KONDISIONAL

OBJEKTIF :

1. Mahasiswa mampu memahami tentang operator.
2. Mahasiswa mampu memahami tentang statement `if`.
3. Mahasiswa mampu memahami tentang statement `if-else`.
4. Mahasiswa mampu memahami tentang statement nested `if`.
5. Mahasiswa mampu memahami tentang statement `switch-case`.

2.1 OPERATOR

Operator merupakan sebuah simbol yang digunakan untuk melakukan operasi atau manipulasi. Operasi yang dimaksud adalah operasi yang terjadi di dalam program. Operator yang akan dibahas yaitu:

- Operator Relasional
- Operator Logika

2.1.1 OPERATOR RELASIONAL

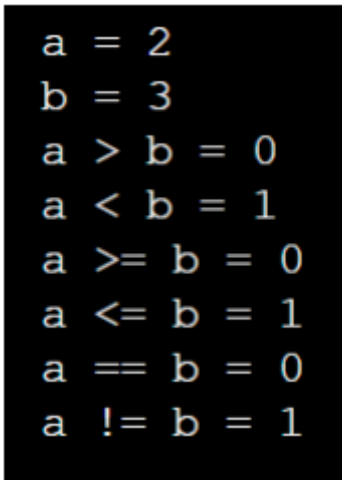
Operator relasional adalah operator yang menguji atau mendefinisikan hubungan antara dua entitas atau variabel. Operator relasional dipakai untuk membandingkan dua buah nilai. Hasil dari operator relasional ini adalah integer **True** atau **False**, karena bahasa C tidak memiliki tipe data `boolean` bawaan, maka, hasilnya adalah `integer` 1 atau 0. Di bawah ini merupakan jenis-jenis operator yang termasuk ke dalam operator relasional:

Operator	Keterangan	Contoh	Hasil
`>`	Lebih besar	`3 > 2`	`0` (false)
`<`	Lebih kecil	`3 < 5`	`1` (true)
`>=`	Lebih besar atau sama dengan	`3 >= 2`	`1` (true)
`<=`	Lebih kecil atau sama dengan	`3 <= 3`	`1` (true)
`==`	Sama dengan (Bukan Penugasan)	`3 == 3`	`1` (true)
`!=`	Tidak sama dengan	`3 != 3`	`0` (false)

Contoh program `kondisi1.c`:

```c
// Program menampilkan hasil relasi kondisi1.c

// File library
#include <stdio.h>
// Fungsi main
int main(){
    int a = 2;
    int b = 3;

    // Mencetak nilai a dan b
    printf("a = %d\n", a);
    printf("b = %d\n", b);

    // Menampilkan hasil operasi dari operator relasional
    printf("a > b = %d\n", a > b);
    printf("a < b = %d\n", a < b);
    printf("a >= b = %d\n", a >= b);
    printf("a <= b = %d\n", a <= b);
    printf("a == b = %d\n", a == b);
    printf("a != b = %d\n", a != b);
}
```

Output program `kondisi1.c`:



2.1.2 OPERATOR LOGIKA

Operator logika digunakan untuk menghubungkan dua ekspresi menjadi satu ekspresi. Operator logika dipakai untuk menghasilkan nilai `boolean` **True** atau **False** dari dua kondisi atau lebih. Di bawah ini merupakan operator yang tergolong ke dalam operator logika:

Operator	Nama	Keterangan
`&&`	And	Akan menghasilkan `1` jika kedua operand `1`
`||`	Or	Akan menghasilkan `1` jika salah satu operand `1`
`!`	Not	Akan menghasilkan `1` jika operand `0`

Di bawah ini merupakan tabel kemungkinan pada operasi yang menggunakan operator logika (|| dan &&):

Ekspresi_1	Ekspresi_2	Hasil ||	Hasil &&
Salah	Salah	Salah	Salah
Salah	Benar	Benar	Salah
Benar	Salah	Benar	Salah
Benar	Benar	Benar	Benar

