


Bab 2

Percabangan

2.1 Operator Relasional

Operator relasional digunakan untuk memeriksa relasi dan membandingkan nilai dari dua operan. Jika benar akan menghasilkan nilai **TRUE** (direpresentasikan angka 1), jika salah maka akan menghasilkan nilai **FALSE** (direpresentasikan angka 0). Berikut adalah operator relasional dalam bahasa C.

Operator	Simbol	Keterangan	Contoh
Sama dengan	==	Digunakan untuk memeriksa apakah kedua operan memiliki nilai yang sama.	$5 == 2$  (FALSE) $5 == 5$ (TRUE)
Tidak Sama dengan	!=	Digunakan untuk memeriksa apakah kedua operan memiliki nilai yang tidak sama.	$5 != 2$ (TRUE) $5 != 5$ (FALSE)
Lebih besar	>	Digunakan untuk membandingkan apakah operan pertama lebih besar nilainya dari operan kedua.	$5 > 2$ (TRUE) $5 > 5$ (FALSE) $2 > 4$ (FALSE)
Lebih kecil	<	Digunakan untuk membandingkan apakah operan pertama lebih kecil nilainya dari operan kedua.	$5 < 2$ (FALSE) $5 < 5$ (FALSE) $2 < 4$ (TRUE)
Lebih besar sama dengan	>=	Digunakan untuk membandingkan apakah operan pertama lebih besar atau sama nilainya dari operan kedua.	$5 >= 2$ (TRUE) $5 >= 5$ (TRUE) $2 >= 4$ (FALSE)
Lebih kecil sama dengan	<=	Digunakan untuk membandingkan apakah operan pertama lebih kecil atau sama nilainya dari operan kedua.	$5 <= 2$ (FALSE) $5 <= 5$ (TRUE) $2 <= 4$ (TRUE)

2.2 Operator Logika

Operator logika digunakan untuk melakukan tes pada kondisi/ekspresi, apakah kondisi tersebut benar atau salah. Operator logika hanya akan menghasilkan nilai **TRUE** (jika benar) atau **FALSE** (jika salah). **TRUE** direpresentasikan oleh angka 1, sedangkan **FALSE** oleh angka 0. Operator-operator logika dalam bahasa C adalah sebagai berikut.

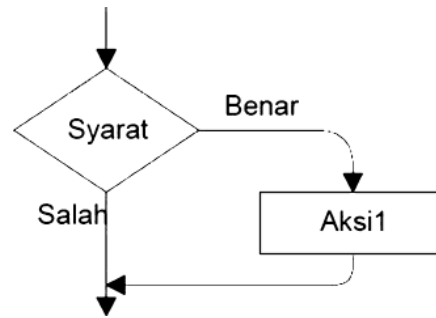
Operator	Simbol	Keterangan	Nilai Kebenaran
Logical NOT	!	Operator NOT digunakan untuk membalikkan kondisi, TRUE menjadi FALSE dan FALSE menjadi TRUE.	!1 = 0 !0 = 1
Logical AND	&&	Operator AND akan menghasilkan nilai TRUE jika kedua operan mempunyai nilai TRUE.	1 && 1 = 1 0 && 1 = 0 1 && 0 = 0 0 && 0 = 0
Logical OR		Operator OR akan menghasilkan nilai TRUE jika salah satu operan mempunyai nilai TRUE.	1 1 = 1 0 1 = 1 1 0 = 1 0 0 = 0

Operator logika pada umumnya digunakan bersamaan dengan operator relasional untuk melakukan tes pada ekspresi yang berhubungan dengan kebenaran suatu kondisi. Penggunaan paling umum adalah untuk melakukan percabangan (akan dipelajari di bagian selanjutnya). Contoh:

```
int a, b, c, d;  
a = 11;  
b = 24;  
c = 11;  
d = ((a == c) && (b > a));           // 1 (TRUE)  
d = ((a >= b) || (a < c));           // 0 (FALSE)  
d = ((b != b) || (b > c)) && (c == a); // 1 (TRUE)
```

2.3 Pernyataan if

Pada model pernyataan if, sebuah aksi akan dikerjakan jika syarat yang diajukan bernilai benar.



