Dua struktur utama, percabangan dan perulangan merupakan struktur yang tidak terpisahkan dengan algoritma dan pemrograman. Struktur perulangan memungkinkan algoritma untuk melakukan serangkaian perintah secara berulang-ulang. Dan untuk memenuhi syarat bahwa algoritma harus finite (terbatas) maka dalam perulangan pasti ada titik pemberhentian. Jika ternyata dalam sebuah kasus perulangan tidak mencapai titik berhenti maka dapat dikatakan algoritma tersebut salah.

Titik pemberhentian dapat diberikan dengan beberapa cara, sebagai berikut.

1. Pemberhentian dengan syarat

   Pemberhentian dengan syarat artinya ada sebuah kondisi yang akan menyebabkan perulangan berhenti. Pemberian syarat ini juga dapat dilakukan dengan dua cara, yaitu

a. Syarat diberikan di awal, di mana selama persyaratan dipenuhi maka dilakukan serangkaian perintah

b. Syarat diberikan di akhir, di mana proses akan diulang-ulang sampai syarat dipenuhi. Bedakan antara kedua kalimat pada a dan b. Ketika syarat diberikan di awal, maka selama persyaratan itu dipenuhi , maka perulangan dilakukan. Jika kondisi sudah tidak dipenuhi maka berhenti. Sedangkan pada kasus syarat di akhir, dikerjakan serangkaian langkah.

   Setiap selesai rangkaian langkah diperiksa apakah kondisi sudah dicapai, jika belum maka proses diulangi lagi.

2. Pemberhentian dengan pencacah

   Pemberhentian dengan pencacah, artinya dari awal sudah ditentukan bahwa perulangan akan

dilakukan berapa kali. Pencacah ini juga ada dua macam cara, yaitu

a. Pencacah naik

   Pemberhentian dengan pencacah naik artinya untuk suatu pencacah, misalkan i dari 1 sampai 100 lakukan rangkain langkah x. Artinya langkah tersebut akan diulangi sebanyak seratus kali

b. Pencacah turun

   Pemberhentian dengan pencacah turun , artinya sebaliknya untuk suatu pencacah i dari 100 sampai 1 lakukan rangkaian langkah x, artinya langkah x akan diulangi sebanyak seratus kali.

   Penggunaan beberapa macam pemberhentian di atas tergantung dari situasi dan kondisi, untuk lebih jelas nya akan dijelaskan pada tiap-tiap kegiatan Belajar.

Baik, pada Kegiatan Belajar pertama ini dimulai dengan struktur perulangan dengan syarat di depan.

Perhatikan kembali dua permasalahan di dabawah.

Kasus 1

Diketahui potongan algoritma berikut

```
//Algoritma Perulangan1
Variabel i : integer
i=1
WHILE (i<=10)
WRITE "Hello"
i=i+1
END
```

Jika ditelusuri maka, akan menghasilkan proses sebagai berikut. Mula-mula nilai i =1 Selama i<=10 maka cetak kata kata “Hello” dan naikkan nilai i = i+1 Dengan demikian urutan perintah yang dilakukan adalah:

- Mulai i=1

- Cek apakah i<=10? Iya karena i=1, berarti cetak “Hello”, tambahkan i = i+1=2
- Cek apakah i<=10? Iya karena i=2, berarti cetak “Hello”, tambahkan i = i+1=3
- Cek apakah i<=10? Iya karena i=3, berarti cetak “Hello”, tambahkan i = i+1=4
- Cek apakah i<=10? Iya karena i=4, berarti cetak “Hello”, tambahkan i = i+1=5
- Cek apakah i<=10? Iya karena i=5, berarti cetak “Hello”, tambahkan i = i+1=6
- Cek apakah i<=10? Iya karena i=6, berarti cetak “Hello”, tambahkan i = i+1=7
- Cek apakah i<=10? Iya karena i=7, berarti cetak “Hello”, tambahkan i = i+1=8
- Cek apakah i<=10? Iya karena i=8, berarti cetak “Hello”, tambahkan i = i+1=9
- Cek apakah i<=10? Iya karena i=9, berarti cetak “Hello”, tambahkan i = i+1=10

- Cek apakah i<=10? Iya karena i=10, berarti cetak “Hello”, tambahkan i = i+1=11
- Cek apakah i<=10? Tidak karena i=11, berarti STOP keluar dari perulangan, END.
- Jadi dengan dmeikian kata “Hello” dicetak 10 kali.

