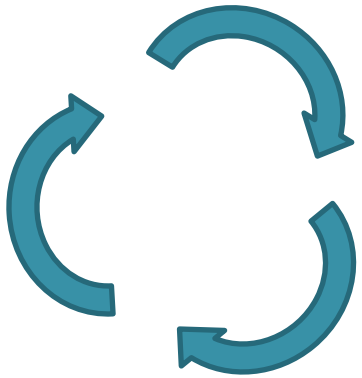
PERULANGAN PROSES

Proses perulangan ditandai dengan mekanisme yang disebut ***loop***

Proses Loop : Proses yang berulang-ulang



Perintah atau notasi dalam struktur pengulangan Melipiuti :

Pernyataan **for**

Pernyataan **while**

Pernyataan **do..while**

Pernyataan **continue dan break**

Pernyataan **go to**

Struktur Perulangan "FOR"

- Digunakan untuk mengulang suatu proses yang telah diketahui jumlah perulangannya

- Pada umumnya looping yang dilakukan oleh for telah diketahui batas awal, syarat looping dan perubahan

- Selama *kondisi* terpenuhi, maka pernyataan akan terus dieksekusi.

```
For (ungkapan 1; ungkapan 2; ungkapan 3)
    pernyataan
```

```
For (inisialisasi; kondisi; perubahan)
    Pernyataan
```

```
For (awal; akhir; peningkatan)
```

Ungkapan 1: digunakan untuk memberikan inisialisasi terhadap pengendali variabel loop

Ungkapan 2: dipakai sebagai kondisi untuk keluar dari loop

Ungkapan 3: diapakai sebagai pengatur kenaikan perubahan nilai variabel pengendali loop

Contoh : Kita akan menampilkan sebuah string (kata : Hallo Selamat belajar C) beberapa kali

Programnya :

```
#include <iostream,h>

void main ()
{
    int X;
    for (X =1;  X<= 10; X++)
     cout<<“Hallo Selamat Belajar C++”<<endl;

}
```

Jika program dijalankan :

