

الفصل الرابع

٤

جمل التفرع والتكرار في لغة الجافا

الحلقات التكرارية Looping

تستخدم الحلقات Loops لتكرار تنفيذ أمر أو أكثر أو كود عدد معين من المرات فالتكرارات تختبر الشرط فإذا كانت قيمته صحيحة يتم تنفيذ الكود المطلوب ثم إعادة اختبار القيمة مرة أخرى فإذا كانت صحيحة يتم إعادة تنفيذ الكود وهكذا، وفي حالة عدم تحقق الشرط يتم التوقف عن تنفيذ الكود وإكمال تنفيذ كود البرنامج، واستخدام الحلقات التكرارية له مزايا عديدة منها سهوله البرمجة واختصار الوقت والجهد

- الجملة التكرارية For Loop

تعتبر جملة For أكثر انتشارا وتستخدم لتكرار تنفيذ عملية أكثر من مرة وتستخدم بكثرة مع المصفوفات لتجنب التعامل مع كل عنصر على حدة
الصيغة Syntax:

```
For (Variable Assignment ; Test Expression ; Variable Increment or Decrement )  
{  
    Statement  
}
```

حيث أن:

Variable Assignment : القيمة الابتدائية initialization التي يبدأ بها التكرار.
Test Expression : شرط التكرار حيث يستمر التكرار طالما أن هذا الشرط صحيح.
Variable Increment or Decrement : مقدار الزيادة أو النقصان في قيمة العداد
Statement : الجمل المطلوب تنفيذها داخل التكرار

مثال:

طباعة الكلمة " Good " ٢٠ مره على أن تكون كل مره على سطر مستقل.

1	For (int i=1; i<=20; i++)
2	System.out.println("good");

```
public static void main(String[] args) {  
    for(int i=1; i<=20 ; i++)  
        System.out.println("good");  
}
```

الفصل الرابع: جمل التكرار والجمل الشرطية في لغة الجافا

ويظهر ناتج التنفيذ كما يلي :

```
Output - JavaApplication1 (run)
good
good
good
good
good
good
good
good
good
good
BUILD SUCCESSFUL (total time: 1 second)
```

مثال :

طباعة الأعداد الفردية من ١ إلى ٣٠

1	For (int i=1 ;i<=30 ;i+=2)
2	System.out.println("java i=" + i)

```
public static void main(String[] args) {
    for(int i=1 ;i<=30 ;i+=2)
        System.out.println("java i=" + i);
}

Output - JavaApplication1 (run)
run:
java i=1
java i=3
java i=5
java i=7
java i=9
java i=11
java i=13
java i=15
java i=17
java i=19
java i=21
java i=23
java i=25
java i=27
java i=29
BUILD SUCCESSFUL (total time: 0 seconds)
```

الفصل الرابع: جمل التكرار والجمل الشرطية في لغة الجافا

مثال :
طباعة الأعداد تنازليا من ٥٠ إلى ٣٠

1	For (int i=50 ;i>=30 ; i--)
2	System.out.println("java i=" + i)

يظهر ناتج التنفيذ كما يلي :

```
Output - JavaApplication1 (run)
> java i=50
java i=49
java i=48
java i=47
java i=46
java i=45
java i=44
java i=43
java i=42
java i=41
java i=40
java i=39
java i=38
java i=37
java i=36
java i=35
java i=34
java i=33
java i=32
java i=31
java i=30
BUILD SUCCESSFUL (total time: 0 seconds)
```

مثال :
استخدام العداد في جملة For لإيجاد حاصل ضرب في القيم 10 ، 100 ، 1000

1	public static void main(String []arg) {
2	int i;
3	for(i=0;i<=10;++i)
4	{
5	System.out.println(i+"\t"+(i*10)+"\t"+(i*100)+"\t"+(i*1000));
6	}
7	}

يظهر ناتج التنفيذ كما يلي :

```
run:
0      0      0      0
1      10     100    1000
2      20     200    2000
3      30     300    3000
4      40     400    4000
5      50     500    5000
6      60     600    6000
7      70     700    7000
8      80     800    8000
9      90     900    9000
10     100    1000   10000
BUILD SUCCESSFUL (total time: 1 second)
```

الجمل الشرطية المتداخلة

قد يوجد أكثر من جملة شرطية متداخلة فإذا كان شرط ما صحيحاً فإنه يجب أن يكون شرط آخر صحيحاً لكي يتم تنفيذ كود معين.

