

# Proses Perulangan (Looping)





# TUJUAN

- Menjelaskan proses perulangan menggunakan pernyataan for, while, dan do-while.
- Menjelaskan penggunaan pernyataan break dan continue, goto.
- Menjelaskan loop di dalam loop (nested loop) dan contoh kasusnya.

# Pernyataan for

- Digunakan untuk membuat looping dengan jumlah perulangan yang ditentukan di awal.

- Sintak:

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

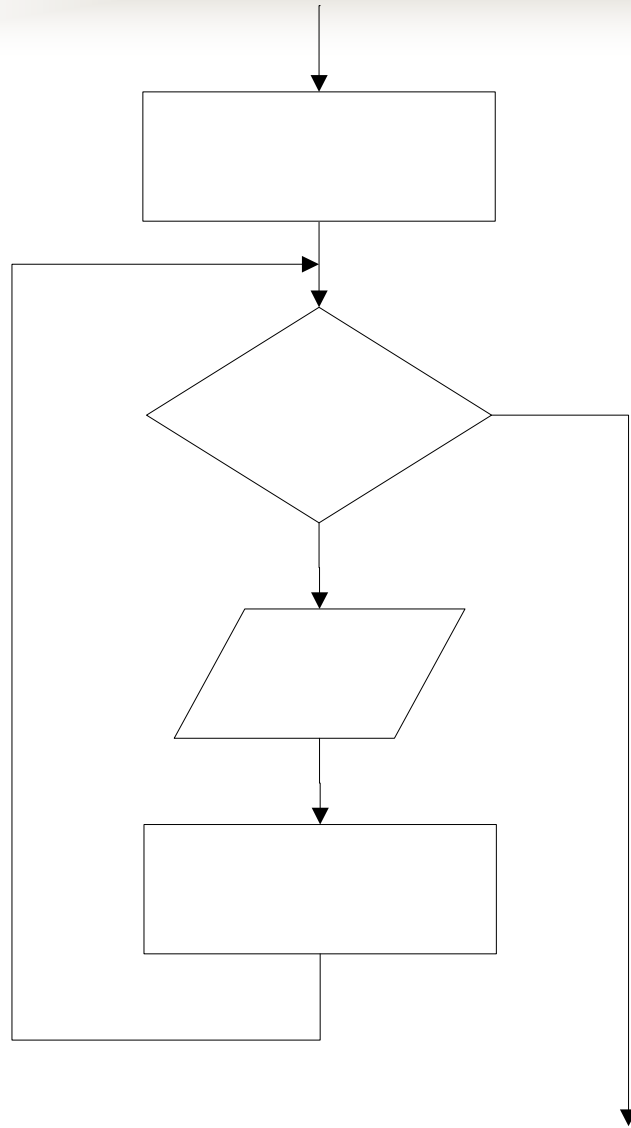
- **Ungkapan1**: digunakan untuk memberikan inisialisasi terhadap variabel pengendali *loop*.
- **Ungkapan2**: dipakai sebagai kondisi untuk keluar dari *loop*.
- **Ungkapan3**: dipakai sebagai pengatur kenaikan nilai variabel pengendali *loop*.

## Contoh penggunaan for

```
for (bil = 1; bil <= 15; bil += 3)
    printf("%d\n", bil);
```

Akan menghasilkan:

```
1
4
7
10
13
```



bil

bil <=

# Pernyataan `while`

- Pengecekan terhadap loop dilakukan di bagian awal.
- Pernyataan didalamnya bisa tidak dikerjakan sama sekali.
- Sintak

```
while (kondisi )
```

```
    pernyataan ;
```

