



Konsep Pemrograman – Pertemuan 5



Perulangan Proses - 1

(Looping)

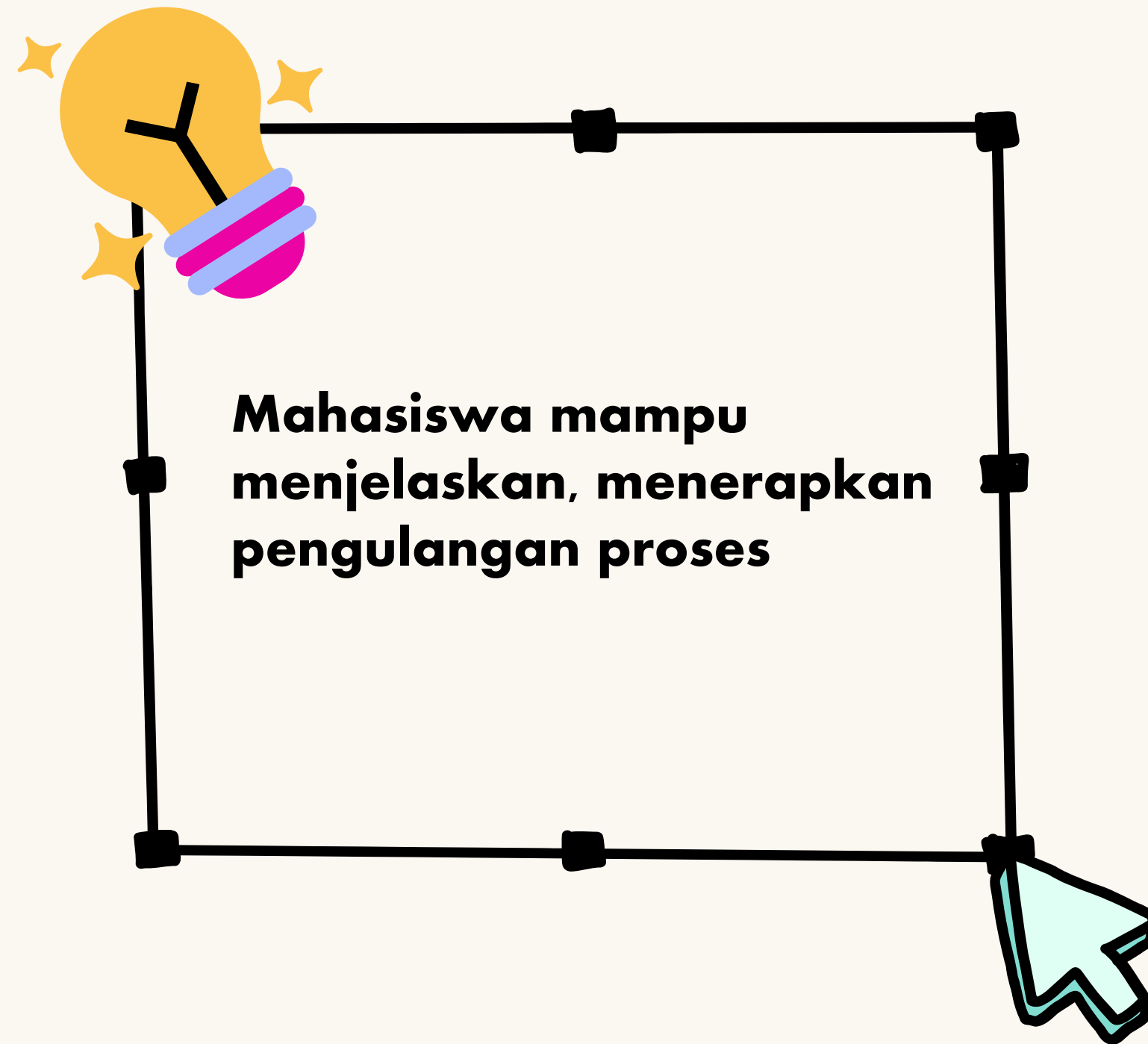


Yunia Ikawati

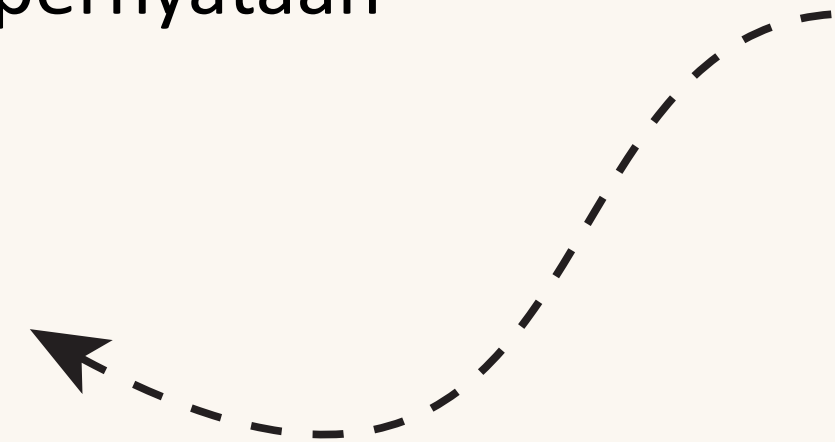
Teknik Informatika-PENS

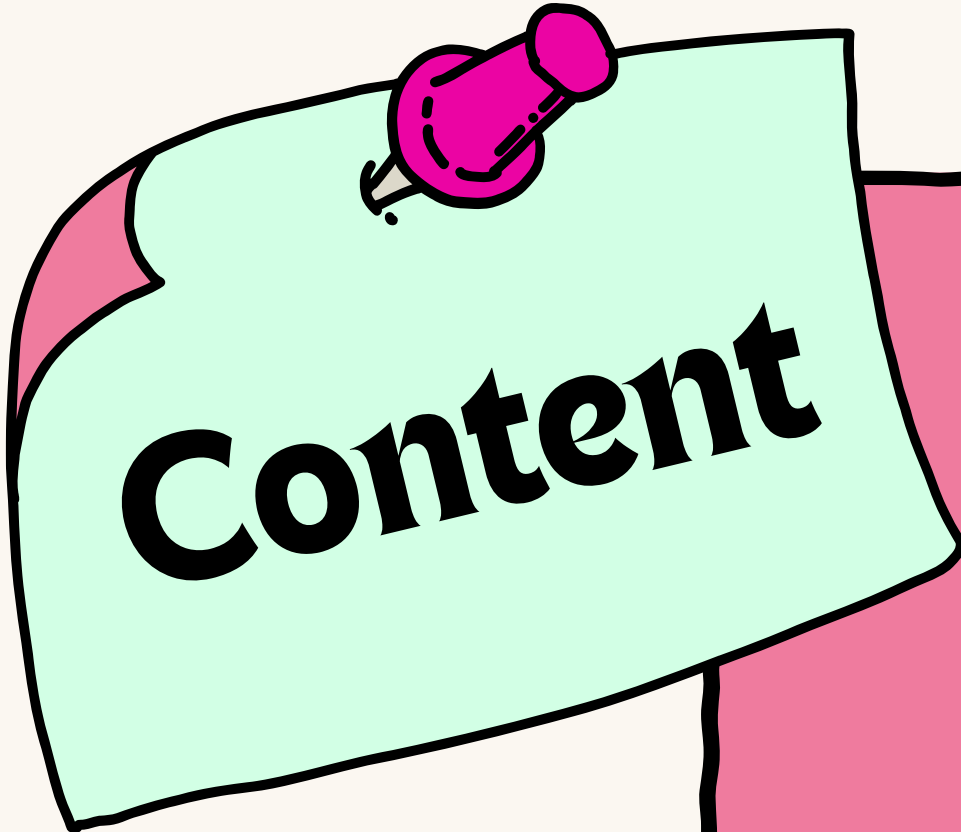


Sub CPMK (Capaian Pembelajaran Mata Kuliah)

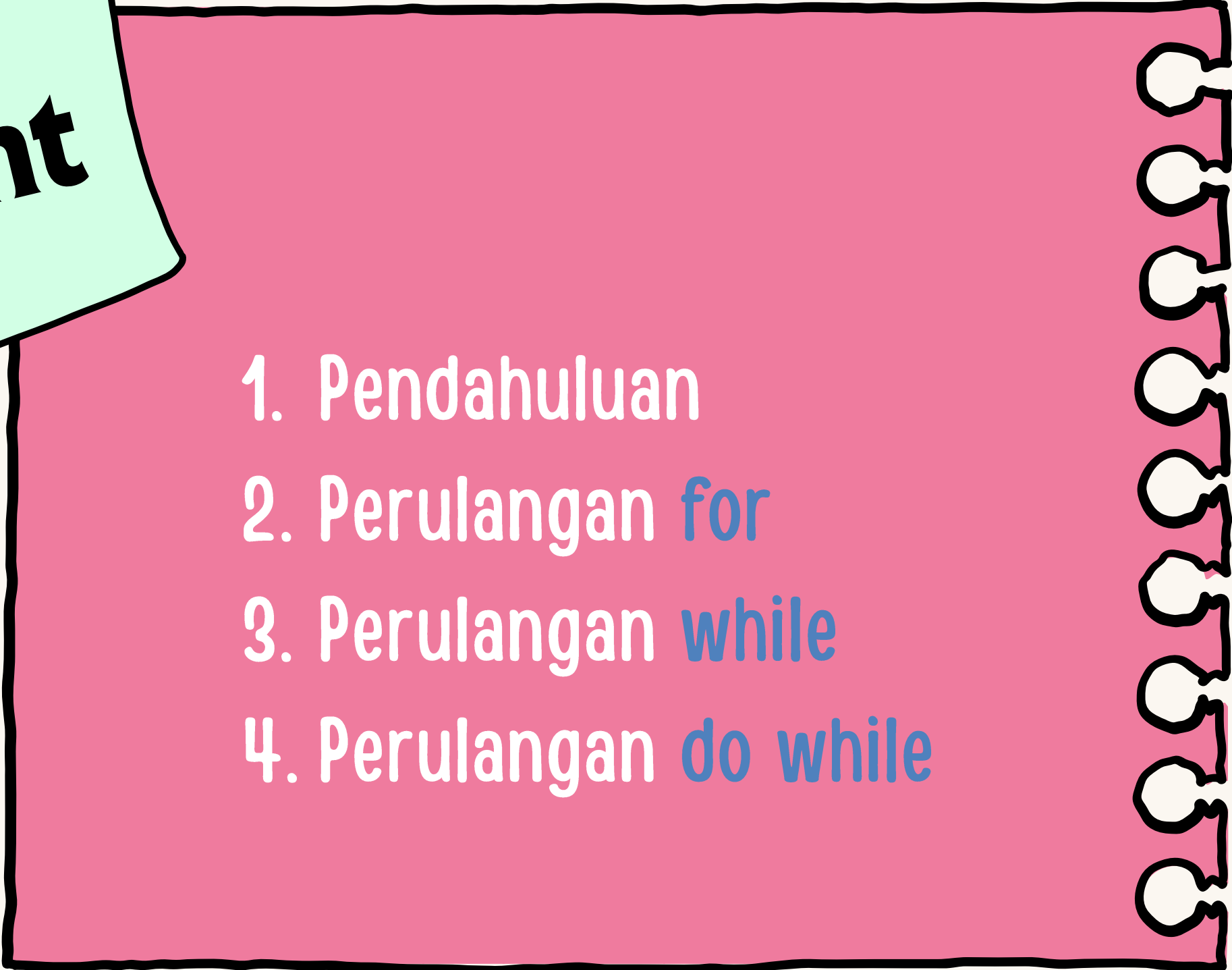
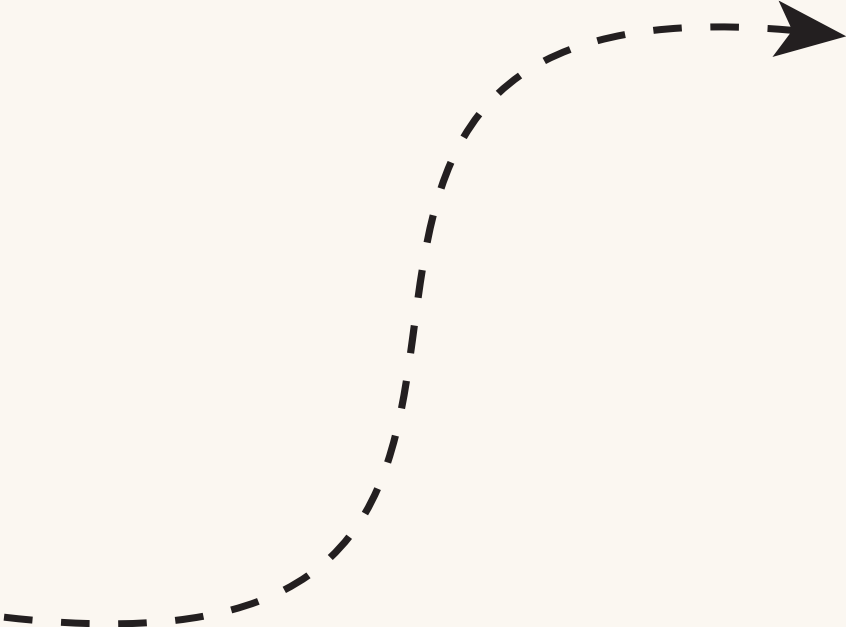