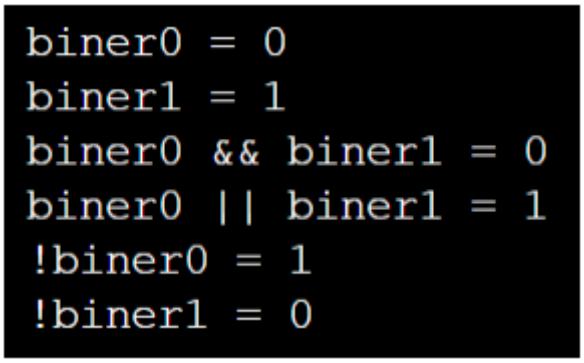
Bentuk susunan operasi logika apabila ditemukan ada lebih dari dua ekspresi:

`(a == b) || (a || c)`

Contoh program `operatorlogika.c`:

```c
#include <stdio.h>

int main(){
    int biner0 = 0; // false
    int biner1 = 1; // true

    printf("biner0 = %d\n", biner0);
    printf("biner1 = %d\n", biner1);

    // logika AND
    printf("biner0 && biner1 = %d\n", biner0 && biner1);

    // logika OR
    printf("biner0 || biner1 = %d\n", biner0 || biner1);

    // logika NOT
    printf("!biner0 = %d\n", !biner0);
    printf("!biner1 = %d\n", !biner1);
}
```

Output program `operatorlogika.c`:



2.2 SELEKSI KONDISI PADA BAHASA C

Dalam Bahasa C, terdapat suatu kasus di mana kita akan dihadapi oleh suatu keadaan atau kejadian untuk memilih suatu pilihan. Kejadian tersebut dinamakan dengan kondisi. Kondisi digunakan untuk pengambilan keputusan. Pengambilan keputusan dalam Bahasa C dapat dilakukan dengan cara:

- Statement `if`
- Statement `switch`
- Operator kondisi `(?:)`

Di bawah ini merupakan pembahasan detail mengenai ketiga cara pengambilan keputusan yang telah dipaparkan di atas.

2.2.1 STATEMENT `if`

Pada `if` sederhana terdapat tanda `( )` tempat kondisi dituliskan. Lalu terdapat tanda `{ }` tempat dimana *statement* ditulis. Kemudian terdapat *statement* yang merupakan proses yang terjadi apabila kondisi terpenuhi. Pada bentuk `if` ini dapat digunakan untuk memproses beberapa *statement* sekaligus dalam suatu blok kondisi jika kondisi yang diseleksi benar.

Bentuk umum dari `if` Sederhana adalah sebagai berikut:

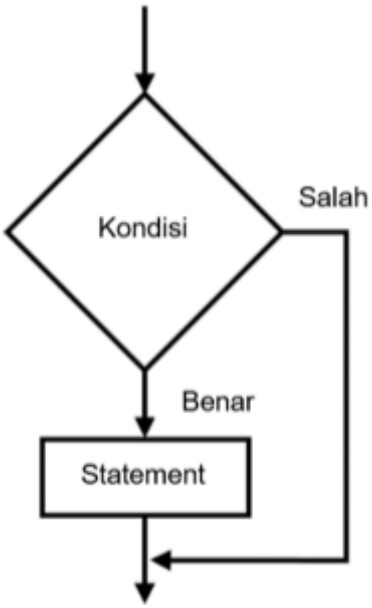
```
if (kondisi)
{
    statement1
    statement2
}
```

Berdasarkan hal di atas, didapat:

- Kondisi adalah suatu kejadian yang digunakan untuk pengambilan keputusan
- *Statement* adalah suatu bagian yang akan dieksekusi apabila kondisi yang menampung *statement* ini bernilai benar. *Statement* dapat berupa sebuah *statement* tunggal atau *statement* majemuk.

Pada bentuk `if` sederhana yang disampaikan di atas, bentuk *statement* dalam kondisinya berupa *statement* majemuk.