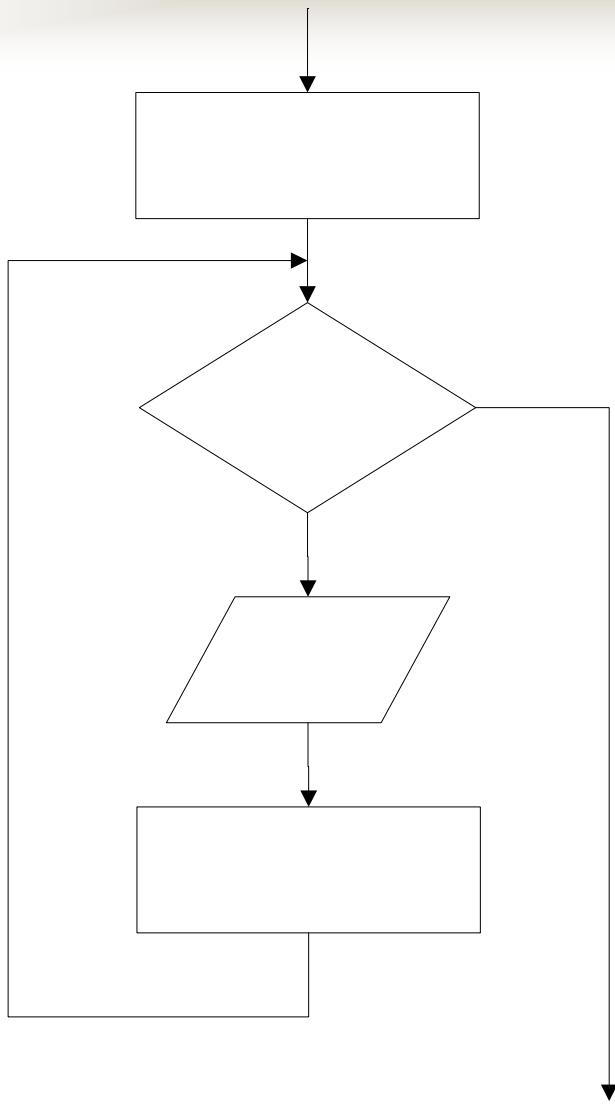
- Selama `kondisi` benar maka `pernyataan` dikerjakan
- Jika `kondisi` salah → keluar dari loop

# Contoh penggunaan while

```
bil = 1;
while (bil <= 15)
{
    printf("%d\n", bil);
    bil = bil + 3;
}
```

Akan menghasilkan:

```
1
4
7
10
13
```



bil

bil <

# Pernyataan do-while

- Pengecekan terhadap loop dilakukan di bagian akhir.
- Pernyataan didalamnya pasti dijalankan (minimal 1 kali).

- Sintak

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

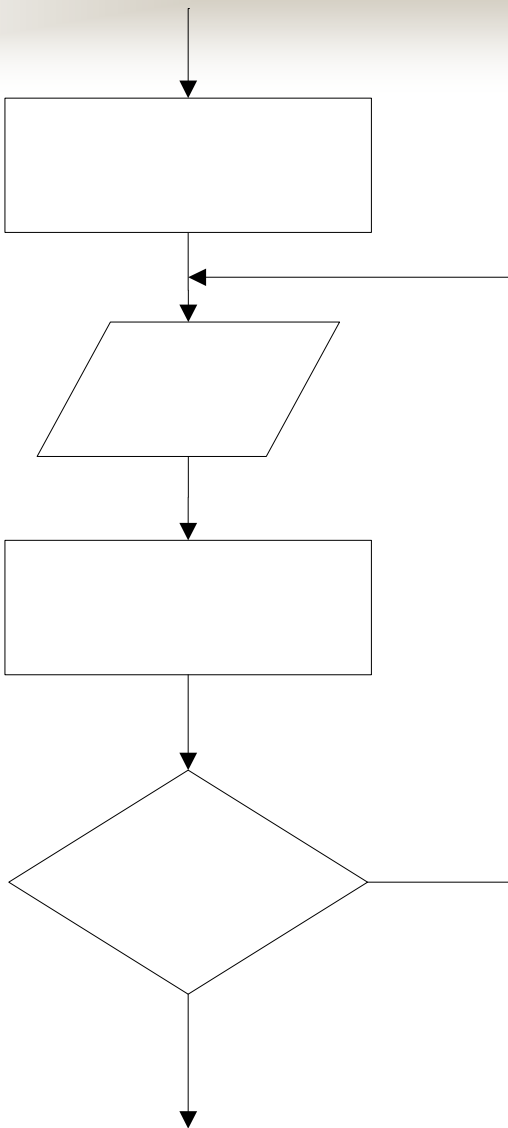
- Mula-mula **pernyataan** dijalankan, selanjutnya **kondisi** diuji jika **benar** dilakukan **perulangan**, jika **salah** maka keluar dari loop

