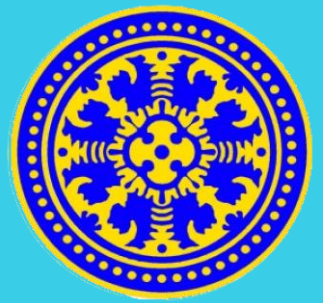
Logika Informatika (Algoritma dan Pemrograman)



Topik 5 – Pengulangan

I Dewa Made Bayu Atmaja Darmawan,S.Kom.M.Cs.

Kuliah Daring SPADA Indonesia

Capaian Pembelajaran

- Mahasiswa dapat memahami konsep dasar dan struktur pengulangan
- Mahasiswa dapat memahami perintah pengulangan dalam C
- Mahasiswa dapat menerapkan sintaks-sintaks pengulangan dalam menyelesaikan persoalan

Pengulangan

- Perulangan atau yang sering disebut dengan “looping”, merupakan proses yang dilakukan secara berulang-ulang dalam batas yang telah ditentukan.
- Pengulangan merupakan sebuah konsep pemrograman yang penting karena konsep ini memungkinkan pengguna menggunakan sekumpulan baris program berulang kali

Komponen Pengulangan

- **Inisialisasi** :menentukan kondisi awal dilakukannya pengulangan.
- **Jumlah iterasi**: menunjukkan berapa kali pengulangan akan dilakukan.
- **Kondisi berhenti** : menentukan kondisi yang dapat mengakhiri pengulangan.

ketiga komponen ini tidak selalu dapat didefinisikan dalam struktur pengulangan, mungkin saja salah satu komponen tersebut tidak didefinisikan, asal komponen yang tidak didefinisikan tersebut dapat diketahui secara tersirat.

Sintaks WHILE

- WHILE merupakan sebuah pengulangan yang dikendalikan oleh suatu kondisi, kondisi tersebut akan dicek pada setiap awal iterasi, lalu kondisi tersebut akan menentukan apakah perulangan itu akan terus dilaksanakan atau dihentikan.
- Struktur pengulangan while:

```
1  {inisialisasi}
2  WHILE (kondisi)
3      aksi
4      ubah pencacah (pencapaian kondisi berhenti)
5  ENDWHILE
```

Contoh 1

- Algoritma mencetak “ * ” sebanyak 5 kali, menggunakan While

```
ALGORITMA Tampil_Bintang
IS : -
FS : Jumlah bintang yg tampil = 5
KAMUS DATA
  i : integer
1   BEGIN
2   i ← 0    {inisialisasi pencacah i}
3   WHILE (i < 5)    {jumlah iterasi}
4   Output ('*') {aksi}
5   i ← i + 1    {pencapaian kondisi berhenti}
6   ENDWHILE
7   END
```

Contoh 1

- Tabel iterasi:

Iterasi ke-	i ke-	Bintang < 5	Cetak	Bintang
1	0	Ya	Ya	1
2	1	Ya	Ya	2
3	2	Ya	Ya	3
4	3	Ya	Ya	4
5	4	Ya	Ya	5
6	5	tidak	tidak	-

- Output dari algoritma tsb:

```
*
*
*
*
*
```

Implementasi While pada Program C

- Implementasi While berdasarkan contoh 1:

```
 1  #include <stdio.h>
 2  int main()
 3  {
 4      int i; //deklarasi variabel i
 5      i=0; //inialisasi nilai i
 6      while(i<5) //jumlah batas iterasi
 7      {
 8          printf("*"); //aksi
 9          printf("\n");
10          i=i+1;//mengubah nilai pencacah
11      }
12      return 0;
13  }
```

Sintaks Do…While

- Bedanya, jika pada sintaks WHILE kondisi dievaluasi/ diuji sebelum aksi pengulangan dilakukan, sedangkan pada sintaks DO...WHILE pengujian kondisi dilakukan setelah aksi pengulangan dilakukan.
- Struktur sintaks do…while :

```
1   {inisialisasi}
2   DO
3   aksi
4   ubah pencacah
5   WHILE (kondisi)
```

Contoh 2

- Algoritma mencetak iterasi angka

```
ALGORITMA Iterasi_Angka
IS : -
FS : Tampil angka 1 hingga 3
KAMUS DATA
   i : integer
1  BEGIN
2  i ← 1 {inisialisasi pencacah i}
3  DO
4  Output ('Ini adalah iterasi ke-',i){aksi}
5  i ← i + 1
6  WHILE (i <= 3) {kondisi berhenti}
7  END
```

Contoh 2

- Tabel iterasi

Iterasi ke-	{aksi}/cetak	i=i+1	i <= 3
1	Ya (1)	2	Ya
2	Ya (2)	3	Ya
3	Ya (3)	4	tidak

- Output algoritma tsb:

```
Ini adalah iterasi ke-1
Ini adalah iterasi ke-2
Ini adalah iterasi ke-3
```

- Pengulangan dihentikan sebelum iterasi ke- 4 karena nilai i = 4.