Kasus 2

Pada kasus 2, diminta membuat algoritma untuk menuliskan semua bilangan kelipatan 5 yang kurang dari 100.

5 10 15 20 25 ... 95

Strategi yang harus dijalankan adalah menentukan:

1. Tentukan iterator, yaitu variabel yang berperan sebagai penggerak perulangan
2. Tentukan sentinel, syarat perulangan yang dibutuhkan agar perulangan dapat berhenti. Untuk memecahkan permasalahan tersebut, ada beberapa strategi yang bisa diterapkan.

Jika 5, 10, 15, adalah suku dari sebuah barisan, bagaimana cara mendapatkan setiap sukunya? Bisa dari rumus suku ke-n atau menghubungkan antara suku ke n dengan suku sebelumnya. Kita tahu bahwa barisan di atas dapat dituliskan ulang dalam bentuk 5x1, 5x2, 5x3, dan seterusnya, maka dalam hal ini bilangan 1, 2, 3 sebagai iterator, selanjutnya dapat dituliskan langkah-langkah nya adalah sebagai berikut.

a. Nilai awal i =1 {i sebagai iterator}
b. Suku awal, a=5
c. Selama a< 100 maka lakukan langkah berikut {syarat a<100 disebut sentinel}
d. Tuliskan a;
e. i=i+1;
f. a=5*i
g. Selesai

Dalam tiap perulangan iterator dan sentinel harus ada. Iterator akan menjamin perulangan dilakukan hingga mencapai kondisi (sentinel) dipenuhi sedangkan sentinel akan menjamin perulangan akan berhenti.

Perhatikan potongan algoritma di atas, kita dapat menelusuri algoritma di atas sebagai berikut.

Pertama nilai i=1 dan a=5

Diperiksa apakah a<100? Iya, maka perintah berikutnya dijalankan Tuliskan a {output: 5}

i=i+1=2 {nilai i menjadi 2}

a=5*i=5*2=10 {nilai a menjadi 10}

Di akhir struktur maka, kembali ke atas untuk diperiksa persyaratan Apakah a<100? Iya, maka perintah berikutny dijalankan lagi

Tuliskan a {output: 10}

i=i+1=3 {nilai i menjadi 3}

a=5*i=5*3=15 {nilai a menjadi 15}

Di akhir struktur maka, kembali ke atas untuk diperiksa persyaratan Apakah a<100? Iya, maka perintah berikutnya dijalankan lagi

Tuliskan a {output: 15}

i=i+1=4 {nilai i menjadi 4}

a=5*i=5*4=20 {nilai a menjadi 20} dan seterusnya sehingga ketika a=95, i=19

Apakah a<100? Iya, maka perintah berikutnya dijalankan lagi Tuliskan a {output: 95}

i=i+1=20{nilai i menjadi 20} a=5*i=5*20=100 {nilai a menjadi 100}

Apakah a<100? Tidak perulangan maka berhenti Jadi keluaran dari algoritma adalah 5 10 15 ... 95

Struktur Perulangan dengan WHILE DO

Dalam berbagai bahasa pemrograman dikenal struktur WHILE DO, yang berarti selama memenuhi kriteri/kondisi tertentu, maka dilakukan serangkaian

proses. Struktur ini dapat diterapkan pada contoh di atas sebagai berikut.

```
Algoritma Kelipatan5
Variabel i, a:integer
  1) i =1 {i sebagai
  iterator} 2) a=5
  3) WHILE (a< 100) DO
  4)   WRITE a
  5)   i=i+1
  6)   a=5*i
  7) END
```

Alternatif Penyelesaian

Selain menggunakan bentuk 5x1, 5x2, ..., 5xn sebagai model perulangan kita juga dapat melihat hubungan sebagai berikut

Suku pertama =5

Suku kedua = 5+5 =10 Suku ketiga = 10+5=15 Suku keempat =15+5=20 Dan seterusnya

Sehingga dapat dilihat hubungan bahwa suku ke-n diperoleh dari suku sebelumnya ditambah dengan 5.