## Contoh penggunaan do-while

```
bil = 1;
do {
    printf("%d\n", bil);
    bil = bil + 3;
} while (bil <= 15);
```

Akan menghasilkan:

```
1
4
7
10
13
```

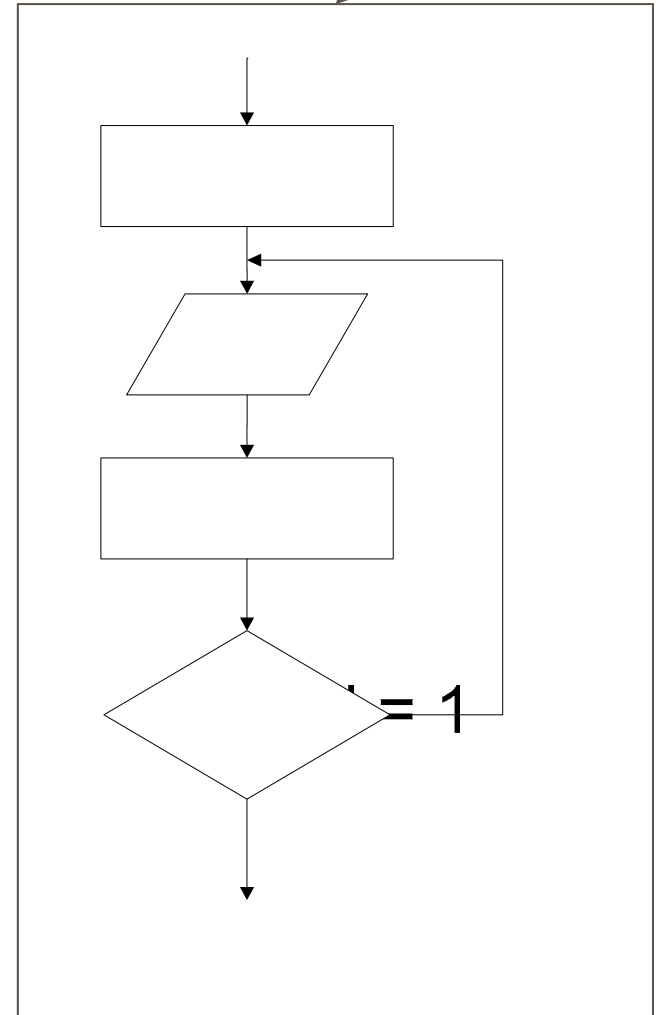
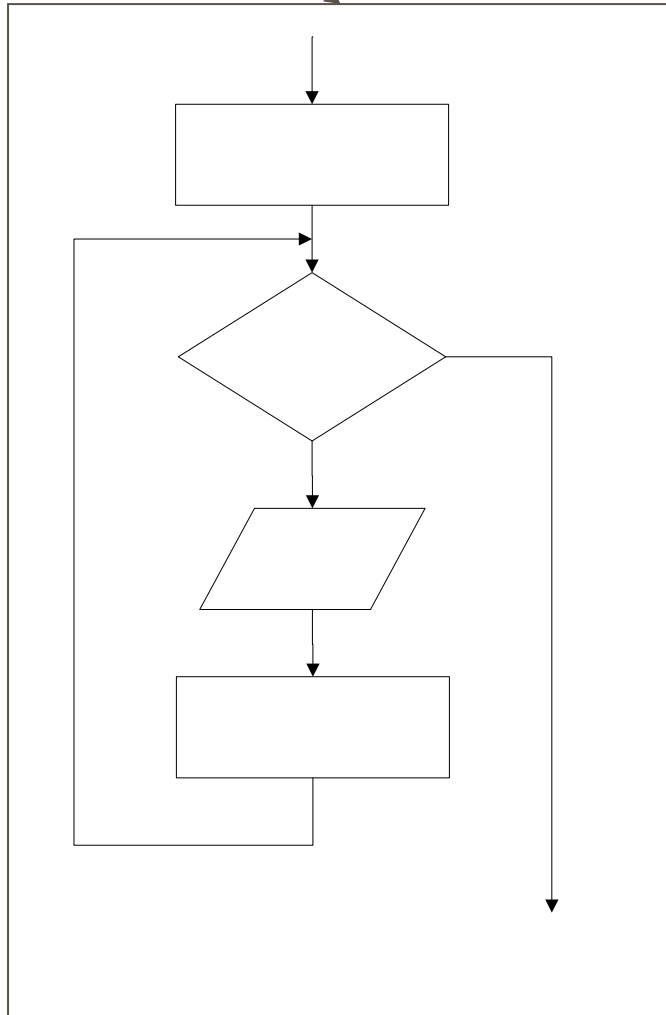


