



# TEKNIK MEMBINA ATUR CARA DENGAN BAHASA C

DAYANG NORHAYATI ABANG JAWAWI  
ROSBI MAMAT



## Bab 5: Struktur Kawalan

© Copyright Universiti Teknologi Malaysia

innovative • entrepreneurial • global

## 5.1 Pengenalan

- Bab 1 telah menerangkan bagaimana hendak menyelesaikan masalah dengan mereka bentuk struktur jujukan, struktur pemilihan, struktur perulangan atau gelung.
- C mempunyai beberapa struktur kawalan untuk mengawal aliran atur cara bagi melakukan pencabangan dan perulangan.
- Struktur-struktur ini berguna untuk mengawal pelaksanaan susunan kenyataan dengan cara yang berbeza. Kata kunci yang digunakan dalam suruhan kawalan adalah seperti di bawah:

```
if     else     switch  case  
for    while   do        goto
```

## 5.1 Pengenalan

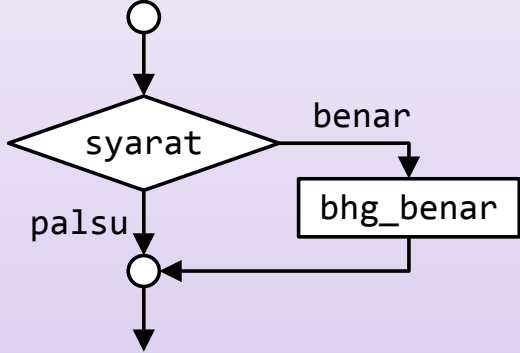
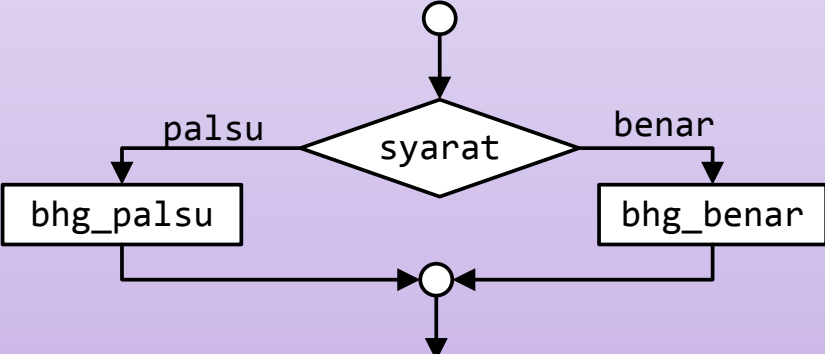
- ❖ **Struktur pemilihan** - arahan-arahan tertentu akan dipilih untuk dilaksanakan dan pemilihan ini bergantung kepada syarat pemilihan.
  
- ❖ C melaksanakan struktur pemilihan dengan tiga cara iaitu:
  1. Pemilihan dengan `if` dan `else`
  2. Pemilihan dengan operator bersyarat
  3. Pemilihan dengan `switch`, `case` dan `break`
  
- ❖ Dalam bahasa C tiga kata kunci digunakan untuk melakukan perulangan atau gelung:
  1. Gelung dengan `while`
  2. Gelung dengan `do` dan `while`
  3. Gelung dengan `for`

## 5.2 Pemilihan dengan `if` dan `else`

- ❖ Kata kunci `if` dan `else` digunakan untuk memilih aliran atur cara atau membuat keputusan bergantung kepada hasil penilaian ungkapan syarat logik sama ada benar atau palsu.
  
- ❖ Terdapat tiga format struktur penggunaan kenyataan pemilihan dengan `if` dan `else`:
  1. pemilihan satu-arah
  2. pemilihan dua-arah
  3. pemilihan multi-arah

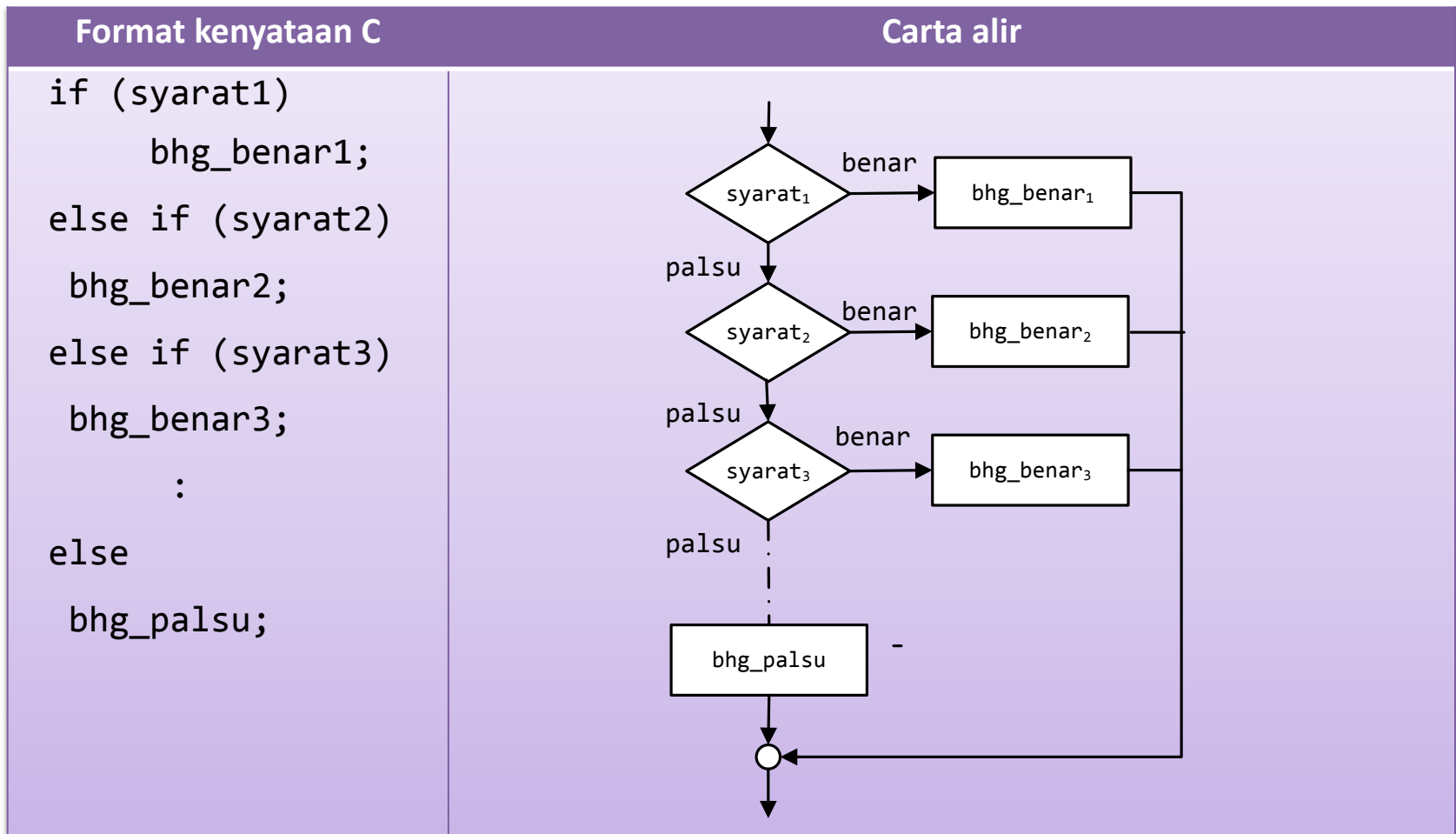
## 5.2 Pemilihan dengan `if` dan `else`

❖ Struktur kawalan pemilihan format kenyataan C dengan format carta alir:

Format kenyataan C	Carta alir
<pre>if (syarat)     bhg_benar;</pre>	 <pre>graph TD; Start(( )) --&gt; Syarat{syarat}; Syarat -- benar --&gt; BhgBenar[bhg_benar]; BhgBenar --&gt; Join(( )); Syarat -- palsu --&gt; Join; Join --&gt; End(( ));</pre>
<pre>if (syarat)     bhg_benar; else     bhg_palsu;</pre>	 <pre>graph TD; Start(( )) --&gt; Syarat{syarat}; Syarat -- palsu --&gt; BhgPalsu[bhg_palsu]; Syarat -- benar --&gt; BhgBenar[bhg_benar]; BhgPalsu --&gt; Join(( )); BhgBenar --&gt; Join; Join --&gt; End(( ));</pre>

## 5.2 Pemilihan dengan `if` dan `else`

❖ Struktur kawalan pemilihan format kenyataan C dengan format carta alir:



## 5.2 Pemilihan dengan `if` dan `else`

- ❖ contoh2 kenyataan pemilihan satu-arah dengan penggunaan kata kunci `if`:

Contoh 1

```
if (x == 10)
printf( "sama dgn 10");
```

Contoh 2

```
if ((x>10) || (y>20))
printf("Memenuhi syarat");
```

## 5.2 Pemilihan dengan `if` dan `else`

❖ contoh2 kenyataan dua-arah:

Contoh 1

```
if (x < 0)
    printf( "negatif");
else
    printf( "positif");
```



## 5.2 Pemilihan dengan `if` dan `else`

❖ contoh2 kenyataan dua-arah (sambungan):

Contoh 2

```
if (x > 10)
{
    printf( "lebih dari 10");
    ++kira;
}
else
{
    printf( "kecil dari 10");
    --kira;
}
```

## 5.2 Pemilihan dengan `if` dan `else`

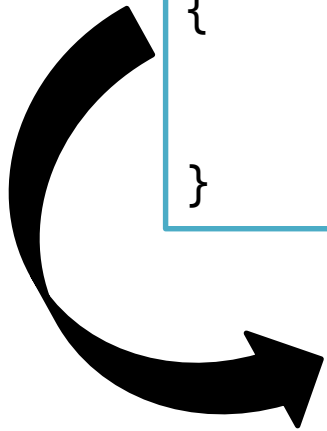
- ❖ **Pengujian fail** input merupakan satu contoh penggunaan kenyataan pemilihan.
- ❖ Tujuan pengujian adalah untuk memastikan tiada ralat dalam proses membuka fail input, pengujian boleh dilakukan terutama bagi fail yg hendak dibaca. Semasa fail `stor12.dat` hendak dibuka & dibaca pengujian dilakukan.

```
FILE *failinp;  
failinp = fopen("stor12.dat", "r");  
  
if (failinp == NULL)  
{  
    printf("Ralat dalam pembukaan fail.\n");  
    exit(-1); /*Tamatkan Atur cara*/  
}
```

## 5.2 Pemilihan dengan `if` dan `else`

❖ Contoh pegujian fail (sambungan):

```
FILE *failinp;  
failinp = fopen("stor12.dat", "r");  
  
if (failinp == NULL)  
{  
    printf("Ralat dalam pembukaan fail.\n");  
    exit(-1); /*Tamatkan Atur cara*/  
}
```



Kod ini boleh ditukar  
spt berikut -  
memberi maksud  
yang sama

```
if (failptr = fopen("stor12.dat", "r"))== NULL  
{  
    puts("Ralat dalam pembukaan fail.");  
    exit(-1); /*Tamatkan Atur cara*/  
}
```

## 5.3 Pemilihan dengan Operator Bersyarat

- ❖ C mempunyai singkatan bagi mengungkap kenyataan if else dengan menggunakan operator bersyarat dengan simbol operator ? dan :.
- ❖ Format penggunaan operator bersyarat dalam atur cara:

```
syarat ? ungkapan_benar : ungkapan_palsu
```

- ❖ Jika syarat adalah BENAR, maka keseluruhan ungkapan ini akan memberikan nilai ungkapan\_benar. Sebaliknya, jika syarat adalah PALSU, maka keseluruhan ungkapan ini akan memberikan nilai ungkapan\_palsu.

## 5.3 Pemilihan dengan Operator Bersyarat

❖ Contoh:

```
harga_tiket = (umur < 12) ? 10 : 20;
```

❖ pembolehubah harga\_tiket akan diberikan nilai 10 jika nilai pembolehubah umur kurang dari 12 sebaliknya ia akan diberikan nilai 20 jika nilai pembolehubah umur lebih dari atau sama dgn 12. Kenyataan `if-else` yg setara:

```
if (umur < 12)
    harga_tiket = 10;
else
    harga_tiket = 20;
```

## 5.3 Pemilihan dengan Operator Bersyarat

- ❖ Contoh aturcara mencari luas segi empat menggunakan operator bersyarat:

```
1: #include <stdio.h>
2:
3: int main () {
4:     float luas, panjang, lebar;
5:
6:     printf ("masukkan nilai panjang dan lebar\n");
7:     scanf ("%f %f", &panjang, &lebar);
8:
9:     if (panjang== 0 || lebar ==0) {
10:         printf("salah satu nilai adalah 0\n");
11:         luas=0;
12:     }
13:     else {
14:         panjang = (panjang < 0) ? -panjang : panjang;
15:         lebar = (lebar < 0 ) ?          -lebar : lebar;
16:         luas = panjang*lebar;
17:     }
18:     printf ("luas segiempat ialah %0.2f\n", luas);
19:     return 0;
20: }
```

## 5.4 Pemilihan dengan `switch`, `case` dan `break`

- ❖ Struktur `if-else-if-...else` digunakan untuk memilih aliran atur cara yang terdiri daripada banyak alternatif. Penggunaan struktur `switch-case-break` akan menjadikan kenyataan-kenyataan struktur berbilang alternatif lebih mudah dilakukan.
- ❖ Format penggunaan `switch-case-break` dalam aturcara:

```
switch (ungkapan_integer)
{
    case pemalar1:  kenyataan21; break;
    case pemalar2:  kenyataan22; break;
    case pemalar3:  kenyataan23; break;
                  :
    default:        kenyataan2n;
}
```

- ❖ Di mana `ungkapan_integer` adalah ungkapan yang menghasilkan nilai integer atau char `innovative` • `entrepreneurial` • `global`

## 5.4 Pemilihan dengan switch, case dan break

❖ Cara operasi:

1. Atur cara akan mencari nilai ungkapan\_integer.
2. Nilai akan dibandingkan dengan nilai-nilai pemalar1, pemalar2 dalam kenyataan case pemalar1, case pemalar2 dan seterusnya. Perbandingan dilakukan satu demi satu sehinggalah atur cara menjumpai nilai pemalar yang sama dengan nilai ungkapan\_integer. Jika sama, kenyataan-kenyataan di antara case break tersebut akan dilaksanakan. Misalnya jika ungkapan\_integer sama dengan pemalar2, maka bahagian kenyataan22 akan dilaksanakan.



## 5.4 Pemilihan dengan switch, case dan break

❖ Cara operasi (sambungan):

3. Jika ungkapan `integer` tidak sama dengan mana-mana pemalar dalam senarai `case` maka kenyataan `default` akan dilaksanakan. Bahagian `default` adalah pilihan, bahagian ini boleh ditinggalkan jika tidak perlu.
4. Kata kunci `break` akan menyebabkan pelaksanaan kenyataan dalam `switch` ditamatkan dan selanjutnya kenyataan di luar daripada blok kenyataan `switch() { ... }` dilaksanakan. Jika `break` ditinggalkan, kenyataan-kenyataan pada bahagian `case` selepasnya akan dilaksanakan.