Implementasi DO...WHILE pada Program C

- Implementasi DO...WHILE berdasarkan contoh 2:

```
 1  #include <stdio.h>
 2  int main()
 3  {
 4      int i; //deklarasi variabel i
 5      i=1; //inialisasi nilai i
 6      do //jumlah batas iterasi
 7      {
 8          printf("ini adalah iterasi ke-%d",i); //aksi
 9          printf("\n");
10          i=i+1;//mengubah nilai pencacah
11      }while(i<=3);//kondisi berhenti
12      return 0;
13  }
14
```

Sintaks FOR

- Sintaks ini serupa dengan sintaks WHILE... DO... dalam hal pengecekan kondisi dilakukan di awal.
- Dalam menggunakan sintaks FOR untuk pengulangan maka pemrogram harus mendefinisikan nilai awal dan nilai akhir pencacah yang menunjukkan jumlah iterasi.
- Struktur sintaks FOR:

```
FOR(Inisialisai;KondisiPengulangan; PengubahNilaiPencacah)
{
      pernyataan/perintah pengulangan
}
ENDFOR
```

Sintaks FOR

Keterangan:

- Inisialisasi : Pemberikan nilai awal untuk variabel pencacah.
- Kondisi Pengulangan : Berisi syarat kondisi pengulangan akan berhenti atau tidak.
- Perubah Nilai Pencacah : pengubahan nilai variabel pencacah untuk mencapai kondisi berhenti, dapat berupa kenaikan ataupun penurunan.
- Pernyataan perintah : aksi yang akan diulang

Contoh 3

- Contoh algoritma penerapan for

```
ALGORITMA Hitung_Mundur
IS :
FS : Menampilkan angka dari 5 hingga 1
KAMUS DATA
hitung : integer
1   BEGIN
2   FOR(hitung ← 5; hitung > 0;bilangan--)
3       Output(hitung)
4   ENDFOR
5   Output ('DOR!!!')
6   END
```

Contoh 3

- Tabel iterasi:

Iterasi ke-	hitung	hitung > 0?	{aksi}/cetak
1	5	Ya	Ya
2	4	Ya	Ya
3	3	Ya	Ya
4	2	Ya	Ya
5	1	Ya	Ya
6	0	ya	tidak

- Output algoritma tsb adalah

```
5
4
3
2
1
DOR!!!
```

Implementasi FOR pada Program C

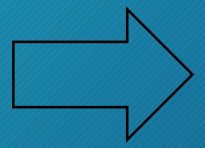
- Implementasi FOR sesuai contoh 3:

```
1   #include <stdio.h>
2   int main()
3   {
4       int hitung;//deklarasi variabel
5       for(hitung=5;hitung>0;hitung--)
6       {
7           printf("%d",hitung);  //aksi
8           printf("\n");
9       }
10      printf("DOR!!!");
11      return 0;
12  }
```

Perulangan Bersarang

- Perulangan bersarang (*Nested Loop*) adalah struktur perulangan yang berada di dalam perulangan lainnya
- Contoh & hasil algoritma perulangan bersarang:

```
ALGORITMA Tampil_Bintang
IS : -
FS : Menampilkan bintang sebanyak
3 kolom dan 5 baris
KAMUS DATA
 i
 :
1   BEGIN
2   FOR(i ← 0; i < 5; i++)
3   FOR(j ← 0; j < 3; j++)
4   Output ('*') {aksi}
5   ENDFOR
6   ENDFOR
7   END
```



```
***
***
***
***
***
```

BREAK dan CONTINUE

- Sintaks BREAK dan CONTINUE dapat digunakan baik di dalam struktur pengulangan WHILE, DO...WHILE, dan FOR.
- Sintaks *BREAK* digunakan untuk menghentikan pengulangan kemudian keluar dari struktur pengulangan tanpa melanjutkan perintah di dalam struktur pengulangan.
- Sintaks *CONTINUE* digunakan untuk kembali ke awal pengulangan tanpa menjalankan perintah berikutnya

