JAVA QUESTIONS

1. Explain about the main method in java?

Ans: The Java main method is usually the first method you learn about when you start programming in Java because its the entry point for executing a Java program. The main method can contain code to execute or call other methods, and it can be placed in any class that’s part of a program.

Syntax:

public static void main(String[] args){

// some code

}

**Public:**

The access modifier of the main method needs to be public so that the JRE can access and execute this method. If a method isn’t public, then access is restricted.

**static**

When the Java program starts, there is no object of the class present. The main method has to be static so that the JVM can load the class into memory and call the main method without creating an instance of the class first.

**void**

Every Java method must provide the return type. The Java main method return type is void because it doesn’t return anything. When the main method is finished executing, the Java program terminates, so there is no need for a returned object.

**main**

The Java main method is always named main. When a Java program starts, it always looks for the main method.

**String[] args**

Java main method accepts a single argument of type String array. Each string in the array is a command line argument. You can use command line arguments to affect the operation of the program, or to pass information to the program, at runtime.

1. What are the different Control flow Statements available in java?

**Ans: Java’s Selection statements:**

* [if](https://www.geeksforgeeks.org/decision-making-javaif-else-switch-break-continue-jump/#if)
* [if-else](https://www.geeksforgeeks.org/decision-making-javaif-else-switch-break-continue-jump/#if-else)
* [nested-if](https://www.geeksforgeeks.org/decision-making-javaif-else-switch-break-continue-jump/#nested-if)
* [if-else-if](https://www.geeksforgeeks.org/decision-making-javaif-else-switch-break-continue-jump/#if-else-if)
* [switch-case](https://www.geeksforgeeks.org/decision-making-javaif-else-switch-break-continue-jump/#switch-case)
* [jump](https://www.geeksforgeeks.org/decision-making-javaif-else-switch-break-continue-jump/#jump) – break, continue, return
* for loop
* while loop
* do while loop

1. if: if statement is the most simple decision-making statement. It is used to decide whether a certain statement or block of statements will be executed or not i.e if a certain condition is true then a block of statements is executed otherwise not.

Syntax:

if(condition)

{

// Statements to execute if

// condition is true

}

For example,

if(condition) //Assume condition is true

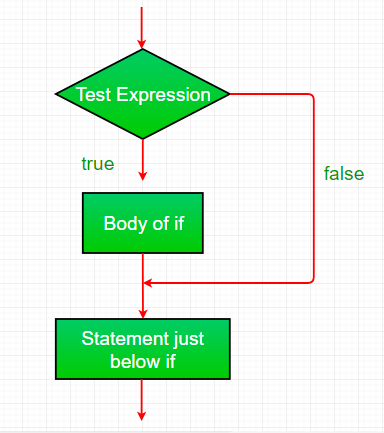
statement1; //part of if block

statement2; // separate from if block

// Here if the condition is true

// if block will consider statement1 as its part and executes in only true condition

// statement2 will be separate from the if block so it will always executes whether the condition is true or false.



**2. if-else**: The if statement alone tells us that if a condition is true it will execute a block of statements and if the condition is false it won’t. But what if we want to do something else if the condition is false? Here comes the else statement. We can use the else statement with the if statement to execute a block of code when the condition is false.

**Syntax**:

if (condition)

{

// Executes this block if

// condition is true

}

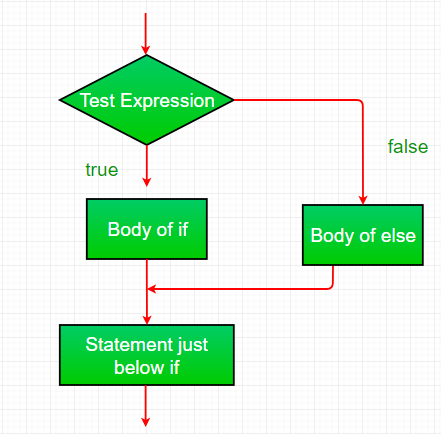
else

{

// Executes this block if

// condition is false

}



**3. nested-if:** A nested if is an if statement that is the target of another if or else. Nested if statements mean an if statement inside an if statement. Yes, java allows us to nest if statements within if statements. i.e, we can place an if statement inside another if statement.