## 5.4 Pemilihan dengan switch, case dan break

- ❖ Aturcara ini akan mencetak nombor 1, 2, atau 3 yang dimasukkan melalui papan kekunci dlm perkataan- satu, dua atau tiga:

```
1: #include <stdio.h>
2:
3: int main () {
4:
5:     int x;
6:     scanf("%d", &x);
7:     switch (x)
8:     {
9:         case 1: puts ("satu"); break;
10:        case 2: puts ("dua"); break;
11:        case 3: puts ("tiga"); break;
12:        default: puts ("bukan satu dua tiga ");
13:    }
14:
15:    return 0;
16: }
```

## 5.4 Pemilihan dengan switch, case dan break

- ❖ Perhatikan output jika break dikeluarkan seperti berikut:

```
switch (x)
{
    case 1: puts ("satu"); break;
    case 2: puts ("dua");
    case 3: puts ("tiga");
    default: puts ("bukan satu dua tiga ");
}
```

<b>Input : 1</b> <b>Output :</b> satu	<b>Input: 3</b> <b>Output:</b> tiga bukan satu dua tiga
<b>Input: 2</b> <b>Output:</b> dua tiga bukan satu dua tiga	<b>Input: 7</b> <b>Output:</b> bukan satu dua tiga

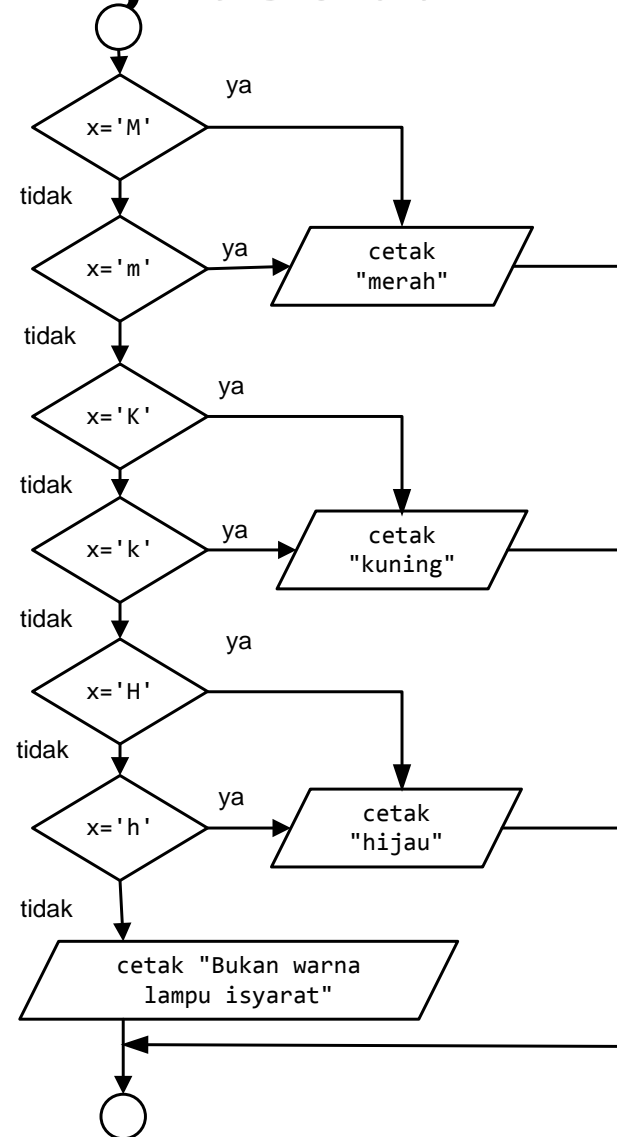
## 5.4 Pemilihan dengan switch, case dan break

- ❖ Contoh atur cara akan mencetak jenis warna pada skrin, ia akan mencetak "merah" jika aksara 'M' atau 'm', "kuning" jika aksara 'K' atau 'k' dan "hijau" jika aksara 'H' atau 'h' dimasukkan melalui papan kekunci.

```
1: #include <stdio.h>
2:
3: int main () {
4:     char x;
5:     scanf("%c", &x);
6:     switch (x) {
7:         case 'M':
8:         case 'm': printf ("merah\n"); break;
9:         case 'K':
10:        case 'k': printf ("kuning\n"); break;
11:        case 'H':
12:        case 'h': printf ("hijau\n"); break;
13:        default: printf ("Bukan warna lampu isyarat\n");
14:    }
15:    return 0;
16: }
```

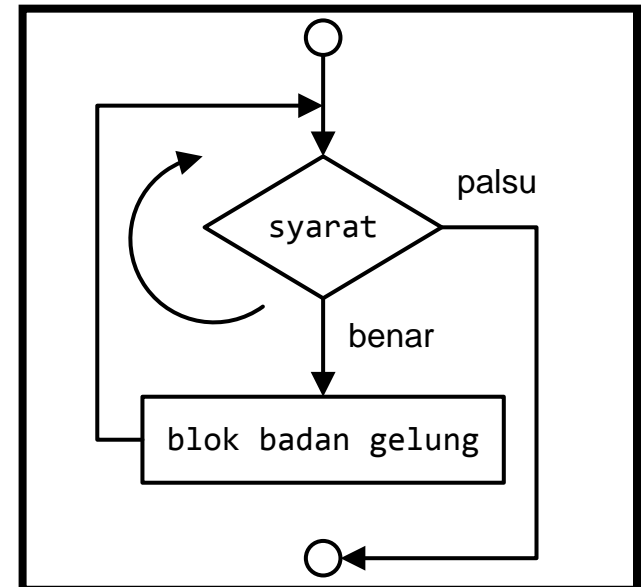
# 5.4 Pemilihan dengan switch, case dan break

❖ Carta alir bagi aturcara sebelum ini:



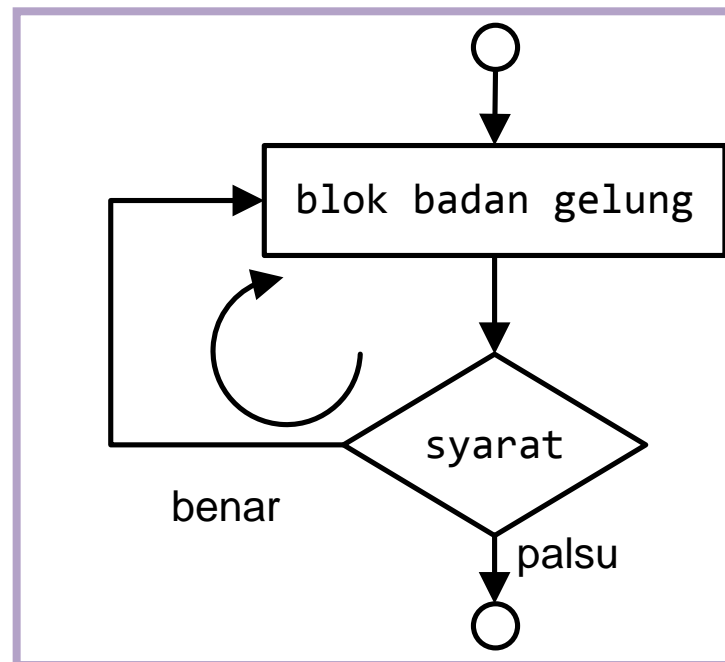
## 5.5 Gelung dengan `while` dan `do while`

- ❖ Kenyataan `while` digunakan untuk melaksanakan operasi gelung dengan mengulang satu blok kenyataan sehingga satu syarat dipenuhi.
- ❖ Perbezaan `while` dan `do while` adalah gelung `while` merupakan gelung pra-uji, manakala gelung `do while` merupakan gelung pasca-uji.
- ❖ Gelung pra-uji akan menilai syarat sebelum memasuki blok badan tidak akan dilaksanakan jika syarat adalah palsu semasa masuk ke gelung.  
Carta alir kawalan gelung pra-uji :



## 5.5 Gelung dengan `while` dan `do while`

- ❖ Gelung pasca-uji akan menilai syarat selepas memasuki blok badan gelung dan juga akan mengulangi kenyataan-kenyataan dalam blok badan gelung selagi syarat masih benar. Carta alir kawalan gelung pasca-uji :



## 5.5 Gelung dengan `while` dan `do while`

□ Format penggunaan suruhan `while` dan `do .. while` dalam aturcara:

- `while` :

```
while (syarat)
{
    kenyataan1;
    kenyataan2;
    :
    kenyataann;
}
```