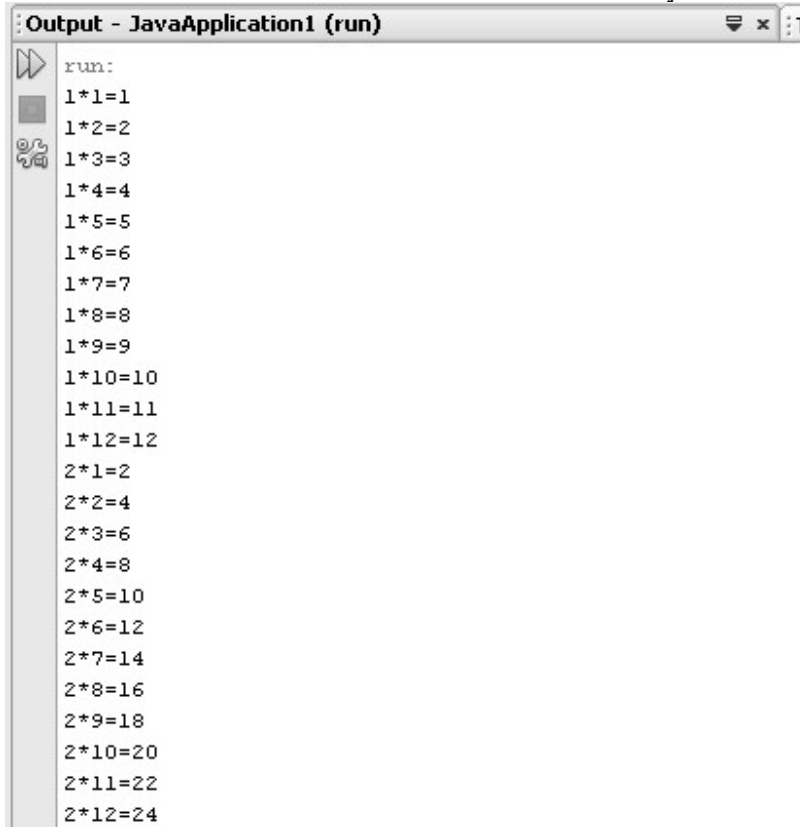
مثال :

طباعة جدول ضرب الأعداد من ١ إلى ١٢

```
1 For ( int w=1 ; w <= 12 ;w++ )
2 For ( z=1 ; z <= 12 ; z++ )
3 System.out.println ( w + " * " + z + " = " + w * z )
```

```
public static void main(String[] args) {
    for(int w=1 ;w<=12 ;w++)
        for(int z=1 ; z<=12 ; z++)
            System.out.println(w + "*" + z + "=" + w * z);
}
```

ويظهر ناتج التنفيذ كما يلي :



```
Output - JavaApplication1 (run)
run:
1*1=1
1*2=2
1*3=3
1*4=4
1*5=5
1*6=6
1*7=7
1*8=8
1*9=9
1*10=10
1*11=11
1*12=12
2*1=2
2*2=4
2*3=6
2*4=8
2*5=10
2*6=12
2*7=14
2*8=16
2*9=18
2*10=20
2*11=22
2*12=24
```

-الجملة التكرارية While Loop

نستخدم الحلقة **while** إذا كنا نريد تنفيذ الكود عدة مرات ولكننا لا نعرف كم مرة بالتحديد لأننا نريد إيقاف التنفيذ إذا تحقق شرط معين ، تتوقف هذه الحلقة عن تكرار نفسها إذا تحقق الشرط الذي تم وضعه لها (طالما أن الشرط لم يتحقق استمر في تكرار الكود)
الصيغة :Syntax

Initialization;

While (Expression)

```
{  
    // Statements ;  
    Decrement Or Increment;  
}
```

Initialization : تعريف متغير (عداد)

مثال :