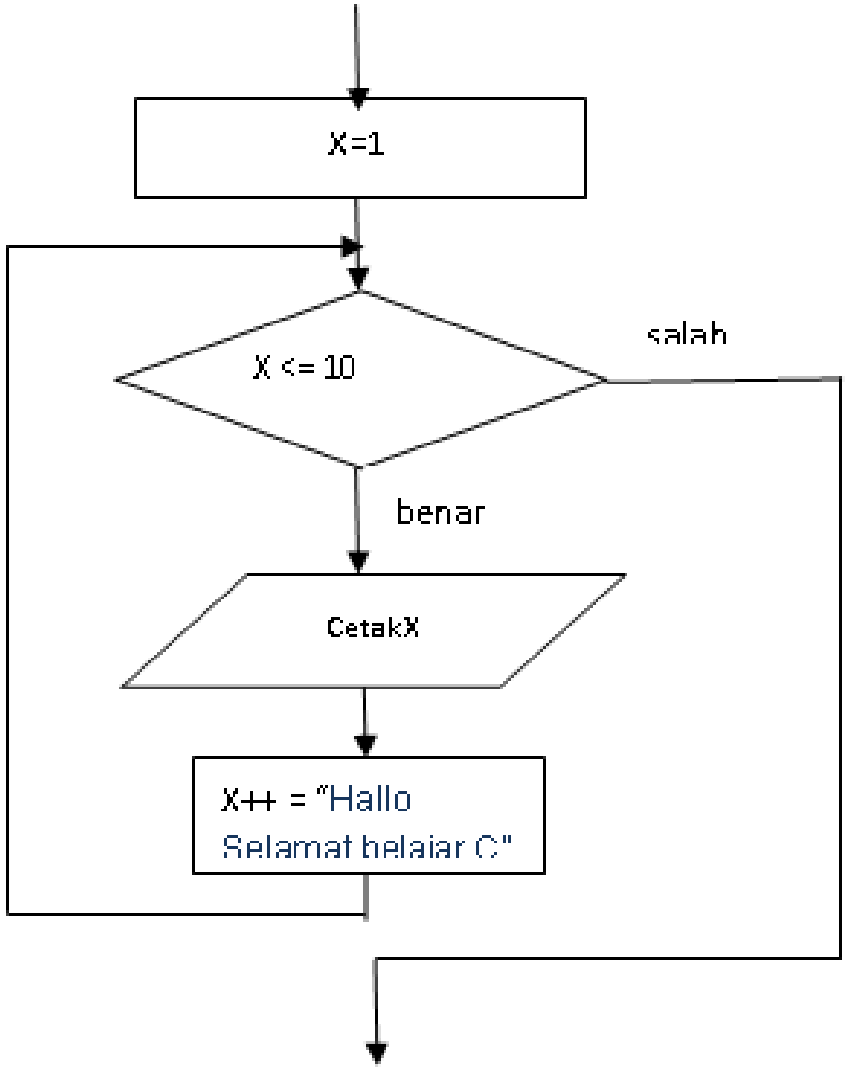
Hallo Selamat belajar C

Hallo Selamat belajar C

Hallo Selamat belajar C

Hallo Selamat belajar C

Diagram Flowchart For :



Contoh bentuk-bentuk perulangan lainnya:

1. Perulangan positif

Perulangan yang peningkatannya positif (meningkat) untuk variabel pengontrol perulangannya (mis: X++) ; (I+=3);

Contoh :

```
For (X = -2; X<= 4; X++) cout <<i<<endl;

For (I = 3;I<= 12; I+=3)  cout <<i<< endl;
```

2. Perulangan negatif

Perulangan dengan penurunan nilai dari besar ke kecil (menurun) untuk variabel pengontrol perulangannya (mis: X--) ; (I-=3);

For (X = 12; X<= 2; X--) cout<<i<< endl;

3. Perulangan dengan blok statement :

Perulangan ini memungkinkan sejumlah statemen di dalam blok diproses berulang-ulang.

```
#include <stdio.h>
        Main ()
        {
            Int I, N;
             Ffloat X, Rate, Total;

                    /*Menanyakan banyaknya data*/
           Printf ("Masukkan banyaknya Data : ");
           Scanf ("%d", &N);
           Printf("\n");

                 /*Memasukkan masing-masing data dari i=1 ke  I =N*/
           Total = 0
           For (I=1; I<=N; I++)
            {
                     Printf("Data Ke %3d ?",I);
                     Scanf("%f", &X);
                    Total = Total + X;
            }
    Rata = Total /N;
        /*Menampilkan hasil */
          Printf("\n");
          Printf ("Banyaknya Data  = %.3d\n", N);
          Printf ("Total Nilai  Data   = %.3f\n", Total);
          Printf ("Rerata Nilai  Data = %10.2f\n", Rata);
}
```

Hasil eksekusi program :

Jika program dijalankan :

Masukkan Banyaknya data :5

Data ke 1 ? 12.34

Data ke 2 ? 12.35

Data ke 3 ? 67.89

Data ke 4 ? 55.15

Data ke 5 ? 55.55

Banyaknya data = 5

Total nilai data = 203.28

Rerata nilai data = 40.66

4. Peningkatan tanpa nilai awal :

Nilai awal dari variabel pengontrol perulangan tidak harus ada di dalam statemen for , tetapi dapat ditentukan sebelumnya.

Contoh :

```
#include <stdio.h>
Main ()
{
    Int I;

    I= 3;
        For (  ; I<=5; I++ )
        {
            Printf ("%d\n", I);

        }
}
```

Statemen For tanpa peningkatan :

Program I

```
#include <stdio.h>
Main ()
{
    Int I:
   For (I=3; I<=5;   )
    Printf ("%d\n, I++);
 }
```

Program II

```
 #include <stdio.h>
Main ()
{
    Int I:
   For (I=3; I<=5;   )
   Printf ("%d\n, ++I);
}
```

Program III

```
 #include <stdio.h>
Main ()
{
   Int I:
  For (I=3; I<=5;   )
 Printf ("%d\n",  I);
 I++;
}
```

5. Statemen For Bersarang (For di dalam For / Neted For).

Syarat : tiap-tiap statemen for harus menggunakan variabel pengontrol perulangan yang berbeda satu dengan lainnya.