Gambar 1 : Pernyataan if dalam bentuk flowchart

Sintaks yang digunakan dalam percabangan menggunakan if adalah sebagai berikut.

```
if (<Ekspresi/Kondisi>) {  
    //kode yang akan dieksekusi jika kondisi tersebut benar  
}
```

Cara kerja percabangan if yaitu memeriksa dan mengevaluasi suatu kondisi untuk menentukan apakah instruksi selanjutnya dalam bracket akan dijalankan atau tidak oleh program.

- Jika kondisi tersebut bernilai **TRUE (1)**, kode yang di dalam bracket **akan dieksekusi**.
- Sebaliknya jika kondisi tersebut bernilai **FALSE (0)**, kode yang di dalam bracket **tidak akan dieksekusi**.

Contoh dari pernyataan if adalah,

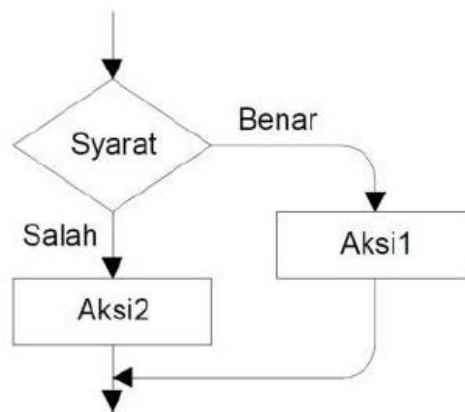
```
int a = 5;  
if(a>0) {  
    printf("Nilai a positif");  
}
```

Output:

Nilai a positif

2.4 Pernyataan if-else

Pada model pernyataan ini, aksi 1 akan dikerjakan jika syarat bernilai benar sedangkan jika salah maka aksi2 yang akan dikerjakan seperti pada flowchart berikut:



Gambar 2 : Pernyataan if...else dalam bentuk flowchart

Sintaks yang digunakan dalam percabangan menggunakan if-else adalah sebagai berikut.

```

if (<Ekspresi/Kondisi>) {
    //kode yang akan dieksekusi jika kondisi tersebut benar
}else {
    //kode yang akan dieksekusi jika kondisi tersebut salah
}
  
```

Cara kerja percabangan if-else diatas yaitu memeriksa kondisi dalam if.

- Jika kondisi tersebut bernilai TRUE (1), Program akan menjalankan kode di dalam bracket if.
- Sebaliknya jika kondisi tersebut bernilai FALSE (0), kode di dalam bracket else yang akan dijalankan.

Contoh dari pernyataan if else adalah,

```

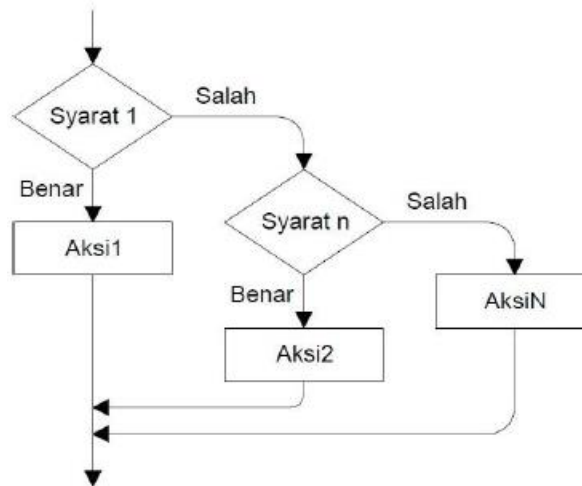
int a = 5;
if(a>0) {
    printf("Nilai a positif");
}else{
    printf("Nilai a tidak positif");
}
  
```

Output:

Nilai a positif

2.5 Pernyataan if-else Bertingkat

Pernyataan if ... else dapat dibuat secara bertingkat sesuai dengan kebutuhan penggunaannya pada program.



