

Object Oriented Programming

Prepared by Dr. Yasser Abdelhamid

IT223
Object Oriented Programming

Lecture 06



OUTLINE

- Java control statements
- Decision making
- If statement
- Switch statement
- For loop
- While loop
- Do-while loop
- Break statement
- Continue statement

CONTROL STATEMENTS

CONTROL STATEMENTS

❖ Control statements are used to change the sequence of executing program code to fulfil the desired logic of the program.

CONTROL STATEMENTS

❖ Java provides three types of control flow statements.

❖ Decision Making statements

- if statements
- switch statement

❖ Loop statements

- do while loop
- while loop
- for loop
- for-each loop

❖ Jump statements

- break statement
- continue statement

DECISION-MAKING STATEMENTS

DECISION-MAKING STATEMENTS

- ❖ According to a conditional expression which gives a Boolean value (True/False), decision-making statements control the program flow.

IF STATEMENT

❖ Syntax:

```
if ( <condition> ) {  
    <statement1> ;  
}
```

Example:

```
public class Student {  
    public static void main(String[] args) {  
        int x = 10;  
        int y = 12;  
        if (x+y > 20) {  
            System.out.println("x + y is greater than 20");  
        }  
    }  
}
```

We use this form when we want to execute a code block when a condition is true.

IF ELSE STATEMENT

❖ Syntax:

```
if ( <condition> ) {  
    <statement1>;  
}  
else {  
    <statement2>;  
}
```

Example:

```
public class Student {  
    public static void main(String[] args) {  
        int x = 10;  
        int y = 12;  
        if(x+y < 10) {  
            System.out.println("x + y is less than 10");  
        } else {  
            System.out.println("x + y is greater than 20");  
        }  
    }  
}
```

We use this form when we want to execute a code block when a condition is true, and execute another code block if it is false.

IF-ELSE-IF

❖ Syntax:

```
if( <condition1> ) {  
    <statement1>;  
}  
else if( <condition2> ) {  
    <statement2>;  
}  
else {  
    <statement3>;  
}
```

We use this form when we have multiple conditions and multiple code blocks to execute when each of these conditions is true.

EXAMPLE:

```
public class Student {  
    public static void main(String[] args) {  
        String city = "Unknown";  
        if(city == "Cairo") {  
            System.out.println("city is Cairo");  
        }else if (city == "Aswan") {  
            System.out.println("city is Aswan");  
        }else if(city == "Alexandria") {  
            System.out.println("city is Alexandria");  
        }else {  
            System.out.println(city);  
        }  
    }  
}
```

We use this form when we have multiple conditions and multiple code blocks to execute when each of these conditions is true.

NESTED IF STATEMENT

❖ Syntax:

```
if(<condition1>) {  
    <statement1>;  
    if(<condition 2>) {  
        <statement 2>;  
    }  
    else {  
        statement 2;  
    }  
}
```

Nested conditions implement
logical AND of multiple
conditions

NESTED IF STATEMENT

```
public class Student {  
    public static void main(String[] args) {  
        String address = "12";  
        if(address.endsWith("2")) {  
            if(address.contains("1")) {  
                System.out.println("String is 12");  
            }else {  
                System.out.println("String ends with 2");  
            }  
        }  
    }  
}
```

SWITCH STATEMENT

- ❖ Switch statement is an easy way to implement multiple if statements.
- ❖ The case variables can be int, short, byte, char, or enumeration.
- ❖ String type is also supported since version 7 of Java
- ❖ Cases cannot be duplicated.
- ❖ Break statement terminates the switch block when the condition is satisfied.
- ❖ Break statement is optional, if not used, next case is executed.

SWITCH STATEMENT

❖ Syntax:

```
switch (<expression>){  
    case <value 1>:  
        <statement1>;  
        break;  
  
    .  
    .  
    .  
    case <value N>:  
        <statement>;  
        break;  
    default:  
        <default statement>;  
}
```

SWITCH STATEMENT

```
public class Student {  
    public static void main(String[] args) {  
        int num = 2;  
        switch (num){  
            case 0:  
                System.out.println("number is 0");  
                break;  
            case 1:  
                System.out.println("number is 1");  
                break;  
            default:  
                System.out.println(num);  
        }  
    }  
}
```

FOR LOOP

LOOP STATEMENTS

- ❖ A loop statement is a code block that needs to be repeated.
 - a specific number of times
 - while a condition holds
 - until a condition occurs
- ❖ An iteration is a single pass of a loop.
- ❖ The Java **for** loop is used to repeat a part of the program several times.
- ❖ It is the perfect choice when the number of iterations is known.