Sehingga yang menjadi iterator di sini adalah a itu sendiri sedangkan sentinelnya tetap. Sehingga dapat dibuat algoritma alternatif untuk permasalahan di atas sebagai berikut

```
Algoritma Kelipatan5
Variabel i, a:integer
  1)  a = 5
  2)  WHILE (a<100)DO
  3)  WRITE a
  4)  a = a + 5
  5)  END
```

Algoritma ini lebih sederhana buka? Coba kita telusuri idenya. Pertama a=5

Periksa apakah a<100, iya maka lakukan

a=a+5=5+5=10

Periksa apakah a<100, iya maka lakukan

a=a+5=10+5=15

Periksa apakah a<100, iya maka lakukan

a=a+5=15+5=20

dan seterusnya

Periksa apakah a<100, iya maka lakukan

a=a+5=5+5=10

Periksa apakah a<100? Tidak maka berhenti

Dari kedua macam solusi di atas, dapat disimpulkan bahwa dalam menentukan struktur perulangan untuk masalah yang sama bisa lebih dari satu. Untuk lebih memperjelas pemahaman kalian mengenai struktur ini, perhatikan satu contoh berikut.

Contoh 1

Kalian pernah belajar barisan dan deret bilangan kan? Perhatikan barisan berikut 2, 4, 6, 8, ...Rancang sebuah algoritma untuk menentukan suku ke -20.

Jawab:

Pada contoh ini, berhentinya perulangan berdasarkan urutan suku ke berapa.

2, 4, 6, 8 .....

Urutan suku ke-n akan menggunakan hubungan antara n dengan suku ke-n. Ini berarti kita harus menggunakan iterator i sebagaimana contoh pertama yang akan berhenti jika i = n.

Untuk menentukan suku ke 20, misalkan dapat dicari dengan algoritma berikut.

```
Algoritma Barisan Genap
Variabel i, a:integer
  1)  i = 1
  2)  a = 2
  3)  WHILE (a<20)DO
      i = i + 1
      a = a + 2
  4)  WRITE a
  5)  END
```

Perhatikan algoritma di atas, mengapa menggunakan sentinel i<20, bukannya i<=20? Misalkan digunakan bentuk sentinelnya i<=20, maka ketika i=20 masih dapat diolah dalam struktur

perulangan hingga nilai i=21 dan nilai a bukan lagi nilai suku ke 20 tapi nilai suku ke 21 sehingga tidak sesuai dengan tujuan

Untuk memeriksa dapat dicek dengan tabel berikut

I	1	2	3	4	...	19	20 (STOP)
A	2	4	6	8	...	38	40

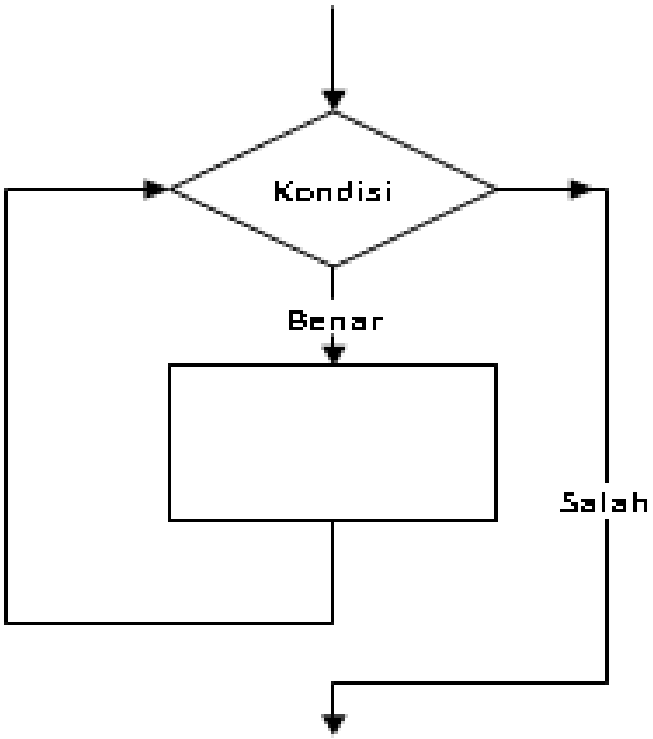
Perintah WRITE diletakkan di luar struktur perulangan, karena yang dicari hanyalah suku ke-20 dari barisan tersebut, berbeda dengan Kasus 1 yaitu menuliskan semua bilangan kelipatan 5 sedangkan pada kasus ini hanya suku ke-20 saja.