- `Do .. while`:

```
do
{
    kenyataan1;
    kenyataan2;
    :
    kenyataann;
} while (syarat)
```



## 5.5 Gelung dengan `while` dan `do while`

- ❑ Perbezaan `while` dan `do while`:

Perbezaan	<code>while</code>	<code>do while</code>
Cara syarat diuji	sebelum masuk blok badan gelung	setelah masukke blok badan gelung
Bilangan ulangan badan gelung jika penilaian kali pertama syarat gelung adalah palsu	0	1

## 5.5 Gelung dengan while dan do while

- Contoh mengira purata senarai nombor dengan gelung while

```
1: #include <stdio.h>
2: #include <conio.h>
3:
4: int main ()
5: {
6:     int pengira=0;
7:     float nom, jumlah = 0;
8:     char kira;
9:
10:    printf("Kira Purata Senarai Nombor\n");
11:    printf("Tekan y untuk tambah nombor > ");
12:    kira = getch();
13:    while ( kira == 'y' || kira == 'Y')
14:    {
15:        ++pengira;
16:        printf("\n%d) Masukan nombor > ", pengira);
17:        scanf("%f", &nom);
18:        jumlah += nom;
19:        printf("Tekan y untuk tambah nombor > ");
20:        kira = getch();
21:    }
22:    printf ("\nJumlah %d nombor ialah %0.2f", pengira, jumlah);
23:    return 0;
24: }
```

## 5.5 Gelung dengan `while` dan `do while`

- Menjejak pelaksanaan gelung `while` dengan satu set input:

<code>kira == 'y'    kira == 'Y'</code> (syarat)	<code>pengira</code>	<code>nom</code>	<code>jumlah</code>
'y'(-)	0	?	0
'y'(benar)	1	2.3	2.3
'y'(benar)	2	4.5	6.8
'y'(benar)	3	20.1	26.9
'n'(palsu)			

- Dengan input `kira` dan `nom` seperti di atas, gelung akan dilaksanakan tiga kali dengan nilai pemboleh ubah `jumlah` juga berubah sebanyak tiga kali. Setelah `kira` dimasukkan input 'n', ulangan gelung akan ditamatkan dan kenyataan `printf()` baris 22 akan mencetak mesej berikut:

Jumlah 3 nombor ialah 26.90

## 5.5 Gelung dengan `while` dan `do while`

- Contoh penyelesaian dengan kawalan pembilang dengan meminta pengguna memasukkan bilangan nombor seperti berikut yang diubah dari aturcara lepas pada baris 11 hingga 21:

```
printf("Masukan bilangan nombor > ");
scanf("%d", &bil);
while (pengira < bil)
{
    ++pengira;
    printf("\n%d) Masukan nombor > ", pengira);
    scanf("%f", &nom);
    jumlah += nom;
}
```

## 5.5 Gelung dengan `while` dan `do while`

- Menjejak pelaksanaan gelung `while` dengan kawalan gelung pembilang:

pengira < bil (syarat)	pengira	nom	jumlah
-	0	?	0
0 < 3 (benar)	1	2.3	2.3
1 < 3 (benar)	2	4.5	6.8
2 < 3 (benar)	3	20.1	26.9
3 < 3 (palsu)			

## 5.5 Gelung dengan `while` dan `do while`

- Jika penyelesaian pada atur cara ditukar dengan menggunakan gelung `do while`, perhatikan penyelesaian kenyataan gelung berikut:

```
do
{
    ++pengira;
    printf("\n%d) Masukan nombor > ", pengira);
    scanf("%f", &nom);
    jumlah += nom;
    printf("Tekan y untuk tambah nombor > ");
    kira = getch();
} while (kira == 'y' || kira == 'Y');
```

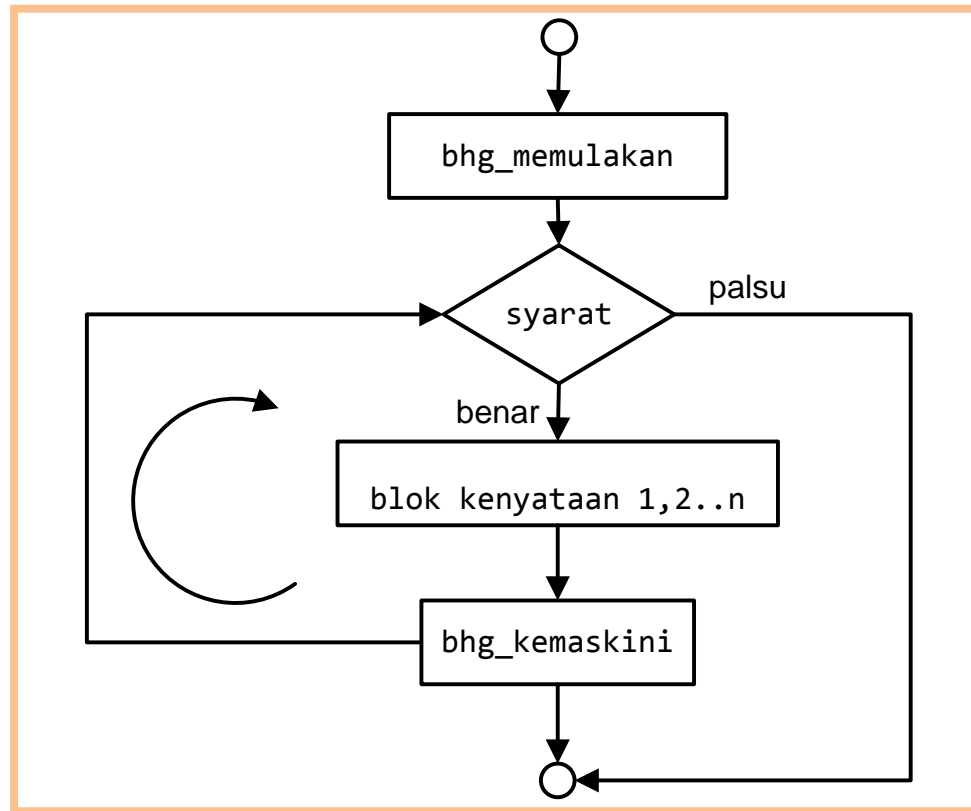
## 5.6 Gelung dengan for

- ❑ Format penggunaan suruhan gelung for dalam atur cara :

```
for (bhg_memulakan ; syarat ; bhg_kemaskini)
{
    kenyataan1;
    kenyataan2;
    :
    kenyataann;
}
```

## 5.6 Gelung dengan for

- Cara aliran operasi gelung for ditunjukkan di carta alir :





## 5.6 Gelung dengan for

- ❑ Pelaksanaan semantik pada carta alir boleh disenaraikan mengikut langkah-langkah berikut:
  1. Pertama, ungkapan di `bhg_memulakan` dilaksanakan. Bahagian dilaksanakan sekali sahaja.
  2. Kemudian syarat diuji yang akan menghasilkan sama ada nilai benar (bukan nilai 0) atau palsu (0).
  3. Jika nilai syarat adalah benar, blok kenyataan-kenyataan  $1, 2 \dots n$  akan dilaksanakan, tetapi jika palsu gelung akan ditamatkan, pelaksanaan seterusnya adalah kenyataan selepas blok kenyataan `for`.
  4. Setelah kenyataan pada blok kenyataan-kenyataan  $1, 2 \dots n$  akan dilaksanakan untuk syarat yang benar, seterusnya ungkapan di `bhg_kemaskini` dilaksanakan dan pelaksanaan akan ulang ke langkah (2).

