Core Java

Interview Questions

Answers

Created by Foram_Chudasama

1. What is Java?

Java is a popular object-oriented programming language. It is defined as a complete collection of objects. By using Java, we can develop lots of

applications such as gaming, mobile apps, and websites.

2. Explain in brief the history of Java?

In the year 1991, a small group of engineers called ‘Green Team’ led by James Gosling, worked a lot and introduced a new programming language called “Java”. This language is created in such a way that it is going to revolutionize the world.

In today’s World, Java is not only invading the internet, but also it is an invisible force behind many of the operations, devices, and applications.

3. What is the difference between C and Java?

The differences between C and Java are as follows:

C- Language	Java
C language was developed by Dennis M. Ritchie in the year 1972.	Java was developed by James Gosling in the year 1995.
C is Procedural-Oriented	Java is Object-Oriented
Functions play a major role in the C language.	Objects play a major role in the Java language.

It is a middle-level language	It is a high-level language.
C language does not support OOPS Concepts.	Java supports OOPS concepts, in which Inheritance the main property used for code reusability.
In C, memory is allocated using Malloc	In Java, memory is allocated using a new keyword.
The threading concept is not supported by C	Java supports Threading
It is not portable	It is portable
Default members of C are public	Default members of Java are private
Storage classes are supported by C	Storage classes are not supported in Java

4. What is the reason why Java came into existence?

Java is the first programming language that is used to write code on a virtual machine, so that is the reason why it is called JVM (Java Virtual Machine). This JVM is a new concept that is introduced in Java. It also provides a new feature called code reusability which is not possible in C.

5. Compare both C++ and Java?

Java	**C++**
Java is platform-independent	C++ is platform dependent

Goto statement is not supported by Java	C++ supports the goto statement
Multiple inheritances are supported in java by using interfaces	Multiple inheritances are supported in C++
Java doesn't support structures and unions	C++ does support unions and structures
Java has built-in support for threading	C++ does not provide built-in support for threading
Java is used for building applications	C++ is used for system programming
Java uses both interpreter and compiler	C++ uses compiler only

6. What are the prominent features of Java?

The following are the notable features in Java:

- **Dynamic:** Java is more dynamic when compared to C++ and C. Java is designed in such a way that it is adaptable to any the evolving environments.
- **Simple:** Java is very easy to learn and code.
- **Object-oriented:** In Java, everything is based on objects.
- **Secure:** Java provides a secure platform to develop safe and virus-free applications.
- **Platform Independent:** Unlike C and C++, when we compile Java code it is compiled into platform-independent bytecode rather than the platform-specific machine code.

- **Portable:** Java provides no implementation aspects for the specifications which make Java portable.
- **Multithreaded:** By this feature, we can perform multiple tasks simultaneously in Java.
- **High-Performance:** With the built-in Just-in-Time compiler, Java provides high performance.
- **Robust:** Java makes more efforts to eliminate errors by concentrating more on runtime checking and compile-time check.

7. What is a Class in Java?

Class is defined as a template or a blueprint that is used to create objects and also to define objects and methods.

8. Define Object in Java?

The instance of a class is called an object. Every object in Java has both state and behavior. The state of the object is stored in fields and the behavior of the objects is defined by methods.

9. How to write a basic Hello World program in Java?

```
class Mindmajix
{
public static void main(String args[ ]) {
System.out.println("Hello World");
}
}
```

10. Define JVM?

JVM is the abbreviation for Java Virtual Machine. It is a virtual machine that provides a runtime environment to write code. JVM is a part of JRE (Java Runtime Environment) and is used to convert the bytecode into machine-level language. This machine is responsible for allocating memory.

