**---------------------------------------------------------------------------------------------------------------------------------------**

**1. What is Java..?**

* Java is a programing language
* Java is based on OOPS concept
* OOPS mean Object oriented programing structute
* Java is introduced by SunMicro System

**---------------------------------------------------------------------------------------------------------------------------------------**

**2. What are the datatype..?**

* String - Declare as String
* Integer - Declare as int
* Boolean - Declare as boolean
* Charactor - Declare as char
* Float - Declare as float

**---------------------------------------------------------------------------------------------------------------------------------------**

**3. How to declare the variable..?**

**DataType variableName = value;**

String name = "Java";

int age = 56;

char initial = 'A';

**---------------------------------------------------------------------------------------------------------------------------------------**

**4. How to print the variable..?**

String name="Java";

System.out.println(name);

**---------------------------------------------------------------------------------------------------------------------------------------**

**5. What is String..?**

* String is one of the Datatype
* It is used to declare the values with collection of characters
* We are declaring the String value by using double quotes(" ")

String topic = “Java”;

**---------------------------------------------------------------------------------------------------------------------------------------**

1. **What is integer..?**

* Integer is one of the datatype
* It is used to declare the value with only numbers
* We can declare the integer directly

int age = 30;

**7**. **What is boolean..?**

* Boolean is one of the datatype
* It is used to declare the value true/false
* We can declare the boolean value directly

Boolean flagTrue = true;

Boolean flagFalse = false;

**---------------------------------------------------------------------------------------------------------------------------------------**

**8. What is Character..?**

* Character is one of the datatype
* It is used to declare the value with single character
* We can declare the character value with single quote('D')

char initial = ‘A’;

**---------------------------------------------------------------------------------------------------------------------------------------**

**9. How to print any statement..?**

System.out.println("This is my name")

**---------------------------------------------------------------------------------------------------------------------------------------**

**10. How to create main method..?**

* Main method is a default method in the Java
* It is used to execute the class

public static void main (String [] args )

{

// Body of the method

}

**---------------------------------------------------------------------------------------------------------------------------------------**

**11. How to create a class..?**

We can create a class by using class keyword in lowercase.

public class ClassName

{

// Body of the class

}

**12. How to create an object for the class..?**

ClassName objectName = new ClassName();

**Example:** Created object for Seenium class

Selenium sel = new Selenium();

**---------------------------------------------------------------------------------------------------------------------------------------**

1. What are the control statement…?

* Control statement is used to control the flow of execution of the program
* There are different types of control statement
* If statement
* If else statement
* else if statement
* For loop
* While loop
* Switch

**13. Explain If condition:**

* It is one type of control statement
* It is used to check the condition if true or false
* If the condition is true mean, the body of the if block will execute.

if(condition)

{

// Body of if block

}

**---------------------------------------------------------------------------------------------------------------------------------------**

**13. Explain If condition:**

* It is one type of control statement
* It is used to check the condition if true or false
* If block gets executed once the condition is true

if(condition)

{

// Body of if block

}

**---------------------------------------------------------------------------------------------------------------------------------------**

**14. Explain If else condition:**

* It is one type of control statement
* It is used to check the condition if true or false
* If block gets executed once the condition is true
* Else block gets executed once the condition is false.

if(condition)

{

// Body of if block

}

else

{

// Body of else block

}

**---------------------------------------------------------------------------------------------------------------------------------------**

1. **What is else if statement..?**

else if statement is used to specify a new condition if the first condition is false.

if(condition)

{

// Body of if block

}

else if(condition)

{

// Body of if else block

}

else

{

// Body of else block

}

**---------------------------------------------------------------------------------------------------------------------------------------**

1. **What is for loop statement..?**

* For loop is used to perform the perticular action for repeated time.
* The Java for loop is used to iterate a part of the program several times. If the number of iteration is fixed, it is recommended to use for loop.

**Syntax**:

**for**(initialization; condition; increment/decrement)

{

//statement or code to be executed

}

for(int I=0; I<10; I++)

{

System.out.println(“Java”)

}

1. ****Initialization****: It is the initial condition which is executed once when the loop starts. Here, we can initialize the variable, or we can use an already initialized variable. It is an optional condition.
2. ****Condition****: It is the second condition which is executed each time to test the condition of the loop. It continues execution until the condition is false. It must return boolean value either true or false. It is an optional condition.
3. ****Increment/Decrement****: It increments or decrements the variable value. It is an optional condition.
4. ****Statement****: The statement of the loop is executed each time until the second condition is false.

**---------------------------------------------------------------------------------------------------------------------------------------**

1. **What is while loop statement..?**

* The [Java](https://www.javatpoint.com/java-tutorial) while loop is used to iterate a part of the [program](https://www.javatpoint.com/programs-list) repeatedly until the specified Boolean condition is true.
* As soon as the Boolean condition becomes false, the loop automatically stops.
* The while loop is considered as a repeating if statement. If the number of iteration is not fixed, it is recommended to use the while [loop](https://www.javatpoint.com/java-for-loop).

