நன்றி

திரு.கி.முத்துராமலிங்கம்

அனைத்து வினாக்களுக்கும்

Thanks to

geeksforgeeks

for answers

1. What is object?

- object is a instance of a class. combination of state(information and data) and behaviors(activities and function or method). object is a real time entity.

2. What Is Class?

- A class is the blueprint from which individual objects are created (eg. Bicycle, Laptop)

3. What is connection between object and class?

- A class is a blueprint from which you can create the instance, i.e., objects. An object is the instance of the class. (eg. you have a class called Vehicle and Car is the object of that class. You can create any number of objects for the class named Vehicle, such as Van, Truck, and Auto.

4. What is method ?

- Method is a collection of statements that perform some specific task and return the result same can perform some specific task without returning anything , it helps to reuse the code without retyping the code.

5. What is Block?

- is a set of code, enclosed within curly braces { } within any class, method, or constructor. It begins with an opening brace ({) and ends with an closing braces (}

6. What is Variable?

- is a data container that saves the data values during Java program execution and In Java, all the variables must be declared before use

7. What is Static variable?

- is variable which belongs to the class and initialized only once at the start of the execution. A single copy to be shared by all instances of the class

8. What is Non-Static variable?

- Non static variables are specific to that instance of a class, is like a local variable and they can be accessed through only instance of a class

9. What is global variable (Fields)?

- Global variables are defined outside of all the functions, usually on top of the program. The global variables will hold their value throughout the life-time of your program. Global variables have static duration, which means they are created when the program starts and are destroyed when it ends

10. What is local variable?

- A variable defined within a block or method or constructor is called a local variable.

11. What is argument?

- An argument is a value passed to a function when the function is called. Whenever any function is called during the execution of the program there are some values passed with the function.

12. What is Parameter?

- Parameters act as variables inside the method. Parameters are specified after the method name, inside the parentheses. You can add as many parameters as you want, just separate them with a comma – (ref: W3S)

13. What is the purpose of void?

- to specify that a method doesn't return anything.

14.What is the use of return key word?

- return is a reserved keyword i.e, we can't use it as an identifier. It is used to exit from a method, with or without a value.

15.Can we return more than one value at a time?

- No, Java doesn't support multi-value returns

16. Explain public static void main (String[] args)

- It is a Syntax of Java main method and is the entry point of any java program, The Java compiler or JVM looks for the main method when it starts executing a Java program.

17. What is constructor?

- A constructor in Java is a special method that is used to initialize objects. The constructor is called when an object of a class is created. It can be used to set initial values for object attributes.

18. What is Default constructor?

- A constructor that has no parameter is known as the default constructor. If we don't define a constructor in a class, then the compiler creates a default constructor(with no arguments) for the class

19. What is Parameterized constructor?

- A constructor that has parameters is known as parameterized constructor. If we want to initialize fields of the class with our own values, then use a parameterized constructor.

20. How can we call constructor explicitly?

- using **this** keyword to invoke a constructor explicitly

21. What is the use of this keyword?

- **this** keyword is used to refer to the current object, i.e. through which the method is called. this() is used to call one constructor from the other of the same class.

22. What is the use of super keyword?

- The **super** keyword in java is a reference variable that is used to refer parent class objects.

23. What is the purpose of this()?

- Using this() to invoke current class constructor

24. What is the purpose of super()?

- The super() in Java is a reference variable that is used to refer parent class constructors.

25. Can we call one constructor from another constructor?

Yes, can call another constructor via the **this(...)** keyword to call a constructor from the same class and the **super(...)** keyword to call a constructor from a superclass

26. Rules for Constructor

- Constructors must have the same name as the class
- Constructors does not return any value

27. What is inheritance?

- It is the mechanism in java by which one class is allowed to inherit the features(fields and methods) of another class. (in other words child class behave like parent class)

28. What does extends keyword do?

- extends keyword is used to extend the functionality of the parent class to the subclass.

29. Can we extend final class?

- A final class cannot be extended(inherited)

30. Can we extend interface?

- Yes, an interface can extend any number of interfaces at a time..

31. What is single inheritance?

- a class is allowed to inherit from only one class. i.e. one sub class is inherited by one base class only.

