Repetition Statement

Computer Engineering

Yusramohammed@tiu.edu.iq

2022 - 2023

while(true)

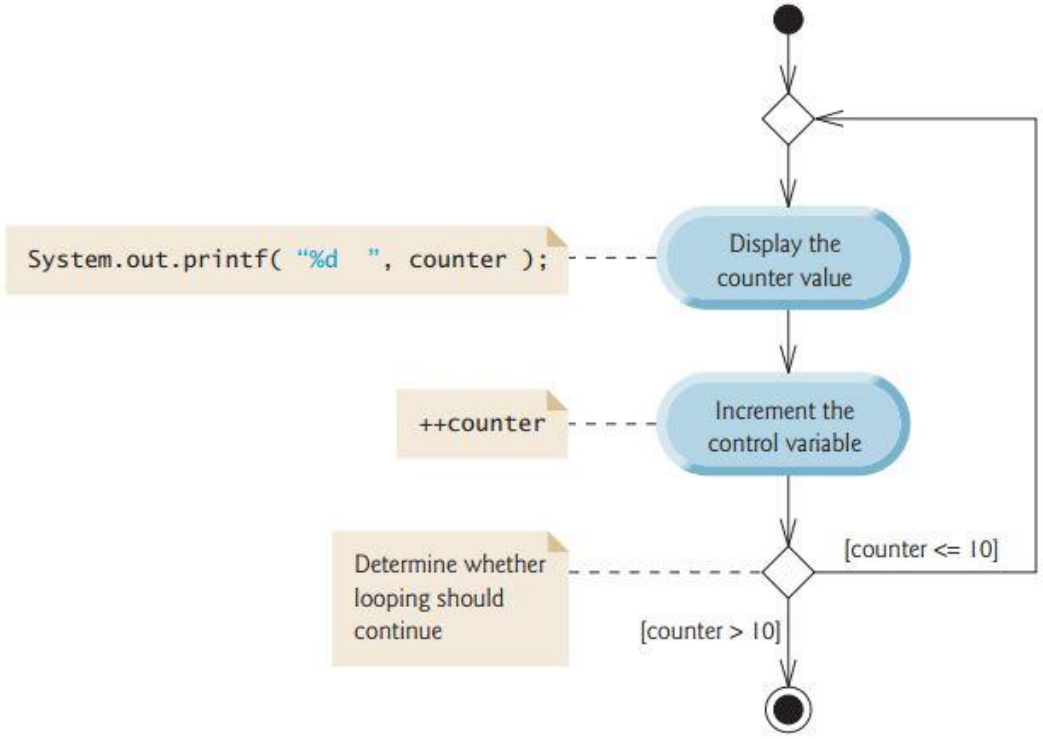
- While true is used to make the program run continuously
- There is no condition and increment or decrement to stop the program.
- We use break statement to stop the while loop when a specific condition is true by using if statement in the while loop.

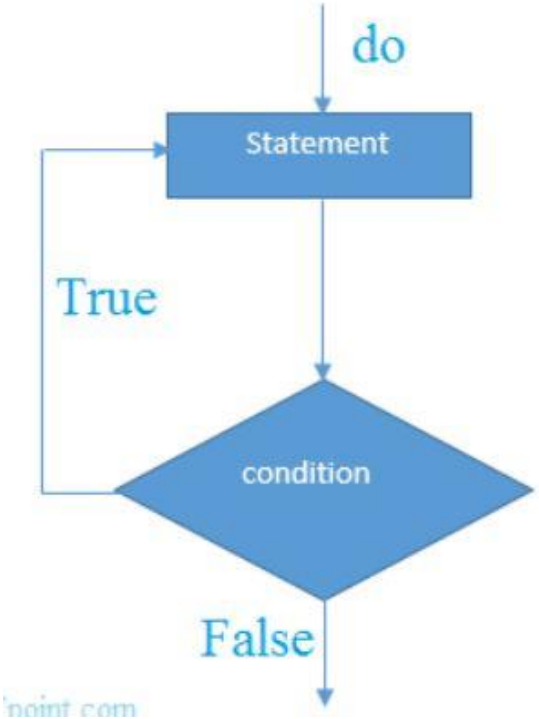
```
Scanner input=new Scanner(System.in);
System.out.println("Enter numbers:");
int sum=0;

while(true)
{
    int number=input.nextInt();

    if(number==-1)
        break;
    else
        sum+=number;


}
System.out.println("Sum of numbers is: "+sum);
```

```
run:
Enter numbers:
4
5
6
7
-1
Sum of numbers is: 22
```

Do-While Repetition statement

- The **do…while repetition statement** is similar to the while statement. In the while, the program tests the loop-continuation condition at the beginning of the loop, before executing the loop's body; if the condition is false, the body *never* executes.
- *The do…while statement tests the loop-continuation condition after executing the loop's body; therefore, the body always executes at least once.*





Do-While Repetition statement

- The syntax structure of do- while statement is:

```
Declaration and Initialization
do
{
      //Your Code
      Increment or decrement

} While (condition)
```