**Syntax:**

if (condition1)

{

// Executes when condition1 is true

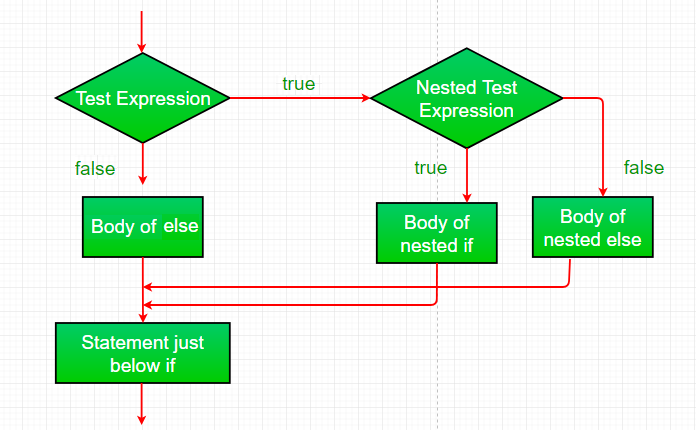
if (condition2)

{

// Executes when condition2 is true

}

}



**4. if-else-if ladder:** Here, a user can decide among multiple options.The if statements are executed from the top down. As soon as one of the conditions controlling the if is true, the statement associated with that ‘if’ is executed, and the rest of the ladder is bypassed. If none of the conditions is true, then the final else statement will be executed. There can be as many as ‘else if’ blocks associated with one ‘if’ block but only one ‘else’ block is allowed with one ‘if’ block.

if (condition)

statement;

else if (condition)

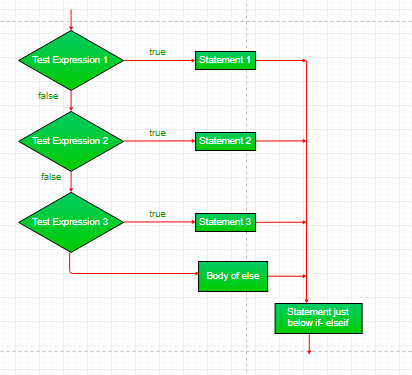
statement;

.

.

else

statement;



**5. switch-case:** The switch statement is a multiway branch statement. It provides an easy way to dispatch execution to different parts of code based on the value of the expression.

**Syntax:**

switch (expression)

{

case value1:

statement1;

break;

case value2:

statement2;

break;

.

.

case valueN:

statementN;

break;

default:

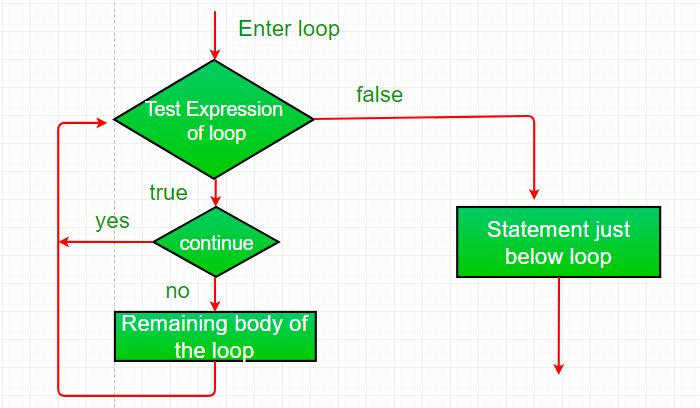
statementDefault;

}

* The expression can be of type byte, short, int char, or an enumeration. Beginning with JDK7, the *expression* can also be of type String.
* Duplicate case values are not allowed.
* The default statement is optional.
* The break statement is used inside the switch to terminate a statement sequence.
* The break statements are necessary without the break keyword, statements in switch blocks fall through.
* If the break keyword is omitted, execution will continue to the next case.

**6. jump:** Java supports three jump statements: **break, continue** and **return**. These three statements transfer control to another part of the program.