## 5.6 Gelung dengan for

- Contoh aturcara mempamerkan sifir bagi 11 dengan gelung for

```
1:  #include <stdio.h>
2:
3:  int main ()
4:  {
5:      int nomb_sifir, pengira, kiraan_max;
6:
7:      nomb_sifir = 11;
8:      kiraan_max =12;
9:
10:     for ( pengira = 1; pengira <= kiraan_max; pengira++ )
11:         printf ("%d X %d  = %d \n", nomb_sifir, pengira,
12:                 pengira*nomb_sifir);
13:
14:     return 0;
15: }
16:
```

Corak output:

```
11 X 1 = 11
11 X 2 = 22
      :
11 X 12 = 132
```

## 5.6 Gelung dengan for

- Menjejak pelaksanaan gelung for sifir darab bagi 11:

pengira <= kiraan_max (syarat)	Output	pengira
-		1
1<=12 (benar)	11 X 1 = 11	2
2<=12 (benar)	11 X 2 = 22	3
3<=12 (benar)	11 X 3 = 33	4
4<=12 (benar)	11 X 4 = 44	5
5<=12 (benar)	11 X 5 = 55	6
6<=12 (benar)	11 X 6 = 66	7
7<=12 (benar)	11 X 7 = 77	8
8<=12 (benar)	11 X 8 = 88	9
9<=12 (benar)	11 X 9 = 99	10
10<=12 (benar)	11 X 10 = 110	11
11<=12 (benar)	11 X 11 = 121	12
12<=12 (benar)	11 X 12 = 132	13
13<=12 (palsu)		

## 5.6 Gelung dengan for

- ❑ Ungkapan-ungkapan `bhg_memulakan` atau `bhg_kemaskini` pada gelung `for` boleh diabaikan tetapi tanda semikolon mesti di kekalkan.
- ❑ contoh jika gelung `for` pada aturcara ditulis semula tanpa mengubah hasil pelaksanaan:

1. Mengabaikan ungkapan `bhg_memulakan`.

```
pengira = 1;
for ( ; pengira <= kiraan_max; pengira++ )
    printf ("%d X %d = %d \n", nomb_sifir, pengira,
pengira*nomb_sifir);
```

2. Mengabaikan ungkapan `bhg_kemaskini`.

```
for ( pengira = 1; pengira <= kiraan_max;)
{
    printf ("%d X %d = %d \n", nomb_sifir, pengira,
pengira*nomb_sifir);
    pengira++;
}
```

## 5.6 Gelung dengan for

- ❑ contoh jika gelung for pada aturcara ditulis semula tanpa mengubah hasil pelaksanaan (sambungan):
  3. Mengabaikan ungkapan `bhg_memulakan` dan `bhg_kemaskini`.

```
pengira = 1;
for (; pengira <= kiraan_max;)
{
    printf ("%d X %d = %d \n", nomb_sifir, pengira,
           pengira*nomb_sifir);
    pengira++;
}
```

## 5.7 Penggunaan Struktur Kawalan Pemilihan dan Gelung

Pertimbangkan satu masalah yang perlu dibangunkan untuk mengira purata markah pelajar-pelajar, yang mana markah-markah pelajar perlu dibaca dari fail markah.dat.

Isi fail markah.dat :

```
12
56
89
76
78
```

1. Ada ulangan dalam PM?  
= Ya
2. Jika soalan 1 'Ya' ,tahukah berapa kali perlu diulang?  
= Tidak
3. Jika soalan 2 'tidak', bilakah harus berhenti atau apa syarat untuk berhenti.  
= Sehingga habis semua markah dibaca dalam fail.

## 5.7 Penggunaan Struktur Kawalan Pemilihan dan Gelung

- ❑ 2 cara untuk memeriksa penghujung fail:
  1. EOF - pemalar penghujung fail
  2. feof() - macro yang memulangkan nilai 0 selagi pemalar EOF belum dijumpai
- ❑ Atur cara:

```
1: #include <stdio.h>
2: #include<process.h>
3:
4: int main (void)
5: {
6:     FILE *masuk;
7:     int jumlah = 0, bil = 0, markah;
8:
9:     if ((masuk = fopen("markah.dat", "r"))== NULL)
10:    {
11:        puts("Ralat dalam pembukaan fail.");
12:        exit(-1); /*Tamatkan Atur cara*/
13:    }
14:    printf("Markah-markah \n");
15:    while ((fscanf(masuk,"%d", &markah))!=EOF)
```

## 5.7 Penggunaan Struktur Kawalan Pemilihan dan Gelung

- Atur cara (sambungan):

```
16:     while ((fscanf(masuk,"%d", &markah))!=EOF)
17:     {
18:         printf("%5d\n", markah);
19:         jumlah+=markah;
20:         bil++;
21:     }
22:     printf("Purata markah pelajar %0.2f", (float)jumlah/bil);
23:
24:     return 0;
25: }
```

- Penyelesaian ini boleh ditukar kepada penggunaan fungsi feof() dengan menukar gelung yang memproses bacaan markah, cetak markah dan kira jumlah markah.



## 5.7 Penggunaan Struktur Kawalan Pemilihan dan Gelung

- Pertukaran kepada fungsi feof():

```
fscanf(masuk,"%d", &markah);  
while (!feof(masuk))  
{  
    printf("%5d\n", markah);  
    jumlah+=markah;  
    bil++;  
    fscanf(masuk,"%d", &markah);  
}
```

- Output aturcara:

Markah-markah

12

56

89

76

78

ir Purata markah pelajar 62.20

## 5.8 Struktur Kawalan Bersarang

- Kedua-dua struktur kawalan boleh digunakan secara bersarang iaitu sama ada **struktur pemilihan bersarang** atau **gelung bersarang**.
- **Pemilihan bersarang** mengandungi satu pemilihan luaran dan satu atau lebih pemilihan dalaman.
- **Gelung bersarang** mengandungi satu gelung luaran dan satu atau lebih gelung dalaman.
- **struktur kawalan bersarang campuran** - melibatkan satu jenis struktur sebagai struktur luaran yang mengandungi satu struktur kawalan lain sebagai struktur kawalan dalaman.

## 5.8.1 Pemilihan Bersarang

- Pemilihan bersarang biasanya dibuat dengan pemilihan `if` dan dua cara `if` bersarang digunakan iaitu keputusan `if` **berbilang alternatif** atau **multi-arah** dan syarat `if` **beraras**
- contoh tiga aras `if` bersarang:

```
if (syarat1)
  if (syarat2)
    if (syarat3)
      bhg_benar;
```

- `if` bersarang keputusan berbilang alternatif digunakan bila hanya satu data atau syarat yang melibatkan satu pemboleh ubah sahaja yang digunakan untuk menentukan bahagian yang benar.

## 5.8.1 Pemilihan Bersarang

- Contoh : masalah untuk mengira luas satu segiempat

### Input

Media: papan kekunci

panjang nilai masukan panjang sisi segiempat

lebar nilai masukan lebar sisi segiempat

### Output

Media: skrin

luas hasil proses luas segiempat

### Proses

1. Mencari magnitud sesuatu nombor:

jika nombor adalah negatif, magnitud =  $-(\text{nombor})$

jika nombor adalah positif, magnitud = nombor

2. luas = panjang x lebar

## 5.8.1 Pemilihan Bersarang

- Reka bentuk penyelesaian masalah untuk mengira luas satu segiempat dengan pemilihan bersarang:

```
Mengira luas satu segiempat
Begin
  baca panjang & lebar dari pengguna
  if panjang = 0 atau lebar = 0 then
    begin
      cetak mesej amaran
      luas = 0
    end
  else
    begin
      if panjang negatif then
        panjang = magnitud panjang
      endif
      if lebar negatif then
        lebar = magnitud lebar
      endif
      luas = panjang x lebar
    end
  endif
  cetak luas
End
```

## 5.8.1 Pemilihan Bersarang

➤ Atur cara:

```
1: #include <stdio.h>
2:
3: int main ()
4: {
5:     float luas, panjang, lebar;
6:
7:     printf ("masukkan nilai panjang dan lebar\n");
8:     scanf ("%f %f", &panjang, &lebar);
9:     if (panjang== 0 || lebar ==0)
10:    {
11:        printf("Salah satu nilai sisi 0\n");
12:        luas=0;
13:    }
14:    else
15:    {
```

## 5.8.1 Pemilihan Bersarang

➤ Atur cara (sambungan):

```
16:         if (panjang < 0)
17:             panjang = -panjang;    /*cari magnitud panjang*/
18:         if (lebar < 0 )
19:             lebar = -lebar;        /* cari magnitud lebar*/
20:             luas = panjang*lebar;
21:         }
22:         printf ("luas segiempat ialah %0.2f\n", luas);
23:
24:     return 0;
25: }
```

## 5.8.2 Gelung Bersarang

- gelung bersarang merujuk kepada struktur ulangan gelung yang mengandungi satu atau gelung lain.
- Contoh atur cara menggunakan gelung bersarang bagi menyelesaikan masalah untuk mempamerkan sifir darab bagi 1 hingga 5 dan hanya hasil darab yang kurang serta sama dengan 5 sahaja akan dicetak.
- Gelung `for` merupakan gelung luaran dan gelung `while` merupakan gelung dalaman.