Example

```
int x=1;
do
{
            System.out.println(x);
            x++;
} While (x<10)
```

- It isn't necessary to use braces in the do…while repetition statement if there's only one statement in the body. However, many programmers include the braces, to avoid confusion between the while and do…while statements. For example:

```
do
   statement
while ( condition );
```

```
do
{
   statement
} while ( condition );
```

Do-While Repetition statement

```
public class dowhile {
    public static void main(String[] args)
        int x=1;
        do{
          System.out.println(x);
          x++;
        }
        while(x<10);
    }}
```

output

```
run:
1
2
3
4
5
6
7
8
9
```

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

output

```
run:
10
```

Do-While (Example)

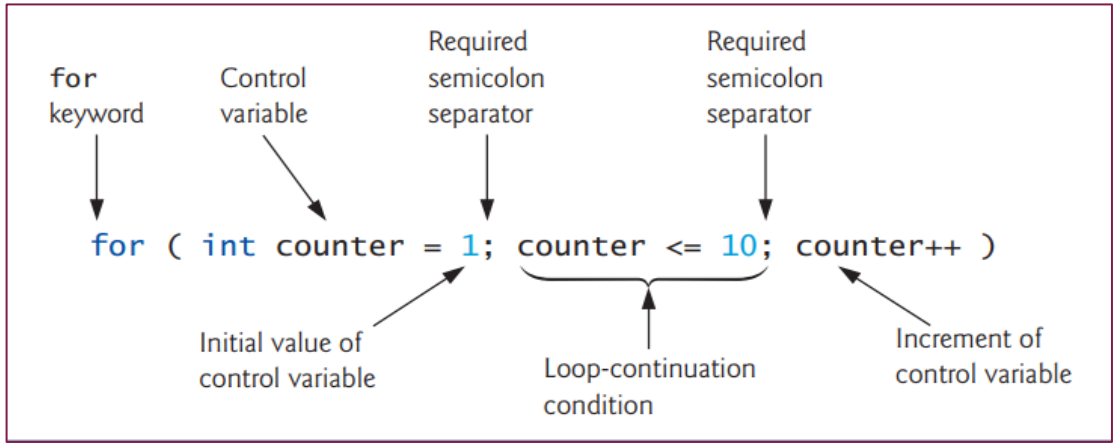
- ***Find the Smallest and Largest Value)*** Write an application that finds the smallest and largest of several integers. Assume that the first value read specifies the number of values to input from the user. If user input -1 the program stops and shows the smallest and largest value among the integers. (Use do-while)

```
Scanner input=new Scanner(System.in);
System.out.print("Enter number or -1 to qiut: ");
int number=input.nextInt();
int max=number;
int min=number;
do{
    if(number>max)
        max=number;
    else if(number<min)
        min=number;

   System.out.print("Enter number or -1 to qiut: ");
   number=input.nextInt();
}while(number!=-1);
System.out.printf("Max number is: %d\n",max);
System.out.printf("Min number is: %d\n",min);
```

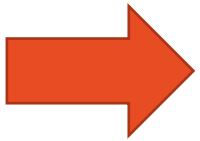
```
run:
Enter a number or -1 to qiut: 32

Enter a number or -1 to qiut: 1
Enter a number or -1 to qiut: 78
Enter a number or -1 to qiut: 65
Enter a number or -1 to qiut: 3
Enter a number or -1 to qiut: 9
Enter a number or -1 to qiut: -1
Max number is: 78
Min number is: 1
```

For Repetition Statement

- The while statement can be used to implement any counter-controlled loop.
- Java also provides the **for repetition statement**, which specifies the counter-controlled-repetition details in a single line of code.
- General format of ***for*** statement.




```
initialization;
while ( loopContinuationCondition )
{
   statement
   increment;
}
```



```
for ( initialization; loopContinuationCondition; increment )
   statement
```

For Repetition Statement

```
public class forloop {
    public static void main(String[] args) {
        for (int i = 0; i <= 10; i++) {
            System.out.println(i);
        }
    }
}
```

```
int i=0;
while(i<=10)
{
    System.out.println(i);
    i++;
}
```

- Both ***for*** and ***while*** has the same output which prints from 0 to 10.
- ***for*** statements are used for counter-controlled repetition, it means the repetition of the loop is known.
- ***while*** statements for sentinel-controlled repetition. However, while and for can each be used for either repetition type. it means the repetition of the loop is NOT known.

For Repetition Statement (Example)

- Write a program to find the average of a no of students entered by user. Use for loop statement.

