Lec 5

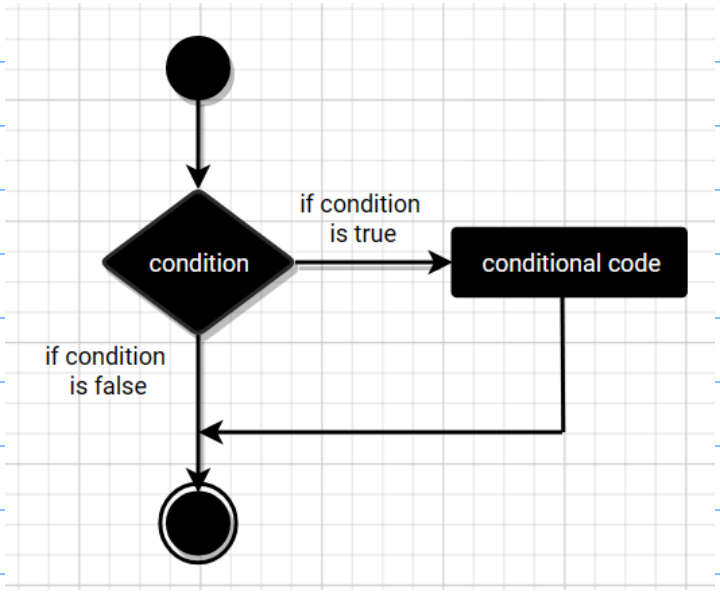
Decision Making

- if statement
- go to statement
- if else statement
- switch statement
- if elseif else stat

1 if statement :-

ممكن شرط واحد
او عده شروط

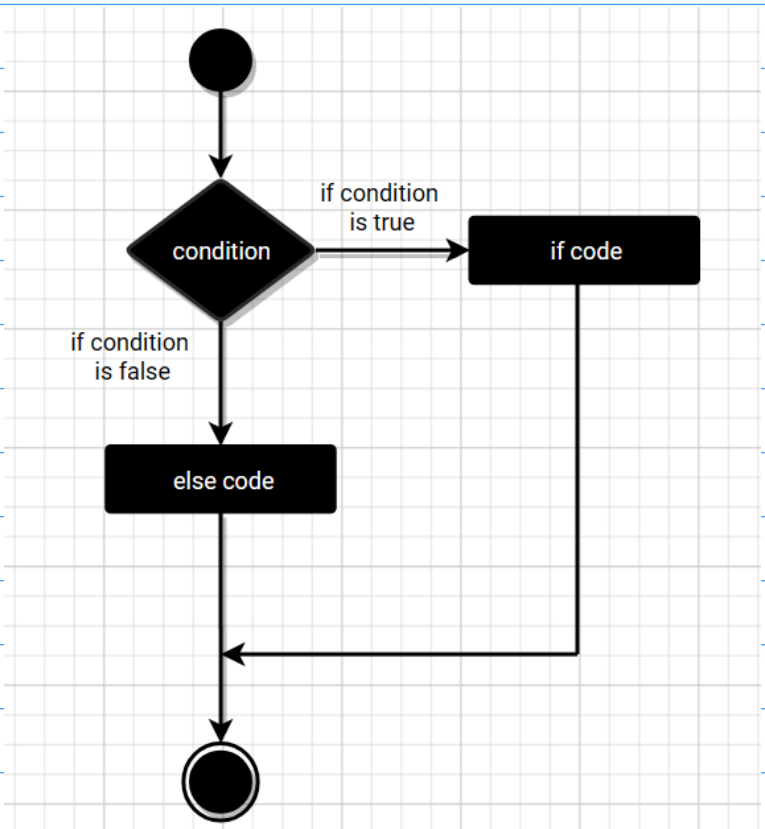
```
if(conditions)
{
/*code*/
}
```



- الـ if statement طريقة عملها كالاتى
  - لو الـ condition اتحقق يعنى بقا True هينفذ العمليات الى البلوك الى بعده { }
  - لو الـ ~ متحققش يعنى بقا False مش هيدخل على { } وهينزل ينفذ باقى الكود ايا كان اى