32. What is Multilevel Inheritance?

- When a class extends a class, which extends anther class then this is called multilevel inheritance. For example class C extends class B and class B extends class A.

33. What is Hierarchical inheritance?

more than one child class inherits the properties of the same parent class is called hierarchical inheritance. There are multiple child classes and a single parent class. All the child classes will inherit the methods and fields present in the parent class.

34. Does java supports multiple inheritance?

- Java doesn’t support Multiple Inheritance, (if necessary we can tell that we can achieve multiple inheritance in java thru interface)

35. What is Hybrid Inheritance?

- 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.

36. Can we extend more than 1 class in java? Explain.

- multiple inheritances are not allowed due to ambiguity. Therefore, a class can extend only one class to avoid ambiguity. Implements: In Java, the implements keyword is used to implement an interface.

37. What is Diamond Problem?

- Java does not allow is multiple inheritance where one class can inherit properties from more than one class. It is known as the diamond problem. In the above figure, we find that class D is trying to inherit form class B and class C, that is not allowed in Java.)

38. What is Polymorphism?

- The word polymorphism means having many forms. In simple words, we can define polymorphism as the ability of a message to be displayed in more than one form. (eg.. man at the same time is a father, a husband, an employee. So the same person possesses different behavior in different situations. This is called polymorphism)

39. What is Compile time Polymorphism?

- a polymorphism that is resolved during the compilation process. Overloading of methods is called through the reference variable of a class. Compile-time polymorphism is achieved by method overloading and operator overloading.

40. What is Runtime Polymorphism?

- It is a process in which a function call to the overridden method is resolved at Runtime. This type of polymorphism is achieved by Method Overriding.

41. What is Method overriding? What is significance of covariant return datatype in Method overriding?

- Overriding is a feature that allows a subclass or child class to provide a specific implementation of a method that is already provided by one of its super-classes or parent classes. -- if subclass overrides any method whose return type is Non-Primitive but it changes its return type to subclass type.

42. What is inner class?

- inner class refers to the class that is declared inside class or interface which were mainly introduced, to sum up, same logically relatable classes as Java is purely object-oriented so bringing it closer to the real world.

43. Can we create interface inside from another interface?

An interface can be declared inside another interface also.

44. Can we override a final method ?

- Final methods can not be overridden .

45. What is connection between Method Overriding and Encapsulation?

- Visibility (eg. Can not reduce the visibility)

46. What is Encapsulation?

- Encapsulation is defined as the wrapping up of data under a single unit. It is the mechanism that binds together code and the data it manipulates, it is a protective shield that prevents the data from being accessed by the code outside this shield.

47. What is the difference between default and protected?

- The Protected access specifier is visible within the same package and also visible in the subclass, whereas the Default is a package level access specifier and it can be visible in the same package.

48. Can we have private class ? Explain

- No, we cannot declare a top-level class as private or protected. It can be either public or default

49. Can we overload main method in class?

- Yes, We can overload the main method in java but JVM only calls the original main method, it will never call our overloaded main method..

50. Can we have Private constructor ? Explain.

- Yes, we can declare a constructor as private. If we declare a constructor as private we are not able to create an object of a class. We can use this private constructor in the Singleton

51. Can we have static constructor ? Explain

- **Java constructor can not be static**

We know static keyword belongs to a class rather than the object of a class. A constructor is called when an object of a class is created, so no use of the static constructor.

52. What is Constructor overloading ? Explain

- we can overload constructors like methods. The constructor overloading can be defined as the concept of having more than one constructor with different parameters so that every constructor can perform a different task.

53. What is the difference between Protected & Public?

- public can be accessed from outside class but protected cannot be accessed from outside class.

54. Can we add access modifiers for local variables? Explain

- Yes, a local variable can be public, private, protected or default

55. What is Abstraction?

- Data Abstraction is the property by virtue of which only the essential details are displayed to the user. The trivial or the non-essentials units are not displayed to the user. Ex: A car is viewed as a car rather than its individual components.

