OBJECT ORIENTED PROGRAMMING INTERVIEW CHEATSHEET(FRESHERS)

Question- -1 What is an Object-Oriented Paradigm?

- Paradigm literally means a pattern or a method. Programming paradigms are the methods to solve a program that is of four types namely, Imperative, logical, functional, and object-oriented. When objects are used as base entities upon which the methods are applied, encapsulation or inheritance functionalities are performed, it is known as an object-oriented paradigm.

Question-0 What is POP?

- In the procedure oriented approach, large programs are divided into smaller programs known as functions.
- In POP, a program is written as a sequence of procedures or function.
- In POP, each procedure (function) contains a series of instructions for performing a specific task.
- During the program execution each procedure (function) can be called by the other procedures.
- To call a procedure (function), we have to write function name only.

Question-1 What are Classes?

- A class is a template for an object
- A class creates a new data type that can be used to create objects. It can't be physical.
- Classes can be inbuilt or user-defined.
- Class in Java contains Fields, Methods, Constructor, Blocks, Nested Class and Interface
- Class is of four types: Derived, Super, Sub, Mixed

Question-2 What are Objects?

- The object is defined as the instance of a class
- Object is an entity that has State and Behaviour
- Object has a Lifecycle: Creation, Accessing, Unreferred, Garbage Collection

Question-3 What are the different ways to create objects in Java?

- Using new keyword
- Using new instance() of Class class
- Using clone() method
- Using Deserialization
- Using the newInstance() method of the Constructor Class

Question-4 What are the Constructors?

- Constructor is a special method that is used to initialize objects
- Every time an object is created using the new() keyword, at least one constructor is called
- The name of constructor is same as of the class

Question-5 What happens if you don't provide a constructor in a class?

- If we don't provide a constructor in a class, the compiler automatically generates a default constructor with no arguments and no operation which is a default constructor. It simply initiates the attributes of class with their default values

Question-6 What happens if you don't provide a constructor in a class?

- If we don't provide a constructor in a class, the compiler automatically generates a default constructor with no arguments and no operation which is a default constructor. It simply initiates the attributes of class with their default values

Question-7 How many types of constructors are used in Java?

- Default Constructor: It is the type that does not accept any parameter value. It is used to set initial values for object attributes.
- Parameterized Constructor: Parameterized Constructor: It is the type of constructor that accepts parameters as arguments. These are used to assign values to instance variables during the initialization of objects.
- Copy Constructor: Unlike other constructors copy constructor is passed with another object which copies the data available from the passed object to the newly created object.

Question-8 What is the purpose of a default constructor?

- Constructors help to create instances of a class or can be said to create objects of a class. Constructor is called during the initialization of objects. A default constructor is a type of constructor which do not accept any parameter, So whatever value is assigned to properties of the objects are considered default values.

Question-9 What are the differences between the constructors and methods?

- Constructors are only called when the object is created but other methods can be called multiple times during the life of an object.
- Constructors do not have a return type, whereas methods have a return type, which can be void or any other type.
- Constructors are used to setting up the initial state but methods are used to perform specific actions.

Question-10 What are Wrapper Class

- A Wrapper class in Java is a class whose object wraps or contains primitive data types.
- When we create an object to a wrapper class, it contains a field and in this field, we can store primitive data types.
- They are Boolean, Byte, Short, Integer, Character, Long, Float, and Double. Further, custom wrapper classes can also be created in Java.

Question-11 Why do we need Wrapper Class?

- Provides methods like valueOf(), parseInt() many other
- Data structures in the Collection framework, such as ArrayList and Vector, store only objects and not primitive types.
- Wrapper classes are final and immutable

Question-12 What are the default values assigned to variables and instances in Java?

- The default value for numeric types (byte, short, int, long, float, and double) is 0.
- The default value for the boolean type is false.
- The default value for object types (classes, interfaces, and arrays) is null.
- The null character, “u0000, ” is the default value for the char type.

Question-13 What is a Class Variable?

- Java, a class variable (also known as a static variable) is a variable that is declared within a class but outside of any method, constructor, or block.
- Class variables are declared with the static keyword, and they are shared by all instances (objects) of the class as well as by the class itself.

Question-14 Explain 'this' keyword.

- this' is a keyword used to reference a variable that refers to the current object.

Question-15 What are the main concepts of OOPs in Java?

- Inheritance
- Polymorphism
- Encapsulation
- Abstraction

Question-16 What is the 'IS-A ' relationship in OOPs Java?

- 'IS-A' is a type of relationship in OOPs Java where one class inherits another class.

Question-17 Define Inheritance.

- Inheritance means creating new classes based on existing ones. A class that inherits from another class can reuse the methods and fields of that class. In addition, we can add new fields and methods to our current class as well.

Question-18 What are the different types of inheritance in Java?