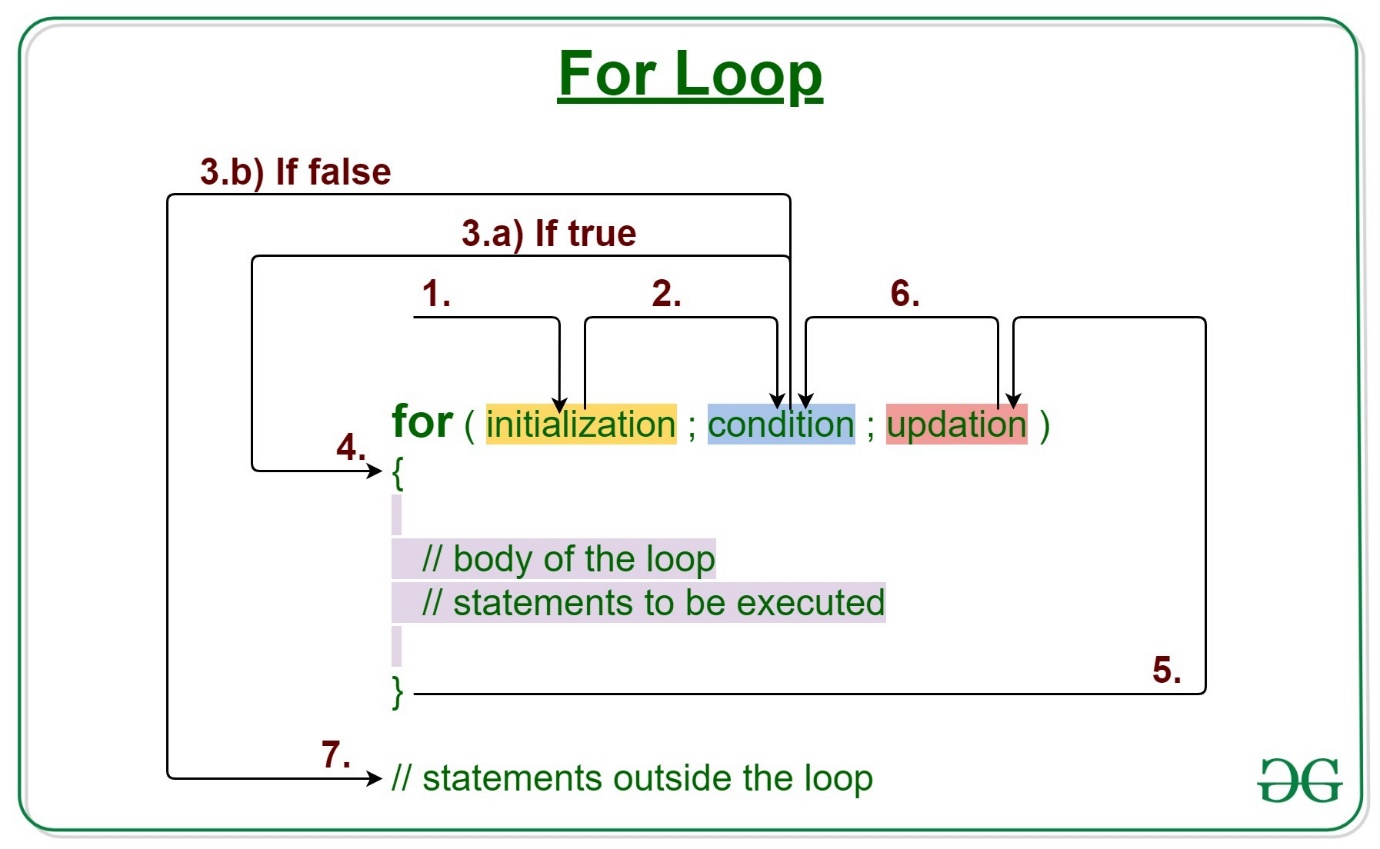
* **Break:** In Java, a break is majorly used for:
  + Terminate a sequence in a switch statement (discussed above).
  + To exit a loop.
  + Used as a “civilized” form of goto.
* **Continue:**Sometimes it is useful to force an early iteration of a loop. That is, you might want to continue running the loop but stop processing the remainder of the code in its body for this particular iteration. This is, in effect, a goto just past the body of the loop, to the loop’s end. The continue statement performs such an action.



* [Return:](https://www.geeksforgeeks.org/return-keyword-java/)The return statement is used to explicitly return from a method. That is, it causes program control to transfer back to the caller of the method.

7. for loop:

The for statement consumes the initialization, condition and increment/decrement in one line thereby providing a shorter, easy to debug structure of looping.



