HW#1 Talha Alçorba

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

We need OOP to transfer objects that we encounter in real life to software. we can determine various characteristics of objects (size, color, brand, weight, etc.) thanks to OOP.

2-) Interface vs Abstract class?

Abstract class is used to gather objects with common properties under one roof. Abstraction is done by writing the word "abstract" in front of the class. Extensions are also used to inherit from an abstract class. Abstract class is used in is-a relationships. A class can only implement an abstract class. Access modifiers used on the abstract class: (public, static, final)

There are only method definitions in the interface. No code snippet is written in it. Interface can inherit from another interface. Interfaces generally work in a "can-do" relationship. Access specifiers are not used in properties and methods in an interface. Everything is considered public.

A class can inherit more than one interface, but only one abstract class can inherit a class. Empty methods can be defined in Interface, but both empty methods and filled methods can be defined in abstract classes. Using Abstract classes provides an advantage in terms of speed. All objects in Interface must be "public", while in Abstract classes not all elements must be "public". Interface constructors do not contain constructors. Abstract class can contain constructor methods. Interface methods cannot be static. Abstract class non-abstract methods can be defined as static.

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

As long as the fields used in the equals comparison remain constant, the hashCode method should always produce the same result when called repeatedly within the same application.

If two objects are equal according to the equals method, the hashCode methods of these two objects must also produce the same integer value.If two objects are not equal according to the equals method, the hashCode method does not have to produce different integer results for these two objects. However, the programmer should know that generating different hash codes for unequal objects can increase hash table performance.

When you don't override the hashCode method with equals, you violate condition 2. equal objects must produce equal hashcodes. In this case, although two different objects are equal according to the equals method, there will not be much similarity between them according to the hashCode method. Therefore, the hashCode method will produce very different values for these two equal objects, contrary to the contract.

4-) Diamond problem in Java? How to fix t?

The diamond problem is a common problem in Java when it comes to inheritance. Inheritance is a very popular property in an object-oriented programming language, such as C++, Java, etc. There are different types of inheritance such as single, multiple, multi-level, and hybrid inheritance. But remember that Java does not support multiple inheritances 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.

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 do we need Garbage Collector? How does it run?

Garbage Collection is an automatic memory management mechanism. This process is based on looking at the heap memory, detecting the used objects and deleting the unreferenced ones. 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 ?

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.

The static keyword is a non-access modifier in Java that is applicable for the following: (Blocks, Variables, Methods Classes)

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

When it comes to object-oriented and functional programming, the concept of immutability is applied at the object level.

It’s to do with the state and how you’re not allowed to modify after it’s been created. Every language has its own system of state management and before we go any further, we need to understand exactly what a state is.

In object-oriented, a state has two parts to it — the properties and the values. The properties are generally static, meaning that they don’t change. The values, however, are expected to be dynamic and therefore changeable.

8-) Composition and Aggregation means and differences ?

Composition: It expresses the situation where there is an A has a B relationship between A and B classes. Objects A and B should have almost the same lifetime.

Aggregation: It is the case of "A holds a B" relationship between A and B classes. It is very similar to Composition. The main difference is that there is no part-whole relationship.

The logical difference between Aggregation and Composition is that Aggregation means that the owned object can exist independently of the owning object. Composition is the inability of the owned object to exist independently of the possessing object.

9-) Cohesion and Coupling means and differences?

Cohesion refers to the degree to which the elements inside a module belong together. In one sense, it is a measure of the strength of relationship between the methods and data of a class and some unifying purpose or concept served by that class. In another sense, it is a measure of the strength of relationship between the class’s methods and data themselves.

Coupling is the degree of interdependence between software modules; a measure of how closely connected two routines or modules are; the strength of the relationships between modules.

The major difference between cohesion and coupling is that cohesion deals with the interconnection between the elements of the same module. But, coupling deals with the interdependence between software modules. Cohesion is defined as the degree of relationship between elements of the same module while Coupling is defined as the degree of interdependence between the modules.

10-) Heap and Stack means and differences?

We can say that Stack and Heap are logical partitions of ram. While value types, pointers and addresses are stored in Stack, reference values are stored in Heap. Access to Stack is faster than Heap and Stack works in LIFO (Last-In-First-Out) logic. So last come first out. For this reason, you cannot remove any element from the middle, they are in relationship with each other. Variables of type Struct are value types and are stored in the Stack. Variables of type Class are reference types and their references are stored in the Stack and itself in the Heap.

While Stack is used for static allocation of memory, Heap provides dynamic allocation. Both are located in the Ram area. The data in the stack is placed directly in the memory, so its access is very fast. Heap, on the other hand, is used at runtime, and since it is a dispersed memory structure, it is not as easy to access as a stack, so it works slowly. While the data in the Stack memory is deleted immediately, the deletion of the data in the Heap memory depends on the Garbage Collector (Garbage collection mechanism). As the stack space is limited, assigning very large numbers and large types of data may cause the memory to become full.

11-) Exception means? Type of Exceptions?

These are conditions that occur unusually during program runtime . If something unusual happens during program runtime, the compiler creates an object. We can call this object an exception object.

Exception Types:

Checked Exception: We need to handle the error throwing status of some code blocks.

Runtime exception(unchecked exception) : These are the error types that are not required to be handled by the compiler. However, if we did not handle the error during the Runtime, we do not have a chance to recover the program.

Try -Catch Finally Block: To catch an error, we need to enclose the code block in a try block.

If an error is received in the try block, an Exception object is thrown. We can handle exception objects in catch blocks

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

Clean code is any software that is easy to understand, free of unnecessary clutter. The understandability of the code should be increased with comments. Structures should be used in accordance with their purpose, there should be no unnecessary 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. It signifies that : The version of a method that is executed will not be determined by the object that is used to invoke it.

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

Here are the points of difference between abstract class and polymorphism:

- Abstract class is a base class. Cannot create instance whether it may be base class or not. But in polymorphism, we can create instance to base class.

- In abstract class, we use abstract keyword to method but in polymorphism uses virtual keyword to base class method.

- In abstract class, we can declare abstract method inside abstract method and can define method which class derived abstract class.

- In polymorphism, declaration and definition are only inside corresponding class.

- In polymorphism, we can access derived class method through base class, but in abstract class we cannot access derived class method threw base class because cannot create an object to base class.

- Abstract method declares only inside abstract class, but we can declare virtual method inside a class as well (it should be a base class).