2 if else statement :

```
if(conditions)
{
/*code*/
}
else
{
    /*code*/
}
```



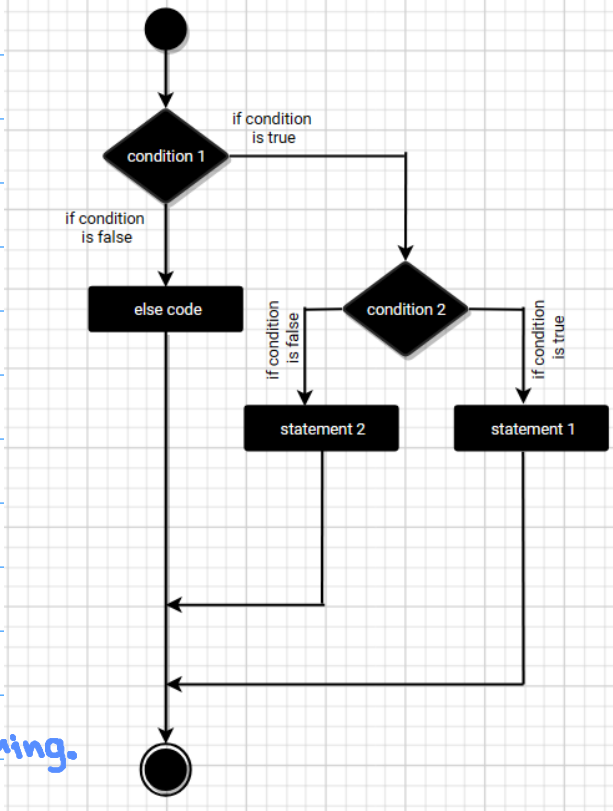
← لو الـ conditions متحققتش هينزل ينفذ فى الـ else على طول

← لو الـ conditions اتحقق.. هينفذ الـ {} تحت الـ if وهيهمل باقى الكود (else).

بستخدم دى علشان أوفر وقت فى الـ compile بدل ما أسأل بـ if كل شرط

*Nested if.....else statement

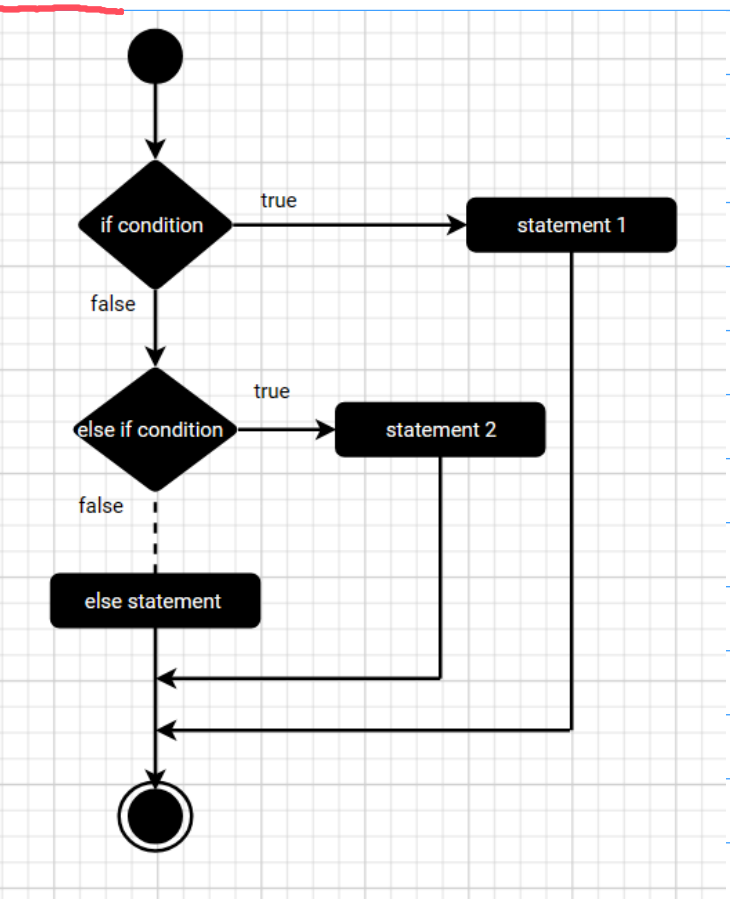
```
if(conditions)
{/*code*/
    if(conditions)
    {
        /*code*/
    }
    else
    {
        /*code*/
    }
/*code*/
}
else
{
    /*code*/
}
```



play an important role in C programming.
it means you can use conditional statements inside another conditional statement