**Syntax:**

for (initialization expr; test expr; update exp)

{

// body of the loop

// statements we want to execute

}

The various **parts of the For loop** are:

1. **Initialization Expression:** In this expression, we have to initialize the loop counter to some value.

**Example:**

int i=1;

2. **Test Expression:** In this expression, we have to test the condition. If the condition evaluates to true then, we will execute the body of the loop and go to update expression. Otherwise, we will exit from the for loop.

**Example:**

i <= 10

3. **Update Expression**: After executing the loop body, this expression increments/decrements the loop variable by some value.

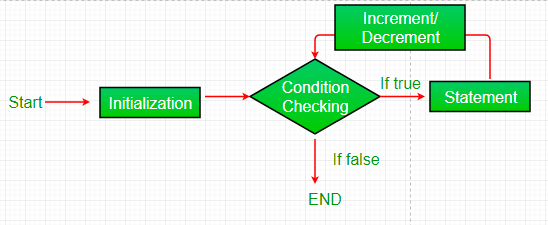
**Example:**

i++;

**How does a For loop execute?**

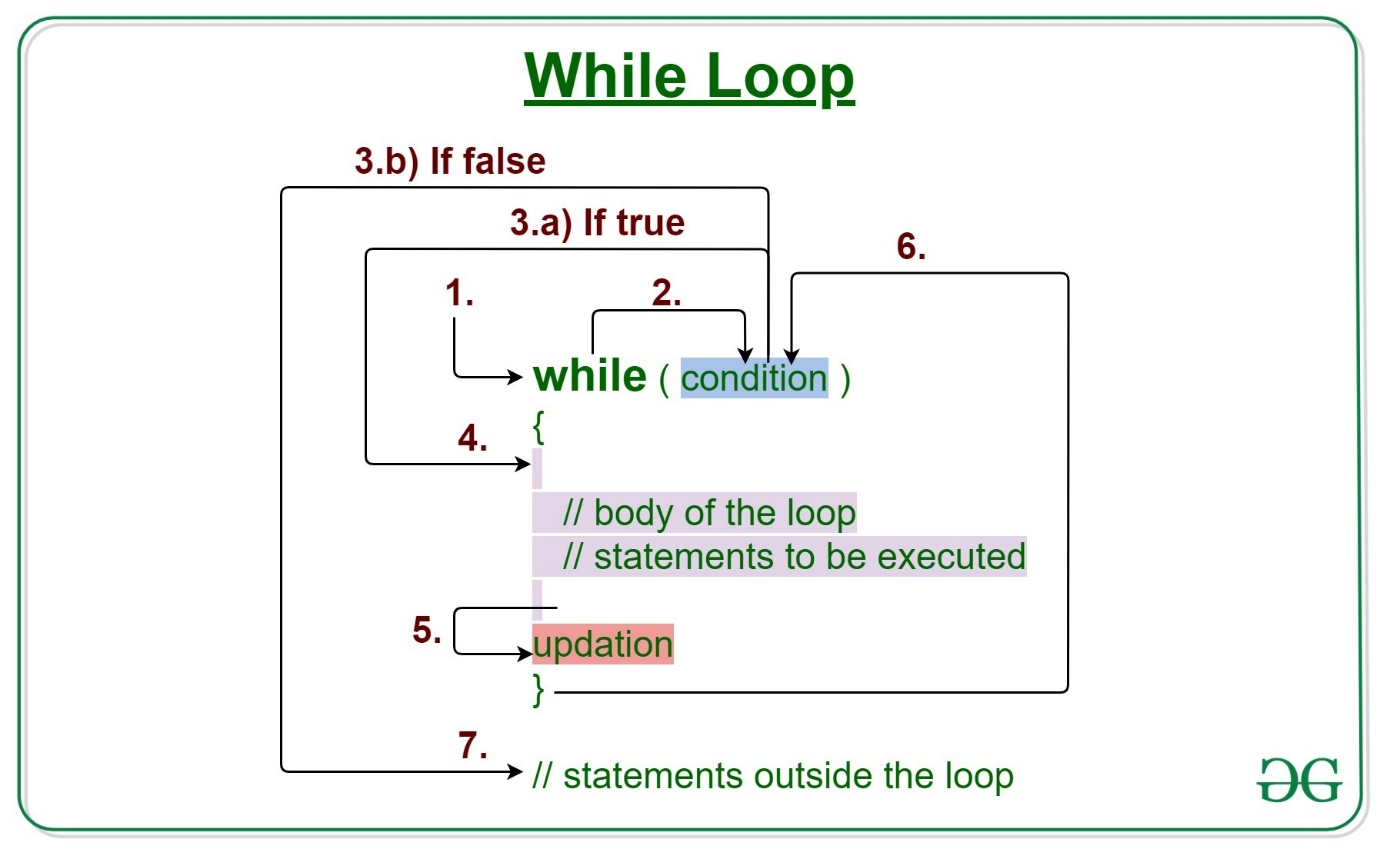
1. Control falls into the for loop. Initialization is done
2. The flow jumps to Condition
3. Condition is tested.
   1. If Condition yields true, the flow goes into the Body
   2. If Condition yields false, the flow goes outside the loop
4. The statements inside the body of the loop get executed.
5. The flow goes to the Updation
6. Updation takes place and the flow goes to Step 3 again
7. The for loop has ended and the flow has gone outside.