Contoh :

```
#include <stdio.h>
Main ()
{
  Int I, J:
  For (I=1; I<=3; I++ )
    {
       For( J=1;J<=3;J++)
       Printf (“  (I=%1d,Y=%1d)”, I, J);
       Printf(“\n”);
     }
}
```

Jika Program dijalankan :

(I=1, Y=1) (I=1, Y=2) (I=1,Y=3)
(I=2, Y=1) (I=2, Y=2) (I=2, Y=3)
(I=3 Y=1) (I=3, Y=2) (I=3, Y=3)

Contoh program lain For Bersarang :

```
#include <stdio.h>
#Define MAKS 8
Main ()
{
    Int baris, kolom, hasil_kali;
    For (baris=1; baris<=MAKS; baris++)
        {
          For(kolom=1; kolom<=MAKS; kolom++)
               {
                  Hasil_kali = baris* kolom;
                   Printf ("%2d", hasil_kali);
               }
          Printf("\n");    /* pindah baris*/
        }
}
```

a. Struktur Perulangan "WHILE"

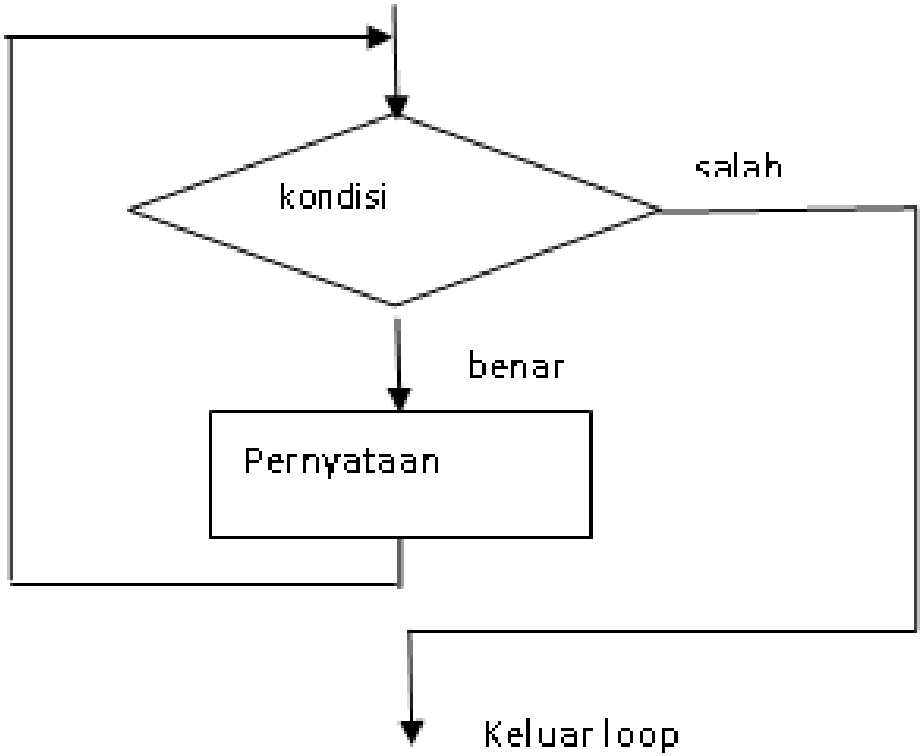
1. Banyak digunakan pada program terstruktur.
2. Perulangan ini banyak digunakan bila jumlah perulangannya belum diketahui.
3. Proses perulangan akan terus berlanjut selama kondisinya bernilai benar (true) dan akan berhenti bila kondisinya bernilai salah.
4. Pengujian terhadap loop dilakukan di bagian awal (sebelum tubuh loop).

Bentuk Umumnya :

```
while (kondisi)
{
    Pernyataan ;
}
```

Pernyataan dapat berupa pernyataan tunggal, majemuk ataupun kosong.