56. What is difference between 100% abstraction class and interface?

- An abstract class permits you to make functionality that subclasses can implement or override whereas an interface only permits you to state functionality but not to implement it. A class can extend only one abstract class while a class can implement multiple interfaces

57. What is interface?

- The interface in Java is a mechanism to achieve abstraction. There can be only abstract methods in the Java interface, not the method body.

58. Can we have object in interface?

- No, we cannot instantiate variables and create an object.

59. Can we have constructor for interface?

- here is no need of making objects for calling methods in the interface and thus ***no point of having constructor in it***.

60. Can we reassign variables present in an Interface?

- **No,** we can't change the value of an interface field because interface fields are final and static by default. We **will get compile time error**, if we try to change the interface field value

61. What is Functional interface?

- A functional interface is an interface that contains only one abstract method. They can have only one functionality to exhibit.

62. What is Dynamic Binding? Explain

- When the compiler resolves the method call binding during the execution of the program, such a process is known as Dynamic or Late Binding in Java..

63. Why do we need getter setter method? Explain

- Getter and Setter are methods used to protect your data and make your code more secure.

64. Predefined java packages – at least 5

- Java.lang, java-io, java-math, java-util, java-sql

65. Predefined java classes - at least 5

- Object class, Math class, System class, String class, Scanner class

66. Explain System.out.println();

- System.out.println() is a statement which prints the argument passed to it. System is the class name, it is declared as final. The out is an instance of the System class and is of type PrintStream. Its access specifiers are public and final. It is an instance of java.io.PrintStream. When we call the member, a PrintStream class object creates internally.

67. Explain java.lang package – at least 5

- Provides classes that are fundamental to the design of the Java programming language. The most important classes are Object, which is the root of the class hierarchy, and Class, instances of which represent classes at run time.

68. Eclipse Shortcuts in keyboard – at least 5

- Ctrl+Shift+L to view the listing for all Eclipse keyboard shortcuts.
- Ctrl + F for find, find/replace
- Ctrl + F4 or Ctrl + w for closing current file
- Ctrl+Shirt+W for closing all files
- Alt + right and Alt + left for going back and forth while editing.

69. Difference between JDK and JVM?

- JDK is a software development kit to develop applications in Java. It is a software bundle which provides Java class libraries with necessary components to run Java code. JVM executes Java byte code and provides an environment for executing it

70. Features in Java

- Simple :- Java is easy to learn and its syntax is quite simple, clean and easy to understand.

- Object-Oriented :- In java everything is Object which has some data and behaviour. Java can be easily extended as it is based on Object Model.

- Portable :- Java Byte code can be carried to any platform.

- Platform independent :- Java is guaranteed to be write-once, run-anywhere language.

- Secured :- With java secure features it enable us to develop virus free, temper free system

71. OOPs Features

- OOP is faster and easier to execute.
- OOP provides a clear structure for the programs.
- OOP helps to keep the Java code DRY "Don't Repeat Yourself", and makes the code easier to maintain, modify and debug.
- OOP makes it possible to create full reusable applications with less code and shorter development time

ref::W3S

72. What is Array?

-An array in Java is a group of like-typed variables referred to by a common name.

73. Java 8 - Features

1) Parallel Operations:- Multiple threads can work at a same time

2) Functional Interface & Lambda Expressions: - An interface with exactly one abstract method becomes Functional Interface. Adding functional capabilities.

3) Method referencing: Now we can refer methods by their name

4) Data time API :- more improved and enhanced version data API is available

5) Default method – Interface to have default method implementation.

6) New tools and compilers are added like "jdeps“

7) JAVASCRIPT engine: new javascript engine named Nashorn is added to execute javascript codes

8) Stream API :- are introduces to run parallel and pipelined processes.

74. Java Keywords – at least 5.

- abstract, byte, break, default, else, final

75. Platform independent- Explain

-The meaning of platform-independent is that the java compiled code(byte code) can run on all operating systems. While you write a Java code and feed it to the compiler a .class file will be generated. Now this .class file/Byte code common for all kind of system whether its Linux or Mac or Windows. Now to run this .class file, we need JRE (which is actually implementation of JVM). SO Mac will have its own version of JRE, Widows will have different and Linux has different one too. But one thing to notice here is that --- the input for all these , means the .class file is same. This is actually Platform Independence.