

# PRAKTIKUM 9

---

## Looping : For

---

### A. TUJUAN PEMBELAJARAN

1. Menjelaskan proses pengulangan menggunakan pernyataan *for*
2. Menjelaskan tentang variasi pernyataan *for*
3. Menjelaskan tentang pernyataan *for* dengan menentukan jumlah langkah

### B. DASAR TEORI

Mengulang suatu proses merupakan tindakan yang banyak dijumpai dalam pemrograman. Pada semua bahasa pemrograman, pengulangan proses ditangani dengan suatu mekanisme yang disebut *loop*. Dengan menggunakan *loop*, suatu proses yang berulang misalnya menampilkan tulisan yang sama seratus kali pada layar dapat diimplementasikan dengan kode program yang pendek.

Pada pemrograman proses perulangan dapat dibagi menjadi 2 bagian utama yaitu:

1. Perulangan yang sudah di ketahui jumlah perulangannya sebelum perulangan tersebut di lakukan.
2. Perulangan yang belum di ketahui jumlah perulangannya sebelum perulangan tersebut di lakukan. Dalam hal ini dapat dibagi menjadi dua bagian yaitu:
  - a) kondisi perulangan diperiksa diawal perulangan.
  - b) kondisi perulangan diperiksa diakhir perulangan.

Untuk kasus 1 seharusnya menggunakan perulangan dengan pernyataan *for* dan akan dipelajari pada praktikum ini. Sedangkan pada kasus 2a dan 2b akan dibahas pada praktikum 10.

Bentuk pernyataan *for* :

```
for (ungkapan1; ungkapan2; ungkapan3)
    pernyataan;
```

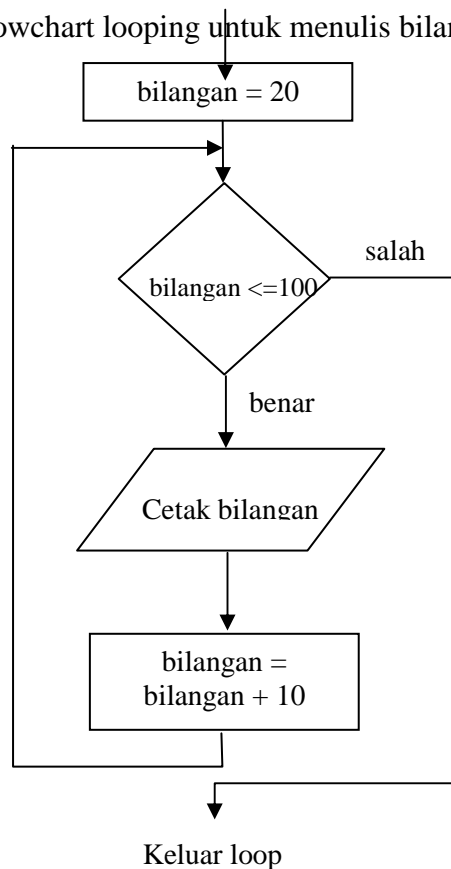
Keterangan:

- Ungkapan1 : digunakan untuk memberikan inialisasi terhadap variabel pengendali *loop*.
- Ungkapan2 : dipakai sebagai kondisi untuk keluar dari *loop*.
- Ungkapan3 : dipakai sebagai pengatur kenaikan nilai variabel pengendali *loop*.

Ketiga ungkapan dalam *for* tersebut harus dipisahkan dengan tanda titik koma (;). Dalam hal ini pernyataan bisa berupa pernyataan tunggal maupun jamak. Jika pernyataannya berbentuk jamak, maka pernyataan-pernyataan tersebut harus diletakkan di antara kurung kurawal buka ({} dan kurung kurawal tutup (}), sehingga formatnya menjadi :

```
for (ungkapan1; ungkapan2; ungkapan3)
{
    pernyataan;
    pernyataan;
    .
    .
    .
}
```

Pada Gambar 9.1 ditunjukkan flowchart looping untuk menulis bilangan 20, 30, ... ,100



Gambar 9.1 Diagram alir for untuk Menulis 20, 30, ..., 100

```
for(bilangan = 20; bilangan <= 100; bilangan += 10)
    printf("%d\n", bilangan);
```

Pada program di atas, kenaikan terhadap variabel pengendali *loop* sebesar 10 (positif), yang dinyatakan dengan ungkapan

```
bilangan += 10
```

Pada contoh yang melibatkan pernyataan *for* di atas, kenaikan variabel pengendali *loop* berupa nilai positif. Sebenarnya kenaikan terhadap variabel pengendali *loop* bisa diatur bernilai negatif.