برنامج لطباعة الأعداد المحصورة بين أي رقمين

```
1 Int w=20 ;  
2 Int z = 30 ;  
3 While ( w<= z )  
4 {  
5     System.out.println(w) ;  
6     W++;  
7 }
```

ويظهر ناتج التنفيذ كما يلي:



```
run:  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
BUILD SUCCESSFUL (total time: 0 seconds)
```


مثال:
طباعة الأعداد من 0 إلى 7

```
1 public static void main(String []arg)
2 {
3     int i=0;
4     while (i<=7)
5     {
6         System.out.println(i);
7         ++i;
8     }
9 }
```

```
public static void main(String []arg)
{
int i=0;
while (i<=7)
{
System.out.println(i);
++i;
}
}
```

ويظهر ناتج التنفيذ كما يلي:

Output - JavaApplication7 (run)

```
run:
0
1
2
3
4
5
6
7
BUILD SUCCESSFUL (total time: 0 seconds)
```

الفصل الرابع: جمل التكرار والجمل الشرطية في لغة الجافا

مثال :
جمع سلسلة من الأعداد داخل نطاق معين.

1	Int w=100 ;
2	Int z = 200 ;
3	Int sum = 0 ;
4	While (w<= z)
5	{
6	Sum = sum + w;
7	W++;
8	}
9	System.out.println(w) ;

```
public static void main(String[] args) {  
    int w = 100;  
    int z = 200 ;  
    int sum = 0 ;  
    while (w<= z)  
    {  
        sum=sum+w;  
        w++ ;  
    }  
    System.out.println(" sum =" + sum );  
}
```

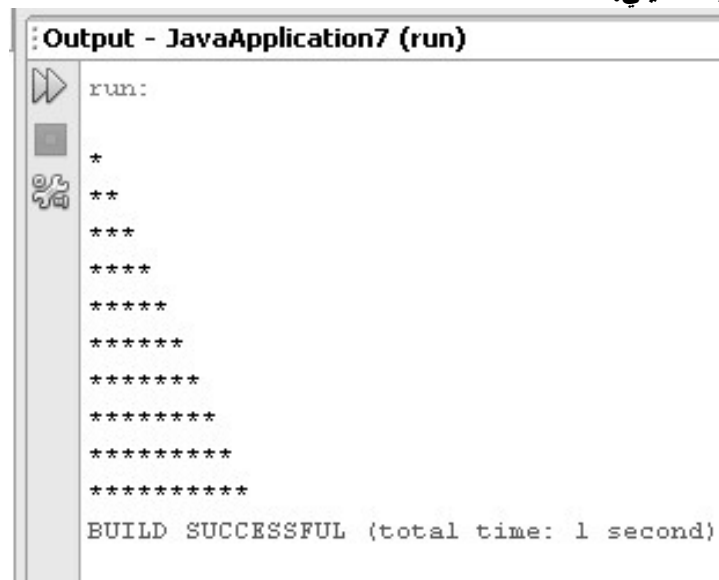
ويظهر ناتج التنفيذ كما يلي :

```
Output - JavaApplication_while2 (run)  
run:  
sum =15150  
BUILD SUCCESSFUL (total time: 0 seconds)
```

مثال:
الحلقات التكرارية المتداخلة

```
1 public static void main(String []arg)
2 {
3     int b,a=0;
4     while (a<=10)
5     {
6         b=1;
7         while(b<=a)
8         {
9             System.out.print ("*");
10            ++b;
11        }
12        System.out.print ("\n");
13        ++a;
14    }
15 }
```

ويظهر ناتج التنفيذ كما يلي:



```
Output - JavaApplication7 (run)
run:
*
**
***
****
*****
*****
*****
*****
*****
*****
*****
*****
BUILD SUCCESSFUL (total time: 1 second)
```

الفصل الرابع: جمل التكرار والجمل الشرطية في لغة الجافا

-الحلقة التكرارية اللانهائية

تكون الجملة الشرطية الخاصة بالعبارة While دائما صحيحة عندئذ يتم تنفيذ العبارات إلى ما نهاية
مثال

```
1 public static void main(String []arg) {
2     int a=1;
3     int b;
4     while (true)
5     {
6         b=1;
7         while(b<=a)
8         {
9             System.out.print(a);
10            ++b;
11        }
12        System.out.print("\n");
13        ++a;
14    }
15 }
```