Penyajian struktur perulangan WHILE DO dengan flowchart

Pada prinsip nya struktur WHILE DO adalah struktur perulangan dengan menggunakan persyaratan (percabnagan di depan). Jika syarat dipenuhi maka proses dilanjutkan pada sebuah blok yang terdiri serangkaian operasi. Di akhir blok maka

aliran proses dikembalikan ke atas, yaitu pada syarat yang ditetapkan di awal. Aliran ini ditunjukkan oleh garis panah. Berikut ini flowcharts secara umum dari struktur.

WHILE (kondisi) DO Pernyataan



Algoritma Kelipatan5

Variabel a:integer1)

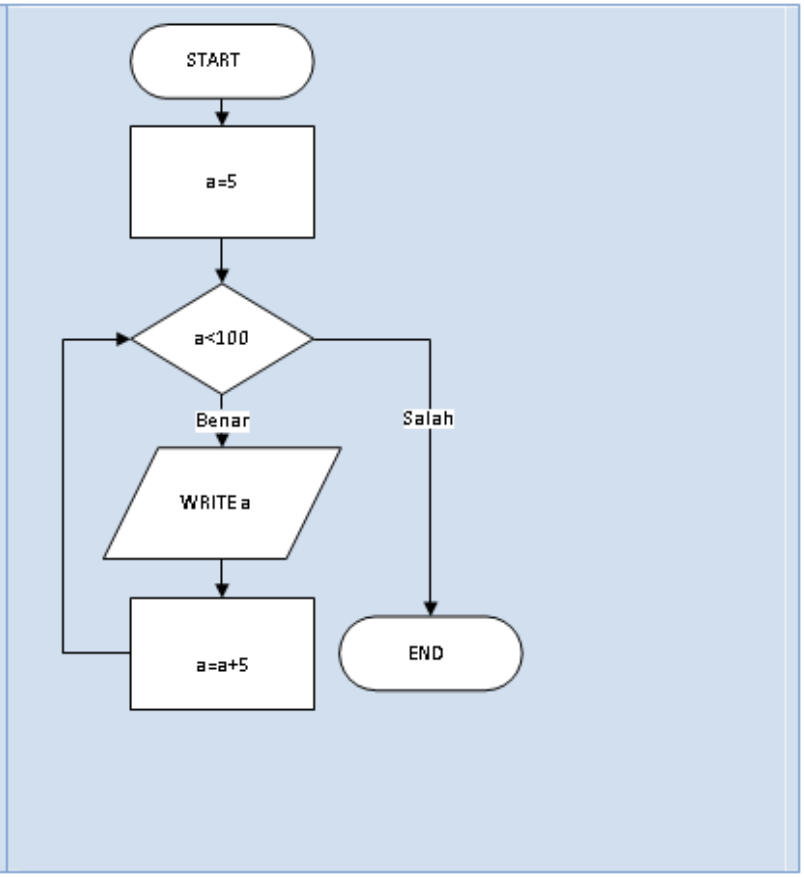
a=5

2) WHILE (a< 100) DO

3) WRITE a

4) a=a+5

5) END



Sedangkan pada kasus dua

Algoritma BarisanGenap

Variabel i, a:integer1)

i=1

2) a=2

3) WHILE i<20 DO 4)

i=i+1

5) a=a+2

6) WRITE a

7) END