Alam Santosa

Teori Algoritma

Perulangan

Algoritma Perulangan

- Seperti pernah dibahas sebelumnya, kemampuan komputer adalah melakukan pekerjaan yang sama tanpa merasa lelah maupun bosan.
- Syarat utama memanfaatkan kemampuan ini adalah menemukan pola perulangannya.

Pola Perulangan

- Untuk menemukan pola perulangan dari sebuah algoritma kita harus mengenali konstanta dan variabel yang terlibat dalam algoritma tersebut.
- Salah satu variabel yang terdapat dalam algoritma perulangan akan muncul dalam bentuk deret yang memiliki selisih tertentu (incremental).
- Variabel-variabel lain mengikuti pola yang sama setiap kali blok perulangan dieksekusi.
- Sedangkan konstanta nilainya selalu tetap selama perulangan tersebut.

Contoh: Deret

- Buatlah deret sebagai berikut:

  0 3 8 15 24 35 48
- Jika kita uraikan deret tersebut maka deret diatas akan

  1 2 3 4 5 6 7 ← Variabel Deret

  0 3 8 15 24 35 48 ← Variabel dependen
- Kita akan menemukan bahwa pola yang berlaku adalah

  1^2-1 2^2-1 3^2-1 4^2-1 … n^2-1 ← Pola Perulangannya

Contoh (lanjutan)

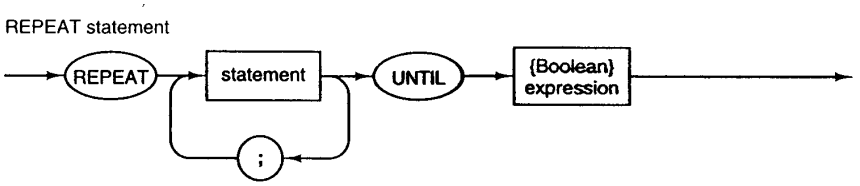
- Algoritma untuk deret tersebut dapat ditulis sebagai berikut:

```
Repeat
    N :=N+1;
    Hasil := N*N-1;
    Write (Hasil);
Until N=7;
```

Algoritma Perulangan

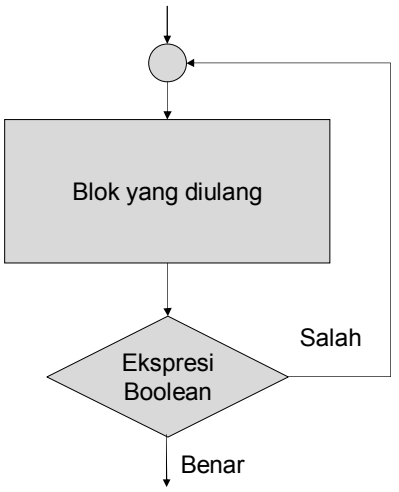
- Algoritma perulangan dapat ditulis dalam tiga macam perintah yaitu:
  - Repeat .. Until
  - While .. Do
  - For .. To .. Do

Repeat .. Until



- **Repeat .. Until** merupakan statement utama dalam perulangan, sebagian besar perulangan dapat diselesaikan dengan algoritma ini.

Flowchart Repeat .. Until



- Dari flowchart disamping ini terlihat bahwa blok yang diulang minimal dilakukan satu kali.
- Perulangan akan berhenti jika kondisi terpenuhi (bernilai benar)
- Format penulisan

```
Repeat
    <perintah yang
    diulang>;
Until <ekspresi boolean>;
```

Contoh: Algoritma Euclidean

Diberikan dua bilangan positif m dan n dimana m ≥ n, Carilah faktor persekutuan terbesar (FPB) dari kedua bilangan tersebut, yaitu bilangan bulat positif terbesar yang habis membagi m dan n.

Deskripsi

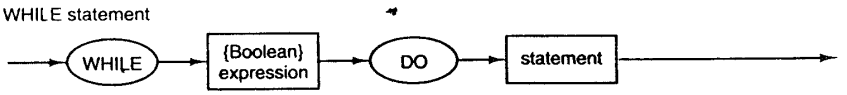
1. Bagilah m dengan n dan misalkan r adalah sisa pembagiannya
2. Jika r = 0 maka
   n adalah jawabannya
   stop
   Tetapi jika r ≠ 0,
   Lanjutkan ke langkah 3
3. Ganti nilai m dengan nilai n dan nilai n dengan nilai r, ulangi langkah 1

```
Program Euclidean;
Uses Crt;
Var m,n,r : Integer;
Begin
  Repeat
        Clrscr;
        Write ('m : ');readln (m);
        Write ('n ; ');readln (n);
  Until m>=n;
  Repeat
     r:=m mod n;
     If r<>0 then
        Begin
           m:=n;
           n:=r;
        End;
  Until r = 0;
  Write ('Pembagi Bersama Terbesar = ',n);
  Readln;
End.
```

Perulangan untuk Memastikan m > n

Algoritma Euclidean

While .. Do



- While .. Do adalah statement perulangan yang letak ekspresi boolean-nya berada di bagian atas, sebelum blok yang diulang.