bil

Ce

I

# while VS do-while



bil <= 15 ?

# Contoh for, while, dan do-while

```
#include <stdio.h>
main()
{
    int bil;

    for(bil = 1; bil <= 15; bil += 3)
        printf("%d\n", bil);

    printf("\n");
    bil = 1;
    while (bil <= 15)
    {
        printf("%d\n", bil);
        bil = bil + 3;
    }

    printf("\n");
    bil = 1;
    do {
        printf("%d\n", bil);
        bil = bil + 3;
    } while (bil <= 15);
}
```




## Pernyataan **break**

- Berfungsi untuk keluar dari loop → untuk looping dengan **for**, **while**, dan **do-while**.
- Berfungsi untuk keluar dari struktur **switch**.
- Sintak:  
`break ;`

# Contoh penggunaan break


- Pada loop:

```
while(kondisi)
{
    break;
}
statement-x;
```



- Pada switch:

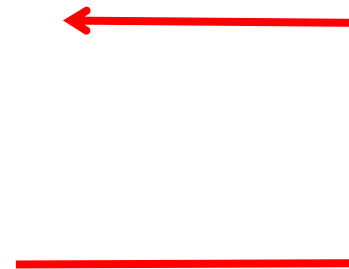
```
switch (ekspresi)
{
    . . . . .
    case konstanta-2:
        pernyataan-21;
        break;
    . . . . .
}
```



# Pernyataan continue

- Pada loop:


```
while(kondisi)
{
    continue;
}
statemen-x;
```



# break VS continue


- Break:

```
while(kondisi)
{
    break;
    statement-x;
}
statement-y;
```



- Continue:

```
while(kondisi)
{
    continue;
    statement-x;
}
statement-y;
```

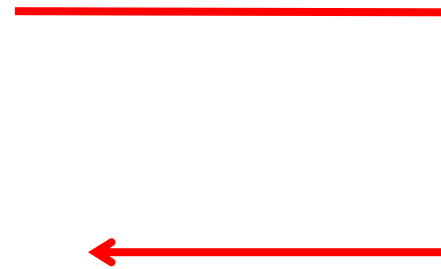


# Pernyataan goto

- Berfungsi untuk mengarahkan eksekusi ke pernyataan yang diawali dengan suatu label.
- Contoh :

**goto nama\_label;**

**label :**



# Loop Di Dalam Loop

	1	2	3	4	5	6	7	8
1	1	2	3	4	5	6	7	8
2	2	4	6	8	10	12	14	16
3	3	6	9	12	15	18	21	24
4	4	8	12	16	20	24	28	32
5	5	10	15	20	25	30	35	40
6	6	12	18	24	30	36	42	48
7	7	14	21	28	35	42	49	56
8	8	16	24	32	40	48	56	64

# Nested loop

```
main() {
    int baris, kolom, hasil_kali;

    for (baris = 1; baris <= 10; baris++)
    {
        for (kolom=1; kolom <= 10; kolom++)
        {
            hasil_kali = baris * kolom;
            printf ("%2d", hasil_kali);
        }
        printf("\n");    /* pindah baris */
    }
}
```



# Exercise

1. Gunakan loop *for* dan *nested while loop* untuk mendapatkan tampilan sbb :

1

22

333

4444

55555

2. Buatlah program untuk menghitung nilai faktorial menggunakan:

- `for`
- `while`
- `do-while`



# Exercise

3. Gunakan loop *for* untuk menjumlahkan seluruh bilangan antara 10 sampai dengan 100 ke dalam sebuah variabel **total**. Asumsikan bahwa variabel **total** tidak diinisialisasi terlebih dahulu dengan nilai nol.