3 if...elseif... else statements :

```
if(conditions)
{
/*code*/
}
else if(conditions)
{
    /*code*/
}
/*any number of else if*/
else
{
    /*code*/
}
```



- بشوف شرط الـ if لو اتحقق
  هينفذ الـ block اللى بعده و هيهمل باقى الكود اللى هو elseif & else
- لو متحققش شرط الـ if هينزل يشوف الشرط اللى جمبه ثم اللى بعده لحد ما شرط يتحقق و ينفذ الـ block بتاعه
- لو مفيش اى شرط اتحقق هينزل ينفذ اللى فى الـ else

4 go to statement :

متستخدمهاش

label

goto

```
label :
/*code*/
goto label;
/*code*/
```

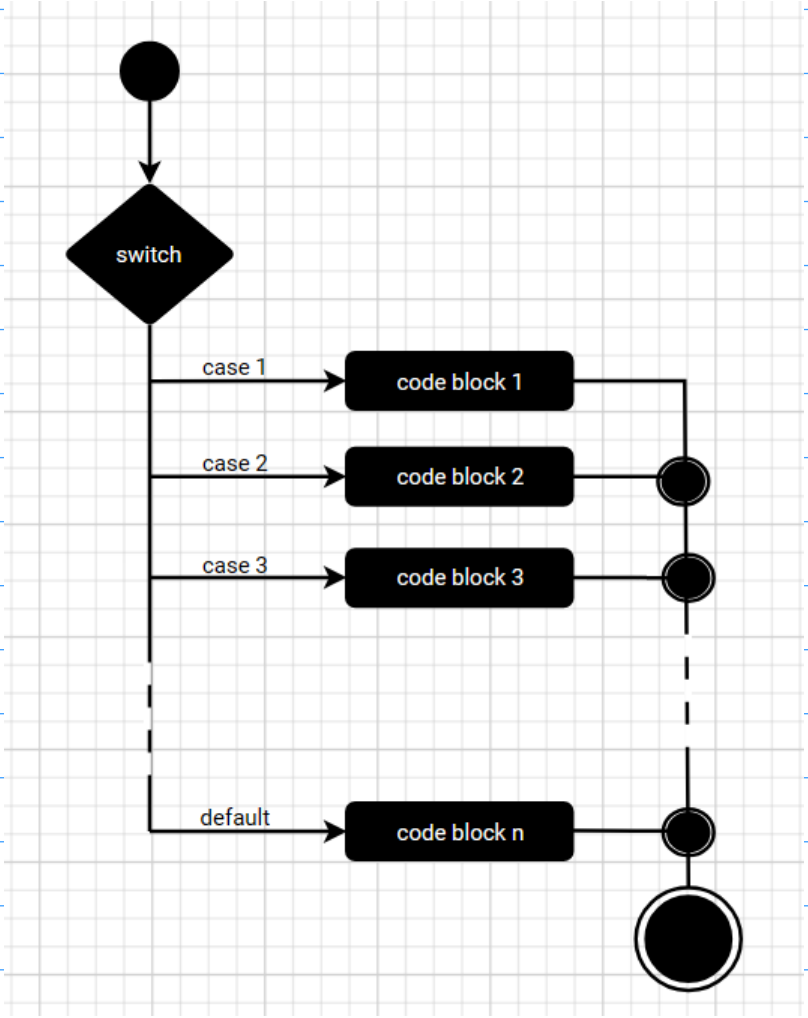
- الـ label ممكن يكون اى اسم (بس مش من الـ Keywords)
- لا يفضل استخدمها (البروسيسور بيبطئ من مكان لمكان) (صعبة فى الـ Debugging)
- عملها كالاتى : بينفذ البرنامج عادى اول مايلاقى goto هيروح للمكان اللى عند الـ label اللى مكتوب جمبها
  بستخدمها عشان اتنقل فى الكود

5 switch statement:

```
switch (variable)
{
    case 1:
      /*code*/
      break;

    case n:
    /*code*/
      break;

    default:
    /*code*/
      break;
}
```



