Aleyna ZENGİN

HW#1

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

As computers started become faster and more powerful, the complexity of the program's developers were writing increased as code became more and more complex, programming languages evolved to suit the needs of developers.

Object-oriented programming (OOP) is a programming paradigm that allows you to package together data states and functionality to modify those data states, while keeping the details hidden away.
There are four major benefits to object-oriented programming:

- **Encapsulation:** Encapsulation provides a high level of protection for data. Preventing direct access to private attributes, allowing them to be accessed only by public methods, ensures that the data can never be modified carelessly. It also ensures that any associated actions also happen.
- **Abstraction:** by using classes, you are able to generalize your object types, simplifying your program.
- **Inheritance:** because a class can inherit attributes and behaviors from another class, you are able to reuse more code.
- **Polymorphism:** one class can be used to create many objects, all from the same flexible piece of code. It allows you to have many different functions, all with the same name, all doing the same job, but on different data

Benefits of OOP include:

- Modularity. Encapsulation enables objects to be self-contained, making troubleshooting and collaborative development easier.
- Reusability. Code can be reused through inheritance, meaning a team does not have to write the same code multiple times.
- Productivity. Programmers can construct new programs quicker through the use of multiple libraries and reusable code.
- Easily upgradable and scalable. Programmers can implement system functionalities independently.
- Interface descriptions. Descriptions of external systems are simple, due to message passing techniques that are used for objects communication.
- Security. Using encapsulation and abstraction, complex code is hidden, software maintenance is easier and internet protocols are protected.
- Flexibility. Polymorphism enables a single function to adapt to the class it is placed in. Different objects can also pass through the same interface.

- Popular pure OOP languages include: Ruby, Scala, JADE, Emerald
- Programming languages designed primarily for OOP include: Java, Python, C++
- Other programming languages that pair with OOP include: Visual Basic .NET, PHP,JavaScript

2 – Interface vs Abstract class ?

Interfaces are rules which works as a common understanding document among various teams in software development.

Interfaces give the idea what is to be done but not how it will be done. So implementation completely depends on developer by following the given rules (means given signature of methods).

Abstract classes may contain abstract declarations, concrete implementations, or both.

Abstract declarations are like rules to be followed and concrete implementations are like guidelines (you can use it as it is or you can ignore it by overriding and giving your own implementation to it).

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

The default implementation of the equals() method compares only the references of objects. But sometimes default implementation is not useful, and we want to compare the objects as per our requirements. In these types of situations, it is better to override the equals() method.

The hashCode() method is defined in Object class which is the super most class in Java. This method returns a hash code value for the object.

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 our 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?

Java does not support multiple inheritance (with classes). Due to this, Java does not support multiple inheritance, you cannot extend more than one other class. Still, if you try to do so, a compile time error is generated.

You can achieve multiple inheritance in Java, using the default methods (Java8) and interfaces.

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

When Java programs run on the JVM, objects are created on the heap, which is a portion of memory dedicated to the program. Eventually, some objects will no longer be needed. The garbage collector finds these unused objects and deletes them to free up memory.

Java garbage collection is an automatic process. The programmer does not need to explicitly mark objects to be deleted. The garbage collection implementation lives in the JVM.

Mark & Sweep Model is the underlying implementation in Java garbage collection. It has two major phases.

1. **Mark**: identify and mark all object references (starting with the GC roots) that are still used and reachable and the rest is considered garbage.
2. **Sweep**: traverse Heap and find unoccupied spaces between the live objects, these spaces are recorded in a free list and are made available for future object allocation.

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.

When you want to have a variable that always has the same value for every object of the class, forever and ever, make it static . If you have a method that does not use any instance variables or instance methods, you should probably make it static.

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

Immutable class 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.

Some of the benefits of immutable objects are:

- Thread safety
- Atomicity of failure
- Absence of hidden side-effects
- Protection against null reference errors
- Ease of caching
- Prevention of identity mutation
- Avoidance of temporal coupling between methods
- Protection from instantiating logically-invalid objects

To make immutable class in java you need to follow the below steps:

- It is necessary to mark the class with the final key so that it cannot be extended.
- Make all fields of the class private so no direct access is allowed.
- Do not provide setter methods for variables.

- Make all mutable fields final so they can only be assigned once.
- Have all fields initialize their values via the constructor.
- Perform cloning of objects outside of all mutable fields.

8 – Composition and Aggregation means and differences ?

Aggregation is an association that describes “has a” relationship between objects. For example, a classroom and student are linked with “has a” relationship.