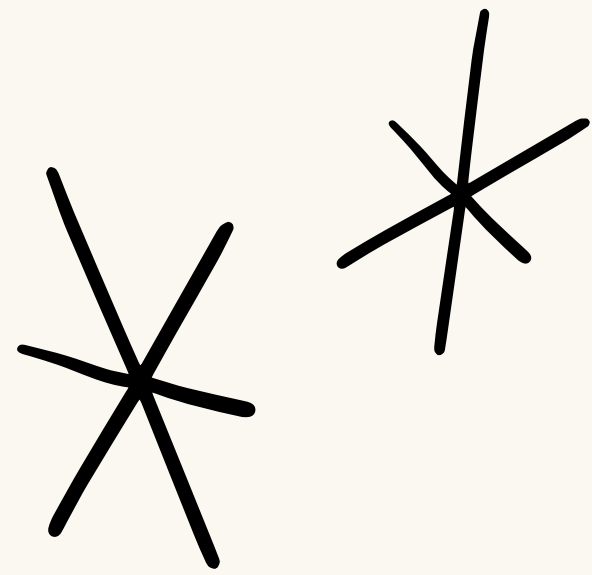
- Mahasiswa mampu memecahkan masalah menggunakan pernyataan **for, while, do-while, nested loop**
- Mahasiswa mampu menjelaskan perbedaan penggunaan pernyataan **for, while dan do-while**





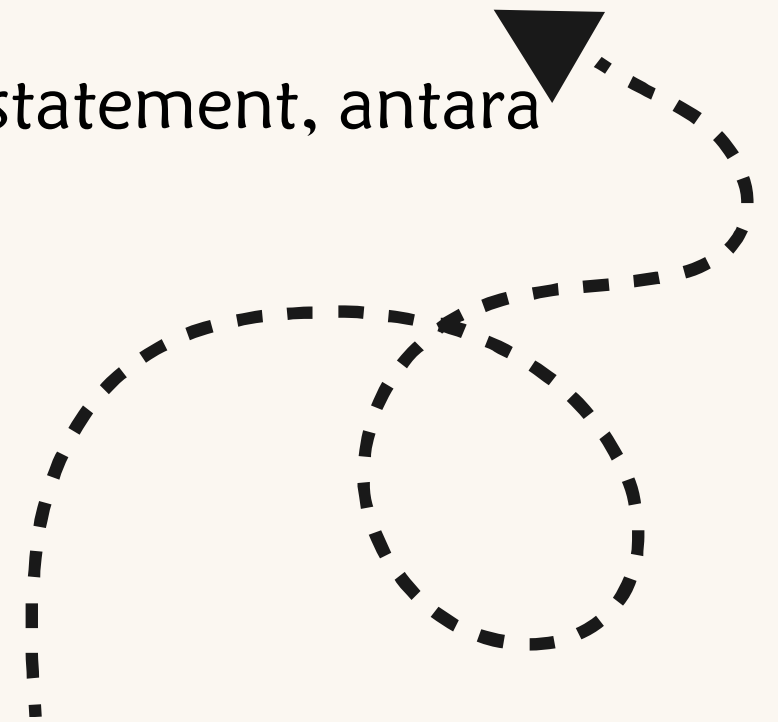
Content

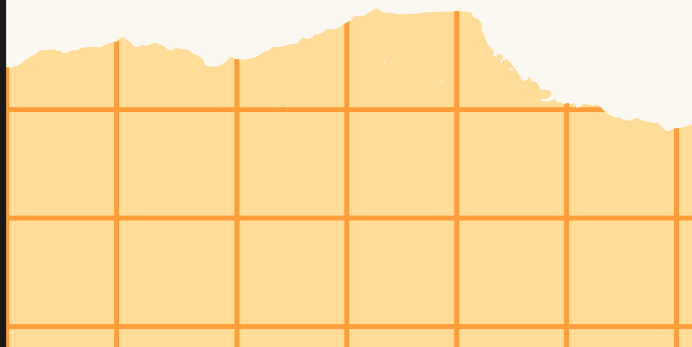
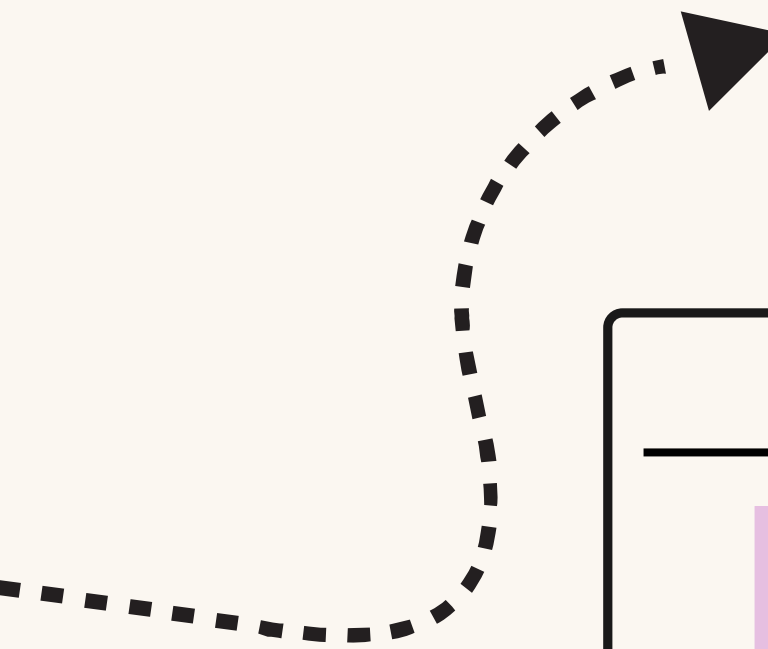
- 
1. Pendahuluan
 2. Perulangan **for**
 3. Perulangan **while**
 4. Perulangan **do while**
- 
- 



Pendahuluan

- ❑ Pada semua bahasa pemrograman, perulangan proses ditangani dengan suatu mekanisme yang disebut looping.
- ❑ Dengan menggunakan looping, suatu proses yang berulang (misalnya menampilkan tulisan yang sama sebanyak seratus kali pada layar) dapat diimplementasikan dengan kode program yang lebih pendek.
- ❑ User tidak perlu melakukan copy-paste kode program yang ingin dijalankan secara berulang berkali-kali.
- ❑ Untuk menerapkan perulangan, Bahasa C menyediakan 3 looping statement, antara lain:
 - ✓ **for**
 - ✓ **while**
 - ✓ **do-while**



- 
- 
- 
- 
- 
- Statement **for** digunakan apabila user sudah mengetahui jumlah perulangan yang ingin dilakukan.
 - Apabila jumlah perulangan belum diketahui, maka user bisa menggunakan salah satu looping statement berikut:
- 
- 

Pemilihan Looping Statement

1. Statement **while**

- ✓ Pengecekan kondisi akan dilakukan di awal
- ✓ Statement yang berada di dalam body loop akan dijalankan selama masih memenuhi kondisi yang ditetapkan
- ✓ Ada kemungkinan statement yang berada di dalam body loop tidak dijalankan sama sekali.

2. Statement **do-while**

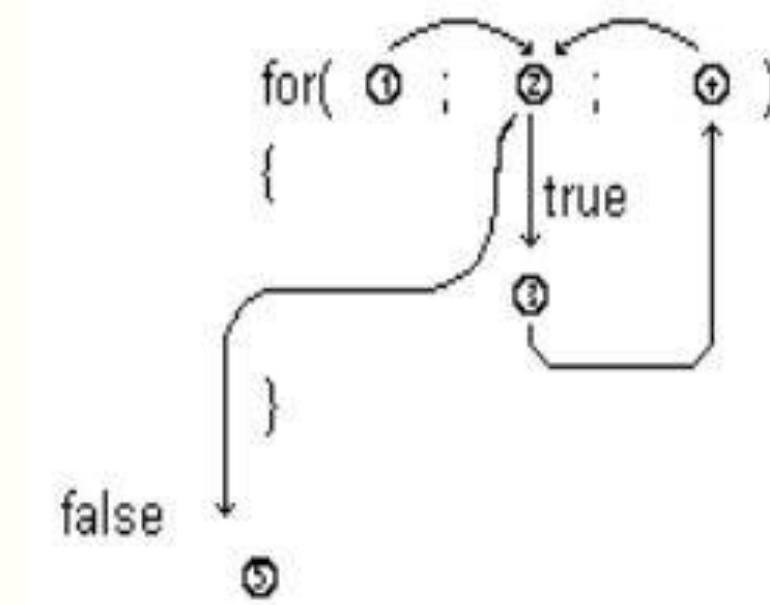
- ✓ Pengecekan kondisi akan dilakukan di akhir
- ✓ Statement yang berada di dalam body loop akan dijalankan selama masih memenuhi kondisi yang ditetapkan
- ✓ Statement yang berada di dalam body loop akan dijalankan minimal 1 kali



Statement for

Bentuk Umum:

```
for(ekspresi1; ekspresi2; ekspresi3) {  
    // body of loop  
    statement;  
    statement;  
    ...  
    statement;  
}
```



- ✓ ekspresi1: inialisasi nilai awal untuk variabel pengendali loop.
- ✓ ekspresi2: continue condition, kondisi yang harus terpenuhi agar loop tetap berjalan.
- ✓ ekspresi3: pengatur perubahan (naik/turun) nilai dari variabel pengendali loop.
- ✓ Ketiga ekspresi dipisahkan dengan tanda titik koma.
- ✓ Apabila hanya ada 1 statement yang berada di dalam body of loop, maka tidak perlu menggunakan kurung kurawal.

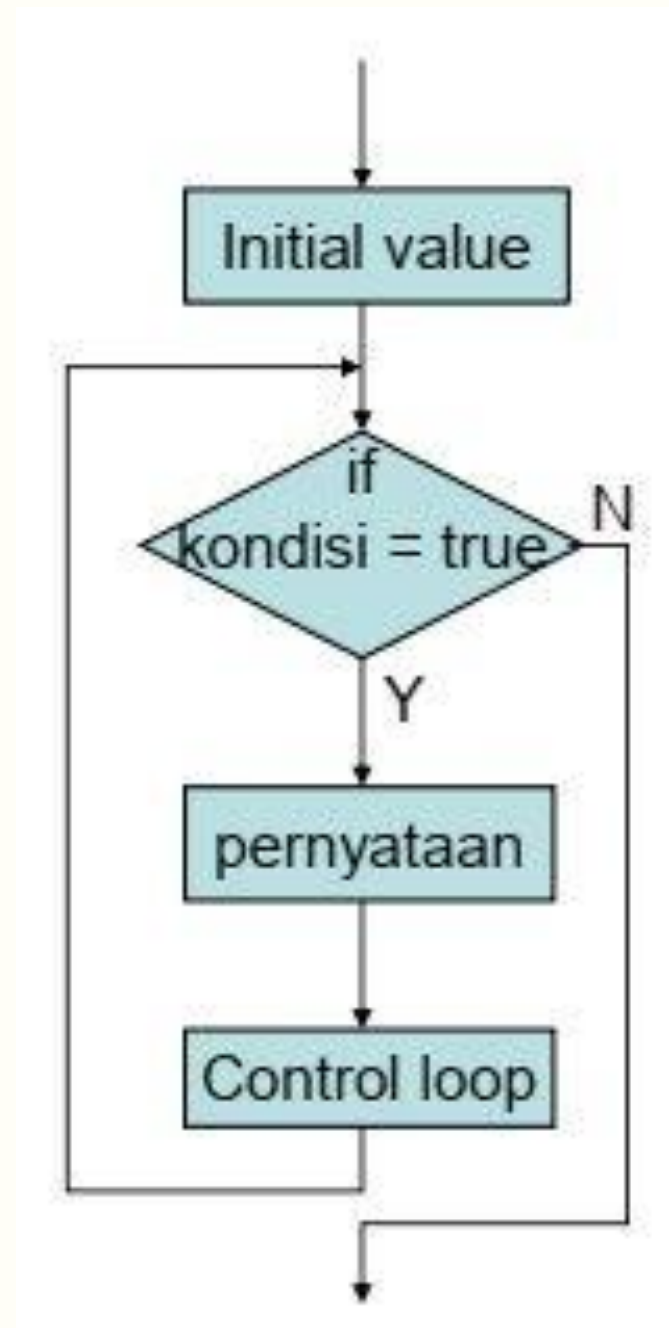
Contoh

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int i;
6      for(i = 1; i<=3; i++){
7          printf("ini baris ke-%d\n", i);
8      }
9  }
```

•ekspresi1: *initial value* → $i = 1$

•ekspresi2: *continue condition* → $i \leq 3$

•ekspresi3: pengatur perubahan nilai dari variabel pengendali *loop*.
→ $i++$



Contoh 1