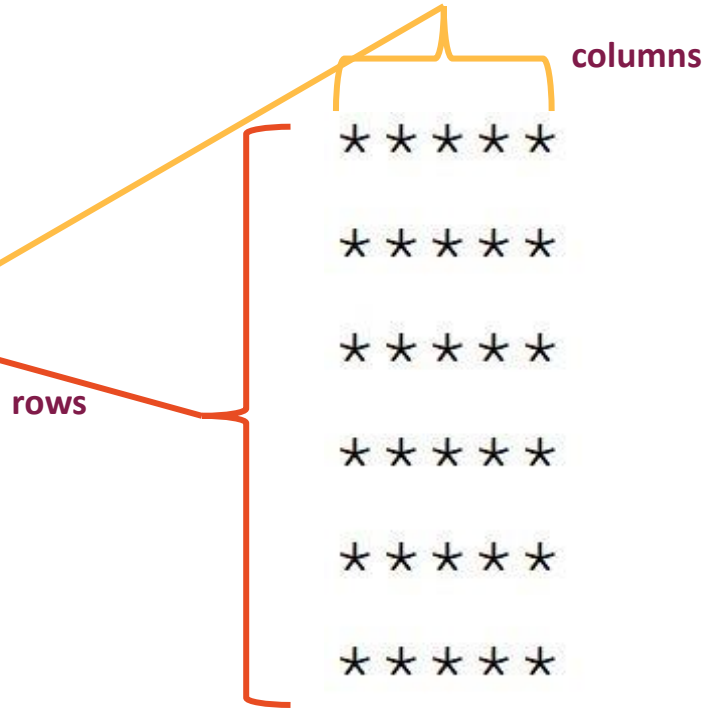
```
public class forloop {
    public static void main(String[] args) {
        Scanner input=new Scanner(System.in);
        System.out.println("Howmany Students do you have");
        int nos=input.nextInt();
        System.out.println("enter grades:");
        int grade;
        int total=0;
        double avg;
        for(int i=1;i<=nos;i++){
            grade=input.nextInt();
            total+=grade;
        }
        avg=total/nos;
        System.out.println("total: "+total);
        System.out.println("average: "+avg);
    }
}
```

```
Howmany Students do you have
4
enter grades:
43
54
65
76
total: 238
average: 59.0
```

Nested For Repetition Statement

- Nested for statement is using a for loop inside another for loop. The output of such loops is in row and coloumns. But not all the time, it depend on the program.
- The first(outer) for loop represent the rows, and the second(inner) for loop represent the columns.

```
public class forloop {
    public static void main(String[] args) {
        for (int i = 0; i <= 5; i++) {
            for (int j = 0; j < 5; j++) {
                System.out.print("*");
            }
            System.out.println("");
        }

    }
}
```



Nested for (Example)

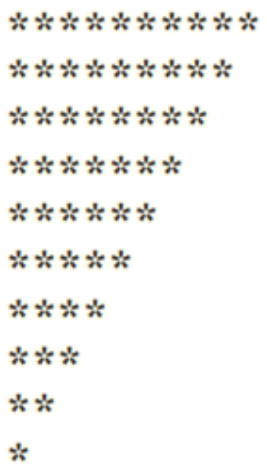
- Write an application that displays the following patterns separately, one below the other. Use for loops to generate the patterns. All asterisks (*) should be printed by a single statement of the form ***System.out.print('*');*** which causes the asterisks to print side by side.

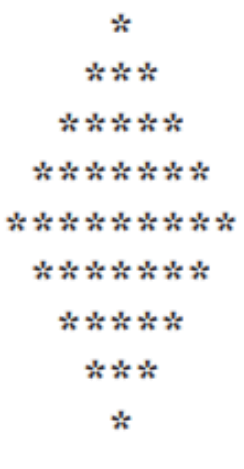
```
public class forloop {
    public static void main(String[] args) {
        for (int row = 0; row <= 10; row++) {
            for (int col = 0; col <=row; col++) {
                System.out.print("*");
            }
            System.out.println("");
        }


    }
}
```

```
(a)
*
**
***
****
*****
******
*******
********
*********
**********
```

Nested for Examples

- Try to write a java program to display the shapes below:





Continue - Break

- ***Break***: The break statement, when executed in a a while, for, do…while or switch , causes immediate exit from that statement.
- **Example**: Write a program to print the numbers between 1 to 10, and break the program if the number is equal to 5.

```
    public static void main(String[] args) {
        int i=1;
        while(i<=10)
        {
            if(i==5)
            {
                System.out.println("Out of loop the number is 5");
                break;
            }
            else
                System.out.println(i);
            i++;
        }
    }
}
```

Continue - Break

- ***Continue***: The continue statement, when executed in while, for or do…while, skips the remaining statements in the loop body and proceeds with the *next iteration* of the loop.
- It is better to use for statement with continue statement. Because in a while statement the increment and decrement follows continue statement, In this case, the increment does *not* execute before the program evaluates the repetition-continuation condition.
- **Example:** Write a program to print out a series of numbers from 1 to 10. if the number is equal to 5, continue the program without printing the 5.

```
public class forloop {
    public static void main(String[] args) {

        for(int i=1;i<=10;i++)
        {
            if(i==5)
                continue;
            System.out.println(i);
        }


    }
}
```

```
1
2
3
4
6
7
8
9
10
```