## 5.8.2 Gelung Bersarang

□ Atur cara:

```
1: #include <stdio.h>
2:
3: int main ()
4: {
5:     int nomb_sifir, hasil_darab, pengira, sifir_max;
6:
7:     sifir_max=5; hasil_darab=0; pengira = 1;
8:
9:     for ( nomb_sifir=1; nomb_sifir<=sifir_max ;nomb_sifir++)
10:    {
11:        printf("\nSifir %d\n", nomb_sifir);
12:        hasil_darab = pengira*nomb_sifir;
13:
14:        while (hasil_darab<=5)
15:        {
16:            printf ("%d X %d = %d \n", nomb_sifir,
17:                pengira, hasil_darab);
18:            pengira++;
19:            hasil_darab = pengira*nomb_sifir;
20:        }
21:        pengira=1;
22:    }
23:    return 0;}
```

## 5.8.2 Gelung Bersarang

□ Menjejak pelaksanaan gelung bersarang sifir 1 hingga 5 :

Gelung luaran		Gelung dalaman			
nomb_sifir	nomb_sifir<=sifir_max (syarat)	hasil_darab <=5 (syarat)	Output	pengira	hasil_darab
1	1<=5 (benar)			**1	*1
1		1<=5 (benar)	1 X 1 = 1	2	2
1		2<=5 (benar)	1 X 2 = 2	3	3
1		3<=5 (benar)	1 X 3 = 3	4	4
1		4<=5 (benar)	1 X 4 = 4	5	5
1		5<=5 (benar)	1 X 5 = 5	6	6
1		6<=5 (palsu)		*1	
2	2<=5 (benar)			1	*2
2		2<=5 (benar)	2 X 1 = 2	2	4
2		2<=5 (benar)	2 X 2 = 4	3	6
2		6<=5 (palsu)		*1	
3	3<=5 (benar)			1	*3
3		3<=5 (benar)	3 X 1 = 3	2	6
4		6<=5 (palsu)		*1	
4	4<=5 (benar)			1	*4
4		4<=5 (benar)	4 X 1 = 4	2	8
4		8<=5 (palsu)		*1	
5	5<=5 (benar)			1	*5
5		5<=5 (benar)	5 X 1 = 4	2	10
5		10<=5 (palsu)		*1	
6	6<=5 (palsu)				

## 5.8.2 Gelung Bersarang

□ output :

Sifir 1

$$1 \times 1 = 1$$

$$1 \times 2 = 2$$

$$1 \times 3 = 3$$

$$1 \times 4 = 4$$

$$1 \times 5 = 5$$

Sifir 2

$$2 \times 1 = 2$$

$$2 \times 2 = 4$$

Sifir 3

$$3 \times 1 = 3$$

Sifir 4

$$4 \times 1 = 4$$

Sifir 5

$$5 \times 1 = 5$$

## 5.8.3 Kawaln Bersarang Campuran

- Kawalan bersarang campuran melibatkan penggunaan kedua-dua struktur kawalan pemilihan dan gelung yang digunakan secara bersarang.
- Berbeza dengan struktur kawalan pemilihan dan gelung, penggunaannya tidak bersarang kerana dua jenis struktur kawalan digunakan pada aras yang sama.
- Penggunaan kawalan bersarang campuran membolehkan struktur kawalan gelung digunakan dalam struktur kawalan pemilihan atau sebaliknya.
- Seperti kawalan gelung dan pemilihan bersarang, kawalan bersarang campuran boleh ditulis lebih daripada dua aras.

## 5.8.3 Kawalan Bersarang Campuran

- Contoh atur cara Mencetak laporan bilangan aksara di dalam fail data:

```
1: #include <stdio.h>
2: #include <process.h>
3: #include <ctype.h>
4:
5: int main(void) {
6:     int bilHbesar=0, bilHkecil=0, bilnom=0, bilLain=0;
7:     char aksara;
8:     FILE *input;
9:
10:    input = fopen("teks.dat", "r");
11:    if (input== NULL) {
12:        puts("Ralat dalam pembukaan fail.");
13:        exit(-1);
14:    }
15:
16:    aksara = fgetc(input);
17:    while (aksara!=EOF) {
18:        putc(aksara, stdout);
19:        if (isupper(aksara))
20:            bilHbesar++;
```

## 5.8.3 Kawaln Bersarang Campuran

- Contoh atur cara (sambungan):

```
21:         else if (islower(aksara))
22:             bilHkecil++;
23:         else if (isdigit(aksara))
24:             bilnom++;
25:         else
26:             bilLain++;
27:         aksara = fgetc(input);
28:     }
29:
30:     printf ("\nBilangan huruf besar %d", bilHbesar);
31:     printf ("\nBilangan huruf kecil %d", bilHkecil);
32:     printf ("\nBilangan digit %d", bilnom);
33:     printf ("\nBilangan aksara lain %d", bilLain);
34:
35:     return 0;
36: }
```

## 5.8.3 Kawaln Bersarang Campuran

- Contoh input fail teks .dat:

```
Universiti Teknologi Malaysia  
81310 UTM, Skudai  
Johor, Malaysia  
http://www.utm.my/
```

- Contoh output fail teks .dat:

```
Universiti Teknologi Malaysia  
81310 UTM, Skudai  
Johor, Malaysia  
http://www.utm.my/  
  
Bilangan huruf besar 9  
Bilangan huruf kecil 52  
Bilangan digit 5  
Bilangan aksara lain 18
```

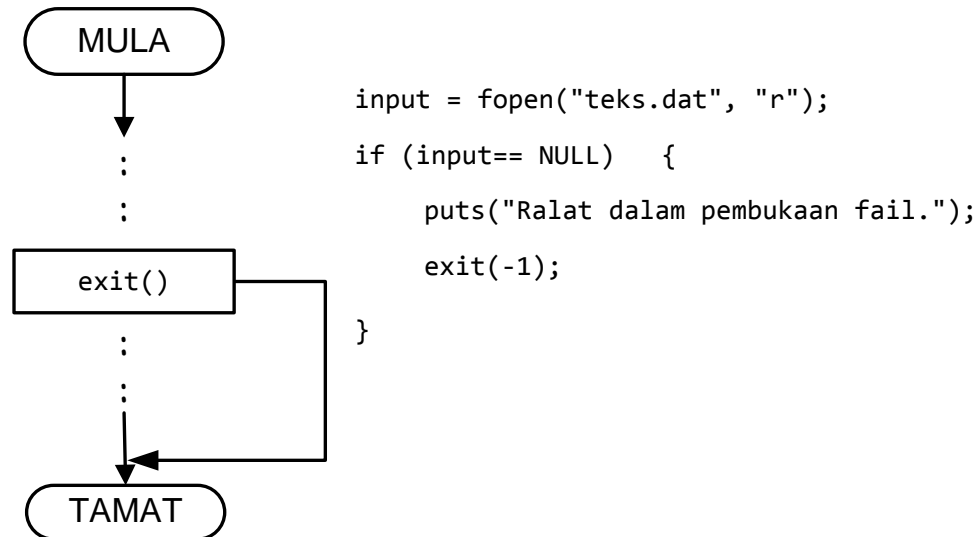
## 5.9 Pintas Struktur

- Tiga struktur kawalan iaitu jujukan, pemilihan dan gelung menentukan aliran pelaksanaan atur cara.
  