بستخدم الـ switch لما يكون عندى variable ليه أكتر من احتمال

كيف تعمل ؟

- الـ switch بتقارن الـ variable مع الـ cases
لحد ما تلاقى case شبه ← فبتنفذ الـ statement اللى عند الـ case دى
و تخرج بالـ break برا الـ switch خالص

- لو ملقاش اى case شبه هيروح ينفذ الـ statement اللى فى الـ default
و يخرج بالـ break برا الـ switch

← فى حالة عدم وجود الـ break هيفضل ينفذ اى حاجه هتقابله لحد ما يلاقى break فيخرج
او يكون الكود فى آخر خالص

← الـ default ممكن تكون فى اى مكان داخل الـ switch مش شرط فى الآخر (عشان كدا يفضل تحط الـ default مع break)

الـ switch اسرع فى التنفيذ من الـ else if

* لازم الـ variable اللى هعمل عليه switch يكون قيمة صحيحة integer لازم
مينفعش يكون float لازم يكون من احدى الـ integer datatype
int - char - long - short - ...

* Nested switch

```
switch(variable)
{
    case 1:
        switch( another variable)
        {
            case 11 :
            /*code*/
            break;
            case n:
            /*code*/
            break;
            default:
            /*code*/
            break;
        }
    break;
    case n :
    /*code*/
    break;
    default:
    /*code*/
    break;
}
```

نادر استخدمها

نفس فكره الـ Neste if

① For loop
② while loop

Looping

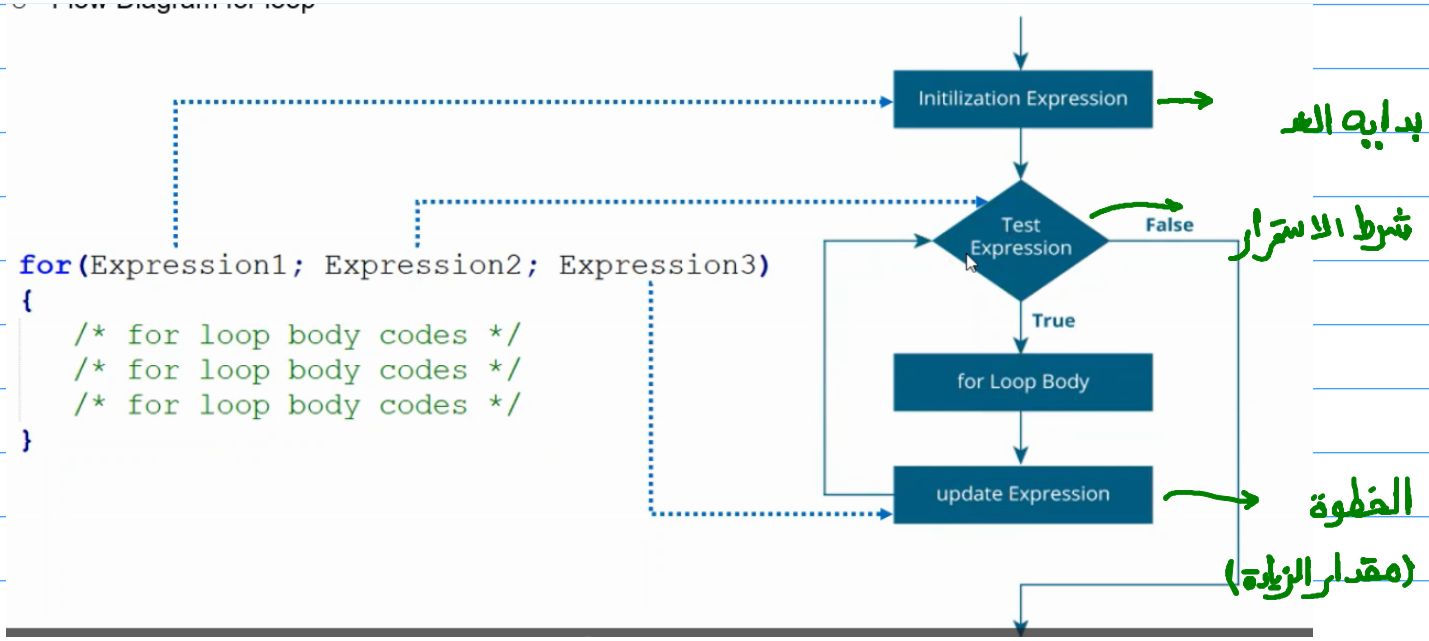
لو عاوز انفذ كود عدد من المرات
(فى اى حاجه Functions)
بستخدم الـ looping