```
1 #include <stdio.h>
2
3 int main()
4 {
5     int i;
6     for(i = 1; i<=3; i++){
7         printf("ini baris ke-%d\n", i);
8     }
9 }
```

Result

```
$gcc -o main *.c
$main
ini baris ke-1
```

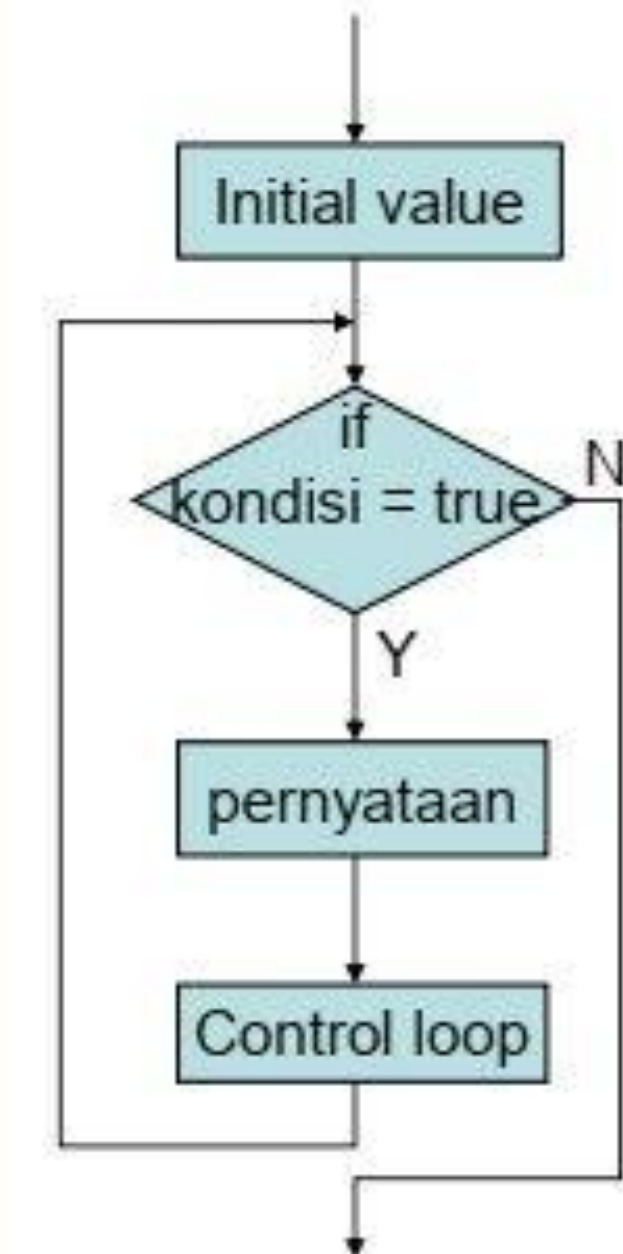
Output iterasi ke-1

• Iterasi 1

$i = 1$

$i \leq 3 \rightarrow 1 \leq 3 \rightarrow \text{TRUE}$

Baris ke-7 dijalankan, sehingga muncul output ke layar "ini baris ke-1" $i++ \rightarrow i = i+1 \rightarrow i = 1+1 = 2$



Contoh 1

```
1 #include <stdio.h>
2
3 int main()
4 {
5     int i;
6     for(i = 1; i<=3; i++){
7         printf("ini baris ke-%d\n", i);
8     }
9 }
```

Result

```
$gcc -o main *.c
$main
ini baris ke-1
ini baris ke-2
```

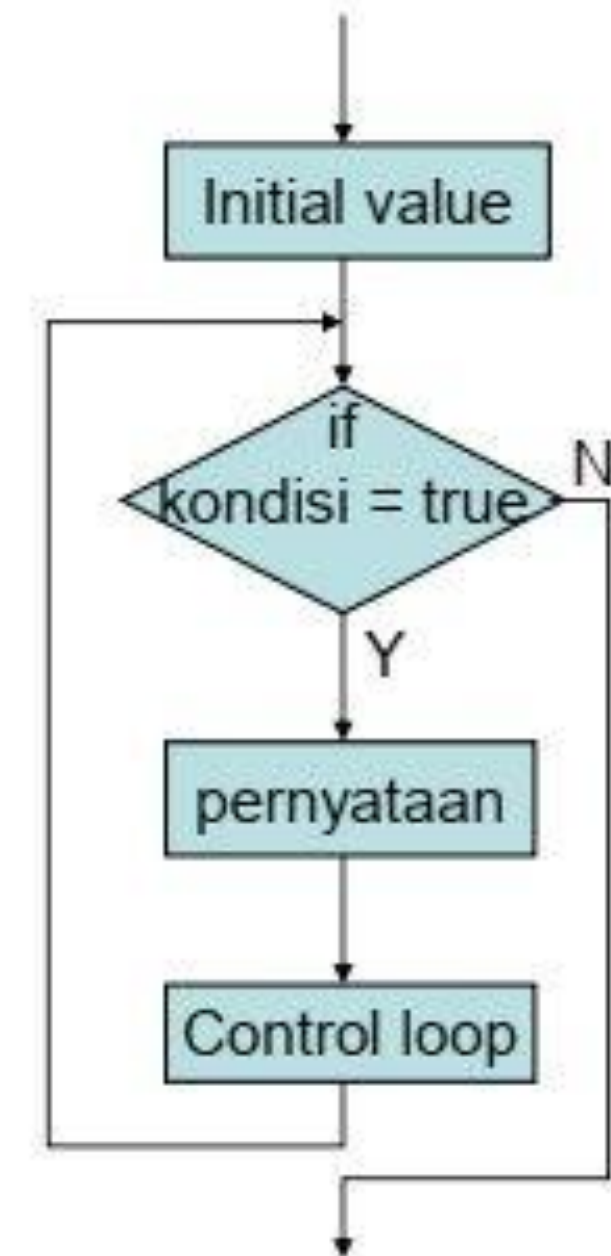
Output iterasi ke-2

• Iterasi 2

$i = 2$

$i \leq 3 \rightarrow 2 \leq 3 \rightarrow \text{TRUE}$

Baris ke-7 dijalankan, sehingga muncul output ke layar "ini baris ke-2" $i++ \rightarrow i = i+1 \rightarrow i = 2+1 = 3$



Contoh 1

```
1 #include <stdio.h>
2
3 int main()
4 {
5     int i;
6     for(i = 1; i<=3; i++){
7         printf("ini baris ke-%d\n", i);
8     }
9 }
```

Result

```
$gcc -o main *.c
$main
ini baris ke-1
ini baris ke-2
ini baris ke-3
```

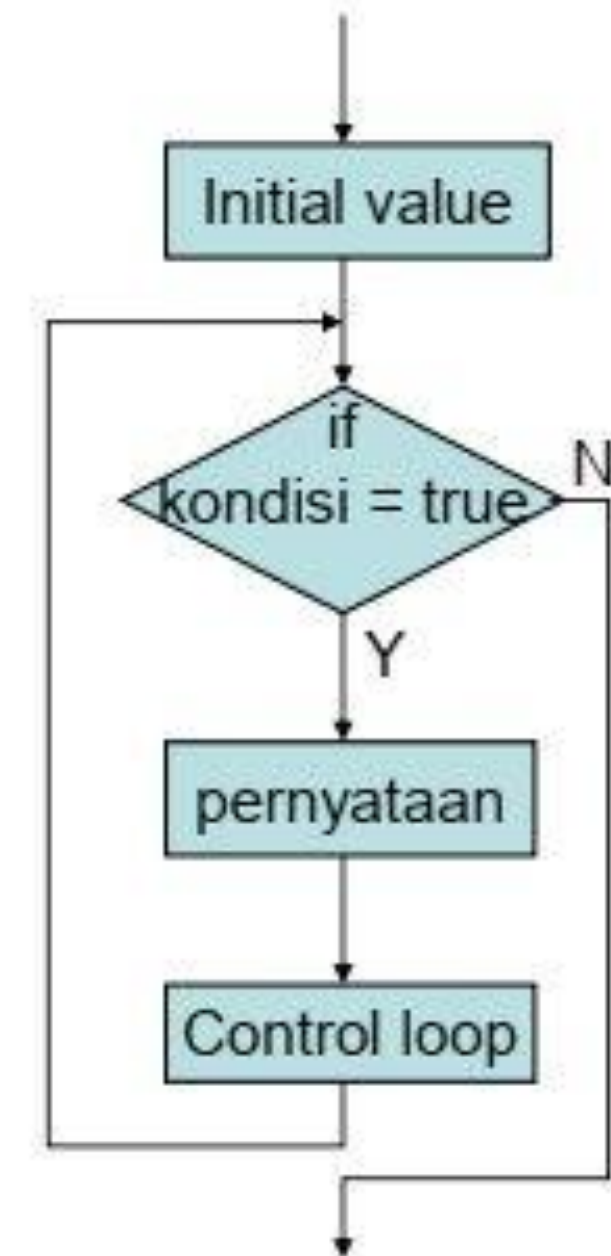
Output iterasi ke-3

• Iterasi 3

$i = 3$

$i \leq 3 \rightarrow 3 \leq 3 \rightarrow \text{TRUE}$

Baris ke-7 dijalankan, sehingga muncul output ke layar "ini baris ke-3" $i++ \rightarrow i = i+1 \rightarrow i = 3+1 = 4$



Contoh 1