**Syntax:**

while (condition)

{

//code to be executed

Increment / decrement statement  ;

}

**Example:**

**int a=10;**

**While(a==10)**

{

System.out.println(“java”);

a++;

}

**---------------------------------------------------------------------------------------------------------------------------------------**

1. **What is break keyword..?**

* When a break statement is encountered inside a loop, the loop is immediately terminated and the program control resumes at the next statement following the loop.
* The Java break statement is used to break loop or [switch](https://www.javatpoint.com/java-switch) statement. It breaks the current flow of the program at specified condition. In case of inner loop, it breaks only inner loop.
* We can use Java break statement in all types of loops such as [for loop](https://www.javatpoint.com/java-for-loop), [while loop](https://www.javatpoint.com/java-while-loop) and [do-while loop](https://www.javatpoint.com/java-do-while-loop).

Syntax: break;

Example:

1. **for**(**int** i=1;i<=10;i++)
2. {
3. **if**(i==5)
   1. {
4. //breaking the loop
5. **break**;
6. }
7. System.out.println(i);
8. }

**---------------------------------------------------------------------------------------------------------------------------------------**

1. **What is the static method..?**

* Static method is created with the help of “**static**” keyword.

public static void tester()

{

// body of the method

}

**---------------------------------------------------------------------------------------------------------------------------------------**

1. **How to access the static method from other class or same class..?**

* We can access the static method by using it’s class name and variable name.

**Syntax**:

ClassName.methodName();

**Example:**

class Job

{

public static void main(String[]args)

{

Job.tester();

}

public static void tester()

{

// body of the method;

}

}

**---------------------------------------------------------------------------------------------------------------------------------------**

1. **What is the static variable..?**

* Static variable is created with the help of “**static**” keyword.

static String name = “tester”;

**---------------------------------------------------------------------------------------------------------------------------------------**

1. **How to access the static method from other class or same class..?**

* We can access the static method by using it’s class name and variable name.

**Syntax**:

ClassName.methodName();

**Example:**

class Job

{

static String name = ”tester”;

public static void main(String[]args)

{

Job.name;

}

}

**---------------------------------------------------------------------------------------------------------------------------------------**

1. **What is the non-static method..?**

* Non-Static method is created with the help of “**static**” keyword.

public void tester()

{

// body of the method

}

**---------------------------------------------------------------------------------------------------------------------------------------**

1. **How to access the non-static method from other class or same class..?**

* We can access the Non-static method by creating the object for the class.

**Syntax:**

ClassName objectName = new ClassName();

objectName.tester();

**Example:**

class Job

{

public static void main(String[]args)

{

Job job = new Job();

Job.tester();

}

public void tester()

{

// body of the method;

}

}

**---------------------------------------------------------------------------------------------------------------------------------------**

1. **What is the non-static variable..?**

* Non-Static variable is created with the help of “**static**” keyword.

String name = “tester”;

**---------------------------------------------------------------------------------------------------------------------------------------**

1. **How to access the non-static method from other class or same class..?**

* We can access the non-static method by creation of object for the class.

**Syntax**:

ClassName objectName = new ClassName();

objectName.tester();

**Example:**

class Job

{

static String name = ”tester”;

public static void main(String[]args)

{

Job job = new Job();

Job.name;

}

}

**---------------------------------------------------------------------------------------------------------------------------------------**

1. **What is an identifiers..?**

* identifiers in Java are symbolic names used for identification.
* They can be a class name, variable name, method name, object name.

**Rules:**

1. We should not special characters except underscore(\_)
2. Spaces should not use in identifiers
3. Number should not use at starting letter

**Class Name:**

1. First letter should start with capital letter

2. Every word should start with capital letter

**Example:** StudentsNameList

**Method Name:**

It should write based on the camel case

**Example:** studentNameList()

**Object Name:**

It should write based on the camel case

**Example:** studentNameList

**Variable Name:**

It should write based on the camel case

**Example:** studentNameList

1. **What is Error…?**

* An Error is a syntax mistake in the Java
* An error is occurs when compiling the programe.

1. **What is an Exception..?**

* An Exception is an un-expected event which is occurs at Runtime.
* The programme gets terminated once the exception occurs
* Exception is one of the Class in the Java.

1. **What are the types of Exception…?**

An Exception is classified into 2 types.

* Checkecd Exception
* Un-checked Exception

1. **Checked Exception:**

* The classes that directly inherit the Throwable class except RuntimeException and Error are known as checked **exceptions**.
* Checked exceptions are checked at compile-time.
* **Example:** IOException, SQLException

1. **Un-Checked Exception:**

* The classes that inherit the RuntimeException are known as unchecked exceptions
* Unchecked exceptions are not checked at compile-time, but they are checked at runtime.
* **Excample:** ArithmeticException, NullPointerException

1. **How can we handle an Exception…?**

We can handle the Exception in 2 ways.