1 For loop

أشهر واحد

executes a statement or block of statements until the given conditions is true

بستخدم الـ For في حاله إني عارف عدد مرات التكرار



ازاى بتشتغل؟

① بيبقا عندى counter بيكون له قيمة ابتدائيه لازم

② جمله الشرط (شرط الدخول لـ body الـ loop) ← شرط التكرار

③ مقدار الزيادة للـ counter الـ incrementer

* الـ counter بيبقه قيمه ابتدائيه و مش بر جعلها تانى
* الشرط لو بـ true بدخل انفذ ولما بخلص بروح للـ increment
  ↳ بيكون بالزياده او النقص (حسب ما المطلوب)
* بعد الـ increment بعمل check على الشرط
  • لو true بكمل وانفذ
  • لو Flase بطلع برا الـ loop
* لو الشرط من البدايه خالص بـ Flase مش بدخل الـ For أصلا"

```
for(initilization ; Test Expression ; counter update)
{
    /*code*/
}
```

```
for(initilization ; Test Expression ; )
{
    /*code*/
    counter update;
}
```

ال increment ممكن اغير مكانه ويبقا جوا ال body

```
for( ;  ; )
{
    /*code*/
}
```

ممكن اعمل For من ال infinite loop

← يعني تتكرر عدد لانهائي من المرات

لو مفيش { } هينفذ أول جمله بعد ال For stat بس

---

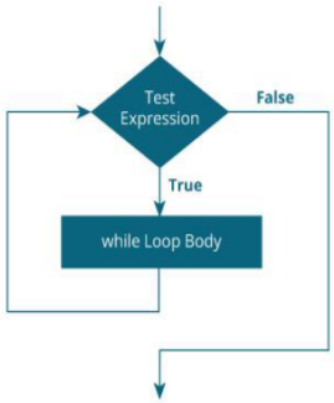
2 while loop

exeutes a statement or block of statement until the given condition is true

بستخدم ال while في حاله اني مش عارف عدد مرات التكرار

← طول ما ال condition بـ true هيفضل ينفذ اللي بين ال { }

```
while()
{
    /*code*/
    Counter++;
}
```



← لو ال condition بـ False

مش هيدخل ينفذ ال body

← بعد كل مره بينفذ فيها ال body

بيرجع يشوف ال condition

بـ true ولا False

* مقدار الزيادة بيكون داخل ال body

Infinit while loop

```
while(1)
{
    /*code*/
}
```

```
while(true)
{
    /*code*/
}
```

هنا بدخل جوا ال loop مش بخرج منها ابدا

إلا لو في مثلا break statement هنشوفها كمان شويه

* اي رقم هيتحط بين ( ) يعتبر true ماعدا الصفر بـ Flase بيطلعني من ال loop

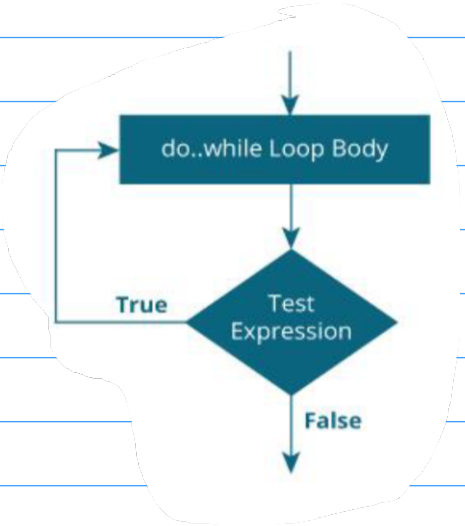
* المشاكل اللي ممكن نقابلها هنا إنك تنسى تحط ال increment يعني ال counter فهيكون دايمًا infinit

3 do ....while( ) loop

- The do... while loop is similar to the while loop with one important difference
  - The body of do .... while loop is executed once, before checking the expression
  - The do... while loop is executed at least once.

```
do
{
   /**
   body 'code'
   */
}
while( );
   ↳ Expression
```

الفرق بينها وبين ال while
إن ال code هينفذ مره واحده
وبعدين هـ check الشرط بـ true هيكمل
ويرجع ينفذ ال body تاني وهكذا
لو الشرط بـ False هيخرج



