HM#1 **Doğukan Bayrak**

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

It is the way many objects we see in real life are transferred to the computer environment. In other words, the display of many features such as the color, condition, name, year of manufacture of an object in a computer environment can be given as an example. Because classes are created once, they can be run with shortcodes instead of rewriting long codes.

Major OOP languages : Java , C# , C++ , Python, Ruby

2. **Interface vs Abstract class ?**

An interface in Java is a blueprint of a class. It has static constants and abstract methods. The interface in Java is a mechanism to achieve abstraction. There can be only abstract methods in the Java interface, not method body. It is used to achieve abstraction and multiple inheritance in Java. In other words, you can say that interfaces can have abstract methods and variables. It cannot have a method body. Java Interface also represents the IS-A relationship. It cannot be instantiated just like the abstract class.

Java abstract class is a class which cannot be instantiated, meaning you cannot create new instances of an abstract class. The purpose of an abstract class is to function as a base for subclasses.

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

The equals() and hashcode() are the two important methods provided by the Object class for comparing objects. Since the Object class is the parent class for all Java objects, hence all objects inherit the default implementation of these two methods.

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

The Diamond problem is about the inheritance. Is known also as a deadly diamond of death. A class try to extends two different classes with same properties , It creates a diamond problem. Bec of that java does not support multiple inheritance.

Still , if we have a diamond problem we can solve that with interfaces. We can override the methods.

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

Garbage Collection is an automatic memory management mechanism. This process is based on looking at memory, detecting used objects and deleting those that are not referenced. The space occupied by unused/non-referenced objects is freed up in memory, freeing up memory space. The mechanism that performs this operation is called the Garbage Collector.

6. **Java 'static' keyword usage?**

We use static keyword in Java because of efficiently manage memory. Generally, if you want to access variables or methods inside a class, you first need to create an instance or object of that class. However, there might be situations where you want to access only a couple of methods or variables of a class and you don't want to create a new instance for that class just for accessing these members. This is where you can use the static keyword in Java.

In Java, it is possible to use the static keyword with methods, blocks, variables, as well as nested classes. In simple words, if you use a static keyword with a variable or a method inside a class, then for every instance that you create for that class, these static members remain constant and you can't change or modify them.

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

Immutable are classes whose contents cannot be changed once objects are created. In Java, all the wrapper classes (like Integer, Boolean, Byte, Short) and String class is immutable. When you send the reference to the other object you don't have to worry about change if it's in immutability class. Also we can create a immutable variable with final keyword.

8. **Composition and Aggregation means and differences?**

Aggretation : It is a special union that represents the "possession" relationship. It has the condition of not being circular. This means that the object cannot relate to itself. Aggregation is a "**has-a**" relationship. Both objects have their own have independent existence.

Composition: A special type of association used for ownership modeling. It is similar to Aggregation, except that it represents the Whole-Part relationship and the part does not have its own independent existence. The object representing the existence of the whole is also responsible for the life of the part. The formation should be understood as a "contains-a" relationship.

The car is made of parts. The car has these parts. If a part is missing, the car is broken. Creates Composition with Car Parts. There are also passengers of the car. Passengers are not Part of the Car. If there is no passenger, the Car is not broken. It does not have Car Passengers. If they're not in the Car, they're elsewhere. Relationship is temporary. Passengers come and go. There is Aggregation relationship between Car and Passengers.

9. **–Cohesion and Coupling means and differences?**

Cohesion is the indication of the relationship within module. It is concept of intra-module. Cohesion has many types but usually highly cohesion is good for software. Coupling is also the indication of the relationships between modules. It is concept of Inter-module. Coupling has also many types but usually low coupling is good for software.

- Cohesion is the concept of intra module., Coupling is the concept of inter module.
- Cohesion represents the relationship within module. Coupling represents the relationships between modules.
- Cohesion represents the functional strength of modules., Coupling represents the independence among modules.

10. Heap and Stack means and differences ?

Stack and Heap are logical structures in memory (ram). Types such as int, short, byte, long, decimal, double, float, which we call primitive type, are called value type and kept in the stack. These values must be known before the runtime, and the operating system allocates a certain place in the stack before the program runs. If this section is exceeded by the person who wrote the code, a stack overflow error may be encountered. The data on the stack is in LIFO logic. It is sorted and no operation can be performed with a value in between. Class type variables are reference types, the model (reference) they refer to is stored in the stack and their values are stored in the heap.

One of the most important differences between Heap and Stack is that the data is stored in a mixed way in the heap, and the stack works in ascending or descending address logic. Accordingly, accessing a data in the heap is more costly than accessing a data in the stack. Another difference is that while the data in the stack is deleted immediately, the data in the heap depends on the Garbage Collector algorithm.

11. Exception means ? Type of Exceptions?

Exceptions are exceptions that occur during program runtime . If something unusual happens during program runtime, the compiler creates an object. We can call this object an exception object. There is two type exception in java. Checked exception and Unchecked exception. There are a number of possible errors that may occur on the basis of our program, and if the program wants us to make a try-catch definition or throws statement while creating this line of code, this error type is called Checked Exception. Those that cannot be checked by the compiler are called unchecked exceptions. Because they are not controlled by the compiler, we may encounter these errors at runtime even if our code has been compiled successfully.

12. How to summarize ‘clean code’ as short as possible ?

It is to comply with the code standards accepted by the majority, namely the rules of naming, classification, comment lines.

13. What is the method of hiding in Java?

If a subclass defines a class method with the same signature as a class method in the superclass, the method in the subclass hides the method in the superclass. Static methods are not overridden due to resolution of method calls made by the compiler at compile time. Therefore, if you define a static method in the base class with the same signature as the one in the superclass, the method in the subclass hides the method inherited from the superclass.

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

Difference between Polymorphism and Abstraction is that Abstraction is implemented using abstract class and interface in Java but Polymorphism is supported by overloading and overriding in Java.

Though overloading is also known as compile-time Polymorphism, method overriding is the real one because it allows a code to behave differently at different runtime conditions, which is known as exhibiting polymorphic behavior.