1. What is Java?

Answer : Java is a high-level, object-oriented programming language that was designed with the principle

of "Write Once, Run Anywhere" (WORA) in mind. Everything in Java is associated with classes and

objects, along with its attributes and methods.

It is used to develop mobile applications, desktop applications, web applications, web servers,

games, database connection and more.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

2. What are the main features of Java?

Answer: Java features include:

-- Simple and easy to learn

==> Java is quite simple to understand and the syntax.

-- Platform Independent

==> Java is platform independent means we can run the same program in any software and

hardware and will get the same result.

-- Object-oriented

==> Java is an object-oriented language that supports the concepts of class, objects, four pillars of OOPS, etc.

-- Secure

==> As we can directly share an application with the user without sharing the actual program makes

Java a secure language.

-- Multithreaded

==> deal with multiple tasks at once by defining multiple threads.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

3. What is the difference between JDK, JRE, and JVM?

Answer: JDK is the Java Development Kit, JRE is the Java Runtime Environment, and JVM is the Java

Virtual Machine. JDK includes tools for developing Java applications, JRE provides the runtime

environment for running Java applications, and JVM executes Java bytecode.

===>> JDK (Java Development Kit):

The JDK is a software development kit provided by Oracle (previously

Sun Microsystems) that includes tools and libraries necessary for Java

application development.

The JDK is used by developers to write, compile, and package Java applications.

It includes the JRE for running Java programs during development.

JDK contains the following components:

\* Java Compiler (javac):

\* Java Runtime Environment (JRE):

\* Development Tools:

===>> JRE (Java Runtime Environment):

The JRE is a subset of the JDK and is required for running Java programs during development.

JRE consists of the following components:

\*Java Virtual Machine (JVM):

\*Core Java Class Libraries:

===>> JVM (Java Virtual Machine):

The JVM is an essential component of both the JDK and the JRE. It is responsible for

executing Java bytecode. WHICH TAKES BYTECODE LANGUAGE INTO ASSEMBLY LANGUAGE.

Its main tasks include:

\* Loading and Verifying:

\* Just-in-Time (JIT) Compilation:

\* Memory Management:

\* Runtime Environment:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

4. Explain the concept of Object-Oriented Programming (OOP) in Java.

Answer: OOP is a programming paradigm based on the concept of "objects." In Java, everything is

treated as an object, and programs are designed using classes and objects. Key principles include

encapsulation, inheritance, and polymorphism.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

5. What is the difference between interface and abstract class in Java?

Answer: Both interface and abstract class are used for abstraction, but interfaces only declare methods

without providing any implementation, while abstract classes can have both abstract and concrete methods.

An Abstract class is a class that is declared with keyword abstract - it may or may not include abstract method.

Abstract classes cannot be instantiated, but they can be subclassed.

An Abstract method is a method that is declared without an implementation (without braces, and followed

by a semicolon)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

6. What is the significance of the final keyword in Java?

Answer: The final keyword can be applied to variables, methods, and classes. It indicates that the

variable cannot be changed, a method cannot be overridden, or a class cannot be extended,

depending on where it is used.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

7. How does exception handling work in Java?

Answer: Exception handling in Java is done through the use of try, catch, finally, and throw keywords.

Code that might throw an exception is placed within the try block, and corresponding exception

handling is implemented in the catch block.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

8. What is the difference between == and .equals() in Java?

Answer: == is used to compare object references, while .equals() is used to compare the content or values

of objects. In other words, == checks for reference equality, and .equals() checks for content equality.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

9. What is a ClassLoader?

Answer: A classloader in Java is a subsystem of Java Virtual Machine, dedicated to loading class files

when a program is executed; ClassLoader is the first to load the executable file.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

10. What are the Memory Allocations available in JavaJava?

Answer: Java has five significant types of memory allocations.

Class Memory

Heap Memory

Stack Memory

Program Counter-Memory

Native Method Stack Memory

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

11. Will the program run if we write static public void main?

Answer: Yes, the program will successfully execute if written so. Because, in Java, there is no specific

rule for the order of specifiers

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

11. What is an Association?

Answer: An Association can be defined as a relationship that has no ownership over another. For example,

a person can be associated with multiple banks, and a bank can be related to various people, but no

one can own the other.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

12. What is a Marker Interface?

Answer: An empty interface in Java is referred to as a Marker interface. Serializable and Cloneable

are some famous examples of Marker Interface.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

13. What is Object Cloning?

Answer: An ability to recreate an object entirely similar to an existing object is known as Object

Cloning in Java. Java provides a clone() method to clone a current object offering the same

functionality as the original object.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

14. What is the constructor?

Answer: Constructor in java is used to create the instance of the class. Constructors are almost similar

to methods except for two things - its name is the same as the class name and it has no return type.

Sometimes constructors are also referred to as special methods to initialize an object.

Types of Contructor : Based on the arguments there are 3 type of Contructor.

i. No arguments Contructor : It is a Contructor that doesn't any arguments or parameters.

ii. Parameterized Contructor : It is a Contructor that conatins arguments or parameters.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

15. What is a class variable?

Answer: A class variable in programming refers to a variable that is associated with a class rather

than with instances (objects) of the class. It is shared among all instances of the class and is

defined within the class but outside any method. Class variables are often used to store data

that is common to all instances of the class.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

16. Define encapsulation?

Answer: Binding data and code together into a single unit are called encapsulation. The capsule

is the best example of encapsulation.

Wrapping data and methods into a single unit (class) to control access and protect the integrity of the data.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

17. What is array? and what are the different types of array?

Answer: An array is a container obeject that holds a fixed number of values of a single type. The

lenght of the array is established when the array is created. After creation, its lenght is fixed.

The main types of arrays in Java are:

1. Single-Dimensional Array:

==> It is a linear collection of elements where each element is accessed using a single index.

Exa : int[] numbers = new int[5];

2. Multidimensional Array:

==> A multidimensional array is an array of arrays. It can have two or more dimensions.

2D Array: A table of elements with rows and columns.

Exa : int[][] matrix = new int[3][4];

3D Array: A cube of elements with multiple layers.

Exa : int[][][] cube = new int[3][4][2];

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

18. What is Class, Method and Object?

Answer: A class is a user-defined type that describes what a certain type of object will look like.

A class description consists of a declaration and a definition.

\* A class in java is a blueprint or a template for creating objects.

\* It defines the structure and behavior of objects that belongs to that class.

--->> 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.

It is also called as functions.

--->> An object is a single instance of a class. You can create many objects from the same class type.

The object is the actual component of programs.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

19. What are the Data types in java?

Answer: Data types are divided into two groups:

-- A primitive data type specifies the size and type of variable values, and it has no additional methods.

There are eight primitive data types in Java:

> Primitive data types - includes byte, short, int, long, float, double, boolean and char

-- Non-primitive data types are called reference types because they refer to objects.

> Non-primitive data types - such as String, Arrays and Classes

\* The main difference between primitive and non-primitive data types are:

i. Primitive types are predefined (already defined) in Java. Non-primitive types are created by the

programmer and is not defined by Java (except for String).

ii. Non-primitive types can be used to call methods to perform certain operations, while primitive types cannot.

iii. A primitive type has always a value, while non-primitive types can be null.

iv. A primitive type starts with a lowercase letter, while non-primitive types starts with an uppercase letter.

v. The size of a primitive type depends on the data type, while non-primitive types have all the same size.