROGRAM UpahKaryawan

{Menentukan upah mingguan seorang karyawan. Data yang di inputkan adalah nama karyawan, golongan, dan jumlah jam kerja. Luaran program adalah upah karyawan tersebut}

DEKLARASI :

const JamNormal = 48 {jumlah jam kerja normal per minggu}

const UpahLembur = 30000 {upah lembur per jam, Rp30000}

nama : **string** {Nama Karyawan}

gol : **char** {golongan karyawan}

JJK : **int** {Jumlah Jam Kerja}

lembur : **int** {Jumlah Jam Lembur}

UpahPerjam : **float** {Upah PerJam}

UpahTotal : **float** {Upah Total Karyawan}

Contoh 4 Kasus

ALGORITMA:

cin (nama, gol, JJK)

if gol = 'A' then

UpahPerjam \leftarrow 40000

else

if gol = 'B' then

UpahPerjam \leftarrow 50000

else

if gol = 'C' then

UpahPerjam \leftarrow 60000

else

if gol = 'D' then

UpahPerjam \leftarrow 75000

end if

end if

end if

end if

if JJK \leq JamNormal then

UpahTotal \leftarrow JJK * UpahPerJam

else {berarti JJK > 48}

lembur \leftarrow JJK - JamNormal

UpahTotal \leftarrow (JamNormal * UpahPerJam) + (lembur * UpahLembur)

end if

cout (UpahTotal)



Contoh : Indeks nilai mahasiswa ditentukan berdasarkan nilai ujian yang diraihinya. Ketentuan pemberian nilai indeks sebagai berikut :

Kasus 1 : jika nilai ujian ≥ 80 , maka indeks nilai = A

Kasus 2 : jika $70 \leq \text{nilai ujian} < 80$, maka indeks nilai = B

Kasus 3 : jika $55 \leq \text{nilai ujian} < 70$, maka indeks nilai = C

Kasus 4 : jika $40 \leq \text{nilai ujian} < 55$, maka indeks nilai = D

Kasus 5 : jika nilai ujian < 40 , maka indeks nilai = E

Contoh 5 Kasus

Buatlah algoritma yang membaca nama mahasiswa dan nilai ujiannya, lalu menentukan indeks nilainya, kemudian mencetak nama mahasiswa, nilai ujian, dan indeksnya.

Contoh 5 Kasus

PROGRAM *IndeksNilai*

{Menghitung indeks nilai ujian mahasiswa}

DEKLARASI :

nama : **string** {Nama mahasiswa}

nilai : **float** {nilai ujian mahasiswa}

indeks : **char** {Indeks Nilai}

ALGORITMA :

cin (*nama*, *nilai*)

if *nilai* \geq 80 **then**

indeks \leftarrow 'A'

else

if (*nilai* \geq 70) **and** (*nilai* < 80) **then**

indeks \leftarrow 'B'

else

if (*nilai* \geq 55 **and** (*nilai* < 70)) **then**

indeks \leftarrow 'C'

else

if (*nilai* \geq 40) **and** (*nilai* < 55) **then**

indeks \leftarrow 'D'

else

indeks \leftarrow 'E'

end if

end if

end if

end if

cout (*nama*, *nilai*, *indeks*)

Struktur Case

- Struktur pemilihan atau percabangan ini secara fungsi tidak jauh berbeda dengan percabangan if-else-else, hanya berbeda pada cara penyajiannya saja.
- Untuk masalah dengan dua kasus atau lebih, konstruksi **CASE** dapat menyederhanakan penulisan **IF-THEN-ELSE** yang bertingkat-tingkat sebagaimana pada contoh-contoh sebelum ini. Konstruksi **CASE** sebagai berikut :

case (ekspresi):

nilai 1 : aksi 1

nilai 2 : aksi 2

nilai 3 : aksi 3

.