ويظهر ناتج التنفيذ كما يلي :

[illegible]

ولإيقاف تلك الحلقة اللانهائية نضغط مفتاحي Ctrl + C

- الحلقة التكرارية Do - While Loop

تتميز جملة التكرار Do - While عن جملة While أنها تقوم بتنفيذ الأمر أولاً ثم اختبار الشرط الصيغة Syntax:

Initialization;

Do

{

Statement;

Decrement Or Increment;

}

While (Expression);

مثال:

طباعة سلسلة من الأرقام المتتالية كل رقم على صف مستقل.

1	Int w=10 ;
2	Int z = 15 ;
3	Do
4	{
5	System.out.println(w) ;
6	W++;
7	{
8	While (w<= z) ;

```
public static void main(String[] args) {  
    int w = 10;  
    int z = 15;  
    do  
    {  
        System.out.println(w);  
        w++;  
    }  
    while (w<=z);  
}
```

ويظهر ناتج التنفيذ كما يلي :

Output - JavaApplication_dowhile (run)

```
run:  
10  
11  
12  
13  
14  
15  
BUILD SUCCESSFUL (total time: 0 seconds)
```

مثال

1	public static void main(String []arg){
2	int i=0;
3	do
4	{
5	System.out.println("HI");
6	++i;
7	}
8	while(i>7);
9	}
10	}

نلاحظ أن الجملة الشرطية بعد While غير متحققة لذا فمن غير المفروض عدم تنفيذ الأوامر الموجودة بين القوسين لذا يظهر ناتج التنفيذ كما يلي:

tput - JavaApplication8 (run)

```
run:
HI
BUILD SUCCESSFUL (total time: 1 second)
```

جمل التحكم Control Statements مع الحلقات التكرارية في الجافا

تستخدم للتحكم في سير تنفيذ الحلقات و مع جملة الشرط Switch

تعريف جملة التحكم Break Statement

تستخدم الجملة break في الحلقات التكرارية و في الجملة switch ، بمجرد ان تنفذ الجملة break فإنها توقف الـ scope بأكمله و تخرج منه و تمسحه من الذاكرة ثم تنتقل للكود الذي يليه في البرنامج.
الصيغة Syntax:

break;

مثال

تعريف حلقة لطباعة الأرقام من 1 إلى 10 مع استخدام الجملة break لجعل الحلقة تتوقف عندما تصبح قيمة العداد i=6

```
1 public class Main {
2     public static void main(String[] args) {
3         // إنشاء حلقة for تتكون من ١٠ دورات في كل دورة تطبع قيمة العداد المستخدم فيها
4         for( int i=1; i<=10; i++ )
5         {
6             // في كل دورة سيتم فحص قيمة العداد و بمجرد أن تساوي ٦ سيتم إيقاف الحلقة نهائياً
7             if( i == 6 ) {
8                 break;
9             }
10            System.out.println( i );
11        }
12    }
13 }
```

يظهر ناتج التنفيذ كما يلي:

```
1
2
3
4
5
```

إذاً الجملة break جعلت الحلقة تتوقف عندما أصبحت قيمة العداد i=6

الفصل الرابع: جمل التكرار والجمل الشرطية في لغة الجافا

تعريف جملة التحكم Continue Statement

نستخدم الجملة `continue` لتجاوز تنفيذ كود معين في الحلقة و نستخدمها تحديداً لإيقاف الدورة الحالية و الانتقال إلى الدورة التالية في الحلقة

الصيغة Syntax:

`Continue;`

مثال

تعريف حلقة تطبع الأرقام من 1 إلى 10 ما عدا الرقم 3 ، تم استخدام الجملة `continue` لجعل الحلقة تتجاوز الدورة الثالثة ، أي لن يتم تنفيذ أمر الطباعة عندما تصبح قيمة العداد `i=3`

```
1 public class Main {
2     public static void main(String[] args) {
3         // إنشاء حلقة for تتألف من ١٠ دورات في كل دورة تطبع قيمة العداد المستخدم فيها
4         for (int i=1; i<=10; i++)
5         {
6             // في كل دورة سيتم فحص قيمة العداد، عندما تصبح تساوي ٣ سيتم الانتقال إلى الدورة التالية في الحلقة
6             // بدون تنفيذ أمر الطباعة الموضوع بعدها
7             if (i == 3) {
8                 continue;
9             }
9             System.out.println(i);
10        }
11    }
12 }
13 }
```

يظهر ناتج التنفيذ كما يلي:

```
1
2
4
5
6
7
8
9
10
```

إذاً الجملة `continue` جعلت الحلقة تتجاوز الدورة الثالثة، لذلك لم تطبع الرقم 3 لأنها لم تنفذ أمر الطباعة في الدورة الثالثة.

الفصل الرابع: جمل التكرار والجمل الشرطية في لغة الجافا

مثال:

تعريف حلقة تطبع جميع الأرقام المفردة من 1 إلى 10 ، استخدمنا الجملة `continue` لجعل الحلقة تتجاوز كل دورة تكون فيها قيمة العداد `i` عبارة عن عدد زوجي

1	<code>public class Main {</code>
2	<code>public static void main(String[] args) {</code>
3	<code>// إنشاء حلقة for تتكون من ١٠ دورات في كل دورة تطبع قيمة العداد المستخدم فيها</code>
4	<code>for (int i=1; i<=10; i++)</code>
5	<code>{</code>
6	<code>// في كل دورة سيتم فحص قيمة العداد، اذا كانت زوجي سيتم الانتقال إلى الدورة التالية في</code>
	<code>الحلقة بدون تنفيذ أمر الطباعة الموضوع بعدها</code>
7	<code>if (i%2 == 0) {</code>
8	<code>continue;</code>
9	<code>}</code>
10	<code>System.out.println(i);</code>
11	<code>}</code>
12	<code>}</code>
13	<code>}</code>

يظهر ناتج التنفيذ كما يلي:

1
3
5
7
9

إذاً الجملة `continue` جعلت الحلقة تتجاوز كل دورة كانت فيها قيمة العداد عبارة عن عدد زوجي.

الجمل الشرطية (جمل التحكم في مسار البرنامج)

الجمل الشرطية من أساسيات أي لغة برمجة حيث تتحكم في خط سير البرنامج فمثلاً عند اختيار المستخدم بين حفظ التغييرات أو إلغاؤها وقام المستخدم باختيار الإلغاء عندئذ يتم تغيير استجابة البرنامج في ضوء رد فعل المستخدم وتستخدم لغة Java الجملة الشرطية IF للتحقق من شرط ما وتنفيذ كود معين إذا كان الشرط صحيحاً True وتنفيذ كود آخر إذا لم يكن صحيحاً False ، ويجب وضع الشرط بين القوسين () ، والكود بين القوسين { } إذا كان يتكون من عدة أسطر أما إذا كان يتكون من سطر واحد فيمكن الاستغناء عن تلك الأقواس.

الحالات المختلفة لجملة IF

الحالة الأولى: جملة IF البسيطة

تستخدم للتحقق من شرط واحد فقط فإذا كان تحقق الشرط يتم تنفيذ الكود المحصور بين القوسين { } ، أما إذا لم يكن صحيحاً فسيتم تجاوزه وتنفيذ الكود الذي يليه الصيغة Syntax:

IF (Expression)

```
{  
    Statement  
}
```

مثال: المقارنة بين رقمين

1	int w=100;
2	int z=60;
3	if(w>z)
4	System.out.println("w greater than z");

```
public static void main(String[] args) {  
    int w=100;  
    int z=60;  
    if(w>z)  
        System.out.println("w greater than z");  
}
```

ويظهر ناتج التنفيذ كما يلي:

Output - JavaApplication_if (run)

```
run:  
w greater than z  
BUILD SUCCESSFUL (total time: 1 second)
```

الفصل الرابع: جمل التكرار والجمل الشرطية في لغة الجافا

مثال:

إذا كانت قيمة المتغير S أكبر من 5 يتم طباعة الجملة: S is bigger than 5

```
1 public class Main {
2     public static void main(String[] args) {
3         int S = 0;
4         if( S > 5 )
5         {
6             System.out.print("S is bigger than 5");
7         }
8     }
9 }
```

الشرط: هل قيمة المتغير S أكبر من 5 ، فكان جواب الشرط لا (false) لذلك لم ينفذ أمر الطباعة الموجود في جملة الشرط ويظهر ناتج التنفيذ كما يلي:

مثال :

إذا كانت قيمة المتغير S أكبر من 5 يتم طباعة الجملة: S is bigger than 5

```
1 public class Main {
2     public static void main(String[] args) {
3         int S = 30;
4         if( S > 5 )
5         {
6             System.out.print("S is bigger than 5");
7         }
8     }
9 }
```

الشرط: هل قيمة المتغير S أكبر من 5؟ فكان جواب الشرط نعم (true) لذلك يتم تنفيذ أمر الطباعة الموجود في جملة الشرط ويظهر ناتج التنفيذ كما يلي:

```
S is bigger than 5
```

الحالة الثانية: جملة IF ----- ELSE

تعني إذا تحقق الشرط ينفذ الكود الأول وإذا لم يتحقق الشرط ينفذ الكود التالي للعبارة ELSE
الصيغة Syntax:

```
IF (Expression)
{
    Statement 1 ;
}
ELSE
{
    Statement 2 ;
}
```

مثال :

```
1 int w=100;
2 int z=500;
3 if(w>z)
4 System.out.println("w greater than z");
5 Else
6 System.out.println("z greater than w");
```

```
} public static void main(String[] args) {
    int w=100;
    int z=500;
    if(w>z)
        System.out.println("w greater than z");
    else
        System.out.println("z greater than w");
}
```

يظهر ناتج التنفيذ كما يلي:

Output - JavaApplication_if (run)

```
run:
z greater than w
BUILD SUCCESSFUL (total time: 0 seconds)
```

مثال:

إذا كانت قيمة المتغير S=5 يتم طباعة الجملة S is equal 5 ، إذا كانت قيمة المتغير S≠5 يتم طباعة الجملة S is not equal 5

```
1 public class Main {
2     public static void main(String[] args) {
3         int S = 5;
4         if( S == 5 ) {
5             System.out.print("S is equal 5");
6         }
7         else {
8             System.out.print("S is not equal 5");
9         }
10    }
11 }
```

يظهر ناتج التنفيذ كما يلي:

S is equal 5

الحالة الثالثة : جملة IF -----ELSEIF

تستخدم تلك الجملة عند زيادة الحالات عن حالتين ويمكن تكرار استخدام العبارة ELSEIF عدد من المرات يتوقف على ما يتطلبه الاستخدام ولكن العبارة ELSE تستخدم مره واحده فقط
الصيغة Syntax:

```
IF (Expression1)
{
    statement ;
}
```

```
ELSEIF (Expression2)
{
    statement ;
}
```

```
ELSEIF(Expression3)
{
    statement ;
}
```

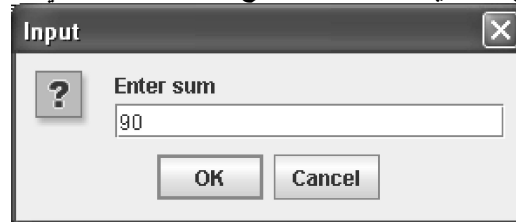
```
ELSE
{
    statement ;
}
```

مثال:
تحويل الدرجة إلى تقدير.

```
1 Public static void main(String[] args) {
2     String msg=" ";
3     int input=0;
4     input=Integer.parseInt(JOptionPane.showInputDialog("Enter Sum"));
5     if(input>=85)msg="ممتاز";
6     else if(input>=75)msg="جيد جدا";
7     else if(input>=65)msg="جيد";
8     else if(input>=50)msg="مقبول";
9     else msg="راسب";
10    JOptionPane.showMessageDialog(null, msg);
11 }
12 }
```

```
public static void main(String[] args) {
    String msg=" ";
    int input=0;
    input=Integer.parseInt(JOptionPane.showInputDialog("Enter sum"));
    if (input>=85)msg="ممتاز";
    else if (input>=75)msg="جيد جدا";
    else if (input>=65)msg="جيد";
    else if (input>=50)msg="مقبول";
    else msg="راسب";
    JOptionPane.showMessageDialog(null, msg);
}
```

