Object Oriented Programming

JAVA CONTROL STATEMENTS

Control Flow Statements

1. Decision Making statements
    - if statements
    - switch statement
2. Loop statements
    - do while loop
    - while loop
    - for loop
    - for-each loop
3. Jump statements
    - break statement
    - continue statement

Decision-making Statements

- Decision-making statements decide which statement to execute and when.

There are two types of decision-making statements in Java.

- If statement
- Switch statement

If Statement

- Use to evaluate a condition
- Gives a Boolean value, either true or false

There are four types of if-statements given below:

1. Simple if statement
2. if-else statement
3. if-else-if ladder
4. Nested if-statement

Simple if statement

- Evaluates a Boolean expression and enables the program to enter a block of code if the expression evaluates to true.

```
if(condition) {
statement 1; //executes when condition is true
}
```

Simple if statement

```
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");
}
}
}
```

If-else statement

- An extension to the if-statement, which uses another block of code.
- The else block is executed if the condition of the if-block is evaluated as false.

```
if(condition) {
statement 1; //executes when condition is true
}
else{
statement 2; //executes when condition is false
}
```

If-else statement

```
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");
}
}
}
```

If-else-if ladder

- Chain of if-else statements that create a decision tree where the program may enter in the block of code where the condition is true.

- We can also define an else statement at the end of the chain.

```
if(condition 1) {
statement 1; //executes when condition 1 is true
}
else if(condition 2) {
statement 2; //executes when condition 2 is true
}
else {
statement 3; //executes when all the conditions are false
}
```

If-else-if ladder

```
public class Student {
public static void main(String[] args) {
String city = “Dhaka";
if(city == “Chittagong") {
System.out.println("city is Chittagong");
}
else if (city == “Dhaka") {
System.out.println("city is Dhaka");
}
else if(city == “Khulna") {
System.out.println("city is Khulna");
}
else {
System.out.println(city);
} } }
```

Nested if-statement

- The if statement can contain a if or if-else statement inside another if or else-if statement

```
if(condition 1) {
statement 1; //executes when condition 1 is true
if(condition 2) {
statement 2; //executes when condition 2 is true
}
else{
statement 2; //executes when condition 2 is false
}
}
```

Nested if-statement

```
public class Student {
public static void main(String[] args) {
String country= " Bangladesh“, city = “Chittagong“;
if(country==“Bangladesh") {
        if(city == " Chittagong ") {
        System.out.println("Your city is Chittagong");
        }
        else if(city == “Khulna") {
        System.out.println("Your city is Khulna");
        }
        else {
        System.out.println(city);
        }
}
else {
System.out.println("You are not living in Bangladesh");
}   }   }
```

Switch Statement

- Contains multiple blocks of code called cases.
- A single case is executed based on the variable which is being switched.
- Easier to use instead of if-else-if statements.

Switch Statement

Points to be noted about switch statement:

- The case variables can be int, short, byte, char. String type is also supported since version 7 of Java
- Cases cannot be duplicate
- Default statement is executed when any of the case doesn't match the value of expression. It is optional.
- Break statement terminates the switch block when the condition is satisfied. It is optional, if not used, next case is executed.
- While using switch statements, we must notice that the case expression will be of the same type as the variable. However, it will also be a constant value.

Switch Statement

```
switch (expression){
    case value1:
     statement1;
     break;
    .
    .
    case valueN:
     statementN;
     break;
    default:
     default statement;
}
```

Switch Statement

```
public class Student implements Cloneable {
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);
} } }
```

Loop Statements

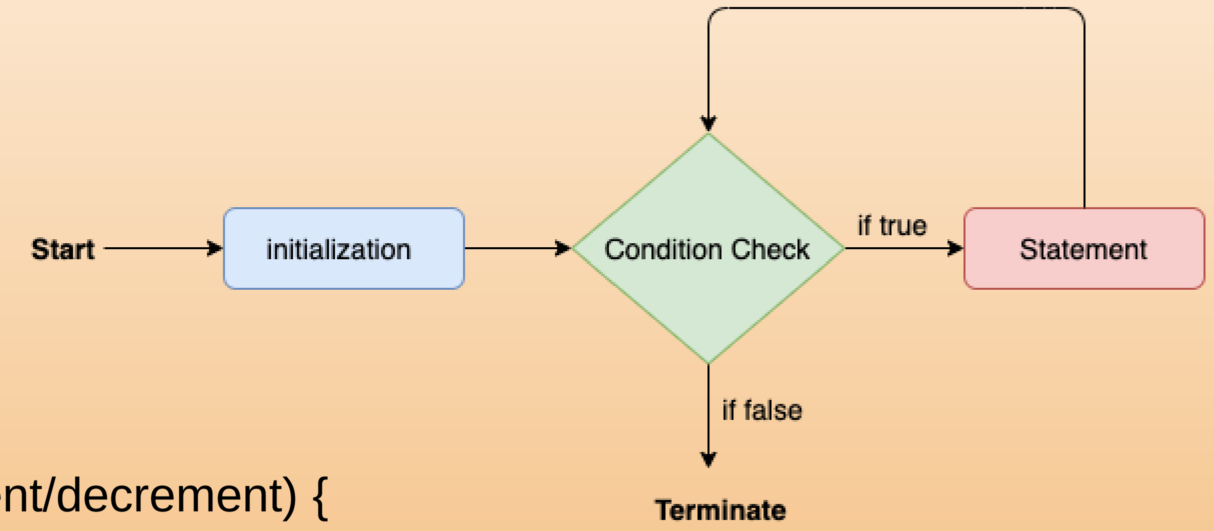
- Execute the block of code repeatedly while some condition evaluates to true.
- Loop statements are used to execute the set of instructions in a repeated order