FOR LOOP STATEMENT

❖ Syntax:

```
for( [<initialization>] ; [<condition>] ; [<increment/decrement>]){  
    <statement>;  
}
```

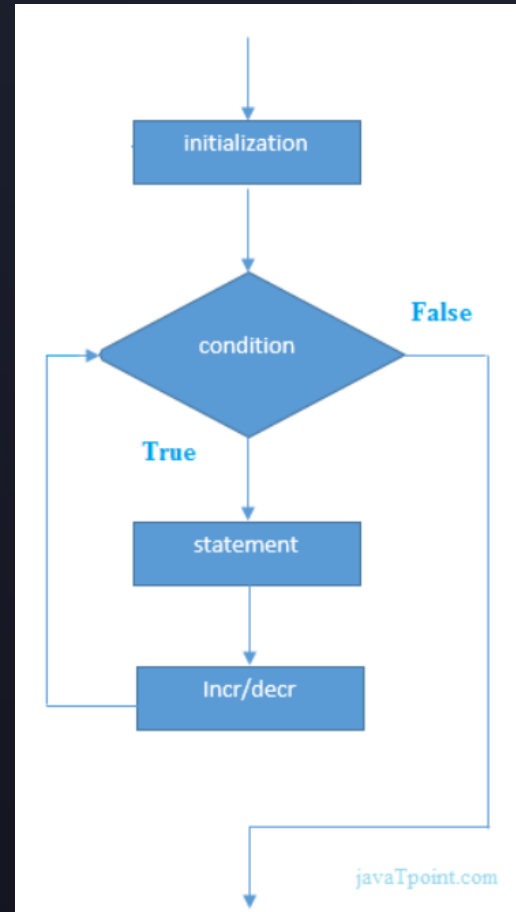
<Initialization> is executed once when the loop starts. Here, we can initialize the iterator variable, or we can use an already initialized variable. **It is an optional condition.**

<condition> is executed each iteration to decide whether to execute the for loop code block or not. It must return boolean value either true or false. **It is an optional condition.**

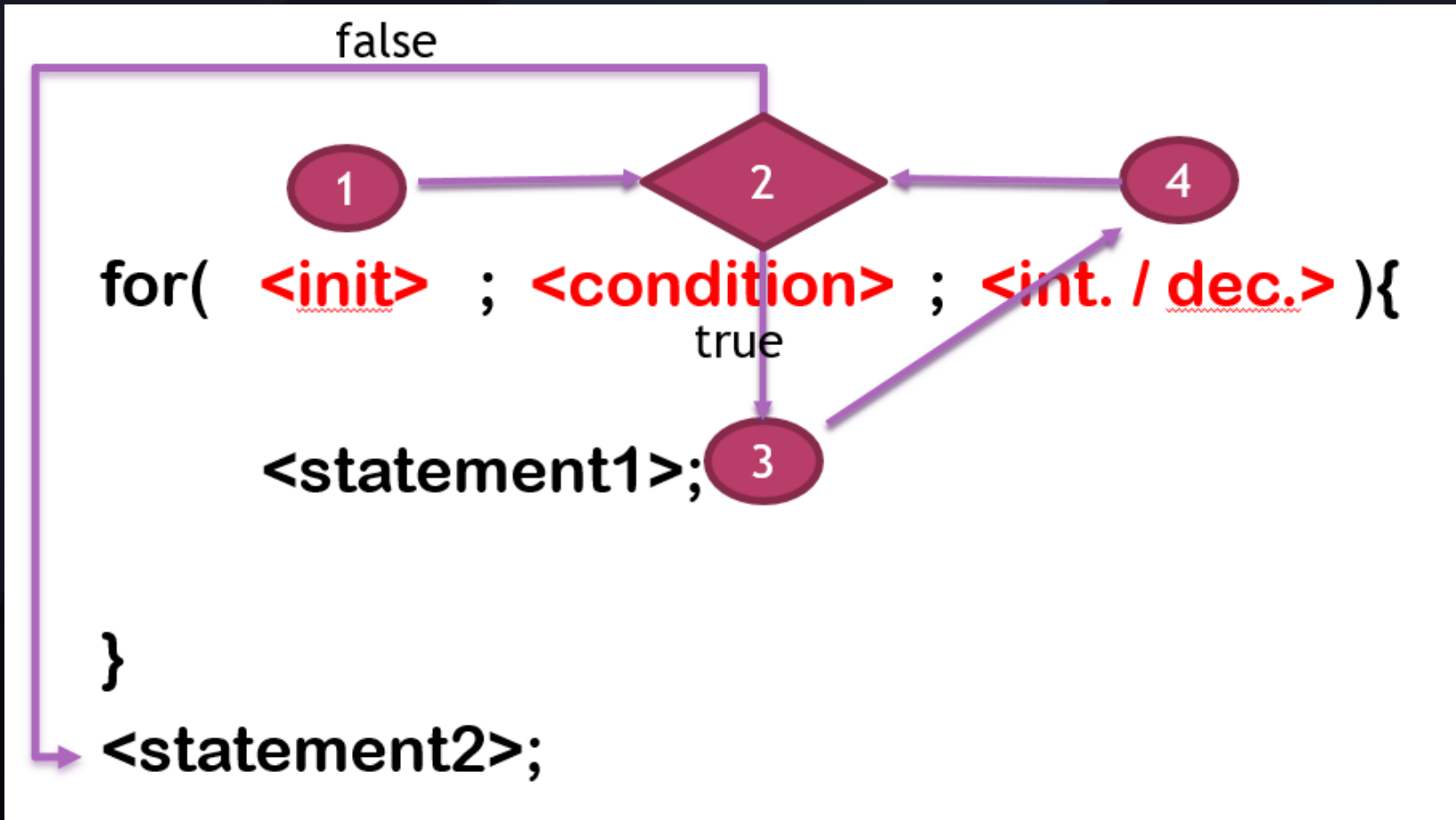
<increment/decrement> : It changes the value of the iterator variable either by incrementing or decrementing its value. **It is an optional condition.**