Java Interview Questions for Freshers

11. Name the memory areas that are allocated by JVM and explain the class loader in JVM?

There are totally five memory areas that are allocated by the JVM, and they are:

- Class Area
- Heap
- Stack
- Native Method Stack
- PC Register

ClassLoader: class loader is a subschema of JVM which is used to load class files. Whenever we run Java programs, the data will be first loaded from the classloader. There are mainly three in-built classloaders in JVM, they are:

- Application Classloader
- Bootstrap Classloader
- Extension Classloader

12. What is JDK?

Java Development Kit is one of the three prominent technology packages used in Java programming. JDK is used as a standalone component to run the Java programs by JVM and JRE. This kit is used to implement Java platform specifications, including class libraries and compiler.

13. What do you mean by JRE?

Java Runtime Environment (JRE) is a collection of software tools that are designed for the development of Java applications. This is a part of JDK, but it can be downloaded separately. JRE is mainly responsible for orchestrating the component activities.

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

We can understand the difference between JVM, JDK, and JRE by the following diagram:

- **JVM:** Java Virtual Machine is the main part of Java programming, which provides platform independence. JRE and JDK both contain JVM in order to run our Java programs.
- **JDK:** This development kit is mainly used for developing programs.
- **JRE:** Java Runtime Environment is mainly used for running Java programs.

15. Define the JIT compiler in Java?

Just In Time Compiler is the component of JRE, which is used to compile the bytecodes of the particular method into the native machine code. This compiled code of the method is directly called by JVM without interpreting it

16. Define variables in Java and explain with example?

Variables in Java can be defined as a basic storage unit of a program. It is a storage unit that holds the value during the program execution. Always the variable is assigned with a datatype.

For Example:

```
int a = 10;
```

17. What are the different types of variables in Java?

There are mainly three different types of variables available in Java, and they are:

- Static Variables
- Local Variables
- Instance Variables

Static Variables: A variable that is declared with the static keyword is called a static variable. A static variable cannot be a local variable, and the memory is allocated only once for these variables.

Local Variables: A variable that is declared inside the body of the method within the class is called a local variable. A local variable cannot be declared using the static keyword.

Instance Variables: The variable declared inside the class but outside the body of the method is called the instance variable. This variable cannot be declared as static and its value is instance-specific and cannot be shared among others.

Example:

```
class A{
int num=30;//instance variable
static char name=pranaya;//static variable
void method(){
int n=90;//local variable
}
}//end of class
```

18. What is Typecasting?

Typecasting in Java is done explicitly by the programmer; this is done to convert one data type into another data type.

Widening (automatically) - conversion of a smaller data type to a larger data type size.

```
byte -> short -> char -> int -> long -> float -> double
```

Narrowing (manually) - converting a larger type to a smaller size type

```
double -> float -> long -> int -> char -> short -> byte
```

19. What is Type Conversion?

Type conversion can be defined as converting one data type to another data type automatically by the compiler.

There are two types of type conversions, and they are:

- Implicit type conversion
- Explicit type conversion.

20. What are the data types in Java?

Datatypes in Java specify the values and sizes that can be stored in the variables. There are mainly two types of data types; they are:

- Primitive Data Types
- Non-primitive Data Types

21. What are primitive data types?

Primitive data types in Java are the major constituents of data manipulation. These are the most basic data types that are available in Java. The primitive data types include int, char, byte, float, double, long, short, and boolean.

22. What are the default values and sizes of primitive data types?

Data Type	Default Size	Default Value
int	4 bytes	0
char	2 bytes	‘u0000’
byte	1 byte	0
byte	2 bytes	0
long	8 bytes	0L
float	4 bytes	0.0f
double	8 bytes	0.0d
Boolean	1 bit	false

23. What are non-primitive data types?

The non-primitive data types are something that is different from primitive data types, and these non-primitive data types include String, arrays, and structures.

24. Why char uses 2 bytes in Java and what is this ‘u0000’ notation called?

This is only due to the reason that Java uses the Unicode system. The notation ‘u0000’ is the lowest range of the Unicode system, and it is the default value of the char data type.

25. Write the syntax for object declaration in Java?

The new keyword is used to create an object.