Di bawah ini merupakan flowchart dari *statement* `if`:



Contoh program `diskon.c`:

```c
#include <stdio.h>

int main()
{
    // Deklarasi variabel
    float totalBelanja,diskon, totalBayar;

    // Cetak output dan input variabel
    printf("Total belanja kamu adalah : Rp. ");
    scanf("%f", &totalBelanja);

    // Kondisi jika totalBelanja lebih besar dari 100000
    if(totalBelanja > 100000)
    {
        // Menugaskan variabel diskon dengan hasil ekspresi 0.1 dikalikan dengan
totalBelanja
        diskon = 0.1 * totalBelanja;
        // Menghitung variabel totalBayar
        totalBayar = totalBelanja-diskon;
        printf("Selamat! Anda mendapatkan diskon sebesar Rp. %.2f\n", diskon);
        printf("Maka total pembayaran kamu adalah : Rp. %.2f", totalBayar);
    }

    return 0;
}
```

- Output program `diskon.c` jika `totalBelanja <= 100000`:

```
Total belanja kamu adalah : Rp. 100000
```

Pada output program di atas tidak menampilkan output apapun. Hal ini dikarenakan pada program di atas hanya ada satu kondisi, yang mana apabila kondisi tersebut tidak terpenuhi, maka tidak ada *statement* yang dijalankan.

- Output program `diskon.c` jika `totalBelanja <= 100000`:

```
Total belanja kamu adalah : Rp. 101000
Selamat! Anda mendapatkan diskon sebesar Rp. 10100.00
Maka total pembayaran kamu adalah : Rp. 90900.00
```

Pada output program di atas menampilkan suatu *statement* yang menyatakan user mendapatkan diskon. Hal ini disebabkan adanya kondisi kedua yaitu `else` yang akan dieksekusi apabila kondisi `if` tidak terpenuhi.

CATATAN:

Operator perbandingan harus ditulis seperti `==`, karena jika hanya ditulis `=`, maka compiler akan menganalisa bahwa ini adalah operasi penugasan bukan perbandingan. Ketika operator penugasan dijalankan pada *statement* di dalam kondisi, outputnya akan menghasilkan error.

2.2.2 STATEMENT `if – else`

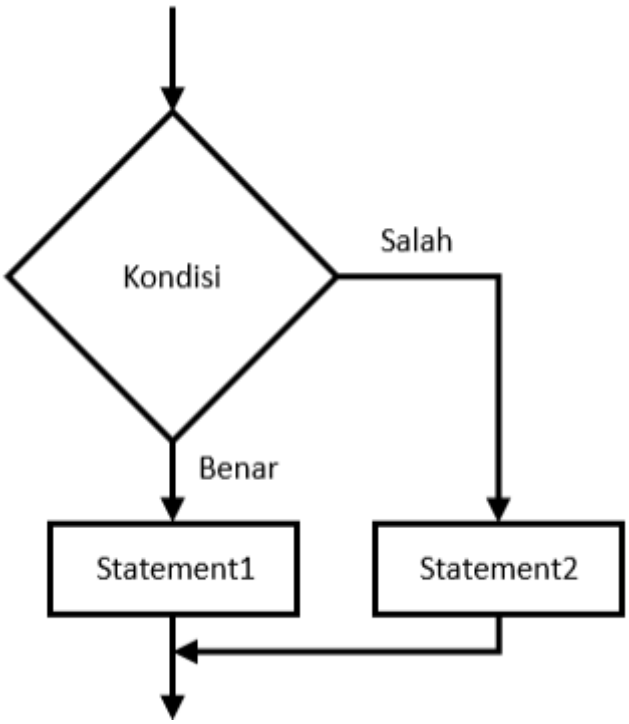
Statement `if – else` merupakan bentuk percabangan yang akan menjalankan *statement* `if` atau *statement* pertama apabila kondisi pertama terpenuhi, dan menjalankan kondisi `else` atau *statement* kedua apabila kondisi pertama tidak terpenuhi.

