W01 HomeWork – Talha Arıç, talharic@gmail.com

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

·We can detect problems more quickly in object-oriented programming languages. Whatever object has a problem, we fix it.

·We avoid code duplication through inheritance.

·Objects are independent of each other. Information security is ensured.

·It allows us to establish flexibility by creating a polymorphic structure in our programs using polymorphism.

·Java, Python, C++, Objective-C, Swift, C#, and PHP

2-Interface vs Abstract class ?

·Abstract class can have abstract and non-abstract methods. Interface can have only abstract methods. Since Java 8, it can have default and static methods also.

·Abstract class doesn't support multiple inheritance. Interface supports multiple inheritance.

· An abstract class can be extended using keyword "extends". interface can be implemented using keyword "implements".

· An abstract class can extend another Java class and implement multiple Java interfaces. An interface can extend another Java interface only.

· The abstract keyword is used to declare abstract class. The interface keyword is used to declare interface.

3- Why we need equals and hashcode ? When to override ?

· The equals method is used to compare two objects. The default equals method is defined in the object class. This implementation is similar to the == operator. Two object references are equal only if they point to the same object. It is possible to override the equals method.

· HashCode is used in hashing to decide which group to categorize an object in. A group of objects can share the same hashCode. A correct hashing function can distribute objects equally among different groups.

· We must override hashCode() in every class that overrides equals(). Failure to do so will result in a violation of the general contract for Object.hashCode(), which will prevent your class from functioning properly in conjunction with all hash-based collections, including HashMap, HashSet, and Hashtable.

4- Diamon problem in Java ? How to fix it?

The diamond problem is a common problem in Java when it comes to inheritance. Java does not support multiple inheritance.

As simple inheritance allows a child class to derive properties from one super-class. for example, if class B inherits properties from only one super-class A, then it is called simple inheritance, and Java supports them.

Multi-level inheritance allows a child class to inherit properties from a class that can inherit properties from some other classes. For example, class C can inherit its property from B class which itself inherits from A class. Java also supports them.

What 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.

It is an ambiguity that can rise as a consequence of allowing multiple inheritance. It is a serious problem for other OPPs languages. It is sometimes referred to as the deadly diamond of death.

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.

The advantage of interfaces is that it can have the same default methods with the same name and signature in two different interfaces. It allows us to implement these two interfaces, from a class. We must override the default methods explicitly with its interface name.

5- Why we need Garbagge Collector ? How does it run ?

Memory management in Java is done in the background with the JVM and the Garbage Collector inside it. Finds and cleans unused objects at any time. We should use it to reduce the possibility of error in software.

6- Java 'static' keyword usage ?

When Static Variables are created, only one place in memory for the class it belongs to is automatic. For an object of its created, it does not take up disk space again. Since they can be purchased without being tied to static objects, they are accessed without creating objects. In other words, after any object is created, files can be downloaded from all packages consisting of any object.

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

All wrapper classes such as String, Boolean, Integer are immutable. Immutable classes are created once and do not change. Immutable classes give us a great transition by being single and here. If we want to make a class thread-safe, we need to make it immutable.

·We have to make the class final.

·We should make all the fields of the class private.

·No setter method is put for variables.

8- Composition and Aggregation means and differences ?

·In aggregation, there is a relationship in which a child can exist independently of the parent.

·It cannot exist independently of the parent in the composition.

9- Cohesion and Coupling means and differences ?

·Cohesion is an indication of the relative functional strength of a module. A cohesive module performs a single task, requiring little interaction with other components in other parts of a program. Stated simply, a cohesive module should (ideally) do just one thing.

·Coupling is an indication of the relative interdependence among modules. Coupling depends on the interface complexity between modules, the point at which entry or reference is made to a module, and what data pass across the interface. A module having high cohesion and low coupling is said to be functionally independent of other modules.

10- Heap and Stack means and differences ?

·It has a smaller area. Simple type data is intended here. Value type settings are in your Factory stack area (int, double, DateTime etc.)

·The heap region of memory exists for storing more complex data. It occupies a larger amount of memory than the Stack region. Reference type variables are kept in the heap region of memory (string, object, Console, etc.).

11- Exception means ? Type of Exceptions ?

Exceptions are events that occur during the execution of programs that disrupt the normal code flow. For example division by zero, non-index array access etc.

Exception classes derive from the Object class like other Java classes. A subclass of the Object class is the Throwable class. In this class, the Exception class derives. The Exception is divided into two parts:

·Check Exception:

It is the exception type that the IDE warns the developer. Checked Exceptions are all handled at compile time. As an example, we are writing a code and the IDE will give a warning if an error is likely to occur here. This error is called Check Exception.

· Uncheck Exception:

It is the exception type that the IDE does not warn the developer. It is an exception class derived from the RuntimeException class.

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

The code is clean, the team members can easily understand and develop the code, apart from the developer who wrote the code.

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.

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

Abstraction is the concept of hiding details. Polymorphism: It describes a language's ability to process objects of various types and classes through a single, uniform interface.