# HW#1

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

It makes easier to develop complex codes. You can create modular structures for programs with OOP.

OOP decreases rewriting same code. It contains code reusability. And it is easy to maintain the existing code. You can always modify it with new features. It is easier to design solutions for a problem with OOP.

Major OOP languages: Java, C#, C++, Python, JavaScript, Kotlin, Swift

## *2 - Interface vs Abstract class?*

- Interfaces are used to achieve abstraction and they have “can-do” relation. Abstract classes offer default functionality for the subclasses and they have “is-a” relation.
- Interfaces can have only abstract methods but abstract classes can have abstract and non-abstract methods.
- An interface can extend another Java interface only but an abstract class can implement multiple Java interfaces and extend another Java class.
- Interface supports multiple inheritance but Abstract class doesn't support multiple inheritance.
- Interface members are public by default but abstract class can have class members like private, protected, etc.
- Interfaces can be implemented using keyword "implements" and abstract classes can be extended using keyword "extends".
- Interface variables are final by default but abstract classes may contain non-final variables.

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

These methods are provided by the Object class for comparing objects. They verifies the equality of two objects.

- Equals default implementation simply checks the object references of two objects to verify their equality. By default for equals, two objects are equal if and only if they are refer to the same memory location.
- Hashcode returns a unique integer value for the object in runtime. By default for hashcode, integer value is derived from memory address of the object in heap and the object’s hash code is used for determining the index location.

***When to override:*** It depends on how we want to compare two objects. The default implementation of Object's equals considers two objects are equal only if they have the same memory address. So if we want to compare two objects which has different memory adresses we should override equals method. And whenever we override equals method we should override hashcode method too. Because we may use Hash based Collections later and if we want to use these collections properly, we should override hashcode method too.

## *4 - Diamond problem in Java? How to fix it?*

Diamond problem is related to multiple inheritance. Inheritance is a relation between two classes, the parent(super-class) and child class(sub-class). The child class inherits all the properties of the parent class. A class can inherit properties of more than one parent class. Diamond problem occurs when there exist methods with the same name and signature in more than one super-classes. When we call the method, the compiler gets confused and can't determine which super-classes method to be called. For to fix we can use interfaces with "super" keyword .

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

Garbage Collection is the process of reclaiming unused memory by destroying the unused objects. In Java garbagge collector destroyes objects that don't contain a reference. So it follows how many references an object has. When the refence number drops to zero the object starts to wait fort o be deleted by garbagge collector. We can not say exactly when garbagge collector starts to work but from time to time it destroyes the objects without references and prevents memory leaks.

## *6 - Java 'static' keyword usage?*

"Static" keyword is an access modifier in Java. It is used generally for memory management. It can be applied with variables, methods, blocks and nested classes. When a member is declared static, it can be accessed before any objects of its class are created and without reference to any object.

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

Immutable objects are instances whose state can not change after it has been initialized. Immutability must be used when we don't want value changes of objects. We can not use Getter methods for immutable objects. For immutable objects, whenever we try to change the state of the object, a new object will be created. Besides objects, classes can be immutable too. For classes we can say that immutable classes are thread-safe.

## *8 - Composition and Aggregation means and differences?*

Aggregation and composition are the types of association. The primary difference between aggregation and composition is that aggregation implicit a relationship where the child can exist independently of the parent. For composition there is a relationship where the child can not exist independent of the parent. Aggregation is kind of “have a” relationship, composition is “part of” relationship. Aggregation is a weak type of association, composition is a strong type of association.

## *9 - Cohesion and Coupling means and differences?*

Cohesion focuses on how single class is designed. It is used to indicate the degree to which a class has a single, well-focused purpose. Coupling is about how classes interact with each other. It is the degree to which one class knows about another class. For ideal OO design, cohesiveness of the class should be higher but coupling degree should be loose coupling.

## *10 - Heap and Stack means and differences?*

Stack is a temporary memory allocation which stores temporary variables created by a function during runtime. Heap is a memory allocation used to store global variables. By default, all global variable are stored in heap memory space. Every time when we create an object it is always created in heap but the referencing information to this object is always stored in stack.

## *11 - Exception means? Type of Exceptions?*

Exception is a problem that occurs during the execution of a program. Many different kinds of errors can cause exceptions.

Common Exceptions of Java:

- *ArithmeticException:* Dividing a number by zero
- *ArrayIndexOutOfBoundsException:* Trying to reference an item in an array with an index doesn't exist
- *NullPointerException:* Trying to call an object which has null value
- *NumberFormatException:* Formatting of any variable or number as mismatched
- *ClassCastException:* Trying to change a String object to a Double or something like this conversion

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

A 'clean code' must have readability, changeability, maintainability, extensibility. The code must be short and simple, not complex.

## *13 - What is the method of hiding in Java?*

When a subclass defines a static class method with the same signature as a static class method in the superclass, the class method in the superclass will be hidden by the class method in the subclass. So the class method in the superclass will be prevented to call because of the subclass. This is called as "method hiding" in Java.

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

Abstraction refers to representing essential features without including the background details or explanations. Polymorphism describes ability to process objects of various types and classes through a single, uniform interface. So we can say that polymorphism is ability to take more than one form.