<statement> is executed each iteration until **<condition>** turns to false.

FOR LOOP STATEMENT



FOR LOOP STATEMENT EXECUTION



EXAMPLES

```
public static void main(String[] args) {  
    for( int i=1; i<=5 ; i++ ){  
        System.out.println( i );  
    }  
}
```

1. What is the output of this method?
2. What is the value of i after the loop?

JAVA NESTED FOR LOOP

```
public static void main(String[] args) {  
    for(int i=1;i< 8;i++){  
        for(int j=1;j< 3;j++){  
            System.out.println(i+" "+j);  
        }  
        // ??? What is the value of j here  
    }  
    // ??? What is the value of i here  
}
```

EXAMPLE

```
public static void main(String[] args) {  
    for(int i=1;i<=5;i++){  
        for(int j=1;j<=i;j++){  
            System.out.print("* ");  
        }  
        System.out.println(); //new line  
    }  
}
```

❖ What is the output of this program?

EXAMPLE

```
public static void main(String[] args) {  
    int k=6;  
    for(int i=1;i<=k;i++){  
        for(int j=k;j>=i;j--){  
            System.out.print("* ");  
        }  
        // What if i put break statement here?  
        System.out.println();  
    }  
}
```

❖ What is the output of this program?

INFINITIVE LOOP

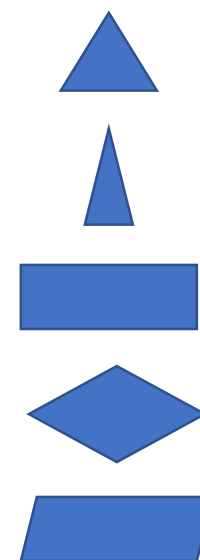
```
for(;;){ }
```

1. Is this a correct statement?
2. What does it do?

ASSIGNMENT 1

Using for loop...

1. Draw an equilateral triangle.
2. Draw an isosceles triangle.
3. Draw a rectangle.
4. Draw a rhombus.
5. Draw a parallelogram.



WHILE LOOP

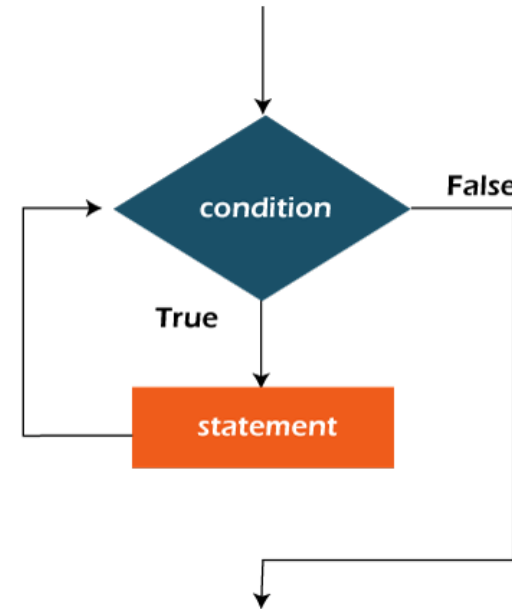
WHILE LOOP

❖ Syntax:

```
while (<condition>){  
    <statement>;  
    <Increment / decrement>;  
}
```

WHILE LOOP

❖ while loop is used to iterate a code block repeatedly as long as a specified condition is true.