Bentuk umum dari `if – else` adalah sebagai berikut:

```
if (condition)
{
    statement1;
}
else
{
    statement2;
}
```

Pada *statement* `if – else` jika kondisi bernilai benar maka `statement1` yang dijalankan, tetapi jika kondisi bernilai salah maka, `statement2` yang dijalankan.

Di bawah ini merupakan flowchart dari *statement* `if – else`:



Contoh program `diskon.c`:

```
#include <stdio.h>

int main()
{
  // Deklarasi variabel
  float totalBelanja, diskon, totalBayar;

  // Cetak output dan input variabel
  printf("Total belanja kamu adalah : Rp. ");
  scanf("%f", &totalBelanja);

    // Kondisi jika totalBelanja lebih besar dari 100000
    if(totalBelanja > 100000)
    {
        // Menugaskan variabel diskon dengan hasil ekspresi 0.1 dikalikan dengan
totalBelanja
        diskon = 0.1 * totalBelanja;
        // Menghitung variabel totalBayar
        totalBayar = totalBelanja-diskon;
        printf("Selamat! Anda mendapatkan diskon sebesar Rp. %.2f\n", diskon);
        printf("Maka total pembayaran kamu adalah : Rp. %.2f", totalBayar);
    }
    // Selain kondisi di atas, yaitu kondisi totalBelanja kurang dari atau sama dengan
100000
    else
    {
        // Menugaskan variabel diskon dengan nilai 0 karena kondisi tidak mendapatkan
diskon
        diskon = 0;
        // Menghitung variabel totalBayar
        totalBayar = totalBelanja-diskon;
```

```
        printf("Maaf, Anda belum mendapatkan diskon.\n");
        printf("Total pembayaran Anda adalah : Rp. %.2f", totalBayar);
    }

    return 0;
}
```

- Output program `diskon.c` jika `totalBelanja <= 100000`:

```
Total belanja kamu adalah : Rp. 100000
Maaf, Anda belum mendapatkan diskon.
Total pembayaran Anda adalah : Rp. 100000.00
```

- Output program `diskon.c` jika `totalBelanja > 100000`:

```
Total belanja kamu adalah : Rp. 101000
Selamat! Anda mendapatkan diskon sebesar Rp. 10100.00
Maka total pembayaran kamu adalah : Rp. 90900.00
```

Pada kedua output di atas, terdapat perbedaan dalam tampilan *statement* yang dieksekusi. Pada output pertama yaitu jika `totalBelanja>totalBelanja <= 100000`, *statement* yang dieksekusi adalah *statement* yang menyatakan bahwa pembeli belum mendapatkan diskon. Sedangkan pada output kedua yaitu jika `totalBelanja > 100000`, maka *statement* yang dieksekusi adalah *statement* yang menyatakan bahwa pembeli mendapatkan diskon.