```
1 #include <stdio.h>
2
3 int main()
4 {
5     int i;
6     for(i = 1; i<=3; i++){
7         printf("ini baris ke-%d\n", i);
8     }
9 }
```

Result

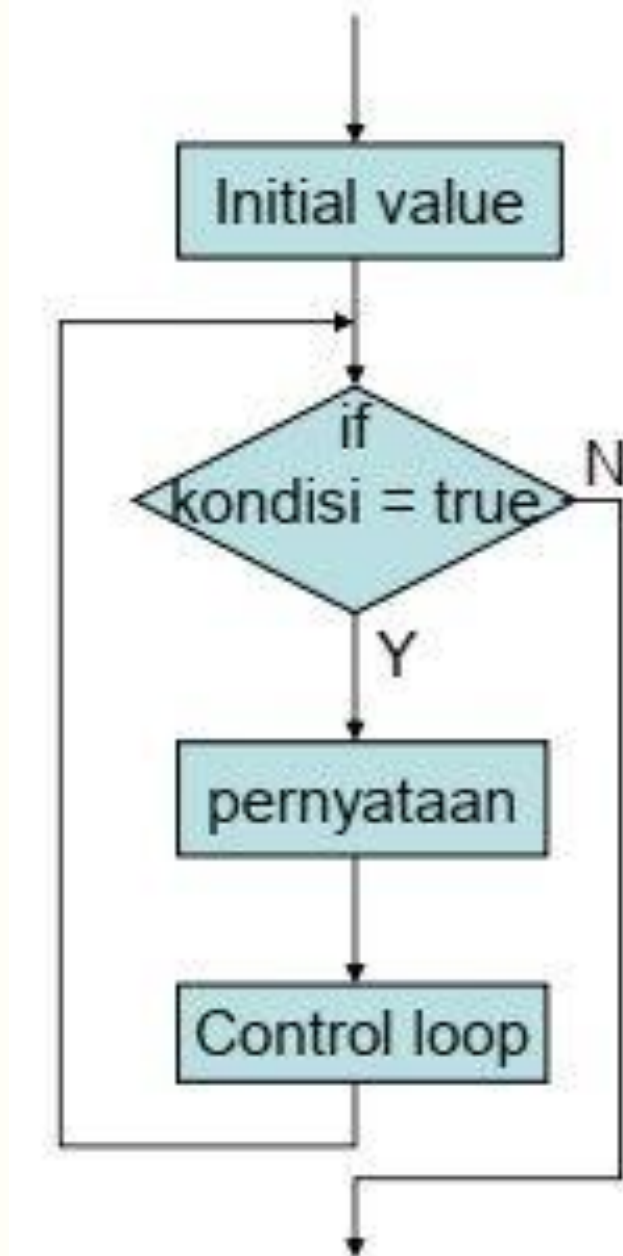
```
$gcc -o main *.c
$main
ini baris ke-1
ini baris ke-2
ini baris ke-3
```

• Iterasi 4

$i = 4$

$i \leq 3 \rightarrow 4 \leq 3 \rightarrow \text{FALSE}$

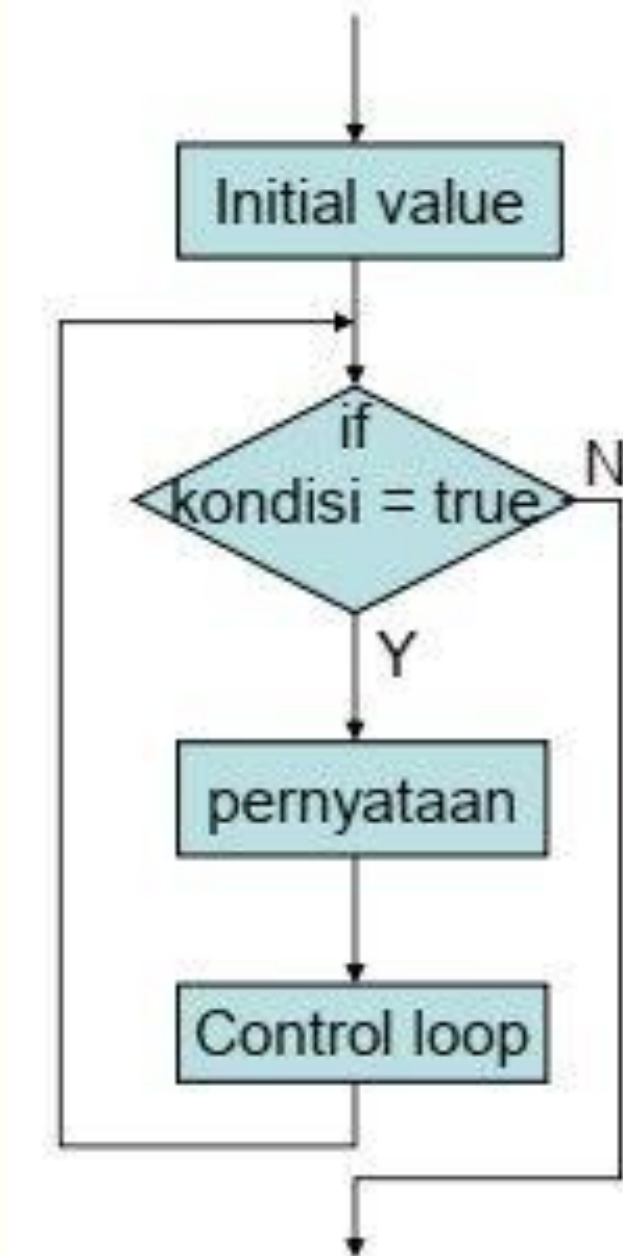
continue condition tidak terpenuhi, maka keluar dari *loop*.



Contoh 2

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int i;
6      for(i = 10; i >= 1; i -= 3){
7          printf("%d\t", i);
8      }
9  }
```

- ekspresi 1: *initial value* → $i = 10$
- ekspresi 2: *continue condition* → $i \geq 1$
- ekspresi 3: pengatur perubahan nilai dari variabel pengendali *loop*.
→ $i = 3$ → $i = i - 3$



Contoh 2

```
1 #include <stdio.h>
2
3 int main()
4 {
5     int i;
6     for(i = 10; i >= 1; i -= 3){
7         printf("%d\t", i);
8     }
9 }
```

Result

```
$gcc -o main *.c
```

```
$main
```

```
10
```

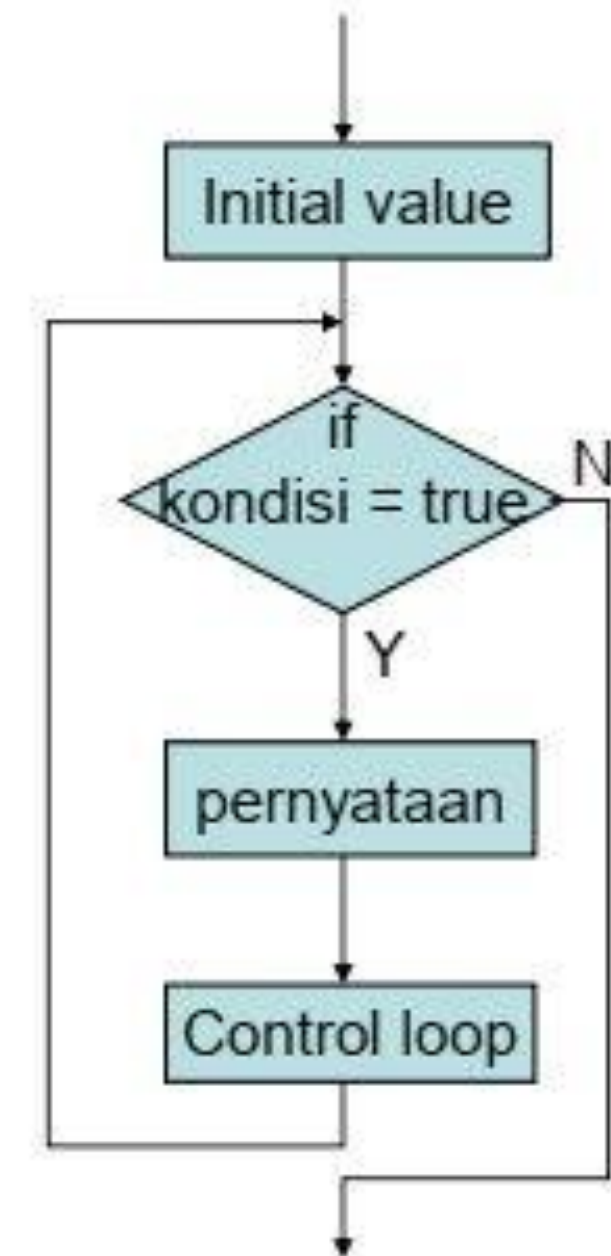
Output iterasi ke-1

- Iterasi ke-1 $i = 10$

$i \geq 1 \rightarrow 10 \geq 1 \rightarrow \text{TRUE}$

Baris ke-7 dijalankan, sehingga muncul output ke layar "10"

$i - 3 \rightarrow i = i - 3 \rightarrow i = 10 - 3 = 7$



Contoh 2

```
1 #include <stdio.h>
2
3 int main()
4 {
5     int i;
6     for(i = 10; i >= 1; i -= 3){
7         printf("%d\t", i);
8     }
9 }
```

Result

```
$gcc -o main *.c
```

```
$main
```

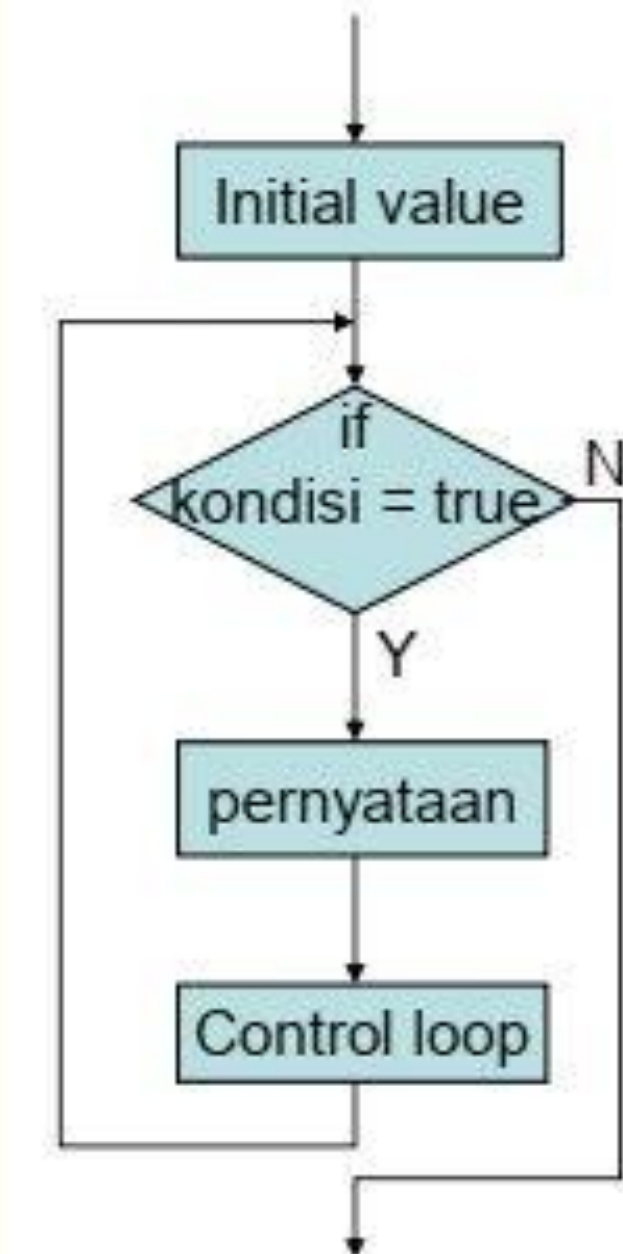
```
10    7
```

Output iterasi ke-2

• Iterasi ke-2 $i = 7$

$i \geq 1 \rightarrow 7 \geq 1 \rightarrow \text{TRUE}$

Baris ke-7 dijalankan, sehingga muncul output ke layar "7 " $i -= 3 \rightarrow i = i - 3 \rightarrow i = 7 - 3 = 4$



Contoh 2

```
1 #include <stdio.h>
2
3 int main()
4 {
5     int i;
6     for(i = 10; i >= 1; i -= 3){
7         printf("%d\t", i);
8     }
9 }
```

Result

```
$gcc -o main *.c
```

```
$main
```

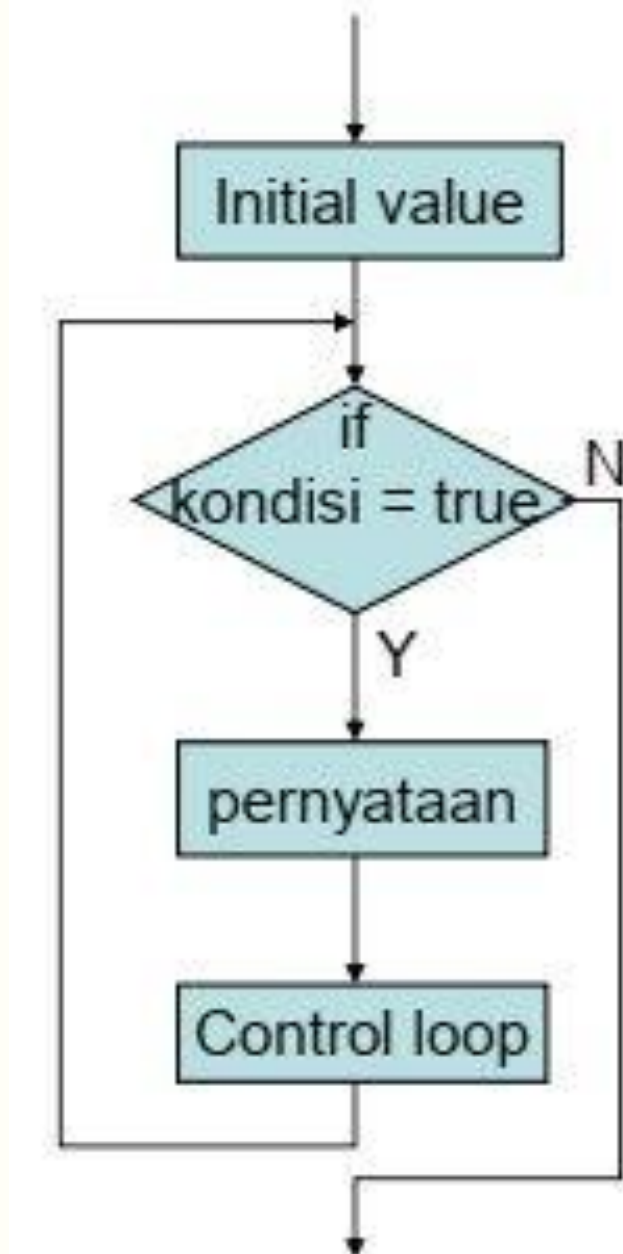
```
10    7    4    1
```

Output iterasi ke-3

• Iterasi ke-3 $i = 4$

$i \geq 1 \rightarrow 4 \geq 1 \rightarrow \text{TRUE}$

Baris ke-7 dijalankan, sehingga muncul output ke layar "4 " $i -= 3 \rightarrow i = i - 3 \rightarrow i = 4 - 3 = 1$