وعند التنفيذ يظهر مربع حوارى نكتب به المجموع وليكن ٩٠ كما يلي :



- نضغط مفتاح Ok عندئذ يظهر النافذة كما يلي :



مثال

إذا كانت قيمة المتغير number=1 يتم طباعة الكلمة one
إذا كانت قيمة المتغير number=2 يتم طباعة الكلمة two
إذا كانت قيمة المتغير number=3 يتم طباعة الكلمة three
إذا كانت قيمة المتغير number أكبر أو تساوي 4 يتم طباعة الجملة four or greater
إذا كانت قيمة المتغير number أصغر من 0 يتم طباعة الجملة negative number

```
1 public class Main {
2     public static void main(String[] args) {
3         int number = 3;
4         if( number == 1 ) {
5             System.out.print("one");
6         }
7         else if( number == 2 ) {
8             System.out.print("two");
9         }
10        else if( number == 3 ) {
11            System.out.print("three");
12        }
13        else if( number >= 4 ) {
14            System.out.print("four or greater");
15        }
16        else {
17            System.out.print("negative
18 number");
19        }
20    }
}
```

يظهر ناتج التنفيذ كما يلي:

three

جملة Switch

- تؤدي نفس وظائف جملة IF ----- ELSEIF ولكن بشكل أيسر حيث تقوم باختبار قيمة متغير ما وإجراء أكثر من افتراض عليه
الصيغة Syntax:

```
Switch (variable)
{
Case State 1 :
    Statement;
    Break ;
Case State 2 :
    Statement;
    Break ;
Case State 3 :
    Statement;
    Break ;
Default ;
    Statement;
}
```

حيث أن:

Switch : لاختبار قيمة المتغير

Case : لتحديد الحالات المختلفة للمتغير

Break : للخروج من الجملة الشرطية عند تحقق حالة ما (لإيقاف تنفيذ جملة Switch يجب
دائما كتابة جملة Break بعد نهاية كل جملة Case ولا يتوجب وجودها في السطر
الذي يحتوي على جملة Default)

Default : لتنفيذ مجموعة من الجمل في حالة عدم تحقق أي حالة


```
1 public static void main(String[] args) {
2     int v=1;
3     switch(v)
4     {
5         case 0:
6             System.out.println("١٦٠٠=السعر");
7             break;
8         case 1:
9             System.out.println("١٧٠٠=السعر");
10            break;
11        case 2:
12            System.out.println("١٨٠٠=السعر");
13            break;
14        case 3:
15            System.out.println("١٩٠٠=السعر");
16            break;
17        case 4:
18            System.out.println("٢٠٠٠=السعر");break;
19        case 5:
20            System.out.println("٢١٠٠=السعر");break;
21        case 6:
22            System.out.println("٢٢٠٠=السعر");break;
23        default:
24            System.out.println("");break;
25    }
26 }
```

مثال

اختبار قيمة المتغير x و الذي نوعه int ، نضع عدة حالات وكل حالة تطبع شيء معين.

```
1 public class Main {
2     public static void main(String[] args) {
3         int x = 2;
4         switch( x ) { // اختبار قيمة المتغير x
5             case 1: // سيتم تنفيذ أمر الطباعة الموضوع فيها
6                 System.out.println("x contain 1");
7                 break;
8             case 2: // سيتم تنفيذ أمر الطباعة الموضوع فيها
9                 System.out.println("x contain 2");
10                break;
11             case 3: // سيتم تنفيذ أمر الطباعة الموضوع فيها
12                 System.out.println("x contain 3");
13                 break;
14             default: // سيتم تنفيذ
15                 // أمر الطباعة الموضوع فيها
16                 System.out.println("x contain a different value");
17             }
18         }
19     }
```

يظهر ناتج التنفيذ كما يلي:

```
x contain 2
```