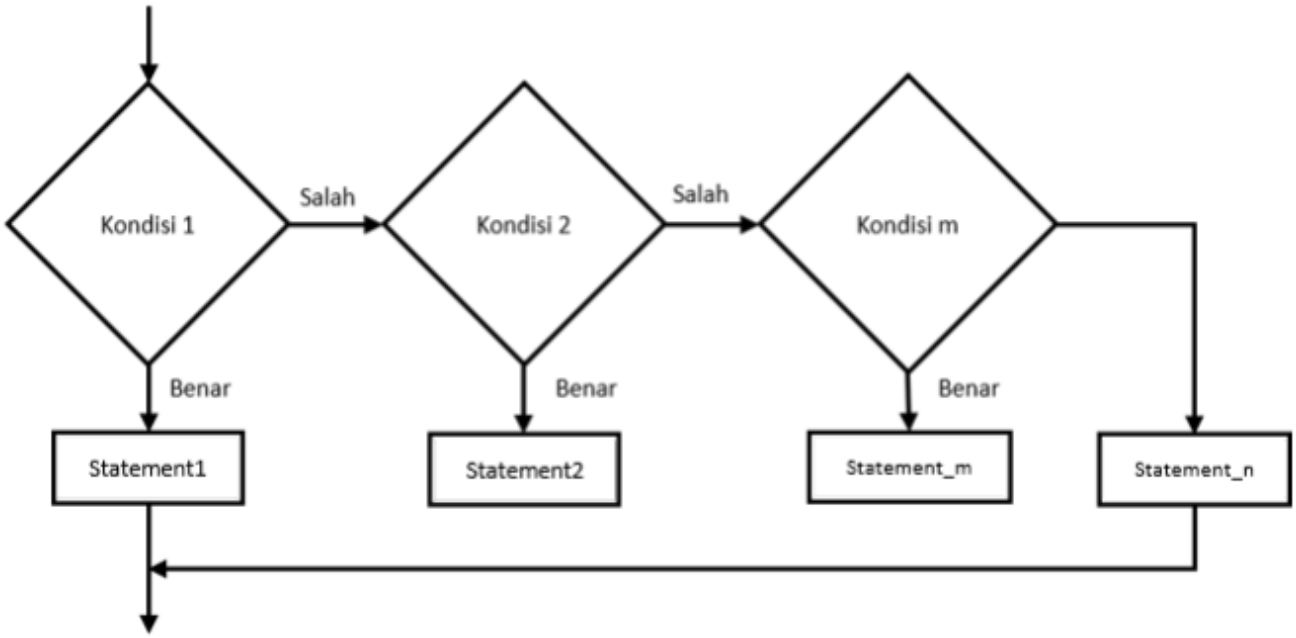
2.2.3 STATEMENT *NESTED* `if`

Percabangan *nested* `if` merupakan bentuk percabangan dimana memungkinkan terdapat lebih dari dua kondisi dalam satu kondisi percabangan. Bentuk umum dari *nested* `if` adalah sebagai berikut:

```
if (kondisi 1)
  statement1;
else
  if (kondisi 2)
    statement2;
  else
    if (kondisi 3)
      statement3;
    else
      if (kondisi m)
        statement_m;
      else              // Opsional
        statement_n;    // Opsional
```

Pada bentuk *nested* `if` kondisi yang akan diseleksi pertama kali adalah kondisi yang terluar (`kondisi1`) terlebih dahulu. Jika kondisi pertama tidak terpenuhi, maka compiler akan secara otomatis menyeleksi kondisi kedua, dan seterusnya, serta *statement* yang akan dieksekusi merupakan *statement* yang terdapat pada kondisi yang terpenuhi.

Di bawah ini merupakan flowchart dari nested `if`:



Contoh program `diskon.c`:

```c
#include <stdio.h>

int main()
{
    // Deklarasi variabel
    float totalBelanja, diskon, totalBayar;

    // Cetak output dan input variabel
    printf("Total belanja kamu adalah : Rp. ");
    scanf("%f", &totalBelanja);

    // Kondisi jika totalBelanja lebih besaar atau sama dengan 100000
    if(totalBelanja <= 100000)
    {
        // Menugaskan variabel diskon dengan nilai 0 karena kondisi tidak mendapatkan
diskon
        diskon = 0;
        // Menghitung variabel totalBayar
        totalBayar = totalBelanja-diskon;
        printf("Maaf, Anda belum mendapatkan diskon.\n");
        printf("Total pembayaran Anda adalah : Rp. %.2f", totalBayar);
    }
    else
    {
      // Kondisi jika totalBelanja antara 100001 sampai 200000
      if((totalBelanja > 100000) && (totalBelanja <= 200000))
      {
        // Menugaskan variabel diskon dengan hasil ekspresi 0.1 dikalikan dengan
totalBelanja
        diskon = 0.1 * totalBelanja;
        // Menghitung variabel totalBayar
        totalBayar = totalBelanja-diskon;
        printf("Selamat! Anda mendapatkan diskon sebesar Rp. %.2f\n", diskon);
```

```
        printf("Maka total pembayaran kamu adalah : Rp. %.2f", totalBayar);
      }
      // Selain kondisi di atas, yaitu total belanja di atas 200000
      else
      {
        // Menugaskan variabel diskon dengan hasil ekspresi  0.2 dikalikan dengan
totalBelanja
        diskon = 0.2 * totalBelanja;
        // Menghitung variabel totalBayar
        totalBayar = totalBelanja-diskon;
        printf("Selamat! Anda mendapatkan diskon sebesar Rp. %.2f\n", diskon);
        printf("Maka total pembayaran kamu adalah : Rp. %.2f", totalBayar);
      }
    }

    return 0;
}
```