For example:

```
Mindmajix m1= new Mindmajix();// create an object for Mindmajix.
```

26. What is the Unicode system?

Unicode is a Universal International Standard Character Encoding which is an adequate resource for representing most of the languages written worldwide.

27. Why Java uses Unicode System?

To overcome the problems present in the previous language standards, the Unicode system has been introduced. Java uses the Unicode system because the character default size provided by Unicode is 2 bytes and Java also needs only 2 bytes for the character.

28. What makes Java ‘Run Anywhere’ in nature?

It is because the Java compiler converts the code into byte code which is the main transitional language between machine code and source code. This byte code is not platform-dependent so that it can be compiled and executed on any platform.

29. Is exit, null, delete, main, and next are the keywords in Java?

No, they are no such keywords in Java.

30. What if we write public static void as static public void?

Both the compilation and execution of the programs are done correctly because in Java specifiers order doesn't matter.

31. Is there any default value for local variables?

No, local variables are not initialized by any default values.

32. How many types of operators are available in Java?

Eight types of operators are available in java, and they are:

1. Arithmetic operators
2. Assignment operators
3. Logical operators
4. Relational operators
5. Bitwise operators
6. Unary operators
7. Ternary operators
8. Shift operators

33. What is operator precedence in Java?

Java provides a set of rules and regulations for particularly specifying the order in which operators are evaluated. If the expression has many numbers of operators then the operator precedence comes into action. This operator precedence evaluates the operators present in the expressions based on the priority. For example, multiplication has the highest priority when compared to addition and subtraction.

34. What are the different logical operators in Java?

Operator	Description
Logical NOT	This operator can be used with logical expressions, boolean type variables, and relational variables, The Logical NOT operator is denoted by the symbol "!"
Logical OR	These logical operators operate only on boolean variable types and the symbol for this operator is "||"
Logical AND	We can combine many relational operations using this operator and the output will be of boolean type. The "&&" is the symbol for logical AND

35. What is the role of the unary operator in Java?

This type of operator has only one operand and is mainly used to perform various operations including negating an expression, either incrementing/ decrementing the value by one, and invention on boolean values.

An example for the unary operator is given below:

```
class UnaryExample{
public static void main(String args[]){
int x=15;
System.out.println(x++);//15 (16)
System.out.println(++x);//17
System.out.println(x--);//12 (16)
System.out.println(--x);//15
}}
```

Java Interview Questions for Experienced

36. What is the difference between ++a and a++ increment operators?

++a is a prefix increment and a++ is the postfix increment. The prefix increment is used to return the value after incrementing the present value. Whereas in postfix increment, the value is returned before incrementing it.

37. Define left shift and right shift operators in Java?

Left Shift: This left shift is a bitwise operator in which bits are moved towards the left side and zeros are placed at the rightmost places.

Example:

```
public class LeftShiftOperator {
```

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

                                int a=2;//
                                int i;
                                i=a<<1;//4
                                System.out.println("the value of a before left
shift is: " +a);
                                System.out.println("the value of a after
applying left shift is: " +i);

        }

}
```

Output: 4

Right Shift: It is also of the bitwise operator in which bits are moved towards the right-hand side and zeros are places at the leftmost places.

Example:

```
public class RightShiftOperator {

        public static void main(String[] args) {
                int a=2;
                int i;
```

```
                i=a>>1;
                System.out.println("the value of a before right shfit is: " +a);
                System.out.println("the value of a after applying right shfit is: "
+i);
  }

}
```

Output: 1

38. What are the bitwise operators in Java?

Bitwise operators are mainly used to work on bits and these operators continue to work on bit-by-bit operations.

The following are the bitwise operators in Java, and they are:

- Bitwise OR (A&B)
- Bitwise AND (A|B)
- Bitwise XOR (A^B)
- Bitwise Complement (~A)
- Left shift (A<<2)
- Right shift (A>>2)
- Unsigned left shift (<<<)
- Unsigned right shift (>>>)

39. What is a ternary operator?

Ternary operator in Java is used to replace the if-else statement. The representation or the syntax for the ternary operator is given as:

```
variable= (expression) ? expression true : expression false
```

40. Is the Empty .java file is a valid source file name?

Yes, Java allows us to save our Java file by .java only, we need to compile it by javac.java and run it by Java class name.

For example:

```
//save by .java only
class A{
public static void main(String args[]){
System.out.println("Hello Mindmajix");
}
}
```

41. What are Java keywords?

Java keywords are also called “Reserved keywords” that act as a key to a code. Keywords in Java are predefined that cannot be used as an object name or variable. There are many keywords in Java, and some of them are:

- abstract
- default
- new
- This
- static
- Extends and many more

42. What are various access specifiers present in Java?

There are four access specifiers present in Java, and they are:

1. **Public:** The methods, classes, and variables that are defined as the public can be accessed by any class or method.
2. **Private:** The methods or classes which are declared as private can be accessible within the same class only.
3. **Protected:** The variables, methods, and classes which are defined as private can be accessed within the same class of the same package or by the subclass of the same class.
4. **Default:** By default, all the classes, variables, and methods are of default scope. The default is accessible within the package only.

43. What are the advantages of packages in Java?

The following are the advantages of packages in Java, and they are:

- Name clashes are avoided using packages
- Packages enable easier access control
- Packages provide an effective and easier way to locate the related classes.

44. What is the output of the following program?

The program is as follows:

```
class Java
{
   public static void main (String args[])
   {
      System.out.println(10 * 50 + "Mindmajix");
      System.out.println("Mindmajix" + 10 * 50);
```

```
    }
}
```

Output:

```
500Mindmajix
Mindmajix500
```

45. List out the control statements in Java?

In Java control statements are divided into three types. They are:

- Selection Statements
- Iterative/looping Statements
- Jump Statements.

46. What are the selection statements in Java?

A selection statement is mainly used to transfer program control to a specific flow based upon the condition either true or false. These selection statements are also called conditional statements.

Selection/Conditional statements in Java include:

- If statement
- If-else statement
- Switch statements

47. What are the different types of iterative statements?

The iterative statements in Java are also called looping statements, these statements are the set of statements that repeat continuously until the condition for the termination is not met.

Looping/iterative statements in Java include:

- For loop
- While loop
- Do-while loop

48. What are the jump statements in Java?

In Java, jump statements are mainly used to transfer control to another part of our program depending on the condition. Moreover, these statements are used to jump directly to other statements.

- Break and Continue are the two jump statements present in Java.

49. Define for each loop in Java?

For-each is another kind of array traversing technique in Java which is the same as that of for loop, while loop. It is most commonly used to iterate over a collection or an array such as ArrayList. An example for for-each loop is as follows:

```
class ForEachPro{
public static void main(String args[]){
  //declaring an array
int arr[]={12,13,14,44};
  //traversing the array with for-each loop
 for(int i:arr){
 System.out.println(i);
  }
 }
```

```
}
```

Output:

```
12

12

14

44
```

50. What is the difference between a while loop and a do-while loop

In the case of a while loop the condition is tested first and then if the condition is true then the loop continues if not it stops the execution. Whereas in the case of the do-while loop first the condition is executed and at the end of the loop, the condition is tested.

The syntax for the while loop is as follows:

```
while(condition){

//code to be executed

}
```

51. What are the comments in Java?

Java comments are statements that are not executed by the interpreter and compiler. These are used to provide information about the class, variables, methods, and any statements. Comments are mainly used to hide program code for a specific time.

There are mainly three types of comments in Java, and they are:

- Documentation comment

- Multi-line comment
- Single-line comment

52. Describe in brief OOPs concepts?

OOPs is an abbreviation for Object-Oriented Programming Language which entirely deals with an object. The main aim of OOPs is to implement real-world entities such as objects, classes, inheritance, polymorphism, and so on. Simula is considered the first object-oriented programming language. The most popular programming languages are Java, Python, PHP, C++, and many others.

The following are the OOPs concepts that are included in Java:

- Class
- Object
- Abstraction
- Encapsulation
- Inheritance
- Polymorphism

53. What is Abstraction?

In simple words, abstraction can be defined as hiding unnecessary data and showing or executing necessary data. In technical terms, abstraction can be defined as hiding internal processes and showing only the functionality.

54. Define encapsulation?

Binding data and code together into a single unit are called encapsulation. The capsule is the best example of encapsulation.

55. What is Inheritance in Java?

When an object of child class has the ability to acquire the properties of a parent class then it is called inheritance. It is mainly used to acquire runtime polymorphism and also it provides code reusability.

56. What is Polymorphism in Java?

Polymorphism in Java provides a way to perform one task in different possible ways. To achieve polymorphism in Java we use method overloading and method overriding. For example, the shape is the task and various possible ways in shapes are triangles, rectangle, circle, and so on.

57. What are the advantages of OOPs?

The following are the advantages of OOPs, and they are:

- OOPs provides data hiding
- OOPs makes development and maintenance easier when compared to a procedural programming language.
- OOPs has the ability to stimulate real-world entities more effectively.

58. What is the major difference between object-oriented language and object-based language?

An object-based programming language provides the most effective way to follow all the features of OOPs concepts except inheritance. VBScript and JavaScript are examples of Object-based programming languages. Whereas Object-oriented programming language supports all the features of OOPs concepts and examples of Object-oriented programming language is Java, Python, and so on.

59. What will be an initial value for an object reference that is defined as an instance variable?

All the object references in Java are initialized to null.

60. What is the Object-oriented paradigm?

It is a paradigm entirely based on objects having defined methods in the class to which it belongs. This is mainly used to incorporate the advantages of reusability and modularity. Objects are defined as instances of classes that interact with one another to design programs and applications. The features of the object-oriented paradigm are as follows:

- Mostly focus on data containing methods to operate on object's data
- Build enhanced real-time approaches such as inheritance, abstraction, and so on
- It follows the bottom-up approach in the program design.

61. What are Java naming conventions?

Java naming convention is a rule to be followed for naming your identifiers such as package, variable, method, and constants. These conventions are supported by many Java communities such as Netscape and Sun Microsystems and mostly all the fields of java programming are given according to Java naming conventions.

62. What are the rules we need to follow to declare a class?

The rules which we need to follow to declare a class are as follows:

- It should start with an uppercase letter.
- The class name must be a noun such as Thread, Class, Java, and so on.

An example for the class is,

```
public class Thread
```

```
{
//code snippet
}
```

63. Define Constructor in Java?

A constructor is a special type of method with a block of code to initialize the state of an object. A constructor is called only when the instance of the object is created. Every time in Java object is created using the new keyword and by default, the constructor is called.

64. What all the rules must be followed while creating a constructor in Java?

The rules we need to follow while creating a constructor are:

- The constructor should have the same name as its class name.
- There is no explicit return type for a constructor.
- In Java, a constructor cannot be synchronized, abstract, final, and static.

65. What are the different types of constructors available in Java?

There are two constructors in Java, and they are:

- Default constructor
- Parameterized constructor

Default constructor: It is also called a no-argument constructor and it is mainly used to initialize the instance variable with the default value. Moreover, it is also used to perform some useful tasks on object creation. This default constructor is implicitly invoked by the compiler if there is no constructor for a particular class.

Parameterized constructor: A parameterized constructor is one type of constructor which is mainly used to initialize the instance variables with the given values. In simple words, the constructor that accepts arguments is called a parameterized constructor.

66. Give examples for both a default constructor and a parameterized constructor?

An example of a default constructor is as follows:

```
//Java Program to create and call a default constructor
class Mindmajix1{
//creating a default constructor
Mindmajix1()
{
System.out.println("Welcome to Mindmajix");
}
//main method
public static void main(String args[]){
//calling a default constructor
Mindmajix1 m=new Mindmajix1();
}
}
```

Output: Welcome to Mindmajix

An example of the parameterized constructor is as follows:

```
//Java Program to demonstrate the use of the parameterized constructor.
class Training{
    int id;
    String name;
    //creating a parameterized constructor
    Training(int i,String n){
   id = i;
    name = n;
    }
    //method to display the values
    void display()
{
System.out.println(id+" "+name);
}
public static void main(String args[]){
//creating objects and passing values
Training t1 = new Training(111,"DevOps");
    Student4 s2 = new Student4(222,"Oracle");
    //calling method to display the values of object
    t1.display();
    t2.display();
   }
}
```

Output:

```
111 DevOps
222 Oracle
```

67. Does the constructor return any value?

Yes, the constructor will return the current or present instance of the class.

68. Can the constructor be inherited?

No, the constructor cannot be inherited.

69. Can the constructor be overloaded?

Yes, It is possible to overload a constructor by changing the number of arguments for each constructor in a particular program or it is possible to overload a constructor by changing the parameter data types.

70. Can we declare a constructor as final?

No, We can not declare the constructor as final, if we declare it as final compiler throws a “modified final not allowed” error.

71. Is there any Constructor class available in Java and what is its purpose of it?

Yes, there is a class called Constructor class available in Java. The purpose of the Constructor class is to get the internal information of the constructor in the class. It is present in java.lang.reflect package.

72. What is the use of a copy constructor in Java?

There is no copy constructor in Java, we can copy the values from one object to another same as that of a copy constructor in C++. In Java, there are many ways to copy the values of one object to another and they are:

- By using Clone() of object class
- By using constructor
- By assigning the value of one object to another

73. Define the method in Java?

In Java method is defined as a set of code that is represented by a name and can be invoked at any point in a program with the help of the method name. Each and every method in the program has its own name which is not the same as that of a class name.

Java Technical Interview Questions

74. Difference between constructor and method?

The difference between constructor and method are as follows:

Constructor	Method
The constructor should have the same name as the class name	The method name is not the same as that of the class name
A constructor has no return type	The method must have a return type
It can be invoked implicitly	It can be invoked explicitly

The constructor is used to initialize the state of the object	A method is used to expose the behavior of an object
Default constructor can be provided by the compiler	The default method is not provided by the Java compiler

75. What is the method signature in Java?

In Java method signature is given the specified format followed by the method name, type, and order of its parameters. Exceptions are not considered as a part of the method signature.

```
return-type  method name (parameter list)
{
//code
}
```

76. What is the role of static keywords in Java?

Keyword static in Java is mainly used for memory management and we can declare block, variable, method, and nested class as static.

77. What is a static variable?

In Java, we can declare a variable as static, if we declare any variable as static the following can be done, and they are:

- A static variable can be mainly used to refer to all the common properties of objects.
- Memory for the static variable is assigned only once in the class area at the time of class loading

Example for static variable:

```
//Program of static variable
 class Mindmajix1{
  int ID;
  String employee name;
  static String office ="Appmajix";

 Mindmajix1(int i,String n){
  ID = i;
  employee name = n;
  }
 void display ()
{
System.out.println(ID+" "+name+" "+office);}
 public static void main(String args[]){
Mindmajix1 m1 = new Mindmajix1(346,"Pranaya");
Mindmajix1 m2 = new Mindmajix1(222,"Lilly");

 m1.display();
 m2.display();
 }
}
```

Output:

346 Pranaya Appmajix

222 Lilly Appmajix

78. What happens if we declare a method as static?

If we declare a method as static, the following operations take place, and they are:

- A static method can be invoked without creating an instance of the class.
- Only a static method can access static data members and can change the value of a particular data member.
- In most of the cases static method belongs to the class rather than the object of the class.

79. What are the cons observed if we declare a method as static in Java?

The restrictions that we face if we declare a method as static are:

- A static method cannot access non-static members and also cannot call the non-static method directly.
- This and super keywords cannot be used in the context of the static method as they are non-static.

80. Why is the main method in Java is static?

The main reason is that the object is not required to call for a static method so, if we declare the main method as non-static we need to create an object first and then call the main() method. In order to save memory, we declare the main method as static in Java.

81. Can we override the static method in Java?

No, we can’t override the static method in Java.

82. What is a static block in Java?

Static block in Java is mainly used to initialize the static data members. The specialty of the static block is that it is executed before the main method at the time of class loading.

An example of the static block is as follows:

```
Class Mindmajix{
  static{System.out.println("static block");
}
  public static void main(String args[]){
   System.out.println("Hello World");
  }
}
```

Output:

```
static block
Hello World
```

83. Can we execute a program in Java without the main() method?

Yes, we can execute the program in Java without the main method using a static block. It was possible only till JDK 1.6 and from JDK 1.7 it is not possible to execute the program without the main method in Java.

84. What happens if we remove the static modifier from the signature of the main() method?

The program will be compiled, but at runtime, it throws NoSuchMethodError error.

85. Can we declare a constructor using static?

As we know that the static context is suitable for only class, variable, and method, not for the object. So the constructors are invoked only when an object is created, so there is no possibility to declare the constructor as static in Java.

86. Can we make abstract methods static in Java?

No, if we declare abstract methods as static it becomes a part of the class, and we can directly call it which is not required. Calling an undefined method is unnecessary. Therefore declaring an abstract method as static is not allowed.

87. Can we declare static methods and variables in the abstract class?

Yes, as we know that there is no need for an object to access the static block, therefore we can access static methods and variables declared inside the abstract class by using the abstract class name. Consider the following example.

```
abstract class Check
{
   static int i = 113;
   static void CheckMethod()
   {
      System.out.println("hi, how are you");
   }
}
public class CheckClass extends Check
```

```
{
   public static void main (String args[])
   {
      Check.CheckMethod();
      System.out.println("i = "+Check.i);
   }
}
```

Output:

```
hi, how are you
i = 113
```

88. Define this keyword in Java?

The main use of this keyword is to refer to the current object in Java programming.

89. What is the usage of this keyword in Java?

The following are the usage provided by this keyword in Java, and they are:

- It is used to refer to the current class instance variable.
- This is also used to invoke the current class constructor
- It is used to invoke the current class method
- It can be passed as an argument in the method call and in the constructor call

90. Can this keyword be used to refer to the current class instance variable?

Yes, this keyword is used to refer to the current class instance variable and it is also used to initiate or invoke the current class constructor.

91. Which class is the superclass for all the classes in Java?

An object class is a superclass for all the classes in Java.

92. What is a singleton class in Java?

Singleton class can be defined as the class that consists of only one single instance. In this class, all the methods and variables belong to only one instance. Singleton class is mainly used in a situation where we need to limit the objects for a class.

93. Why we use inheritance in Java?

In Java, we use inheritance mainly for two uses, and they are:

- For code reusability
- For method overriding

94. Write the syntax for inheritance in Java?

The syntax for inheritance in Java is as follows:

```
class Superclass extends subclass
{
//code
}
```

95. How can we generate random numbers in Java?

By using the Math.random() method we can generate random numbers in Java ranging from 0.1 to less than 1.0. Moreover, the random numbers can be generated by using the Random class present in java. util package.

96. Can the main() method in Java return any data?

The main() method in Java doesn’t return any data because it is declared with a void return type.

97. What are Java packages?

In Java, the package is defined as a collection of interfaces and classes that are bundled together and related to each other. Usage of packages in the program will help developers to group the code for proper re-use. In Java, packages are used by importing them into different classes.

98. What is the reason why multiple inheritances are not supported in Java?

Java doesn’t support multiple inheritances in order to reduce the complexity and simplify the language. An example of multiple inheritances in Java is given below:

```
class X{
void msg()
{
System.out.println("Hello");}
}
class Y{
void msg()
```