- wujud situasi yang memerlukan atur cara untuk keluar dari struktur tertentu dan C menyediakan arahan untuk tujuan ini, antaranya:
  1. fungsi `exit()`
  2. pernyataan `break`
  3. pernyataan `continue`



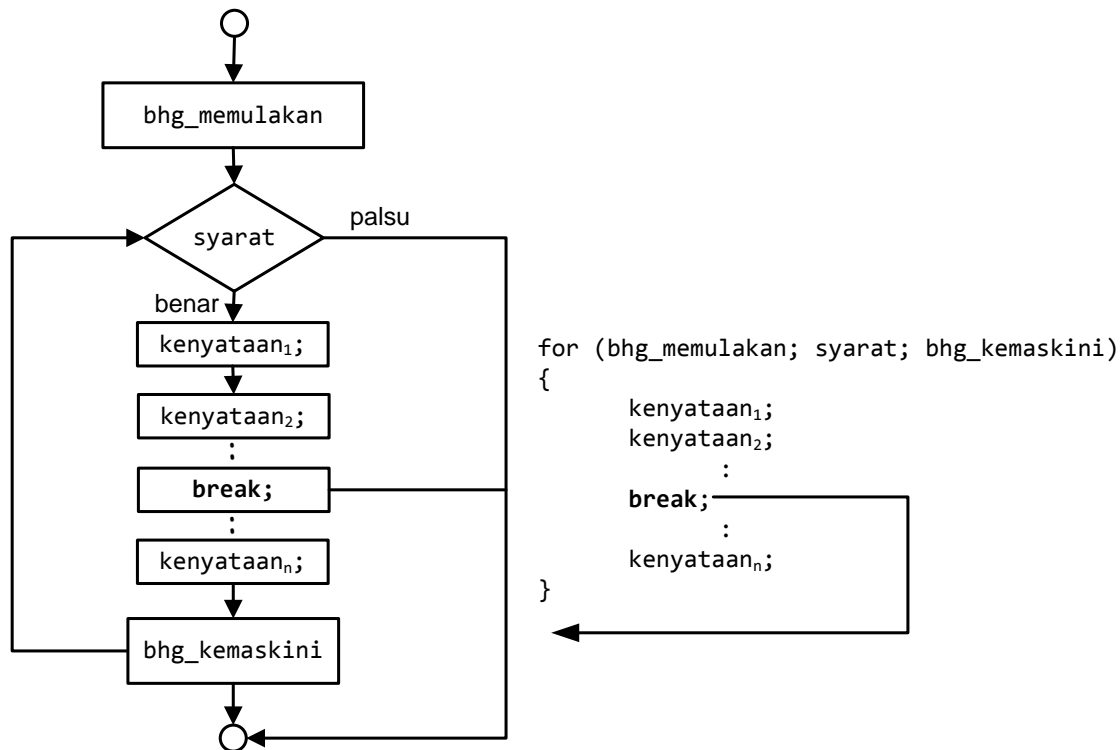
## 5.9 Pintas Struktur

- Contoh aliran pintas dengan fungsi exit():



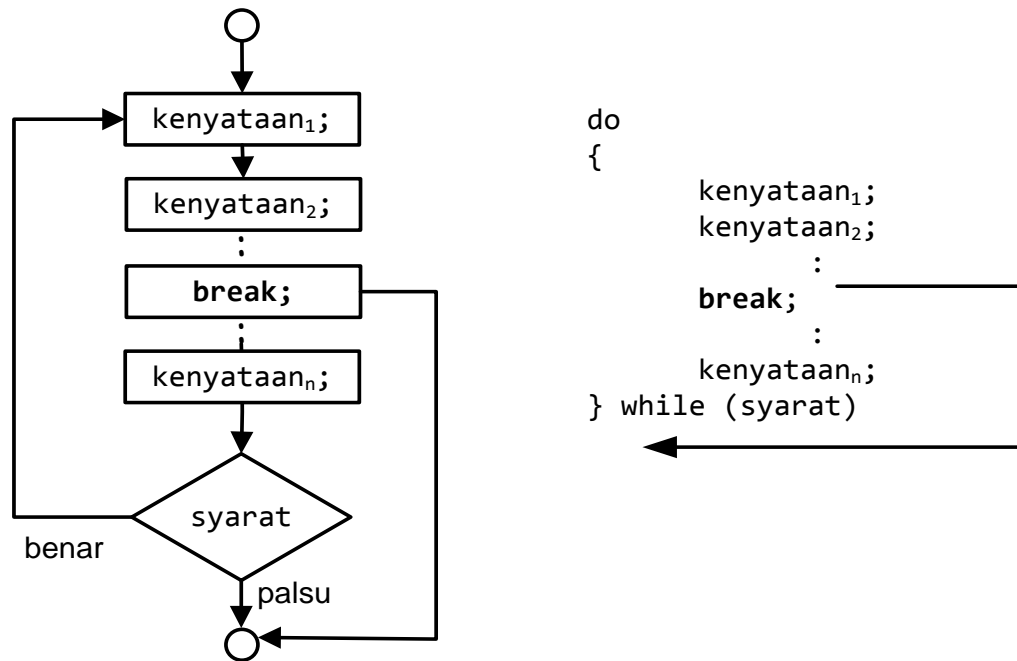
# 5.9 Pintas Struktur

- Contoh aliran pintas break dengan gelung for:



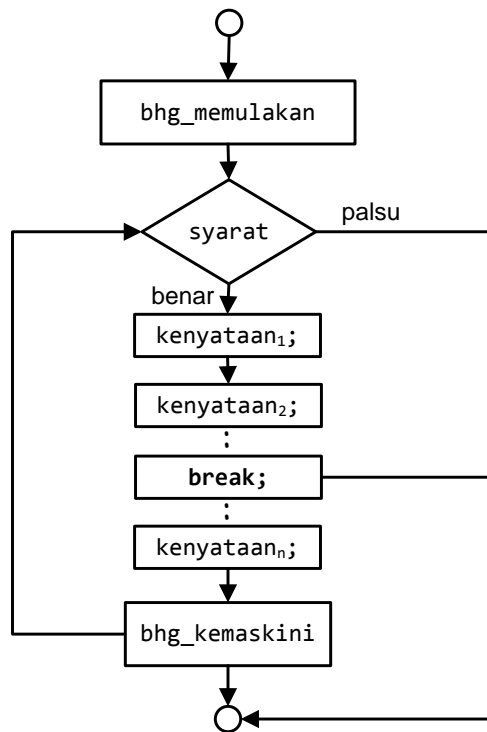
# 5.9 Pintas Struktur

- Contoh aliran pintas break dengan gelung while:



# 5.9 Pintas Struktur

- Contoh aliran pintas break dengan gelung do while:



```

for (bhg_memulakan ; syarat ; bhg_kemaskini)
{
    kenyataan1;
    kenyataan2;
    :
    break;
    :
    kenyataan_n;
}
  
```

## 5.9 Pintas Struktur

- Contoh kenyataan break pada gelung for :

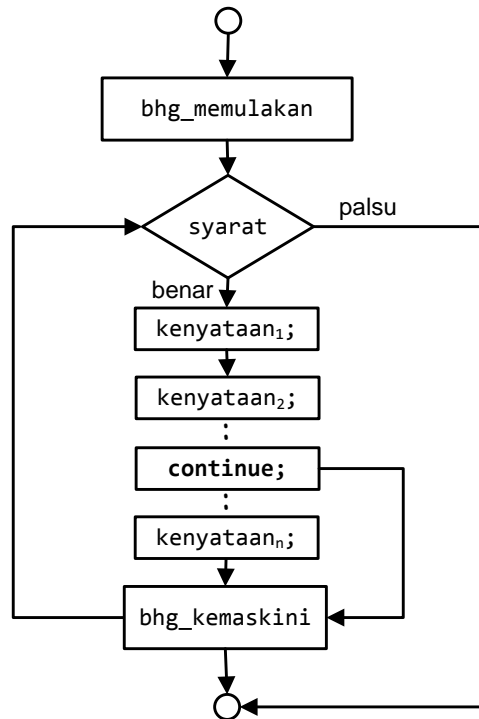
```
for (i=1; i<=3; i++)
{
    printf("%d\n", i);
    if (i==2)
        break;
    printf("Bahagian bawah gelung\n");
}
printf("Keluar dari gelung");
```