EXAMPLE

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

1. What is the output of this program?
2. Reverse it.

ASSIGNMENT 2

- ❖ Implement an infinite loop using while statement.
- ❖ Redo assignment 1 using while statement.

DO-WHILE LOOP

DO-WHILE LOOP STATEMENT

❖ Syntax:

```
do {  
    <statement>;  
    <increment/decrement>;  
} while ( <condition>;
```

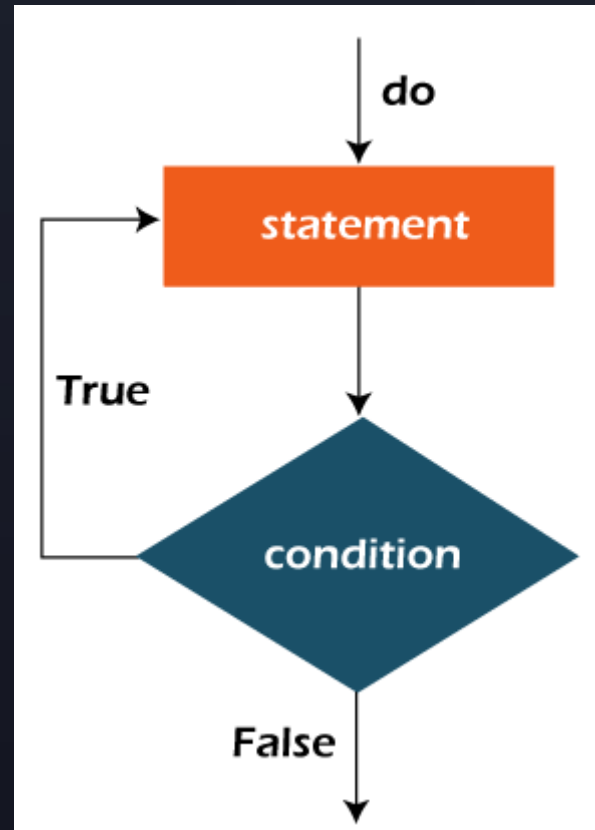
<statement> is executed each iteration until **<condition>** turns to false.

<condition> is executed each iteration to decide whether to execute the do-while loop code block or not. It must return boolean value either true or false.

<increment/decrement> : It changes the value of the iterator variable either by incrementing or decrementing its value.

Notice that the do-while code block is executed at least once.

DO-WHILE LOOP STATEMENT



EXAMPLES

```
public class DoWhileExample {  
    public static void main(String[] args) {  
        int i=1;  
        do{  
            System.out.println(i);  
            i++;  
        } while( i<=3 );  
    }  
}
```

1. What is the output of this method?
2. What is the value of i after the loop?

BREAK STATEMENT

```
public class BreakExample {  
    public static void main(String[] args) {  
  
        for(int i=1;i<=10;i++){  
            if(i==5){  
                break;  
            }  
            System.out.println(i);  
        }  
    }  
}
```

What is the output of this program?
What is the value of i after the loop?

EXAMPLE

```
public class BreakExample {  
    public static void main(String[] args) {  
        for(int i=1;i<=3;i++){  
            for(int j=1;j<=3;j++){  
                if(i==2&& j==2){  
                    break;  
                }  
                System.out.println(i+" "+j);  
            }  
        }  
    }  
}
```

CONTINUE STATEMENT

❖ The Java **continue** statement is used to continue the loop.

❖ Example:

```
public class ContinueExample {  
    public static void main(String[] args) {  
        for(int i=1;i<=10;i++){  
            if(i==5){  
                continue;  
            }  
            System.out.println(i);  
        }  
    }  
}
```

EXAMPLE

```
public class ContinueExample2 {  
    public static void main(String[] args) {  
        for(int i=1;i<=3;i++){  
            for(int j=1;j<=3;j++){  
                if(i==2&&j==2){  
                    continue;  
                }  
                System.out.println(i+" "+j);  
            }  
        }  
    }  
}
```

SUMMARY

- Java control statements
- Decision making
- If statement
- Switch statement
- For loop
- While loop
- Do-while loop
- Break statement
- Continue statement



Thank You

www.midocean.ae