Conditions and looping

① (break statement) `break;`

ال break بستخدمها مع :

1 switch

2 looping
- For
- while
- do...while

و بستخدمها علشان اخرج برا الـ switch أو برا ال loop
← اهم حاجه بتكون جوا if( ) مش لوحدها

```
while(test expression)
{
    /*code*/
    if(condition to break)
    {
        break;
    }
    /*code*/
}
```

```
do
{
    /*code*/
    if(condition to break)
    {
        break;
    }
    /*code*/
}
while(test expression);
```

```
for(init ; testExpression ; update)
{
    /*code*/
    if(condition to break)
    {
        break;
    }
    /*code*/
}
```

Nested For loop

```
for(init ; testExpression ; update)
{
    for(init ; testExpression ; update)
    {
        /*code*/
        if(condition to break)
        {
            break;
        }
        /*code*/
    }
}
```

لو Neste For فهو هيخرج من ال loop
الى هو فيها بس مش برا خالص

if جوا if
For جوا For
وكدا

Nested if
Nested for
Nested switch
Nested while

Nested اى حاجه جوا حاجه بسميها

Nested For loop

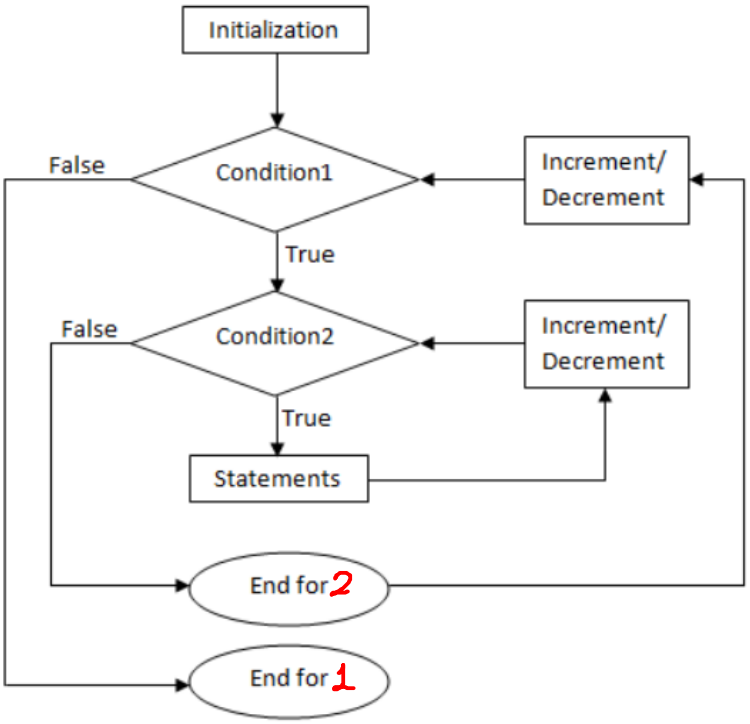


Fig: Flowchart for nested for loop

Conditions and looping

② (continue statement) `continue;`

* تستخدم مع الـ looping
* بتكون جوا if condition

```
for(init ; testExpression ; update)
{
    /*code*/
    if(testExpression)
    {
        continue;
    }
    /*code*/
}
```

```
while(testExpression)
{
    /*code*/
    if(testExpression)
    {
        continue;
    }
    /*code*/
}
```

```
do
{
    if(testExpression)
    {
        continue;
    }
    /*code*/
}
while(testExpression);
```

عكس الـ break
الـ break كانت بتطلعك برا الـ loop خالص

الـ continue بقا بتنقلك لشرط الـ loop
و تبدأ اللوب و بتكمل تنفيذ اى حاجه بعدها

```
int main()
{
    int i;
    for(i=0 ; i<=5 ; i++)
    {
        if(3==i)
        {
            continue;
        }
        printf("i= %i \n",i);

    }
```

```
D:\Embedded\test\test\here.e
i= 0
i= 1
i= 2
i= 4
i= 5

Process returned 0 (0x0)
Press any key to continue
```

بمجرد الـ i=3
دخل الـ if
ونفذ الـ continue
وطلع لشرط الـ if
ومكملش تنفيذ باقى
الكود ومطبعش الـ 3

6/10/23