- Output program `diskon.c` jika `totalBelanja > 100000 && totalBelanja <=200000`:

```
Total belanja kamu adalah : Rp. 200000
Selamat! Anda mendapatkan diskon sebesar Rp. 20000.00
Maka total pembayaran kamu adalah : Rp. 180000.00
```

- Output program `diskon.c` jika `totalBelanja > 200000`:

```
Total belanja kamu adalah : Rp. 200001
Selamat! Anda mendapatkan diskon sebesar Rp. 40000.20
Maka total pembayaran kamu adalah : Rp. 160000.80
```

Selain bentuk di atas, *statement nested* `if` juga dapat disusun dengan bentuk lain, di bawah ini merupakan bentuk lain dari penyusunan *statement nested* `if`:

```
if (kondisi 1)
    statement1;
else if (kondisi 2)
    statement2;
else if (kondisi 3)
    statement3;
else if (kondisi m)
    statement_m;
else            // Opsional
    statement_n;    // Opsional
```

Contoh program `matakuliah.c`:

```
#include <stdio.h>

int main()
{
    // Deklarasi variabel
    float totalBelanja, diskon, totalBayar;
```

```
    // Cetak output dan input variabel
    printf("Total belanja kamu adalah : Rp. ");
    scanf("%f", &totalBelanja);

    if(totalBelanja <= 100000)
    {
        // Menugaskan variabel diskon dengan nilai 0 karena kondisi tidak mendapatkan
diskon
        diskon = 0;
        // Menghitung variabel totalBayar
        totalBayar = totalBelanja-diskon;
        printf("Maaf, Anda belum mendapatkan diskon.\n");
        printf("Total pembayaran Anda adalah : Rp. %.2f", totalBayar);
    }
    // Kondisi jika totalBelanja antara 100001 sampai 200000
    else if((totalBelanja > 100000) && (totalBelanja <= 200000))
    {
        // Menugaskan variabel diskon dengan hasil ekspresi 0.1 dikalikan dengan
totalBelanja
        diskon = 0.1 * totalBelanja;
        // Menghitung variabel totalBayar
        totalBayar = totalBelanja-diskon;
        printf("Selamat! Anda mendapatkan diskon sebesar Rp. %.2f\n", diskon);
        printf("Maka total pembayaran kamu adalah : Rp. %.2f", totalBayar);
    }
    // Kondisi jika totalBelanja diatas 200000
    else
    {
        // Menugaskan variabel diskon dengan hasil ekspresi 0.2 dikalikan dengan
totalBelanja
        diskon = 0.2 * totalBelanja;
        // Menghitung variabel totalBayar
        totalBayar = totalBelanja-diskon;
        printf("Selamat! Anda mendapatkan diskon sebesar Rp. %.2f\n", diskon);
        printf("Maka total pembayaran kamu adalah : Rp. %.2f", totalBayar);
    }

    return 0;
}
```

- Output program `diskon.c` jika `totalBelanja <= 100000`:

```
Total belanja kamu adalah : Rp. 100000
Maaf, Anda belum mendapatkan diskon.
Total pembayaran Anda adalah : Rp. 100000.00
```