.

nilai n : aksi n

otherwise aksi x

end case

Struktur *Case*

- Ekspresi adalah sembarang ekspresi (aritmetika atau boolean) yang menghasilkan suatu nilai (konstanta). Konstruksi **CASE** memeriksa apakah nilai dari evaluasi ekspresi tersebut sama dengan salah satu dari nilai 1, nilai 2,, nilai n (catatan : semua nilai-nilai ini harus berbeda).
- Jika nilai ekspresi sama dengan nilai k, maka aksi k dilaksanakan. Aksi yang bersesuaian dengan nilai k dapat lebih dari satu, karena itu ia berupa runtutan. Jika tidak ada satu pun nilai ekspresi yang cocok, maka aksi x sesudah otherwise dikerjakan.
- Kasus *otherwise* bersifat *optional*, artinya ia boleh ditulis atau tidak di dalam konstruksi **CASE**.

Struktur Case

Contoh : Konversi angka ke teks

Buatlah algoritma yang menerima masukan sebuah bilangan bulat yang nilainya terletak antara 1 sampai 4, lalu mencetak tulisan angka tersebut dalam kata-kata. Misalkan bila nilai yang dibaca angka 1, maka di cetak tulisan “satu”, bila di baca 2, maka tercetak di layar tulisan “dua”, demikian seterusnya. Jika angka yang dimasukkan selain 1 sampai 4, tuliskan pesan bahwa angka yang di masukkan salah.

Penyelesaian :

Dengan struktur *IF-THEN-ELSE*, algoritma mencetak kata untuk angka yang bersesuaian sebagai berikut :

Struktur Case

PROGRAM KonversiAngkaKeTeks

{Mencetak kata untuk angka 1 sampai 4}

DEKLARASI

angka : int {angka yang dibaca}

ALGORITMA

cin (angka)

if angka = 1 then

cout('satu')

else

if angka = 2 then

cout('dua')

else

if angka = 3 then

cout('tiga')

else

if angka = 4 then

cout('empat')

else

cout('angka yang dimasukkan harus 1 sampai 4')

end if

end if

end if

end if

Struktur Case

Dengan konstruksi *CASE*, algoritma untuk masalah sebelumnya dapat dibuat menjadi lebih singkat sebagai berikut :

PROGRAM KonversiAngkaKeTeks

{Mencetak kata untuk angka 1 sampai 4}

DEKLARASI

angka : int {angka yang dibaca}

ALGORITMA

cin (angka)

case (angka):

1 : cout('satu')

2 : cout('dua')

3 : cout('tiga')

4 : cout('empat')

otherwise cout('angka yang dimasukkan salah')

end case

Struktur


Case

Contoh : Mencetak nama bulan

Buatlah algoritma yang menerima input nomor bulan (1 sampai 12), lalu menuliskan nama bulan sesuai angka bulannya. Misalnya jika dibaca bulan 8, maka tercetak “Agustus”.

Penyelesaian :

Masalah ini memiliki 13 buah kasus karena nama bulan berbeda-beda bergantung pada nomor bulan yang diberikan (ada 12 bulan dalam kalender Masehi). Satu kasus tambahan adalah bila nomor bulan yang dimasukkan di luar rentang 1 sampai 12.



Struktur Case

PROGRAM NamaBulan

{Mencetak nama bulan berdasarkan nomor bulan(1 sampai 12)}

DEKLARASI

NomorBulan : int

ALGORITMA

cin (NomorBulan)

case (NomorBulan):

1 : cout('januari')

2 : cout('februari')

3 : cout('maret')

4 : cout('april')

5 : cout('mei')

6 : cout('juni')

7 : cout('juli')

8 : cout('agustus')

9 : cout('september')

10 : cout('oktober')

11 : cout('november')

12 : cout('desember')

otherwise cout('Bukan bulan yang benar')

end case



Terima
kasih

Latihan Soal

1. Sebuah proyek dikerjakan selama X hari. Tulislah algoritma untuk mengkonversi berapa tahun, berapa bulan dan berapa hari proyek tersebut dikerjakan. Asumsikan : 1 tahun = 365 hari, 1 bulan = 30 hari. Keluaran (tahun, bulan, hari) ditampilkan ke piranti keluaran.
2. Pasar swalayan XYZ memberikan diskon harga bagi pembeli yang nilai total belanjanya lebih dari Rp. 100.000. Tulislah algoritma untuk menentukan nilai belanja setelah dikurangi diskon. Data masukan adalah nilai total belanja pembeli, sedangkan keluarannya adalah diskon harga dan nilai belanja setelah di kurangi diskon.