**Flow chart for loop (For Control Flow):** 



8) while loop:

The while loop is considered as a repeating if statement. If the number of iterations is not fixed, it is recommended to use the while loop.



**Syntax:**

while (test\_expression)

{

// statements

update\_expression;

}

The various **parts of the While loop** are:

**1. Test Expression:** In this expression, we have to test the condition. If the condition evaluates to true then we will execute the body of the loop and go to update expression. Otherwise, we will exit from the while loop.

**Example:**

i <= 10

**2. Update Expression**: After executing the loop body, this expression increments/decrements the loop variable by some value.

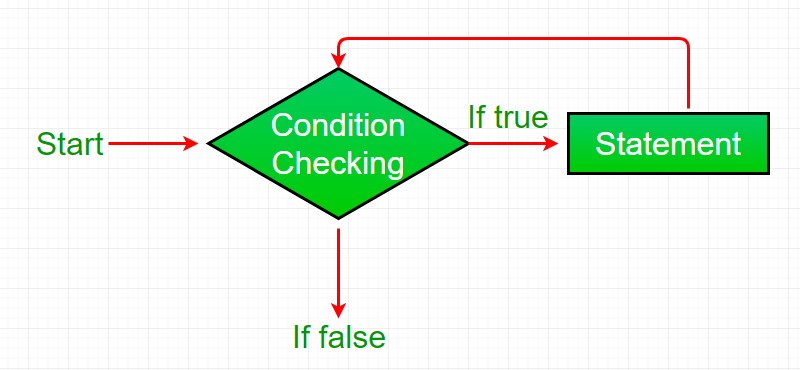
**Example:**

i++;

**How Does a While loop execute?**

1. Control falls into the while loop.
2. The flow jumps to Condition
3. Condition is tested.
   * If Condition yields true, the flow goes into the Body.
   * If Condition yields false, the flow goes outside the loop
4. The statements inside the body of the loop get executed.
5. Updation takes place.
6. Control flows back to Step 2.
7. The while loop has ended and the flow has gone outside.

**Flowchart For while loop (Control Flow):** 



9) do while loop:

The **do-while loop** is an **Exit control loop**. Therefore, unlike ***for*** or ***while*** loop, a do-while check for the condition after executing the statements of the loop body.