Contoh 2

```
1 #include <stdio.h>
2
3 int main()
4 {
5     int i;
6     for(i = 10; i >= 1; i -= 3){
7         printf("%d\t", i);
8     }
9 }
```

Result

```
$gcc -o main *.c
```

```
$main
```

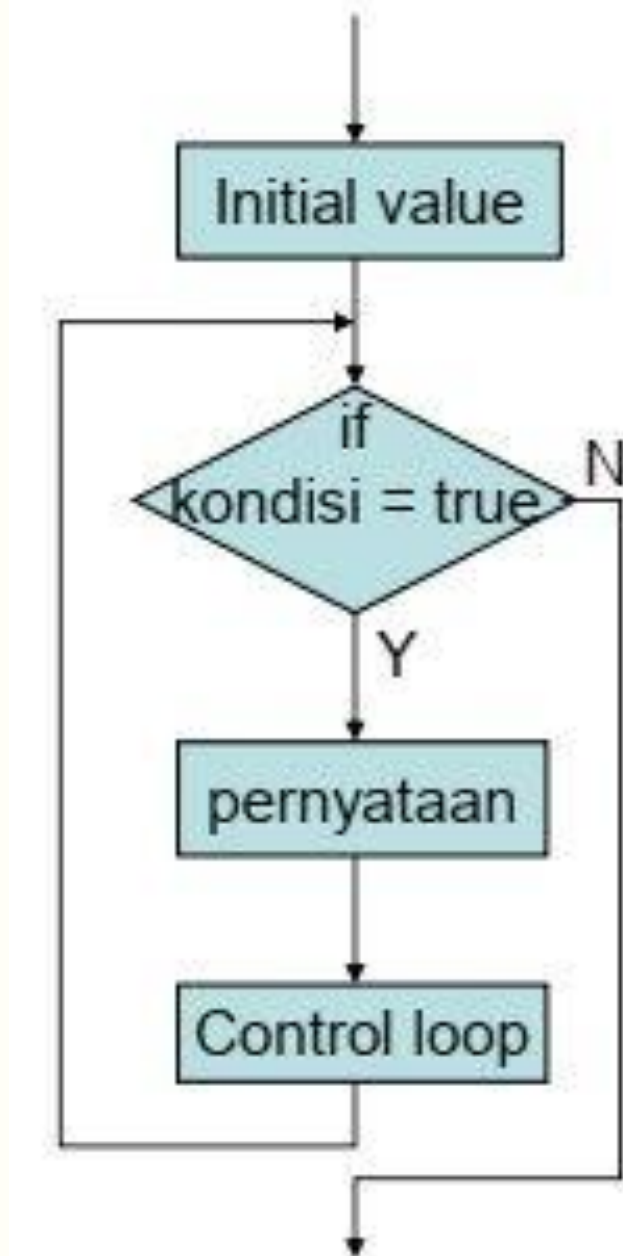
```
10      7      4      1
```

Output iterasi ke-4

• Iterasi ke-4 $i = 1$

$i \geq 1 \rightarrow 1 \geq 1 \rightarrow \text{TRUE}$

Baris ke-7 dijalankan, sehingga muncul output ke layar "1 " $i -= 3 \rightarrow i = i - 3 \rightarrow i = 1 - 3 = -2$



Contoh 2

```
1 #include <stdio.h>
2
3 int main()
4 {
5     int i;
6     for(i = 10; i >= 1; i -= 3){
7         printf("%d\t", i);
8     }
9 }
```

Result

```
$gcc -o main *.c
```

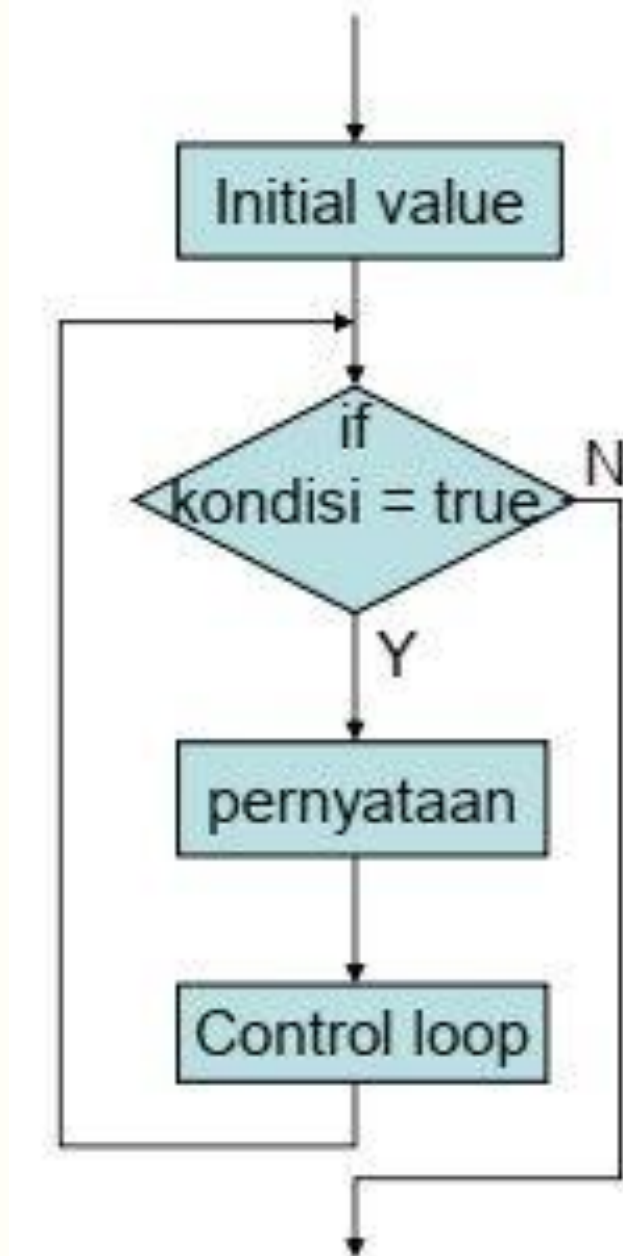
```
$main
```

```
10    7    4    1
```

• Iterasi ke-5 $i = -2$

$i \geq 1 \rightarrow -2 \geq 1 \rightarrow \text{FALSE}$

continue condition tidak terpenuhi, maka keluar dari *loop*.





**Statement
while**



Statement `while`

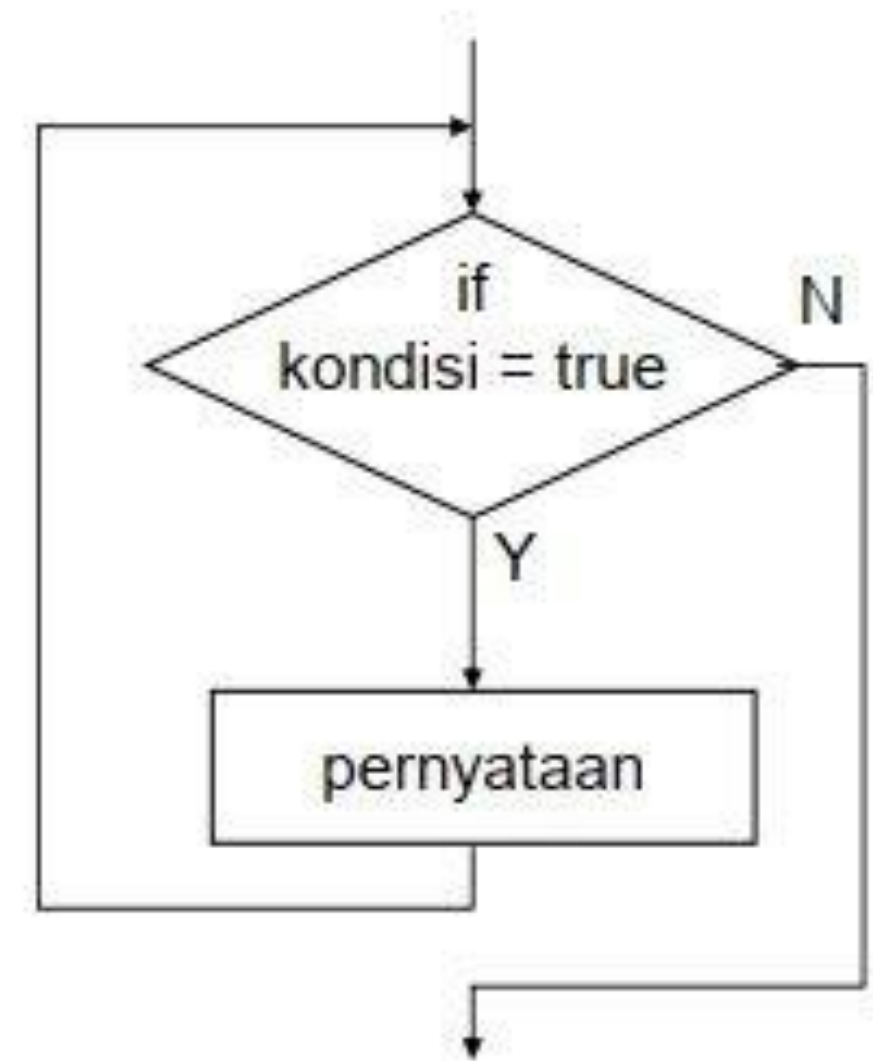
- Bentuk umum:

```
while(continue_condition) {  
    //body_of_loop  
    statement;  
    statement;  
    ...  
    statement;  
}
```

- `continue_condition`: kondisi yang harus terpenuhi agar *loop* tetap berjalan.
- Apabila hanya ada 1 *statement* yang berada di dalam *body of loop*, maka tidak perlu menggunakan kurung kurawal.
- Agar *loop* dapat berhenti, maka di dalam *body of loop* perlu ada *statement* yang bisa merubah nilai kondisi sehingga *looping* dapat berhenti.

Contoh 1

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int i = 10;
6      while(i>=1){
7          printf("%d\t", i);
8          i -= 3;
9      }
10 }
```



- *Continue condition*: $i \geq 1$
- Variabel yang dicek pada *continue condition* perlu diberi nilai awal, yakni i : __
- *Statement* yang merubah nilai kondisi: $i -= 3$

Contoh 1

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int i = 10;
6      while(i>=1){
7          printf("%d\t", i);
8          i -= 3;
9      }
10 }
```

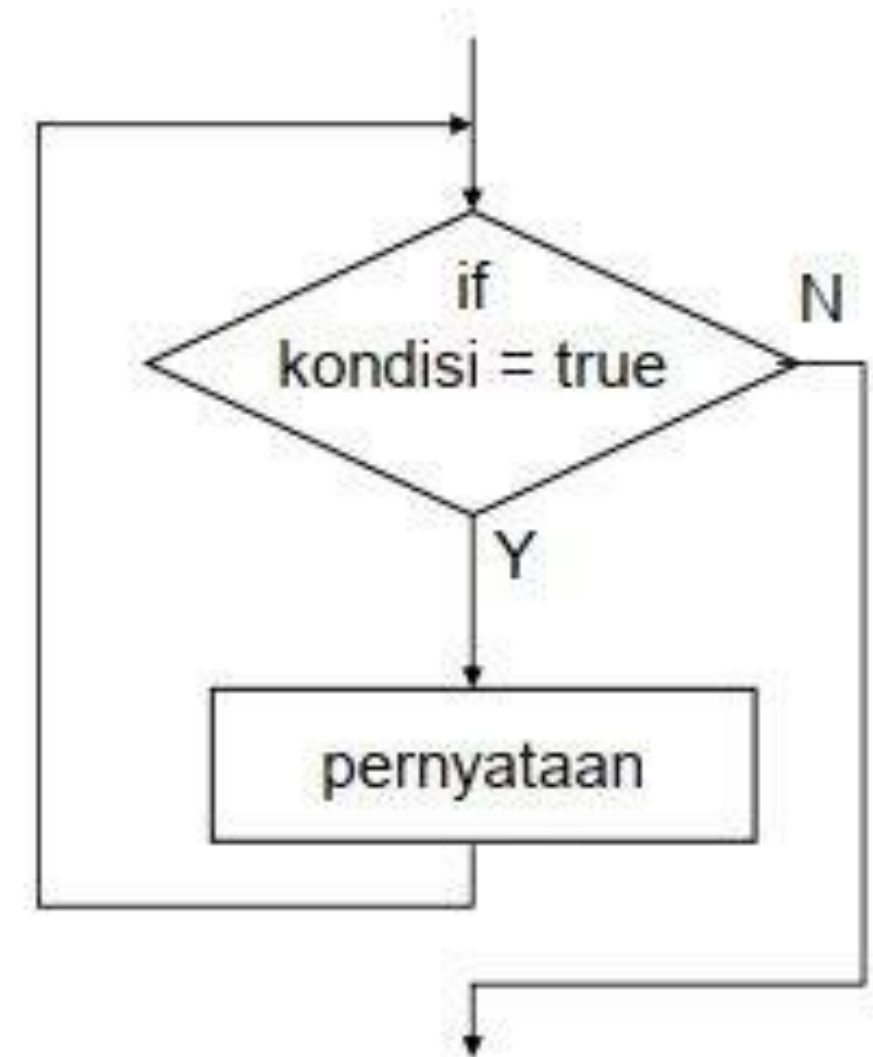
Result

```
$gcc -o main *.c
```

```
$main
```

```
10
```