```
{
System.out.println("Welcome");}
}
class Z extends X,Y{//suppose if it were
public static void main(String args[]){
  Z obj=new Z();
  obj.msg();//Now which msg() method would be invoked?
}
}
```

Output:

Compile-time error

So, the above example shows that Java doesn’t support multiple inheritances.

99. Can we declare the main() method of our class as private?

In Java, main() method must be always public static to run any application or program correctly. Suppose, if the main method is declared as private there will be no complications but it will give a runtime error.

100. Can a class in Java have multiple constructors?

Yes, the classes present in Java can have multiple constructors with different parameters.

101. What is the method overloading in Java?

If a class in Java has multiple numbers methods with the same name and different parameters, then it is called as method overloading. The main

advantage of method overloading is that it increases the readability score of a program.

102. What are the different ways to overload a method?

There are two different ways to overload a method, and they are:

- By changing the data type
- By changing the number of arguments

103. Write a program to demonstrate the method overloading by changing data types?

```
class Addition{
static int add(int x,int y)
{
return x+y;
}
static int add(int x,int y,int z)
{
return x+y+z;
}
}
class TestOverloading1{
public static void main(String[] args){
System.out.println(Adder.add(10,20));
System.out.println(Adder.add(10,20,30));
```

```
}}
```

Output:

```
30

60
```

104. Write a program to demonstrate the method overloading by changing a number of arguments?

```
class Addition{

static int add(int x, int y)

{

return x+y;

}

static double add(double x, double y)

{

return x+y;

}

}

class TestOverloading2{

public static void main(String[] args){

System.out.println(Adder.add(22,22));

System.out.println(Adder.add(12.5,12.5));

}}
```

Output:

```
44

25
```

105. Why method overloading is not possible only by changing the return type of method only?

Ambiguity is the main reason why method overloading is not possible by changing the return type of method only.

106. Can we overload main() method in Java?

Yes, we can overload the main() method in Java by using method overloading but, JVM only calls the main() method that receives string array as arguments only.

107. What is method overriding in Java?

If the subclass in the program has the same method as declared in superclass then it is known as method overriding.

108. What is the usage of method overriding in Java?

Method overriding is used to achieve run-time polymorphism and also it is used to provide a specific implementation for a method that is already given by its subclass.

109. What are the rules we need to follow during method overriding?

The rules are as follows:

- The method must have the same parameters as present in the parent class.
- There must be an IS-A relationship which is called inheritance.
- The method should have the name same as that of the class name.

110. Write a program to demonstrate method overriding?

```
//Java Program to illustrate the use of Java Method Overriding
//Creating a parent class.
Class Shape{
  //defining a method
  void run()
{
System.out.println("Shape is ready");
}
}
//Creating a child class
class Rectangle extends Shape{
  //defining the same method as in the parent class
  void run()
{
System.out.println("Rectangle is drawn");
}
  public static void main(String args[]){
  Rectangle obj = new Rectangle();//creating object
  obj.run();//calling method
  }
}
```

Output:

Rectangle is drawn

111. Can we override a static method?

No, a static method cannot be overridden, it is because a static method is bounded with a class and an instance method is bounded with an object. So, the static belongs to the class area whereas, the instance method belongs to the heap area.

112. Can we override a main() method in Java?

No, because the main() method is static.

113. What is the difference between aggregation and composition?

Aggregation is built to represent the weak relationship whereas, the composition is built to represent the strong relationship.

114. Why does Java not support pointers?

Pointer is a variable that mainly refers to the memory address. Java doesn't support pointers because they are complex to understand and unsecured.

115. What is the super keyword in Java?

Super keyword in Java is used to conjure immediate prompt parent class object. It is also called a reference variable.

116. What is the usage of super keywords in Java?

The uses of the super keyword in Java are as follows:

- Super can be used to conjure an immediate parent class method.
- It is also used to indicate the immediate superclass instance variable.

- Super can be used to conjure an immediate parent class constructor.

117. Can we have virtual functions in Java?

Yes, all the functions in Java are virtual by default.