While .. Do

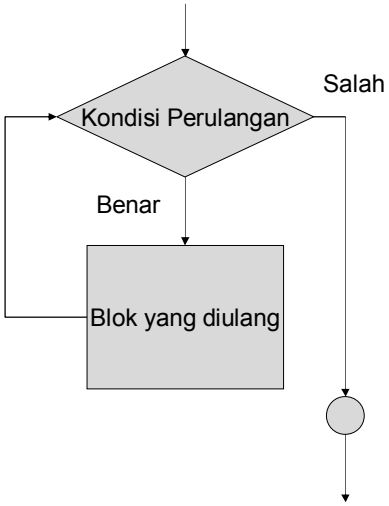
- Format Penulisannya sbb:

```
While <ekspresi boolen> Do
<perintah yang diulang>;
```

atau

```
While <ekspresi boolean> Do
Begin
<blok yang diulang>;
End;
```

Flowchart While .. Do



- Beberapa algoritma yang mengharuskan pengkondisian sebelum melaksanaan blok perulangan tidak dapat/sulit diakomodasi dengan statement Repeat .. Until.
- Minimal perulangan dengan statemen While .. Do adalah nol kali.
- Perulangan akan dilakukan jika kondisi benar.

Contoh Algoritma Euclidean dengan While .. Do

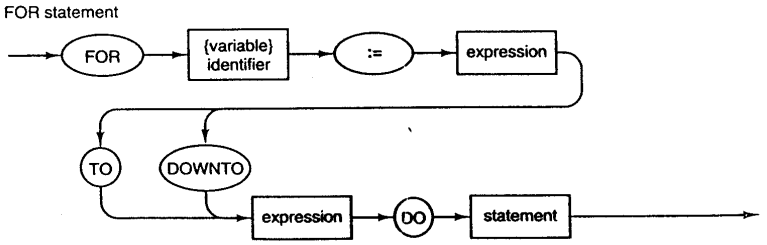
```
Program Euclidean_While;
Uses Crt;
Var m,n,r : Integer;
Begin
  n:=1;
  While m<n do
  Begin
        Clrscr;
        Write ('m : ');readln (m);
        Write ('n ; ');readln (n);
  End;
  r:=1;
  While r<>0 Do
  Begin
     r:=m mod n;
     If r<>0 then
        Begin
           m:=n;
           n:=r;
        End;
  End;
  Write ('Pembagi Bersama Terbesar = ',n);
  Readln;
End.
```

Perulangan untuk Memastikan m > n

Algoritma Euclidean

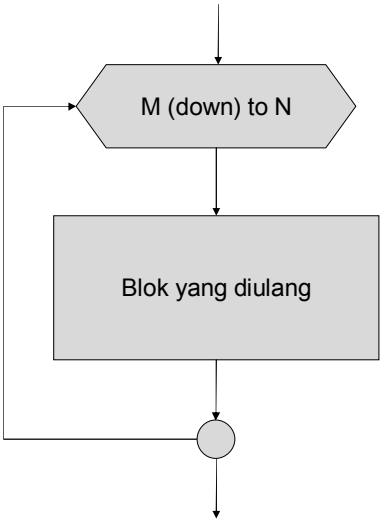
Logika pada bagian ini pun sedikit lebih rumit menggunakan While .. Do dibanding Repeat .. Until, karena sebelum perulangan harus di inisiasi **r tidak sama dengan 0**.

For .. To .. Do



- Perulangan ini adalah yang paling sederhana terutama digunakan untuk perulangan yang bersifat matematis, seperti deret.

Flowchart For .. To .. Do



- Statemen For .. To .. Do mungkin akan membuat penulisan algoritma perulangan lebih mudah namun hanya berlaku pada jenis perulangan tertentu, yaitu yang pola perulangannya berupa perulangan matematis diskrit (hanya menggunakan bilangan bulat)

Format For .. To .. Do

```
For <nilai awal> To <nilai akhir> do
    <Perintah diulang>;
```

Atau

```
For <nilai awal> To <nilai akhir> do
Begin
    <Blok diulang>;
End;
```

Atau

```
For <nilai awal> DownTo <nilai akhir> do
    <Perintah diulang>;
```

Atau

```
For <nilai awal> DownTo <nilai akhir> do
Begin
    <Blok diulang>;
End;
```

Perbandingan Repeat & For

```
Repeat
   N :=N+1;
   Hasil := N*N-1;
   Write (Hasil);
Until N=7
```

```
For N:=1 To 7 Do
Begin
  Hasil := N*N-1;
  Write (Hasil);
End;
```

Contoh: Membuat deret bilangan bulat

```
Program
  Deret_Bilangan_Bulat;
Uses Crt;
Var I: Byte;
Begin
Clrscr;
   For I:= 1 To 10 Do
      Write (I,' ');
Readln;
End.
```

```
Program
  Deret_Bilangan_Bulat;
Uses Crt;
Var I: Byte;
Begin
Clrscr;
   For I:= 10 DownTo 1 Do
      Write (I,' ');
Readln;
End.
```

Contoh: Menghitung Rata-Rata

```
Program Hitung_Rata_Rata_10_Angka;
Uses Crt;
Var I                  :byte;
    Angka,Jumlah       :Real;
Begin
Clrscr;
   Repeat
      I:=I+1;
      Write ('Angka ke-',I,': ');Readln(Angka);
      Jumlah:=Jumlah+Angka;
   Until I>=10;
   Write('Rata-rata dari ',jumlah:6:2,' adalah
   ',jumlah/I:10:2);
Readln;
End.
```

Modifikasi Program diatas menggunakan statement **While..Do** dan **For..To..Do**

Latihan

- Menampilkan Deret Fibonnaci (1,2,3,5,8,13,21,34,…)
- Menghitung Nilai Pangkat Majemuk
- Menghitung Faktorial
- Menghitung Permutasi
- Menghitung Kombinasi
- Menghitung Nilai Akar
  1. Y adalah setengahnya dari X yang akan dicari akarnya.
  2. Hitung nilai Z dengan rumus (Y + X/Y)/2.
  3. Jika Pangkat Z sama dengan bilangan X maka Z adalah akar dari X jika tidak
  4. Ganti nilai Y dengan nilai Z dan ulangi langkah 2 – 3