Output iterasi ke-1



- Iterasi ke-1

continue condition → $i \geq 1$ → $10 \geq 1$ → TRUE

baris ke-7 dijalankan, sehingga muncul output “10”

baris ke-8 dijalankan, sehingga nilai i berubah menjadi $10 - 3 = 7$

Contoh 1

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int i = 10;
6      while(i>=1){
7          printf("%d\t", i);
8          i -= 3;
9      }
10 }
```

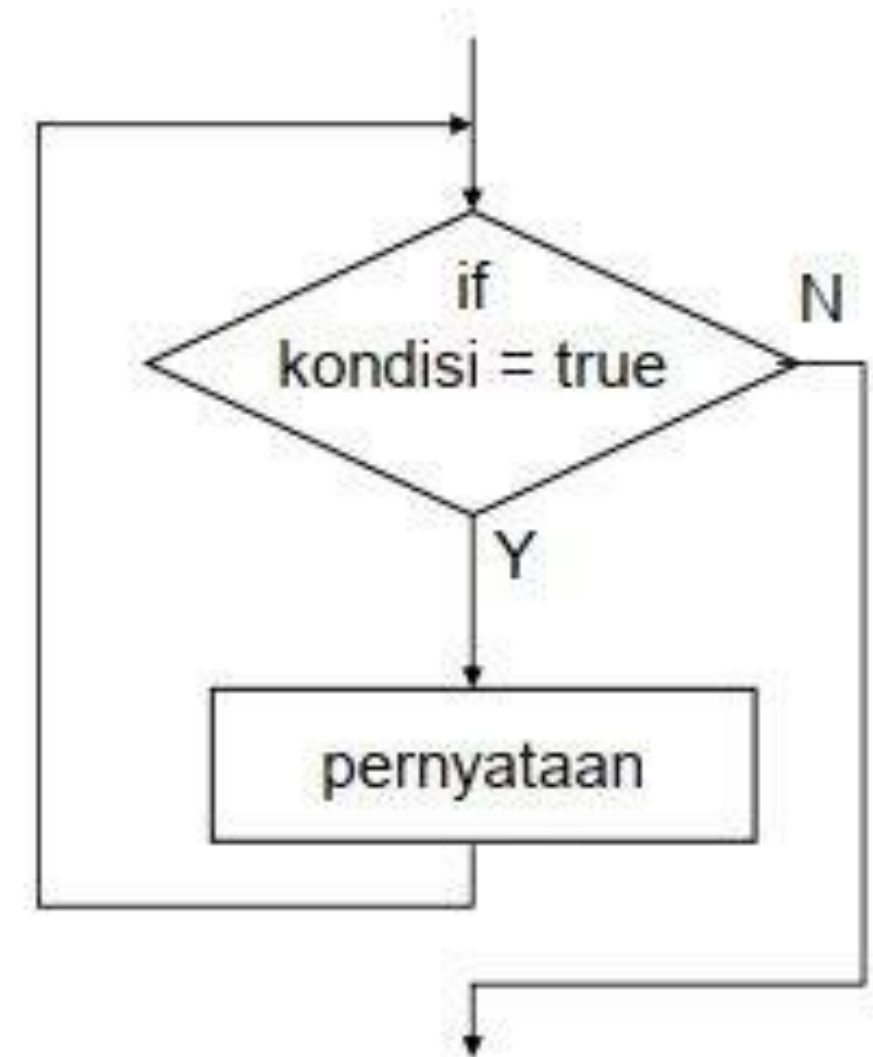
Result

```
$gcc -o main *.c
```

```
$main
```

```
10      7
```

Output iterasi ke-2



- Iterasi ke-2

continue condition → $i \geq 1 \rightarrow 7 \geq 1 \rightarrow \text{TRUE}$

baris ke-7 dijalankan, sehingga muncul output “7”

baris ke-8 dijalankan, sehingga nilai i berubah menjadi $7-3 = 4$

Contoh 1

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int i = 10;
6      while(i>=1){
7          printf("%d\t", i);
8          i -= 3;
9      }
10 }
```

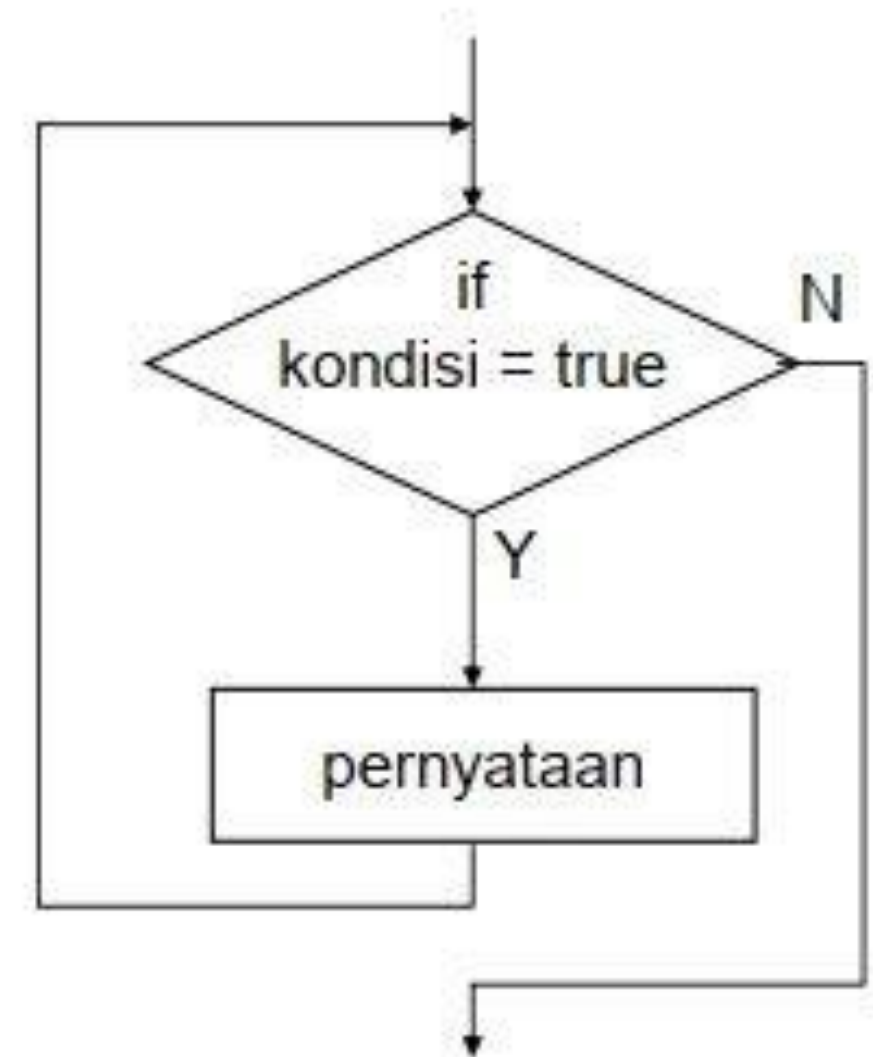
Result

```
$gcc -o main *.c
```

```
$main
```

```
10      7      4
```

Output iterasi ke-3



- Iterasi ke-3

continue condition → $i \geq 1$ → $4 \geq 1$ → TRUE

baris ke-7 dijalankan, sehingga muncul output “4”

baris ke-8 dijalankan, sehingga nilai i berubah menjadi $4 - 3 = 1$

Contoh 1

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int i = 10;
6      while(i>=1){
7          printf("%d\t", i);
8          i -= 3;
9      }
10 }
```

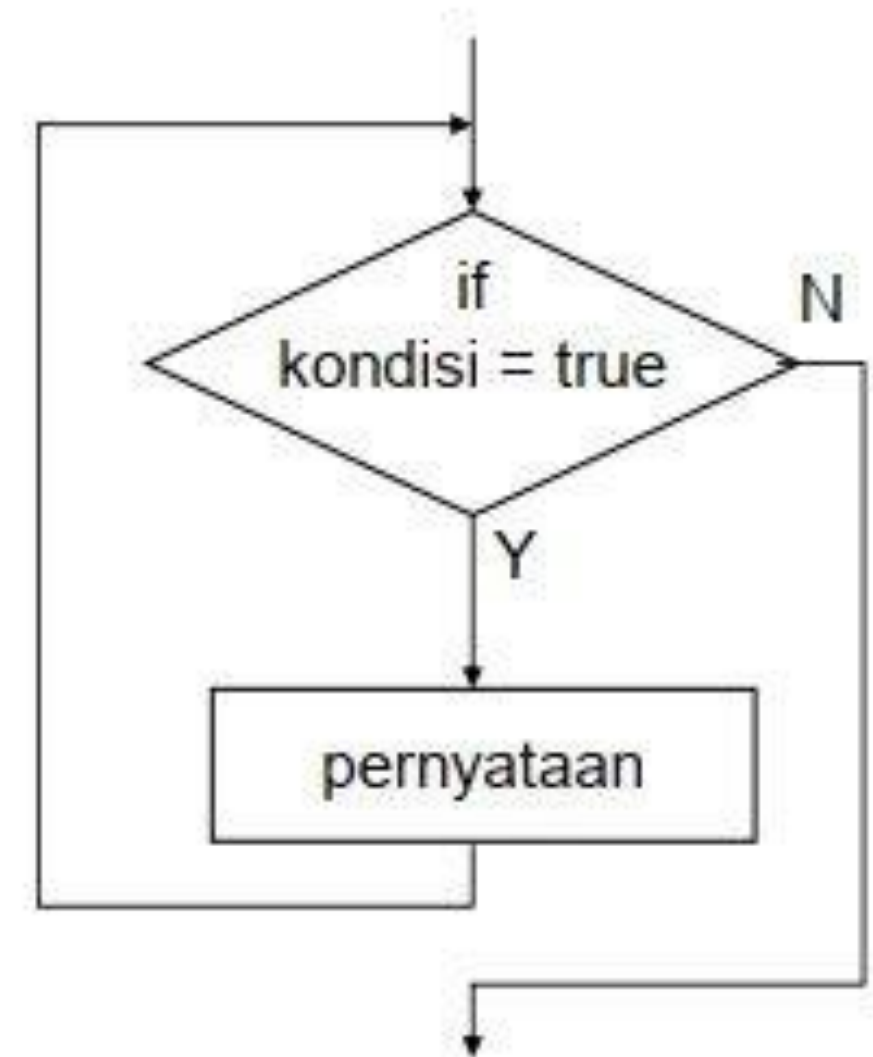
Result