Proses perulangannya sebagai berikut :



Contoh program While :

```
#include <stdio.h>
Main ()
{
    Int i;
    i = 1
    While (i<5)
        Printf (" %1d, i++);
}
```

Prog. 1

```
#include <stdio.h>
Main ()
{
    Int i;
    i = 7
    While (i<5)
        Printf (" %1d, i++);
}
```

Prog. 2

Bagaimana Jika Prog. 2 dijalankan

Program While Bersarang :

```
#include <stdio.h>
#include <math.h>
Main ()
{
    Int X, I;

    Printf (" X   1/X    X^2    X^3 \n");
    Printf (" -------------------------");
    X=1;
    While (X<=10)
    {
        Printf (" %1d  %5.8f", X, 1.0/X);

       I=2;
       While (I<=3)
       {
          Printf (" %8.0f", pow(X,I));
          I++;
       }
       Printf("\n");
       X++;
    }
```

Jika dijalankan :

X	1/X	X^2	X^3
1	1.00000	1	1
2	0.50000	4	8
.			

3. Struktur Perulangan "DO.....WHILE…"

Pada dasarnya struktur perulangan do....while sama saja dengan struktur while, hanya saja pada proses perulangan dengan while, seleksi berada di while yang letaknya di atas

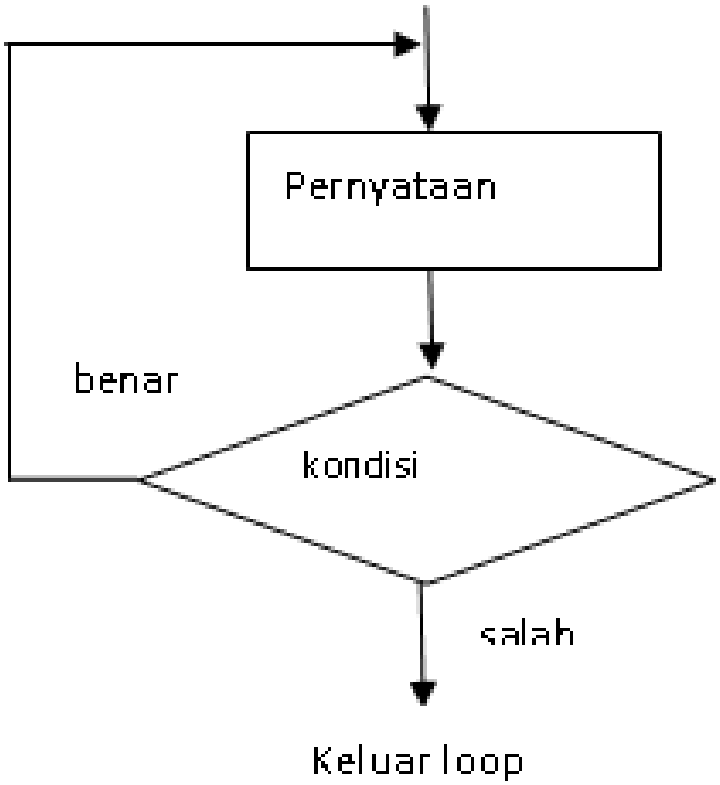
Sementara pada perulangan do....while, seleksi while berada di bawah batas perulangan.

Jadi dengan menggunakan struktur do…while sekurang-kurangnya akan terjadi satu kali perulangan.

3. Struktur Perulangan "DO.....WHILE…"

. Bentuk Umum :

```
do
{
   pernyataan ;
} while(kondisi);
```



3. Struktur Perulangan "DO.....WHILE…"

```
#include <stdio.h>
Main ()
{
    Int X;

    X=9;
    Do
        Printf (" %1d\n", X);
    While (X<5);
 }
```

Hasil :

```
#include <stdio.h>
Main ()
{
    Int X;

    X=3;
    Do
        Printf (" %1d\n", X);
    While (X<5);
 }
```

Hasil :

Hati-hati ??

```
#include <stdio.h>
Main ()
{
    Int X;

    X=3;
    Do
    {
        Printf (" %1d\n", X);
        X++;
    }
    While (X<5);
 }
```

Hasil :