We have three types of loops that execute similarly:

1. For loop
2. While loop
3. Do-while loop

For loop

Flowchart:



Example:

```
for(initialization, condition, increment/decrement) {
//block of statements
}
```

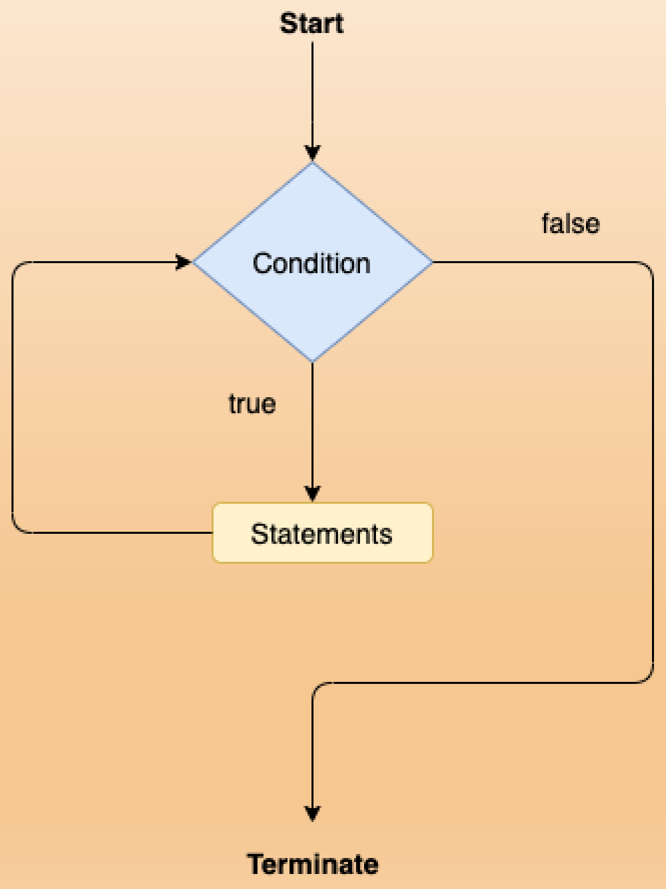
For loop

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

int sum = 0;
for(int j = 1; j<=10; j++) {
    sum = sum + j;
    }
System.out.println("The sum of first 10 natural numbers is " + sum);
}  }
```

While loop

Flowchart:



Example:

```
while(condition){
//looping statements
}
```

While loop

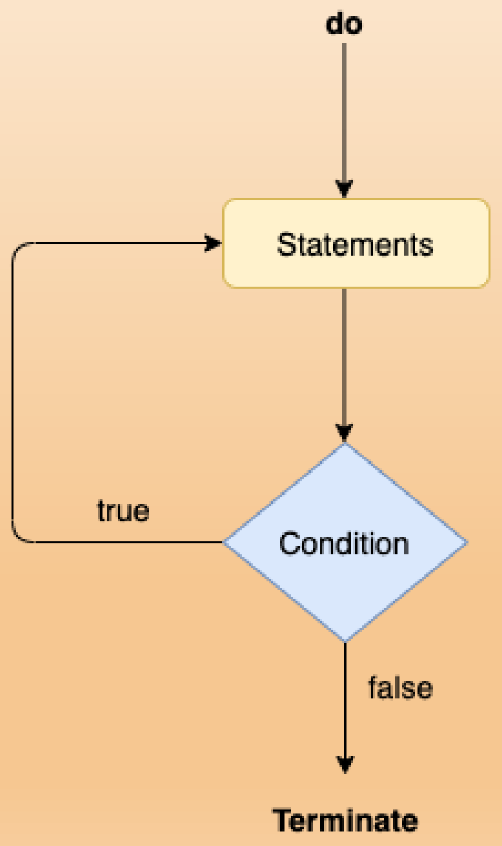
```
public class Calculation {
public static void main(String[] args) {

int i = 0;
System.out.println("Printing the list of first 10 even numbers \n");

while(i<=10) {
    System.out.println(i);
    i = i + 2;
}   }   }
```

Do-while loop

Flowchart:



Example:

```
do
{
//statements
} while (condition);
```

Do-while loop

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

int i = 0;
System.out.println("Printing the list of first 10 even numbers \n");
do {
System.out.println(i);
i = i + 2;
}
while(i<=10);
}   }
```

Jump Statements

- Jump statements are used to transfer the control of the program to the specific statements.
- Transfer the execution control to the other part of the program.

There are two types of jump statements in Java.

1. Break
2. Continue

Break Statement

Example:

```
public class BreakExample {

public static void main(String[] args) {

for(int i = 0; i<= 10; i++) {
System.out.println(i);
     if(i==6) {
          break;
} } }
}
```

Output:

```
0
1
2
3
4
5
```

Continue Statement

Example:

```
public class ContinueExample {

public static void main(String[] args) {
for(int i = 0; i<= 2; i++) {

for (int j = i; j<=5; j++) {

if(j == 4) {
continue;
}
System.out.println(j);
}  }  }
}
```

Output:

```
0
1
2
3
5
1
2
3
5
2
3
5
```

Thank You