**Syntax:**

do

{

// Loop Body

Update\_expression

}

// Condition check

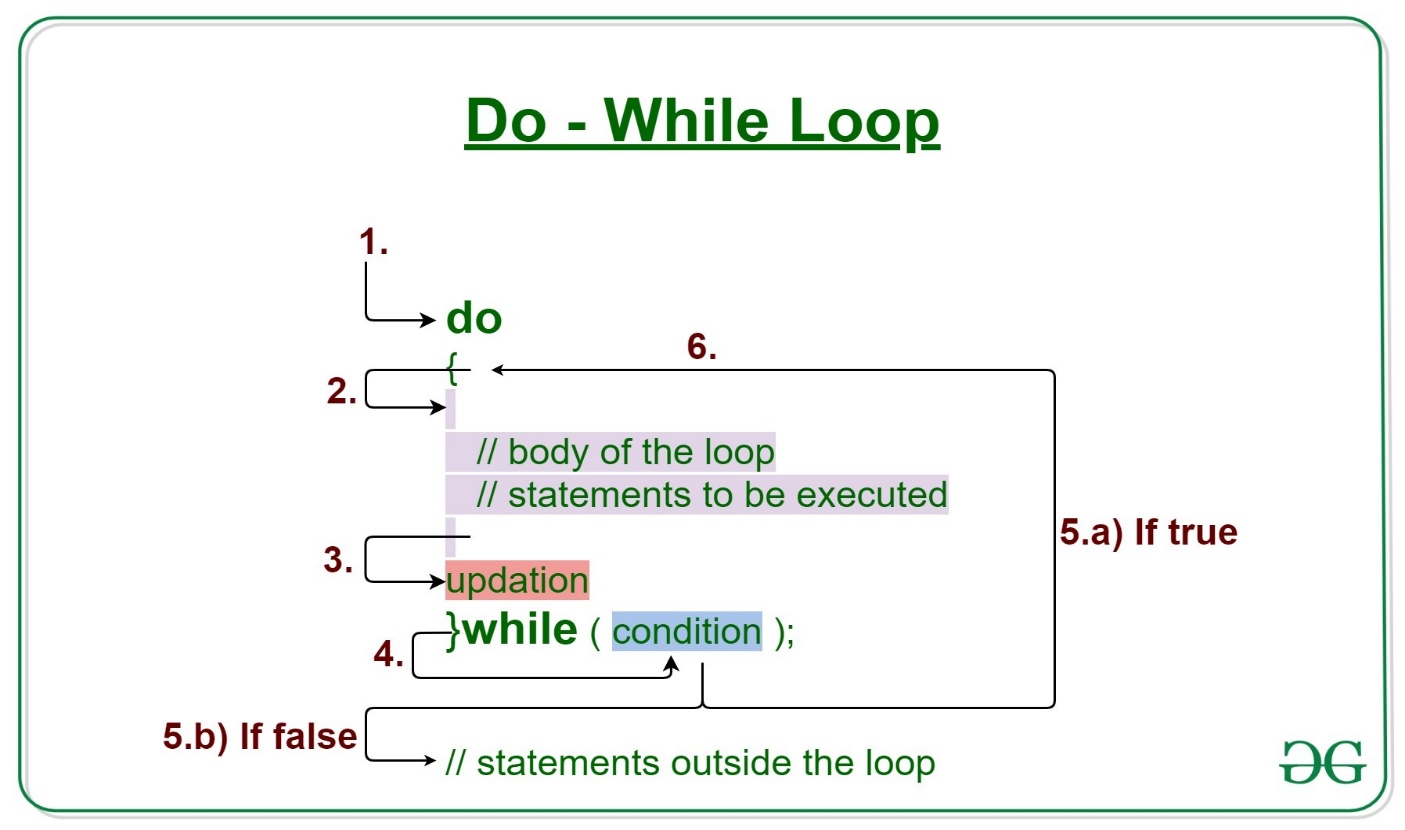
while (test\_expression);

***Note:****The****test\_expression****for the do-while loop must return a****boolean****value , else we would get compile-time error.*

**Application of do-while :** Its example application is showing some kind of menu to the users.

For example:

You are implementing a game where you show some options to the user, press 1 to do this .., press 2 to do this .. etc and press ‘Q’ to quit the game. So here you want to show the game menu to the user at least once, so you write the code for the game menu inside the do-while loop.



**Components of do-while Loop**

**A. Test Expression:** In this expression, we have to test the condition. If the condition evaluates to true then we will execute the body of the loop and go to update expression. Otherwise, we will exit from the while loop. For example:

i <= 10

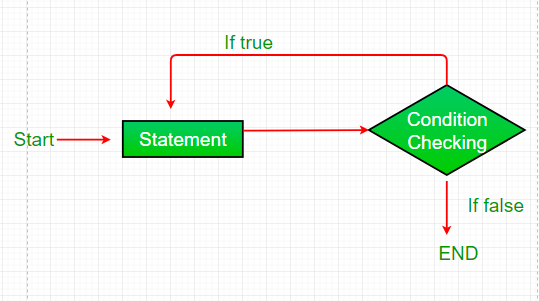
**B. Update Expression**: After executing the loop body, this expression increments/decrements the loop variable by some value. For example:

i++;

**Execution of do-While loop**

1. Control falls into the do-while loop.
2. The statements inside the body of the loop get executed.
3. Updation takes place.
4. The flow jumps to Condition
5. Condition is tested.
   1. If Condition yields true, go to Step 6.
   2. If Condition yields false, the flow goes outside the loop
6. The flow goes back to Step 2.