- Output:

```
1
Bahagian bawah gelung
2
Keluar dari gelung
```

# 5.9 Pintas Struktur

- Contoh aliran pintas kenyataan continue dengan gelung for:

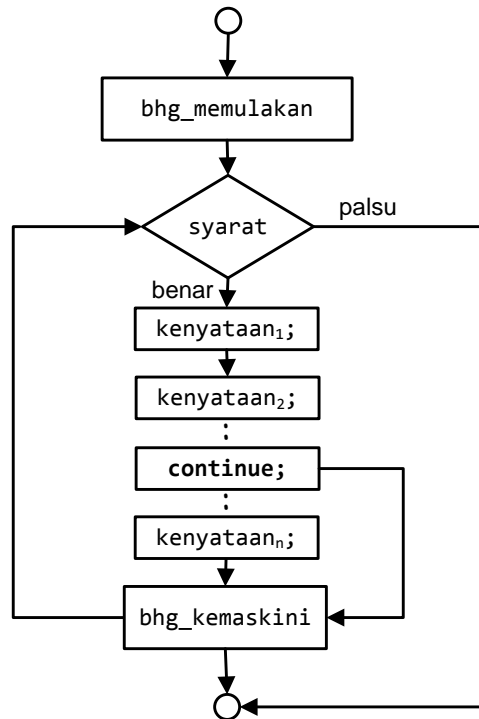


```

for (bhg_memulakan ; syarat ; bhg_kemaskini)
{
    kenyataan1;
    kenyataan2;
    :
    continue;
    :
    kenyataan_n;
}
  
```

# 5.9 Pintas Struktur

- Contoh aliran pintas kenyataan continue dengan gelung for:



```

for (bhg_memulakan ; syarat ; bhg_kemaskini)
{
    kenyataan1;
    kenyataan2;
    :
    continue;
    :
    kenyataan_n;
}
  
```

## 5.9 Pintas Struktur

- Contoh kenyataan continue didalam gelung for :

```
for (i=1; i<=3; i++)  
{  
    printf("%d\n", i);  
    if (i==2)  
        continue;  
    printf("Bahagian bawah gelung\n");  
}  
printf("Keluar dari gelung");
```

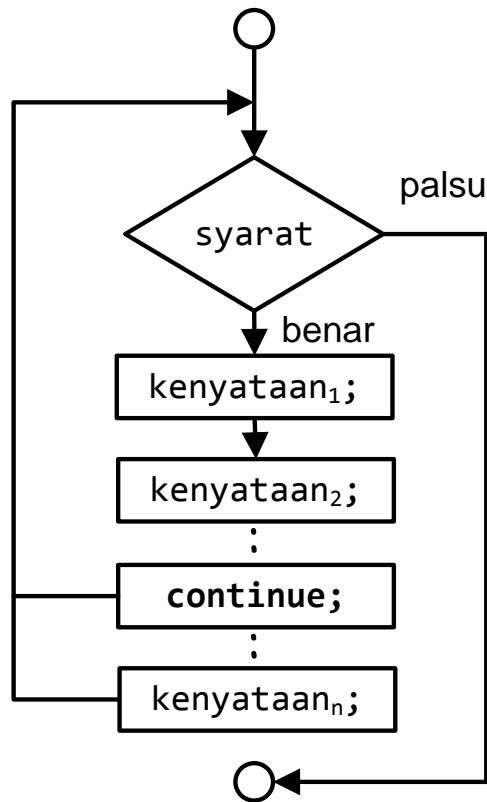
- Output:

```
1  
Bahagian bawah gelung  
2  
3  
Bahagian bawah gelung  
Keluar dari gelung
```



## 5.9 Pintas Struktur

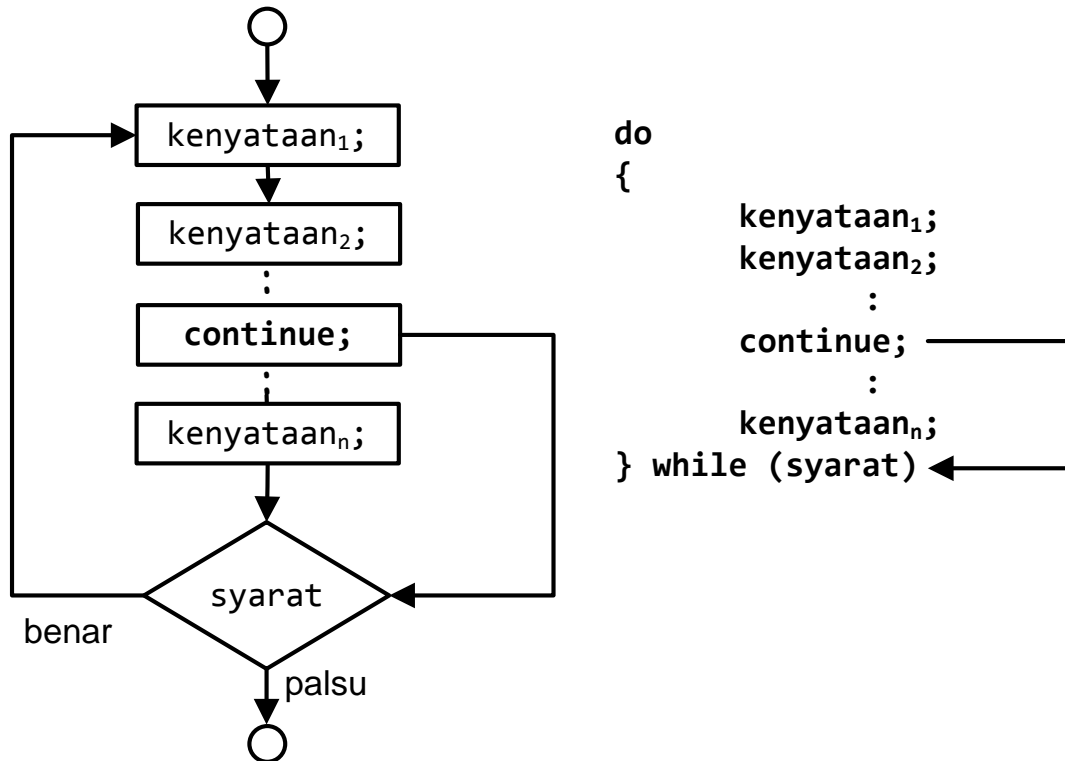
- Contoh aliran pintas kenyataan continue dengan gelung while:



```
while (syarat)
{
    kenyataan1;
    kenyataan2;
    :
    continue;
    :
    kenyataann;
}
```

# 5.9 Pintas Struktur

- Contoh aliran pintas kenyataan continue dengan gelung do while:



## 5.9 Pintas Struktur

- Contoh gelung dengan kenyataan `break` dan `continue`:

```
int i=0;
while (i<10)
{
    if (i<3)
    {
        i+=2;
        printf("%d\n",i);
        continue;
    }
    if (i>6){
        printf("%d\n",i);
        break;
    }
    printf("%d\n",++i);
    printf("Bahagian bawah gelung\n");
}
printf("Keluar dari gelung");
```

## 5.9 Pintas Struktur

- Contoh output gelung dengan kenyataan `break` dan `continue`:

```
2  
4  
5  
Bahagian bawah gelung  
6  
Bahagian bawah gelung  
7  
Bahagian bawah gelung  
7  
Keluar dari gelung
```

© Copyright Universiti Teknologi Malaysia

innovative • entrepreneurial • global