Gambar 3 : Pernyataan if...else bertingkat dalam bentuk flowchart

Sintaks yang digunakan dalam percabangan menggunakan if-else if adalah sebagai berikut.

```

if (<Ekspresi/Kondisi>) {
    //kode yang akan dieksekusi jika kondisi tersebut benar
}else if (<Ekspresi/Kondisi>) {
    //kode yang akan dieksekusi jika kondisi tersebut salah
}
// boleh menambahkan else{} apabila perlu
  
```

Cara kerja percabangan if-else bertingkat yaitu memeriksa kondisi dalam if.

- Jika kondisi tersebut bernilai TRUE (1), Program akan menjalankan kode di dalam bracket if.
- Apabila kondisi pertama tidak memenuhi, maka ia akan memeriksa kondisi didalam else if, apabila bernilai TRUE (1), maka ia akan menjalankan perintah dalam bracket tersebut, apabila tidak maka ia akan menjalankan sequence selanjutnya.
- Apabila kita menyediakan statement else di akhir, maka ketika seluruh kondisi if dan else if tidak memenuhi atau FALSE (0), maka secara otomatis ia akan menjalankan perintah di dalam else tersebut.

Contoh dari pernyataan if-else bercabang adalah,

```

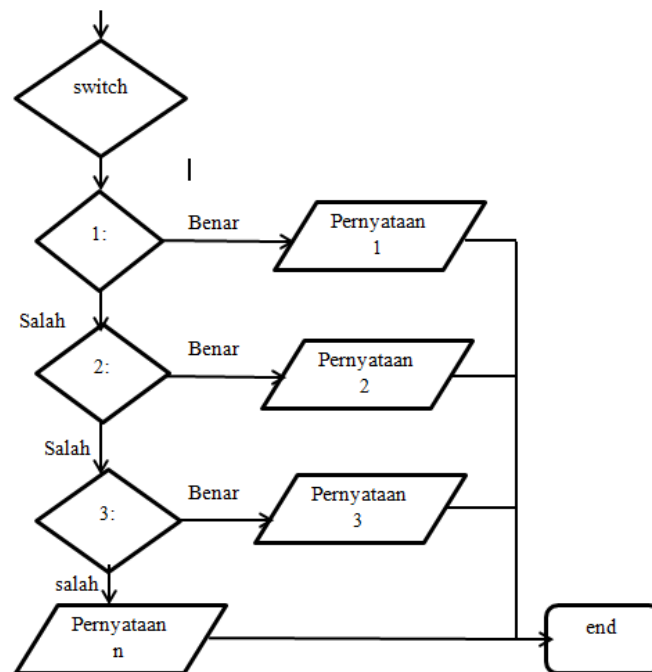
int a = -1;
if(a>0) {
printf("Nilai a positif");
}else if(a<0){
printf("Nilai a negatif");
}else{
printf("Nilai a nol");
}
  
```

Output:

Nilai a negatif

2.6 Pernyataan switch case

Selain penggunaan statemen if untuk memilih diantara banyak alternatif, terdapat pula statemen switch yang memiliki fungsi yang sama, untuk memilih diantara banyak alternatif berdasarkan sebuah kondisi. Kondisi pada statemen switch berisi ekspresi yang dapat menggunakan sebuah variabel tunggal bertipe int yang akan diperiksa nilainya di setiap blok case.



Gambar 4 : Contoh flowchart switch case

Sintaks untuk Case Switch:

switch (ekspresi)

{

case ekspresi-konstan :

statement;

break;

case ekspresi-konstan :

statement;

break;

//anda bisa memiliki jumlah case sebanyak mungkin

//default ketika tidak ada case yang memenuhi

default :

statement;

}

Setiap blok, case harus ditambahkan statement break, karena apabila tidak maka ia akan tetap menjalankan blok case di bawahnya hingga bertemu break lain atau pada akhir blok switch.

Contoh dari pernyataan switch case:

```
switch(pilihan)
{
    case 1:
        printf("bruh\n");
        break;

    case 2:
        printf("bruhy\n");
        break;

    case 3:
        printf("brah\n");
        break;

    default:
        printf("brahy\n");
}
```

Dalam contoh diatas, **ekspresi** yang digunakan adalah **Plat Nomor**, dimana **case-case** nya adalah, huruf plat nomor tersebut, **L,B,D,N**, dan sebagainya.