```
$gcc -o main *.c
```

```
$main
```

```
10      7      4      1
```

Output iterasi ke-4



- Iterasi ke-4

continue condition → $i \geq 1$ → $1 \geq 1$ → TRUE

baris ke-7 dijalankan, sehingga muncul output “1”

baris ke-8 dijalankan, sehingga nilai i berubah menjadi $1-3 = -2$

Contoh 1

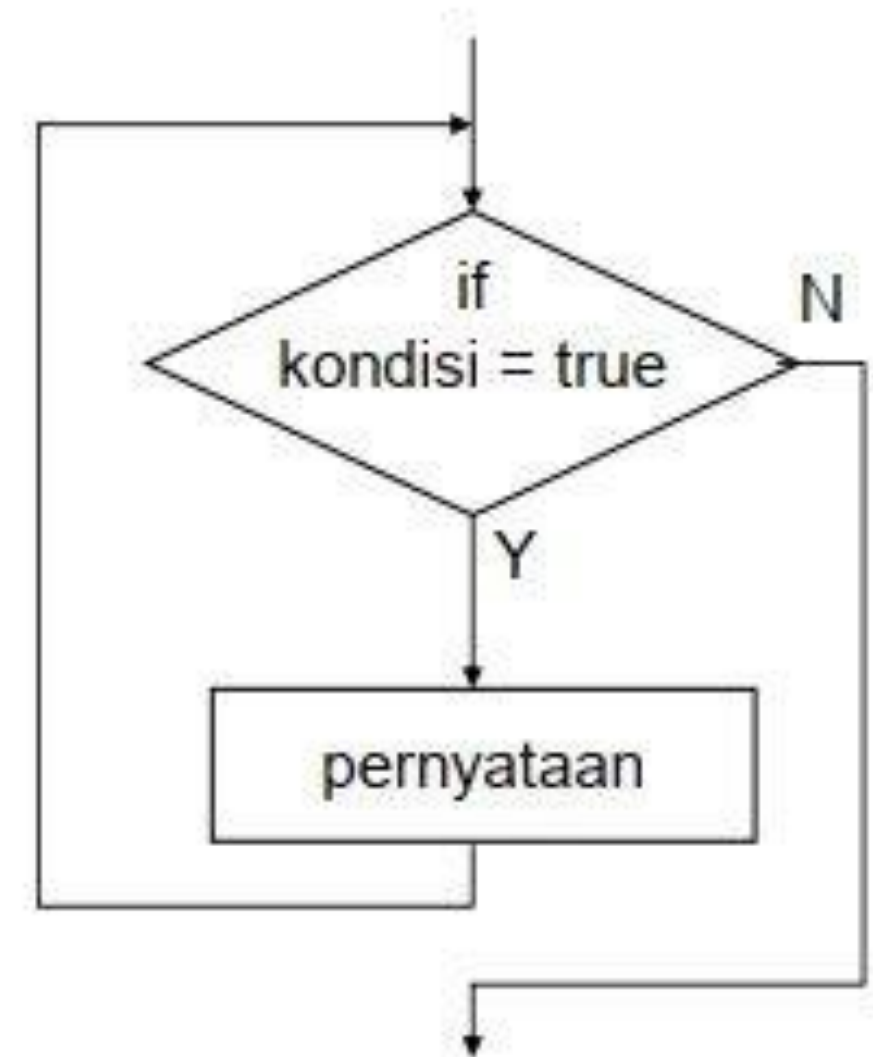
```
1  #include <stdio.h>
2
3  int main()
4  {
5      int i = 10;
6      while(i>=1){
7          printf("%d\t", i);
8          i -= 3;
9      }
10 }
```

Result

```
$gcc -o main *.c
```

```
$main
```

```
10      7      4      1
```



- Iterasi ke-5

continue condition → $i \geq 1 \rightarrow -2 \geq 1 \rightarrow \text{FALSE}$

keluar dari *looping*, sehingga baris ke-7 dan ke-8 tidak dijalankan

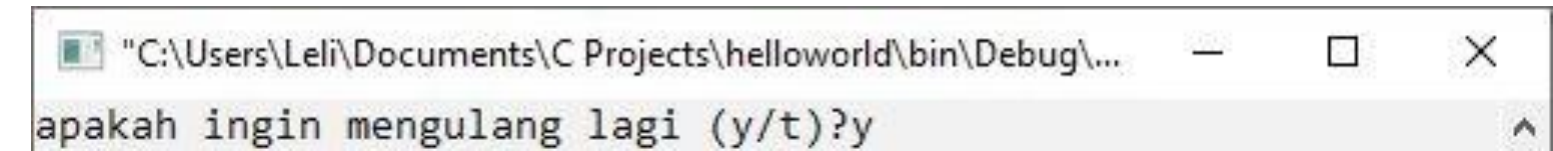
Contoh 2

```
1  #include <stdio.h>
2
3  int main()
4  {
5      char jawab = 'y';
6      while(jawab == 'y'){
7          printf("apakah ingin mengulang lagi (y/t)?");
8          scanf("%c", &jawab);
9          fflush(stdin);
10     }
11 }
```

- *Continue condition*: `jawab == 'y'`
- Variabel yang dicek pada *continue condition* perlu diberi nilai awal, yakni `jawab = 'y'`
- *Statement* yang merubah nilai kondisi:
input nilai untuk variabel `jawab` → `scanf("%c", &jawab);`

Contoh 2

```
1  #include <stdio.h>
2
3  int main()
4  {
5      char jawab = 'y';
6      while(jawab == 'y'){
7          printf("apakah ingin mengulang lagi (y/t)?");
8          scanf("%c", &jawab);
9          fflush(stdin);
10     }
11 }
```



Output iterasi ke-1

- Iterasi ke-1

jawab = 'y'

jawab == 'y' → 'y' == 'y' → TRUE

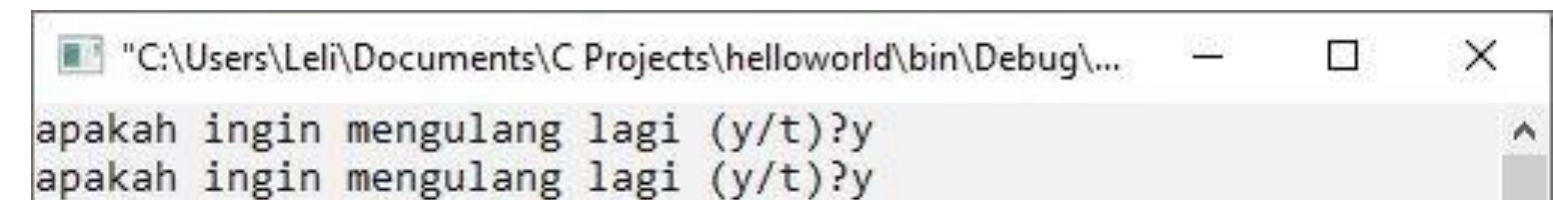
Baris ke-7 dijalankan, sehingga muncul output ke layar “apakah ingin mengulang lagi(y/t)?”

Baris ke-8 dijalankan, sehingga user bisa memasukkan karakter, yakni 'y'

Baris ke-9 dijalankan, sehingga input dari user dihapus dari input *buffer*

Contoh 2

```
1  #include <stdio.h>
2
3  int main()
4  {
5      char jawab = 'y';
6      while(jawab == 'y'){
7          printf("apakah ingin mengulang lagi (y/t)?");
8          scanf("%c", &jawab);
9          fflush(stdin);
10     }
11 }
```



"C:\Users\Leli\Documents\C Projects\helloworld\bin\Debug\...
apakah ingin mengulang lagi (y/t)?y
apakah ingin mengulang lagi (y/t)?y

Output iterasi ke-2

- Iterasi ke-2

jawab = 'y'

jawab == 'y' → 'y' == 'y' → TRUE

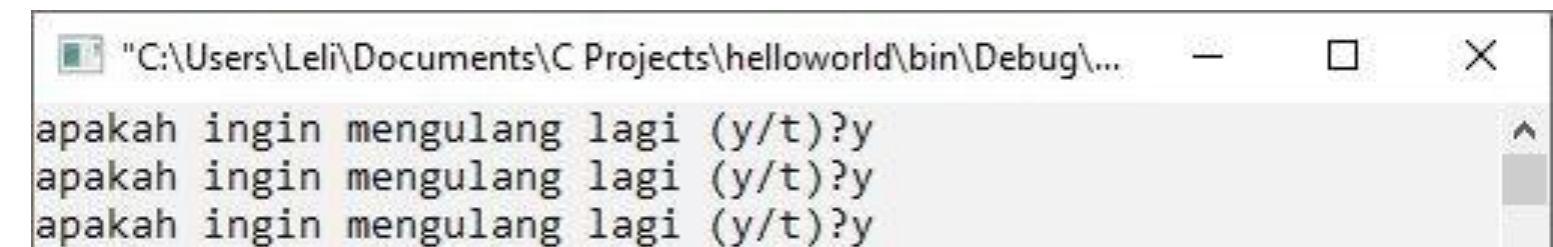
Baris ke-7 dijalankan, sehingga muncul output ke layar "apakah ingin mengulang lagi(y/t)?"

Baris ke-8 dijalankan, sehingga user bisa memasukkan karakter, yakni 'y'

Baris ke-9 dijalankan, sehingga input dari user dihapus dari input *buffer*

Contoh 2

```
1  #include <stdio.h>
2
3  int main()
4  {
5      char jawab = 'y';
6      while(jawab == 'y'){
7          printf("apakah ingin mengulang lagi (y/t)?");
8          scanf("%c", &jawab);
9          fflush(stdin);
10     }
11 }
```



```
"C:\Users\Leli\Documents\C Projects\helloworld\bin\Debug\..."
apakah ingin mengulang lagi (y/t)?y
apakah ingin mengulang lagi (y/t)?y
apakah ingin mengulang lagi (y/t)?y
```

Output iterasi ke-3

- Iterasi ke-3

jawab = 'y'

jawab == 'y' → 'y' == 'y' → TRUE

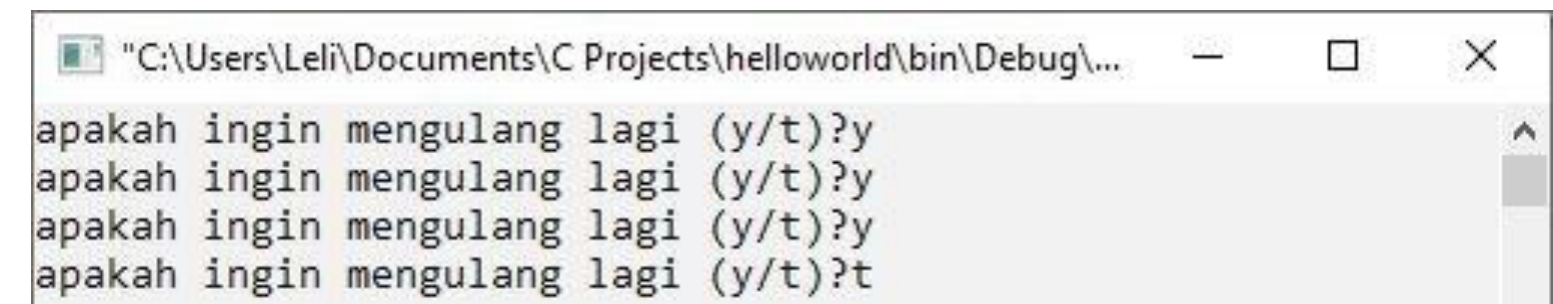
Baris ke-7 dijalankan, sehingga muncul output ke layar "apakah ingin mengulang lagi(y/t)?"

Baris ke-8 dijalankan, sehingga user bisa memasukkan karakter, yakni 'y'

Baris ke-9 dijalankan, sehingga input dari user dihapus dari input *buffer*

Contoh 2

```
1  #include <stdio.h>
2
3  int main()
4  {
5      char jawab = 'y';
6      while(jawab == 'y'){
7          printf("apakah ingin mengulang lagi (y/t)?");
8          scanf("%c", &jawab);
9          fflush(stdin);
10     }
11 }
```



```
"C:\Users\Leli\Documents\C Projects\helloworld\bin\Debug\...
apakah ingin mengulang lagi (y/t)?y
apakah ingin mengulang lagi (y/t)?y
apakah ingin mengulang lagi (y/t)?y
apakah ingin mengulang lagi (y/t)?t
```

Output iterasi ke-4

- Iterasi ke-4

jawab = 'y'

jawab == 'y' → 'y' == 'y' → TRUE

Baris ke-7 dijalankan, sehingga muncul output ke layar "apakah ingin mengulang lagi(y/t)?"

Baris ke-8 dijalankan, sehingga user bisa memasukkan karakter, yakni 't'

Baris ke-9 dijalankan, sehingga input dari user dihapus dari input *buffer*

Contoh 2

```
1  #include <stdio.h>
2
3  int main()
4  {
5      char jawab = 'y';
6      while(jawab == 'y'){
7          printf("apakah ingin mengulang lagi (y/t)?");
8          scanf("%c", &jawab);
9          fflush(stdin);
10     }
11 }
```



```
"C:\Users\Leli\Documents\C Projects\helloworld\bin\Debug\..."
apakah ingin mengulang lagi (y/t)?y
apakah ingin mengulang lagi (y/t)?y
apakah ingin mengulang lagi (y/t)?y
apakah ingin mengulang lagi (y/t)?t