Contoh 4

- Contoh penggunaan sintaks BREAK di dalam pengulangan dengan menggunakan FOR

```
ALGORITMA Iterasi_Angka_Break
IS : -
FS : Menampilkan angka 1 hingga 6 tetapi berhenti
di angka 3
KAMUS DATA
   i : integer
1   BEGIN
2   FOR(i ← 1; i <= 6;i++)
3   IF (i=4)
4   BREAK
5   ENDIF
6   Output (‘Ini adalah iterasi ke-’,i){aksi}
7   ENDFOR
8   OUTPUT(‘Akhir pengulangan’)
END
```

Contoh 4

- hasil algoritma tersebut adalah:

```
Ini adalah iterasi ke-1
Ini adalah iterasi ke-2
Ini adalah iterasi ke-3
Akhir pengulangan
```

- Ketika variabel i mencapai angka 4 maka akan memenuhisyarat untuk masuk ke struktur IF, kemudian perintah BREAK dijalankan yaitu keluar dari struktur pengulangan FOR, dan menampilkan perintah pada baris ke-8 yaitu mencetak “akhir perulangan”.

Contoh 4

- Ini adalah implementasi contoh 4 ke dalam program c

```
1   //TampilAngkaBreak.c
2   #include <stdio.h>
3   main()
4   {
5   int i;
6   for(i=1; i<=6; i++)
7   {
8   if (i==4)
9   {
10 break;
11 }
12 printf("\nIni adalah iterasi ke-%d",i);
13 }
14 printf("\nAkhir pengulangan");
15 }
```

Contoh 5

- Contoh penggunaan sintaks CONTINUE di dalam pengulangan dengan menggunakan FOR

```
ALGORITMA Iterasi_Angka_Continue
IS : -
FS : Menampilkan angka 1 hingga 6
KAMUS DATA
   i : integer
1  BEGIN
2  FOR(i ← 1; i <= 6;i++)
3  IF (i=4)
4  CONTINUE
5  ENDIF
6  Output (‘Ini adalah iterasi ke-’,i){aksi}
7  ENDFOR
8  OUTPUT(‘Akhir pengulangan’)
9  END
```

Contoh 5

- Hasil algoritma tersebut adalah:

```
Ini adalah iterasi ke-1
Ini adalah iterasi ke-2
Ini adalah iterasi ke-3
Ini adalah iterasi ke-5
Ini adalah iterasi ke-6
Akhir pengulangan
```

- Ketika variabel i mencapai angka 4 maka akan memenuhi syarat untuk masuk ke struktur IF, kemudian perintah CONTINUE dijalankan yaitu kembali ke awal pengulangan dimana akan dilakukan penambahan nilai variabel pencacah dan pengecekan kondisi. Ketika perintah CONTINUE dijalankan, maka perintah berikutnya yaitu perintah pada baris ke-6 tidak dijalankan, sehingga tampilan Ini adalah iterasi ke-4 tidak keluar. Setelah kembali ke awal pengulangan, variabel i akan ditambah menjadi 5 kemudian masuk lagi ke struktur pengulangan.

Contoh 5

- Implementasi contoh 5 pada program c

```
1  //TampilAngkaContinue.c
2  #include <stdio.h>
3  main()
4  {
5  int i;
6  for(i=1; i<=6; i++)
7  {
8  if (i==4)
9  {
10 continue;
11 }
12 printf("\nIni adalah iterasi ke-%d",i);
13 }
14 printf("\nAkhir pengulangan");
15 }
```

Rangkuman

- Struktur pengulangan memungkinkan program melakukan satu atau lebih aksi beberapa kali.
- Tiga komponen pengendali aksi adalah inisialisasi, jumlah iterasi, dan kondisi berhenti.
- Tiga struktur pengulangan yang dapat digunakan adalah struktur pengulangan dengan sintaks WHILE, DO...WHILE, dan FOR.
- Struktur pengulangan dapat dibuat bersarang dengan sintaks pengulangan yang sama atau berbeda, bahkan dapat digabungkan dengan struktur pemilihan.
- Untuk keluar dari struktur pengulangan sebelum kondisi berhenti, kita dapat menggunakan sintaks BREAK
- Untuk melanjutkan struktur pengulangan ke awal pengulangan maka dapat digunakan sintaks CONTINUE
- Hal yang terpenting dari struktur pengulangan adalah kondisi berhenti yang akan memberikan kondisi apakah pengulangan dilakukan atau tidak.

Pertanyaan?