- Output program `diskon.c` jika `totalBelanja > 100000 && totalBelanja <= 200000`:

```
Total belanja kamu adalah : Rp. 200000
Selamat! Anda mendapatkan diskon sebesar Rp. 20000.00
Maka total pembayaran kamu adalah : Rp. 180000.00
```

- Output program `diskon.c` jika `totalBelanja > 200000`:

```
Total belanja kamu adalah : Rp. 200001
Selamat! Anda mendapatkan diskon sebesar Rp. 40000.20
Maka total pembayaran kamu adalah : Rp. 160000.80
```

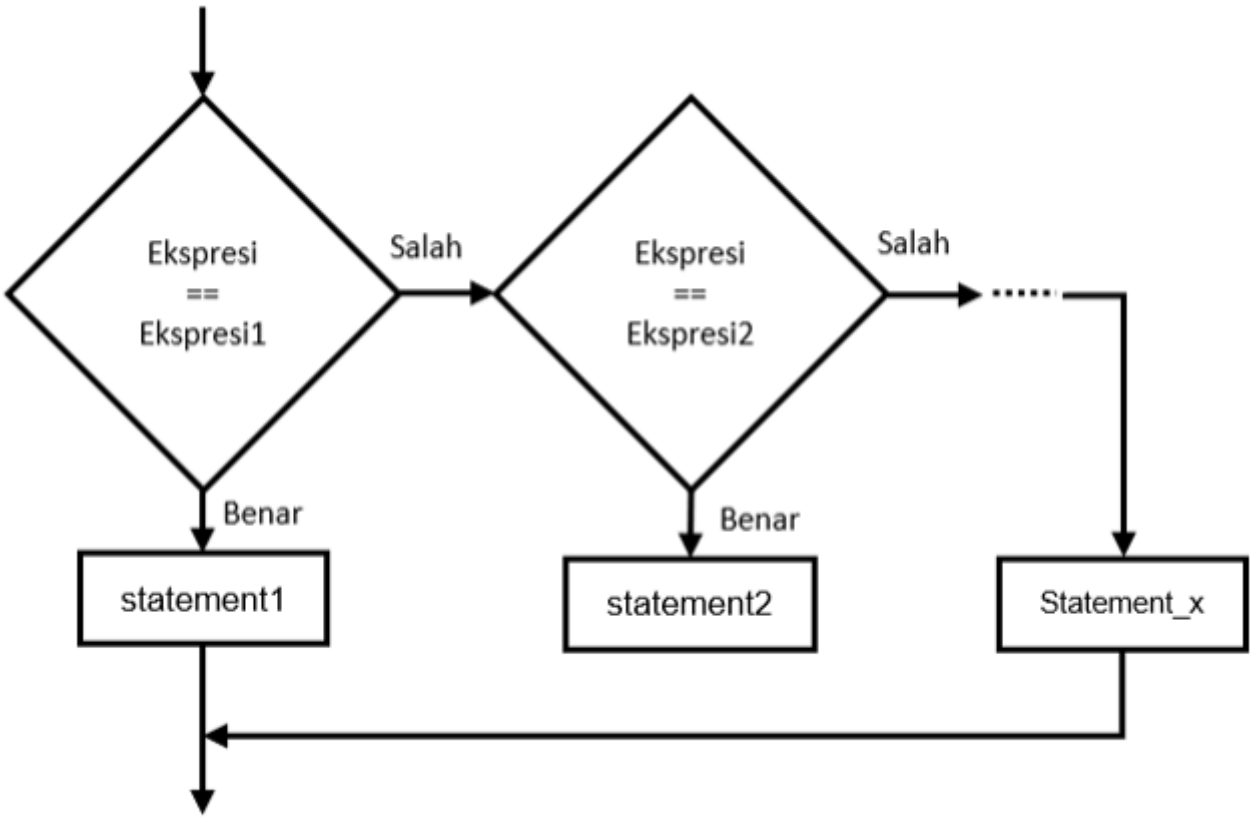
2.2.4 STATEMENT `switch - case`

Statement `switch` adalah *statement* yang digunakan untuk menjalankan salah satu *statement* dari beberapa kemungkinan *statement*, berdasarkan nilai dari sebuah ekspresi dan nilai penyeleksi. Bentuk umum dari `switch` adalah sebagai berikut:

```
switch (condition) {
  case 'pilihan 1':
     statement1;
    break;
  case 'pilihan 2':
     statement2;
    break;
  case 'pilihan 3':
     statement3;
    break;
  ...
  ...
  default:
     statement_n;
}
```

Pada bentuk `switch` sebuah kondisi akan dibandingkan dengan setiap nilai pada case yang ada. Jika kondisi bernilai benar maka, *statement* pada case tersebut akan langsung dijalankan. Apabila setiap kondisi bernilai salah maka, *statement* `default` yang akan dikerjakan.

Di bawah ini merupakan flowchart dari *statement* `switch - case`:

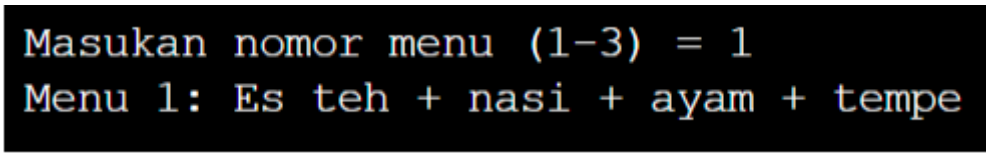


Contoh program `menu_makanan.c`:

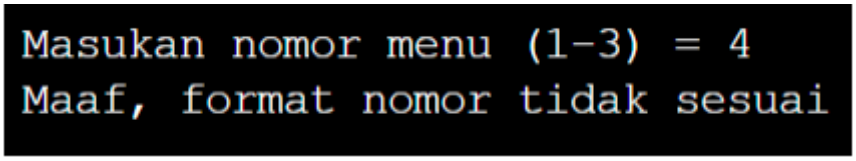
```c
#include <stdio.h>