Process returned 0 (0x0)   execution time : 5.808 s
Press any key to continue.
```

- Iterasi ke-5
jawab = 't'
jawab == 'y' → 't' == 'y' → FALSE
Karena *continue condition* tidak terpenuhi, maka *looping* berhenti. Sehingga baris ke-7 s.d ke-9 tidak lagi dijalankan.

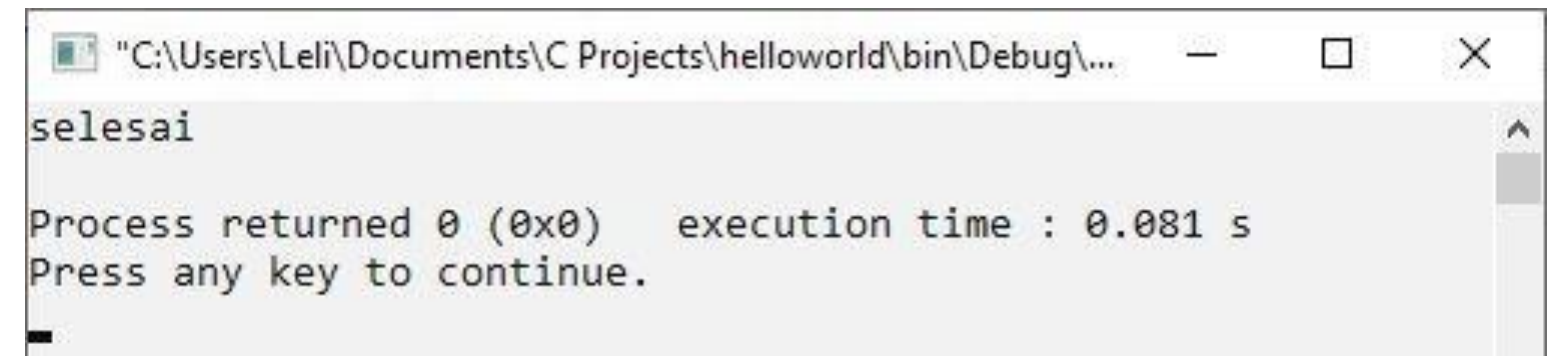
Contoh 3

```
1  #include <stdio.h>
2
3  int main()
4  {
5      char jawab = 't';
6      while(jawab == 'y'){
7          printf("apakah ingin mengulang lagi (y/t)?");
8          scanf("%c", &jawab);
9          fflush(stdin);
10     }
11     printf("selesai\n");
12 }
```

- *Continue condition*: `jawab == 'y'`
- Variabel yang dicek pada *continue condition* perlu diberi nilai awal, yakni `jawab = 't'`
- *Statement* yang merubah nilai kondisi:
input nilai untuk variabel `jawab` → `scanf ("%c", &jawab) ;`

Contoh 3

```
1 #include <stdio.h>
2
3 int main()
4 {
5     char jawab = 't';
6     while(jawab == 'y'){
7         printf("apakah ingin mengulang lagi (y/t)?");
8         scanf("%c", &jawab);
9         fflush(stdin);
10    }
11    printf("selesai\n");
12 }
```



```
"C:\Users\Leli\Documents\C Projects\helloworld\bin\Debug\..."
selesai
Process returned 0 (0x0)   execution time : 0.081 s
Press any key to continue.
```

- Iterasi ke-1
jawab = 't'
jawab == 'y' → 't' == 'y' → FALSE
Karena *continue condition* tidak terpenuhi, maka *looping* berhenti. Sehingga baris ke-7 s.d ke-9 tidak dijalankan.



Statement do-while



Statement

do-while

- Bentuk umum:

```
do{  
  //body_of_loop  
  statement;  
  statement;  
  ...  
  statement;  
} while(continue_condition);
```

- `continue_condition`: kondisi yang harus terpenuhi agar *loop* tetap berjalan.
- Apabila hanya ada 1 *statement* yang berada di dalam *body of loop*, maka tidak perlu menggunakan kurung kurawal.
- Agar *loop* dapat berhenti, maka di dalam *body of loop* perlu ada *statement* yang bisa merubah nilai kondisi sehingga *loop* dapat berhenti.

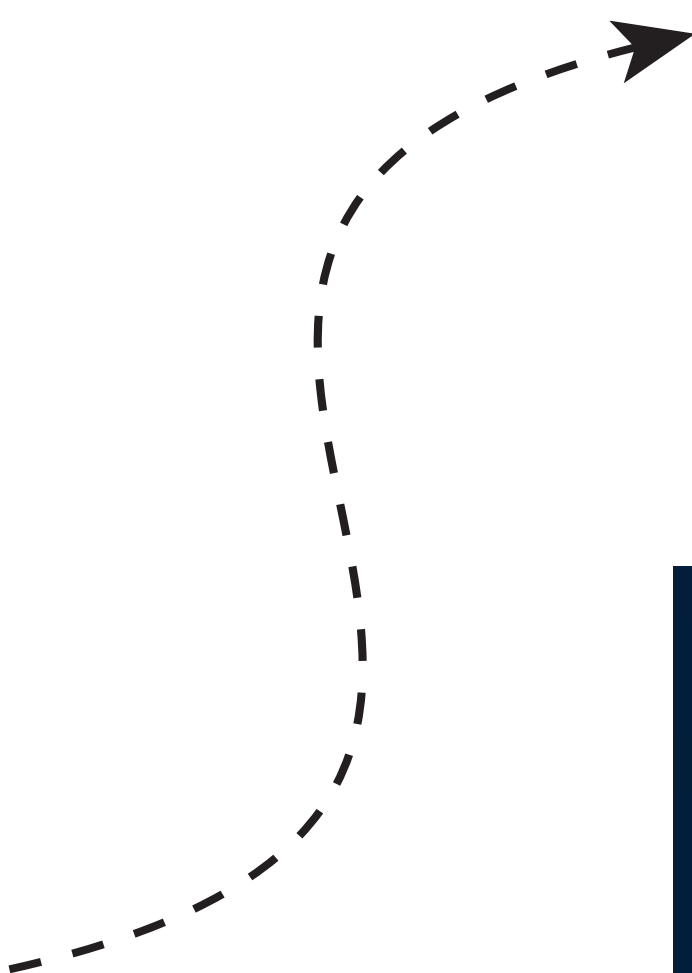


Statement do-while

- Bentuk umum:

```
do{  
    //body_of_loop statement;  
    statement;  
    ...  
    statement;  
} while(continue_condition);
```

- `continue_condition`: kondisi yang harus terpenuhi agar *loop* tetap berjalan.
- Apabila hanya ada 1 *statement* yang berada di dalam *body of loop*, maka tidak perlu menggunakan kurung kurawal.
- Agar *loop* dapat berhenti, maka di dalam *body of loop* perlu ada *statement* yang bisa merubah nilai kondisi sehingga *loop* dapat berhenti.



Contoh



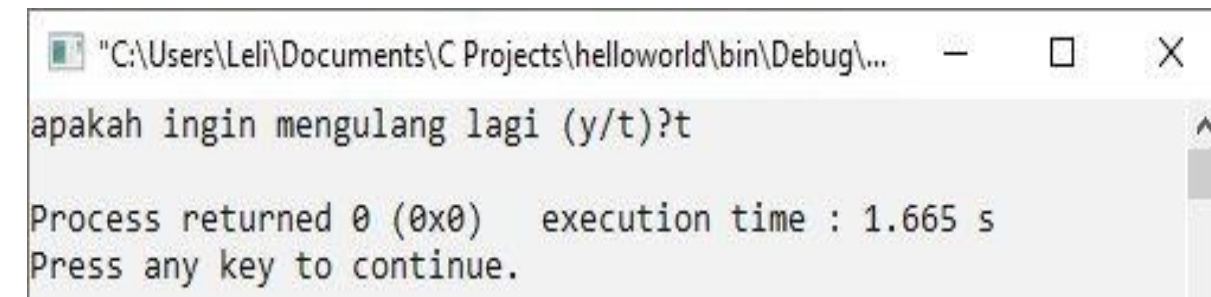
```
1 #include <stdio.h>
2
3 int main()
4 {
5     char jawab;
6     do{
7         printf("apakah ingin mengulang lagi (y/t)?");
8         scanf("%c", &jawab);
9         fflush(stdin);
10    }while(jawab == 'y');
11 }
```

- *Continue condition:* jawab == 'y'
- *Statement yang merubah nilai kondisi:*

input nilai untuk variabel jawab ➔ `scanf ("%c", &jawab);`

Contoh

```
1 #include <stdio.h>
2
3 int main()
4 {
5     char jawab;
6     do{
7         printf("apakah ingin mengulang lagi (y/t)?");
8         scanf("%c", &jawab);
9         fflush(stdin);
10    }while(jawab == 'y');
11 }
```



"C:\Users\Leli\Documents\C Projects\helloworld\bin\Debug\..."
apakah ingin mengulang lagi (y/t)?t
Process returned 0 (0x0) execution time : 1.665 s
Press any key to continue.

Output iterasi ke-1

- Iterasi ke-1

Baris ke-7 dijalankan, sehingga muncul output ke layar “apakah ingin mengulang lagi(y/t)?” Baris ke-8 dijalankan, sehingga user bisa memasukkan karakter, yakni ‘t’

Baris ke-9 dijalankan, sehingga input dari user dihapus dari input *buffer*

Cek *continue condition* → jawab == ‘y’ → ‘t’ == ‘y’ → FALSE Keluar dari *looping*

Latihan

1.



Jelaskan hasil outputnya

```
#include <stdio.h>
```

```
int main() {  
    int i;  
    for (i = 1; i <= 10; i++) {  
        printf("7 x %d = %d\n", i, 7 * i);  
    }  
    return 0;  
}
```

Latihan

1.



Jelaskan hasil outputnya

```
#include <stdio.h>

int main() {
    int i;
    for (i = 1; i <= 10; i++) {
        printf("7 x %d = %d\n", i, 7 * i);
    }
    return 0;
}
```

```
C:\DATAKU\Materi Kuliah\202 x + v
7 x 1 = 7
7 x 2 = 14
7 x 3 = 21
7 x 4 = 28
7 x 5 = 35
7 x 6 = 42
7 x 7 = 49
7 x 8 = 56
7 x 9 = 63
7 x 10 = 70
-----
Process exited after 0.02932 seconds with return value 0
Press any key to continue . . . |
```


Latihan

2.



Buat programnya:

Seorang penjaga parkir ingin menghitung **jumlah mobil yang masuk** ke area parkir. Ia mencatat setiap mobil yang masuk, dan proses pencatatan **akan berhenti jika jumlah mobil sudah mencapai kapasitas maksimal parkir, yaitu 5 mobil.**

Buatlah program dalam bahasa C menggunakan **perulangan while** untuk menghitung jumlah mobil yang masuk, dan tampilkan pesan setiap kali ada mobil masuk.

```
Mobil ke-1 masuk.  
Mobil ke-2 masuk.  
Mobil ke-3 masuk.  
Mobil ke-4 masuk.  
Mobil ke-5 masuk.  
Parkir penuh! Jumlah mobil = 5
```

Latihan

2.



Buat programnya:

Seorang penjaga parkir ingin menghitung **jumlah mobil yang masuk** ke area parkir. Ia mencatat setiap mobil yang masuk, dan proses pencatatan **akan berhenti jika jumlah mobil sudah mencapai kapasitas maksimal parkir, yaitu 5 mobil.**

Buatlah program dalam bahasa C menggunakan **perulangan while** untuk menghitung jumlah mobil yang masuk, dan tampilkan pesan setiap kali ada mobil masuk.

```
Mobil ke-1 masuk.  
Mobil ke-2 masuk.  
Mobil ke-3 masuk.  
Mobil ke-4 masuk.  
Mobil ke-5 masuk.  
Parkir penuh! Jumlah mobil = 5
```

```
#include <stdio.h>
```

```
int main() {  
    int mobil = 1;  
    while (mobil <= 5) {  
        printf("Mobil ke-%d masuk.\n", mobil);  
        mobil++;  
    }  
    printf("Parkir penuh! Jumlah mobil = 5\n");  
    return 0;  
}
```


Latihan

3.



Jelaskan hasil outputnya:

```
#include <stdio.h>

int main() {
    int harga, total = 0;
    char lanjut;

    do {
        printf("Masukkan harga barang: ");
        scanf("%d", &harga);
        total += harga;

        printf("Tambah barang lagi? (y/n): ");
        scanf(" %c", &lanjut);
    } while (lanjut == 'y' || lanjut == 'Y');

    printf("Total harga belanja: %d\n", total);
    return 0;
}
```

Latihan

3.



Jelaskan hasil outputnya:

```
#include <stdio.h>

int main() {
    int harga, total = 0;
    char lanjut;

    do {
        printf("Masukkan harga barang: ");
        scanf("%d", &harga);
        total += harga;

        printf("Tambah barang lagi? (y/n): ");
        scanf(" %c", &lanjut);
    } while (lanjut == 'y' || lanjut == 'Y');

    printf("Total harga belanja: %d\n", total);
    return 0;
}
```

Masukkan harga barang: 15000
Tambah barang lagi? (y/n): y
Masukkan harga barang: 20000
Tambah barang lagi? (y/n): y
Masukkan harga barang: 30000
Tambah barang lagi? (y/n): n
Total harga belanja: 65000

THANK
YOU!