**Flowchart do-while loop:**



1. What is the Difference betweeen break and continue statements?

Ans:

| **Sr. No.** | **Key** | **Break** | **Continue** |
| --- | --- | --- | --- |
| 1 | Functionality | Break statement mainly used to terminate the enclosing loop such as while, do-while, for or switch statement wherever break is declared. | Continue statement mainly skip the rest of loop wherever continue is declared and execute the next iteration. |
| 2 | Executional flow | Break statement resumes the control of the program to the end of loop and made executional flow outside that loop. | Continue statement resumes the control of the program to the next iteration of that loop enclosing 'continue' and made executional flow inside the loop again. |
| 3 | Usage | As mentioned break is used for the termination of enclosing loop. | On other hand continue causes early execution of the next iteration of the enclosing loop. |
| 4 | Compatibility | Break statement can be used and compatible with 'switch', 'label'. | We can't use continue statement with 'switch','lablel' as it is not compatible with them. |

1. what is an array? how to declare an array in java?

Ans:

By definition, an array is a collection of data of the same type.

An array is usually declared so you can have multiple values in the same memory – unlike variables where you can only have one value in the memory.

Syntax :

With new keyword

dataType [] nameOfArray = new dataType [size];

Array literal

datatype[] nameOfArray = {value1,value2,….valuen};

1. when will you get array index out of bound exception in java?

The ArrayIndexOutOfBoundsException is one of the most common errors in Java. It occurs when a program attempts to access an invalid index in an array i.e. an index that is less than 0, or equal to or greater than the length of the array.

Since a Java array has a range of [0, array length - 1], when an attempt is made to access an index outside this range, an ArrayIndexOutOfBoundsException is thrown.

1. Define the syntax to create an object for a class?

Ans: The syntax for creating an object:

ClassName object = new ClassName();

Name conventions followed while creating a class:

Class names should be nouns, in mixed case with the first letter of each internal word capitalized. Try to keep your class names simple and descriptive. Use whole words-avoid acronyms and abbreviations (unless the abbreviation is much more widely used than the long form, such as URL or HTML).

Explain method:

A method in Java is a block of code that, when called, performs specific actions mentioned in it. For instance, if you have written instructions to draw a circle in the method, it will do that task. You can insert values or parameters into methods, and they will only be executed when called

Explain Variable:

Variables are containers for storing data values. In Java, there are different types of variables, for example: String - stores text, such as "Hello". String values are surrounded by double quotes.

1. What is variable? How will you declare a variables in java?

Ans: Variables are the basic unit of storage in a [programming language](https://www.toppr.com/guides/computer-amplitude-and-knowledge/computer-applications/programming-languages/). These variables consist of a data type, the variable name, and the value to be assigned to the variable. Unless and until the variables are declared and initialized, they cannot be used in the program.

Syntax:

*data\_type variable\_name = value;*

where,

type = Data type of the variable

identifier = Variable name

value = Data to be stored in the variable (Optional field)

1. What is string in java? Is it a data type?

Ans: String is a class in java and reference data type. String is a sequence of character and it is not a primitive data type.

1. What are the different ways to create the string object in java?

Ans: There are two ways to create a String object:

1. **By string literal** : Java String literal is created by using double quotes.

Syntax: String variable name=”value”;

1. **By new keyword** : Java String is created by using a keyword “new”.

**Syntax**: String variable name = new String(“value”);

1. What is the difference between .Equals and ==?

Ans:

|  |  |  |
| --- | --- | --- |
| **S.No.** | **== Operator** | **Equals() Method** |
| 1. | == is considered an operator in Java. | Equals() is considered as a method in Java. |
| 2. | It is majorly used to compare the reference values and objects. | It is used to compare the actual content of the object. |
| 3. | We can use the == operator with objects and primitives. | We cannot use the equals method with primitives. |
| 4. | The == operator can’t compare conflicting objects, so at that time the compiler surrenders the compile-time error. | The equals() method can compare conflicting objects utilizing the equals() method and returns “false”. |
| 5. | == operator cannot be overridden. | Equals() method and can be overridden. |