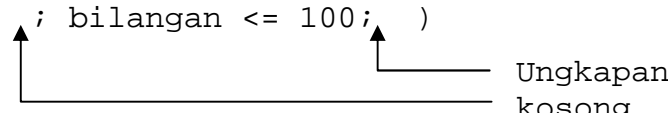
```
for (bilangan = 60; bilangan >= 10; bilangan -= 10)
    printf("%d\n", bilangan);
```

Kadang-kadang dijumpai adanya pernyataan *for* yang tidak mengandung bagian ungkapan yang lengkap (beberapa ungkapan dikosongkan). Dengan cara ini, pernyataan

```
for (bilangan = 20; bilangan <= 100; bilangan += 10)
    printf("%d\n", bilangan);
```

dapat ditulis menjadi :

```
bilangan = 20;          /* inisialisasi di luar for */
for ( ; bilangan <= 100; )
{
    printf("%d\n", bilangan);
    bilangan += 10;
}
```



Pengosongan ini juga dilakukan pada ungkapan yang biasa dipakai untuk menaikkan nilai variabel pengendali *loop*. Sebagai gantinya, di dalam tubuh *loop* diberikan pernyataan untuk menaikkan nilai variabel pengendali *loop*, yaitu berupa

```
bilangan += 10;
```

Ungkapan yang tidak dihilangkan berupa `bilangan <=100`. Ungkapan ini tetap disertakan karena dipakai sebagai kondisi untuk keluar dari *loop*.

Sesungguhnya ungkapan yang dipakai sebagai kondisi keluar dari *loop* juga bisa dihilangkan, sehingga bentuknya menjadi

```
for ( ; ; )
    pernyataan
```

Suatu pertanyaan mungkin timbul “Lalu bagaimana caranya kalau ingin keluar dari loop pada bentuk di atas?”. Caranya adalah dengan menggunakan pernyataan yang dirancang khusus untuk keluar dari loop. Mengenai hal ini akan dibahas pada praktikum selanjutnya.

### C. TUGAS PENDAHULUAN

Buatlah desain flowchart untuk setiap soal dalam percobaan

### D. PERCOBAAN

1. Gunakan loop *for* untuk membuat program sebagai berikut:

input : n

output : 1 3 4 5 ... m ( m = bilangan ganjil ke n)

2. Hitunglah bilangan triangular dari masukan pengguna, yang dibaca dari keyboard dengan menggunakan *scanf()*. Bilangan triangular adalah penjumlahan dari bilangan masukan dengan seluruh bilangan sebelumnya.

Contoh Input : 7

Output : 7 + 6 + 5 + 4 + 3 + 2 + 1 = 28

3. Gunakan loop *for* dengan kenaikan variabel negatif untuk menampilkan seluruh karakter dari Z sampai dengan A dalam baris-baris yang terpisah.

4. Gunakan loop *for* untuk membuat program sebagai berikut:

input : n

output : 1 -2 3 -4 5 -6 7 -8 ... n

5. Gunakan loop *for* untuk mendefinisikan apakah sebuah bilangan adalah bilangan prima atau bukan

input : 27

output : Bilangan adalah bilangan prima

6. Dengan menggunakan looping dan switch case atau else if buatlah program dalam C untuk menghitung Indeks Prestasi Semester seorang mahasiswa, dimana yang

diinputkan adalah nilai huruf dari 5 mata kuliah yang diikutinya dan jumlah jam mata kuliah tsb.

Dimana konversi nilai huruf ke angka untuk menghitung IPS adalah sebagai berikut:

A -> 4, B->3, C->2, D->1, E->0 dan rumus IPS = jumlah (nilai \* jam)/jumlah jam keseluruhan

Contoh :

Input :

Nilai Mata Kuliah 1 : A    jumlah jam : 2

Nilai Mata Kuliah 2 : C    jumlah jam : 2

Nilai Mata Kuliah 3 : B    jumlah jam : 3

Nilai Mata Kuliah 4 : A    jumlah jam : 3

Nilai Mata Kuliah 5 : C    jumlah jam : 3

Output:

Indeks Prestasi Semester : 3

Output di atas didapatkan dari :  $(4*2 + 2*2 + 3*3 + 4*3 + 2*3)/(2+2+3+3+3) = 39/13$

## **E. LAPORAN RESMI**

1. Untuk setiap listing program dari percobaan-percobaan di atas, ambil *capture* outputnya.
2. Tuliskan kesimpulan dari percobaan yang telah anda lakukan.