- Single Inheritance: When a child or subclass extends only one superclass, it is known to be single inheritance. Single-parent class properties are passed down to the child class.
- Multilevel Inheritance: When a child or subclass extends any other subclass a hierarchy of inheritance is created which is known as multilevel inheritance. In other words, one subclass becomes the parent class of another.
- Hierarchical Inheritance: When multiple subclasses derive from the same parent class is known as Hierarchical Inheritance. In other words, a class that has a single parent has many subclasses.
- Multiple Inheritance: When a child class inherits from multiple parent classes is known as Multiple Inheritance. In Java, it only supports multiple inheritance of interfaces, not classes.

Question-19 Is there any limitation to using Inheritance?

- Yes, there is a limitation of using Inheritance in Java, as because of inheritance one can inherit everything from super class and interface because of which subclass is too clustered and sometimes error-prone when dynamic overriding or dynamic overloading is done in certain situation.

Question-20 Can the constructor be inherited?

- No. We can't inherit a constructor.

Question-21 What is Polymorphism?

- Polymorphism allows us to perform a single action in different ways.
- In Java Polymorphism is mainly divided into two types: Compile-time Polymorphism and Runtime Polymorphism

Question-22 What is Method Overloading?

- In Java, Method Overloading allows different methods to have the same name, but different signatures where the signature can differ by the number of input parameters or type of input parameters, or a mixture of both.
- Method overloading in Java is also known as **Compile-time Polymorphism, *Static Polymorphism, or* Early binding**.
- In Method overloading compared to the parent argument, the child argument will get the highest priority.

Question-23 What is method overriding? Can we override the private methods?

- Method overriding is a method to achieve Run-time polymorphism in Java. Method overriding is a feature that allows a child class to provide a specific implementation of a method that is already provided by one of its parent classes. When a method in a child class has the same name, the same parameters or signature, and the same return type(or sub-type) as a method in its parent class, then the method in the subclass is said to override the method in the superclass.
- It is not possible to override the private methods in Java.

Question-24 Can we override the overloaded method?

- Yes, since the overloaded method is a completely different method in the eyes of the compiler. Overriding isn't the same thing at all.

Question-25 Can we overload the main() method?

- Yes in Java we can overload the main method to call the main method with the help of its predefined calling method.

Question-26 Can we override the static method?

- No, as static methods are part of the class rather than the object so we can't override them.

Question-27 Can you have virtual functions in Java?

- Yes, Java supports virtual functions. Functions are by default virtual and can be made non-virtual using the final keyword.

Question-28 What is Abstraction?

- Abstraction refers to the act of representing essential features without including background details. The detailed information or the implementation is hidden. The most common example of abstraction is a car, we know how to turn on the engine, accelerate and move, however, the way engine works, and its internal components are complex logic hidden from the general users. This is usually done to handle the complexity.

Question-29 What do you mean by Data Encapsulation?

- Encapsulation is achieved by declaring the instance variables of a class as private, which means they can only be accessed within the class. In other class they are accessed using getter and setter.
- It is a way of restricting the access of our data members by hiding the implementation details. Encapsulation also provides a way for data hiding. The user will have no idea about the inner implementation of the class.

Question-30 What is Abstract Class? When Abstract methods are used?

- A class declared as abstract, cannot be instantiated i.e., the object cannot be created. It may or may not contain abstract methods but if a class has at least one abstract method, it must be declared abstract.
- An abstract method is used when we want to use a method but want to child classes to decide the implementation in that case we use Abstract methods with the parent classes.

Question-31 What is an Interface?

- An interface in Java is a collection of static final variables and abstract methods that define the contract or agreement for a set of linked classes. Any class that implements an interface is required to implement a specific set of methods. It specifies the behaviour that a class must exhibit but not the specifics of how it should be implemented.

Question-32 What is a Marker Interface?

- An Interface is recognized as an empty interface (no field or methods) it is called a marker interface. Examples of marker interfaces are Serializable, Cloneable, and Remote interfaces.

Question-33 What are the advantages of passing this into a method instead of the current class object itself?

- this is the final variable because of which this cannot be assigned to any new value whereas the current class object might not be final and can be changed.

Question-34 What will be the initial value of an object reference which is defined as an instance variable?

- The initial value of an object reference which is defined as an instance variable is a NULL value.

Question-35 What are Brief Access Specifiers and Types of Access Specifiers?

- Access Specifiers in Java help to restrict the scope of a class, constructor, variable, method, or data member. There are four types of Access Specifiers in Java are Public, Private, Default and Protected.

Question-36 How is the 'new' operator different from the 'newInstance()' operator in Java?

- The new operator is used to create objects, but if we want to decide the type of object to be created at runtime, there is no way we can use the new operator. In this case, we have to use the newInstance() method.

Question-37 What are the advantages of Object Cloning?

- In Java, the '=' assignment operator cannot be used for cloning as it simply creates a copy of reference variables. To overcome such discrepancy the clone() method of Object class can be used over the assignment operator.
- The clone() method is a protected method of class Object which means that only the Employee class can clone Employee objects. This means no class other than Employee can clone Employee objects since it does not know the Employee class attributes.
- Code size decreases as repetition decreases.