1. . By using **throws** keyword
2. . By using **try, catch, finally** block
3. **By using throws keyword:**

* We can use throws keyword in the method to handle an exception

public static void method() thorows IOException

{

Thread.sleep(2000)

}

1. **By using try, catch, finally block:**

* We can use try, catch, finally block to handle an Exception mannually.

1. **try:**

* Java ****try**** block is used to enclose the code that might throw an exception. It must be used within the method.
* If an exception occurs at the particular statement in the try block, the rest of the block code will not execute. So, it is recommended not to keep the code in try block that will not throw an exception.
* Java try block must be followed by either catch or finally block.

1. **catch:**

* Catch block get’s executed once the exception is occcured in the try block
* Control gets moved from try block to catch block once the exception occurred
* We can add multiple catch block after the try block
* We should pass the Exception name as the argument in the catch block
* It is an alternative block if the exception is occurred.

1. **finally:**

* ****Java finally block**** is a block used to execute important code such as closing the connection, etc.
* Java finally block is always executed whether an exception is handled or not. Therefore, it contains all the necessary statements that need to be printed regardless of the exception occurs or not.
* finally block in Java can be used to put "****cleanup****" code such as closing a file, closing connection, etc.
* The important statements to be printed can be placed in the finally block.

**Syntax:**

try

{

// body of try block

}

catch(Exception e)

{

// body of catch block

}

finally

{

// body of finally block

}

1. **What is an Inheritance…?**

* Inheritance is nothing but once class is acquired the properties of another class
* Properties mean the variable and method of the another class
* Inheritance is done by using **extends** keyword
* We can classified the classes in inheritance into 2 types

1. Super class (parent class)
2. Sub class (child class)

* We can not inherite the final class
* We can not inherite the Non-static and private members

**Types of Inheritance:**

The inheritance is classified into differennt types

1. Single Inheritance
2. Multi-level inheritance
3. Hierarchical inheritance
4. Multiple inheritance
5. Hybrid inheritance
6. **Single Inheritance:**

* In single inheritance, subclasses inherit the features of one superclass.
* In below example, class Daughter is inherited the only one class named Mother.

class Mother

{

public static void saree()

{

System.out.println(“Saree”)

}

}

class Daughter extends Mother

{

public static void main(String [] args)

{

Saree();

}

}

1. **Multi-level Inheritance:**

* In multi-level inheritance, one class (child) inherites the properties of another class (parent) and the same class inherited by other class

class Parent

{

public static void saree()

{

System.out.println(“Saree”)

}

}

class Child extends Parent

{

public static void main(String [] args)

{

Saree();

}

}

class GrandChild extends Child

{

public static void main(String [] args)

{

Saree();

}

}

1. **Hierarchical inheritance**

* In Hierarchical Inheritance, one class serves as a superclass (base class) for more than one subclass

**class A**

{

    public void print\_A()

{

System.out.println("Class A"); \

}

}

**class B extends A**

{

    public void print\_B()

{

System.out.println("Class B");

}

}

**class C extends A**

{

    public void print\_C()

{

System.out.println("Class C");

}

}

**class D extends A**

{

     public void print\_D

{

System.out.println("Class D");

}

}

1. **Mutliple Inheritance:**

* In Multiple inheritances, one class can have more than one superclass and inherit features from all parent classes.
* Please note that Java does not support [multiple inheritances](https://www.geeksforgeeks.org/java-and-multiple-inheritance/) with classes.
* In java, we can achieve multiple inheritances only through Interfaces.

interface one

{

public void print\_geek();

}

interface two

{

public void print\_for();

}

interface three extends one, two

{

public void print\_geek();

}

class child implements three

{

@Override

public void print\_geek()

{

System.out.println("Geeks");

}

public void print\_for()

{

System.out.println("for");

}

}

1. **Hybrid Inheritance:**

* Hybrid Inheritance(Through Interfaces): It is a mix of two or more of the above types of inheritance.
* Since java doesn’t support multiple inheritances with classes, hybrid inheritance is also not possible with classes.
* In java, we can achieve hybrid inheritance only through [Interfaces](https://www.geeksforgeeks.org/interfaces-in-java/).

**Important terminology used in Inheritance:**

**Default superclass:** Except Object class, which has no superclass, every class has one and only one direct superclass (single inheritance). In the absence of any other explicit superclass, every class is implicitly a subclass of the Object class.

**Superclass can only be one:** A superclass can have any number of subclasses. But a subclass can have only one superclass. This is because Java does not support multiple inheritances with classes. Although with interfaces, multiple inheritances are supported by java.

**Inheriting Constructors:** A subclass inherits all the members (fields, methods, and nested classes) from its superclass. Constructors are not members, so they are not inherited by subclasses, but the constructor of the superclass can be invoked from the subclass.

**Private member inheritance:** A subclass does not inherit the private members of its parent class. However, if the superclass has public or protected methods(like getters and setters) for accessing its private fields, these can also be used by the subclass.