3

Contoh program Do –While :

```
/*Untuk membaca tombol Y atau T*/
#include <stdio.h>
Main ()
{
    Char pilihan;
    Inst sudah_benar;

    Printf ("pilihlah Y atau T.: ");
        /*Program hanya mau dilanjutkan jika tombol Y, y T atau t yang ditekan*/

    Do
       {
        Pilihan = getche();   /* baca tombol */
        Sudah_benar = (pilihan =='Y') || (pilihan =='y') || (pilihan =='T') || (pilihan =='t');
       }

    While (! Sudah_benar);
    Switch (pilihan)
        {
          Case 'Y':
          Case 'y':
               Puts ("\n {ilihan anda adalah Y");
               Break;
          Case 'T':
          Case 't':
               Puts ("\n {ilihan anda adalah T");
               Break
        }
     }
```

4. PERNYATAAN *continue* dan *break*

- Digunakan untuk pergi ke bagian awal dari blok loop untuk memulai iterasi berikutnya.
- Pernyataan continue menyebabkan perulangan kembali ke awal mulainya perulangan dengan mengabaikan statemen –statemen berikutnya.
- Statemen continue bisa digunakan untuk perulangan : for, while dan do- -while.

.

4. PERNYATAAN *continue* dan *break*

Berikut contoh program continue.

Jika ada nilai negatif ((nilai<0), maka pernyataan continue akan dijalankan.

```
#include <stdio.h>
    Main ()
    {
        Int I, N;
        Ffloat Nilai, Rata, Total;
       /*Menanyakan banyaknya data*/
       Printf ("Masukkan banyaknya Data : ");
       Scanf ("%d", &N);
       Printf("\n");

            /*Memasukkan masing-masing data dari i=1 ke  I =N*/
       Total = 0
       For (I=1; I<=N; )
            {
              Printf("Data Ke %3d ?",I);
              Scanf("%f", &Nilai);
                    /*Periksa apakah nilai yang dimasukkan negatif*/
              IF (Nilai < 0)
                 Continue;

             Total = Total + Nilai;
             I++;
           }

     Rata = Total /N;
```

Lanjutan program …:

```
        Rata = Total /N;

            /*Menampilkan hasil */
        Printf("\n");
        Printf ("Banyaknya Data     = %.3d\n", N);
        Printf ("Total Nilai  Data     = %.3f\n", Total);
        Printf ("Rerata Nilai  Data   = %10.2f\n", Rata);
}
```

Contoh program contimnue yang lain :

Contoh lain :

```
/*Program continue untuk melihat bilangan bil ganjil antara 1 sd 25 kecuali 15*/
#include <stdio.h>
    Main ()
    {
        Int x;
    For (x = 7; x<=25; +=2)   /*menampilkan bil gasal*/
        {
         IF ( x== 15)
              Continue;
         Printf("%d ",x);
        }
  }
```

Lanjutan program …:

Pernyataan *break* akan selalu terlihat digunakan bila menggunakan pernyataan *switch*.

Pernyataan ini juga digunakan dalam loop.

Bila pernyataan ini dieksekusi, maka akan mengakhiri loop dan akan menghentikan iterasi pada saat tersebut.

Contoh :

```
#include <stdio.h>
    Main ()
    {
        Int I;
    For (I = 0; I<10 ;I++)
        {
         IF ( I== 4)
              Continue;
         Printf("Bilangan %d ",x);
        Else
              IF( I ==8)
                 Break;
         }
   }
```

Perulangan dari suatu bilangan sebanyak 10 kali. Tetapi, pada perulangan I =4, ada perintah continue. Program langsung meloncat ke loop berikutnya dan ketika sampai perulangan i = 8, ada perintah break. Otomatis program akan berhenti dan tidak sampai ke I =10.

Dan program akan mencetak bilangan 0, 1, 2, 3, 5, 6 ,7 ,8.

e. **PERNYATAAN** *go to (Lompatan)*

Diperlukan untuk melakukan suatu lompatan ke suatu pernyataan berlabel yang ditandai dengan tanda " **:** ".
Bentuknya : **Goto label:**

Bentuk umumnya adalah :

```
goto bawah;
        pernyataan1;
        pernyataan2;
bawah : pernyataan 3;
```

: Program akan melompat pernyataan berlabel *bawah* dan melakukan pernyataan 3.

Contoh :

```
#include <stdio.h>
Main()
{
   Int I;

   I = 1;
   Perulangan ;
      Printf("%d\n", I++);
      If (I<=5)
      Goto perulangan ;
}
```

TERIMA KASIH