İbrahim AKIN

1. Why we need to use OOP? Some major OOP languages?

OOP insists that you think about what you expose to the outside world, it lets you change the implementation of an object without affecting any other code. It allows you to have many different functions, all with the same name, all doing the same job, but on different data. It lets you write generic code: which will work with a range of data, so you don't have to write basic stuff over, and over again. It lets you write a set of functions, then expand them in different direction without changing or copying them in any way.

Some OOP languages: Java, C++, C#, Kotlin

2. Interface vs Abstract class?

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.

3. Why do we need equals and hashcode? When to override?

The Object class defines both the equals() and hashCode() methods which means that these two methods are implicitly defined in every Java class. The default implementation of equals() in the class Object says that equality is the same as object identity. And income and expenses are two distinct instances.

We can override it when we want it to not consider only object identity, but rather also the value of the relevant properties.

4. Diamond problem in Java? How to fix it?

The diamond problem is a common problem in Java when it comes to inheritance. There are different types of inheritance such as, single, multiple, multi-level, and hybrid inheritance. But Java does not support the multiple inheritance because of the diamond problem.

The solution to the diamond problem is default methods and interfaces. We can achieve multiple inheritance by using these two things. The default method is similar to the abstract method. The only difference is that it is defined inside the interfaces with the default implementation. We need not to override these methods. Because they are already implementing these interfaces.

5. Why we need Garbage Collector? How does it run?

We need Garbage Collector because it frees developers from having to manually release memory, allocates objects on the managed heap efficiently, reclaims objects that are no longer being used, clears their memory, and keeps the memory available for future allocations. Managed objects automatically get clean content to start with, so their constructors don't have to initialize every data field, provides memory safety by making sure that an object cannot use for itself the memory allocated for another object.

6. Java ‘static’ keyword usage?

The static keyword in Java is mainly used for memory management. The static keyword in Java is used to share the same variable or method of a given class. The users can apply static keywords with variables, methods, blocks, and nested classes. The static keyword belongs to the class than an instance of the class. The static keyword is used for a constant variable or a method that is the same for every instance of a class.

7. Immutability means? Where, How and Why to use it?

Immutability in java means that once an object is created, we cannot change its content. In Java, all the wrapper classes (like Integer, Boolean, Byte, Short) and String class is immutable. We can create our own immutable class as well. This is useful as it means you can pass references to the object around, without worrying that someone else is going to change its contents. Especially when dealing with concurrency, there are no locking issues with objects that never change.

8. Composition and Aggregation means and differences?

Aggregation and Composition are subsets of association meaning they are specific cases of association. In both aggregation and composition object of one class "owns" object of another class. But there is a subtle difference:

Aggregation implies a relationship where the child can exist independently of the parent. Example: Class (parent) and Student (child). Delete the Class and the Students still exist.

Composition implies a relationship where the child cannot exist independent of the parent. Example: House (parent) and Room (child). Rooms don't exist separate to a House.

9. Cohesion and Coupling means and differences?

Cohesion is defined as the degree to which the elements of a particular module are functionally related. Basically, cohesion is used to measure the functional strength of a module. For example, the systems having high cohesion will have elements such as instructions, groups of instructions, the definition of data, etc. strongly connected to each other. This helps in improving the focus on a given task and thus, high cohesion is preferred.

Coupling is defined as the degree to which the two modules are dependent on each other. It measures the strength of relationships between modules. For example, loosely coupled systems have modules that hardly depend on other modules and can focus better on their assigned tasks.

10. Heap and Stack means and differences?

Stack is a linear data structure whereas Heap is a hierarchical data structure. Stack memory will never become fragmented whereas Heap memory can become fragmented as blocks of memory are first allocated and then freed. Stack accesses local variables only while Heap allows you to access variables globally. Stack variables can’t be resized whereas Heap variables can be resized. Stack memory is allocated in a contiguous block whereas Heap memory is allocated in any random order.

11. Exception means? Type of Exceptions?

An exception is an event, which occurs during the execution of a program, that disrupts the normal flow of the program's instructions. When an error occurs within a method, the method creates an object and hands it off to the runtime system. The object, called an exception object, contains information about the error, including its type and the state of the program when the error occurred. Creating an exception object and handing it to the runtime system is called throwing an exception. There are mainly two types of exceptions in Java, Checked exception and Unchecked exception. Checked exceptions are also known as compile-time exceptions. The unchecked exceptions are those exceptions that occur during the execution of the program. Hence they are also referred to as Runtime exceptions.

12. How to summarize 'clean code' as short as possible?

Clean code is code that is easy to understand and easy to change.

13. What is the method of hiding in Java?

Method hiding means subclass has defined a class method with the same signature as a class method in the superclass. In that case the method of superclass is hidden by the subclass. It signifies that: The version of a method that is executed will NOT be determined by the object that is used to invoke it. In fact, it will be determined by the type of reference variable used to invoke the method.

14. What is the difference between abstraction and polymorphism in Java?

Their operating model is very similar and based upon the relationship of parent and child classes. In fact, Polymorphism needs the great support of Abstraction to power itself, without Abstraction you cannot leverage the power of Polymorphism. Let's understand this by what Abstraction and Polymorphism provide to an object-oriented program. Abstraction is a concept to simplify the structure of your code. Abstraction allows you to view things in more general terms rather than looking at them as they are at the moment, which gives your code flexibility to deal with the changes coming in the future.