int main()
{
  // Deklarasi variabel angka
  int angka;

  // Mencetak dan input nilai ke variabel angka
  printf("Masukan nomor menu (1-3) = ");
  scanf("%d",&angka);

  // Memulai kondisi percabangan variabel angka
  switch (angka)
  {
    // Kondisi jika user input angka 1
    case 1:
      // Statement yang akan dieksekusi apabila case 1 terpenuhi
      printf("Menu 1: Es teh + nasi + ayam + tempe \n");
      break;
    // Kondisi jika user input angka 2
    case 2:
      // Statement yang akan dieksekusi apabila case 1 terpenuhi
      printf("Menu 2: Es teh + nasi + ayam \n");
      break;
    // Kondisi jika user input angka 3
    case 3:
      // Statement yang akan dieksekusi apabila case 1 terpenuhi
      printf("Menu 3: Nasi + ayam \n");
```

```
      break;
    // Kondisi jika ketiga kondisi diatas tidak terpenuhi
    default:
      // Statement yang akan dieksekusi apabila case 1, 2, dan 3 tidak terpenuhi
      printf("Maaf, format nomor tidak sesuai \n");
  }

  return 0;
}
```

- Output program `menu_makanan.c` jika format nomor yang dimasukkan **benar**:



- Output program `menu_makanan.c` jika format nomor yang dimasukkan **salah**:



Adanya `break` pada program berfungsi untuk mengeluarkan program dari fungsi `switch`. Sedangkan `default` mempunyai fungsi untuk membatasi apabila suatu inputan yang dimasukkan oleh user di luar batas jangkauan case-nya.

REFERENSI

[1] Abdul Kadir. 2015. *From Zero to a Pro*. Yogyakarta. Andi