The composition is a special type of aggregation that describes ownership. For example;

AGGREGATION VERSUS COMPOSITION

AGGREGATION	COMPOSITION
An association between two objects which describes the “has a” relationship	The most specific type of aggregation that implies ownership
Destroying the owning object does not affect the containing object	Destroying the owning object affects the containing object
Diamond symbol represents the aggregation in UML	Highlighted diamond symbol represents the composition in UML

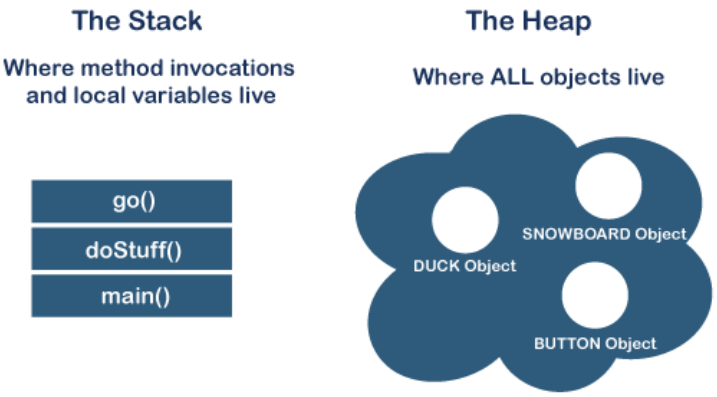
9 – Cohesion and Coupling means and differences ?

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	Coupling
Cohesion is defined as the degree of relationship between elements of the same module.	Coupling is defined as the degree of interdependence between the modules.
It's an intra-module approach	It's an inter-module approach
High cohesion is preferred due to improved focus on a particular task.	Low Coupling is preferred as it results in less dependency between the modules.
Cohesion is used to indicate a module's relative functional strength.	Coupling is used to indicate the relative independence among the modules.
In cohesion, the module focuses on a particular task.	In coupling, a particular module is connected to other modules.
Cohesion is also known by the name 'Intra-module Binding'.	Coupling is also known by the name 'Inter-module binding'.

10 - Heap and Stack means and differences ?

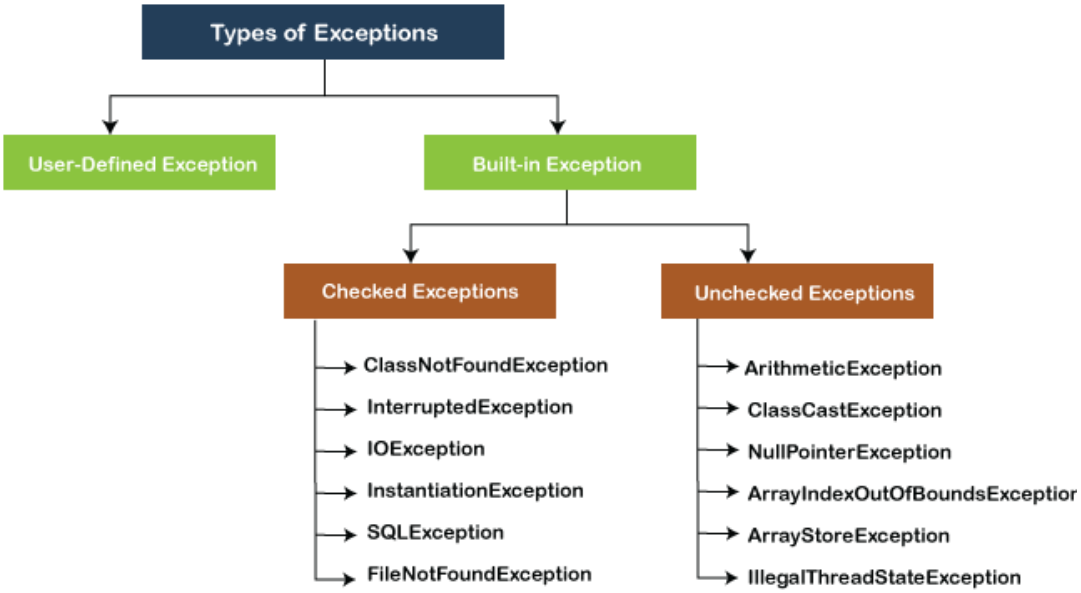
The major difference between Stack memory and heap memory is that the stack is used to store the order of method execution and local variables while the heap memory stores the objects and it uses dynamic memory allocation and deallocation. In this section, we will discuss the differences between stack and heap in detail.



Stack Vs Heap

11 – Exception means ? Type of Exceptions ?

Exceptions are the unwanted errors or bugs or events that restrict the normal execution of a program. Each time an exception occurs, program execution gets disrupted. An error message is displayed on the screen.



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 can be defined as, "if a subclass defines a static method with the same signature as a static method in the super class, in such a case, the method in the subclass hides the one in the superclass." The mechanism is known as method hiding. For Example;

```java
public class Animal {
    public static void foo() {
        System.out.println("Animal");
    }
}

public class Cat extends Animal {
    public static void foo() {  // hides Animal.foo()
        System.out.println("Cat");
    }
}
```

Here, Cat.foo() is said to hide Animal.foo(). Hiding does not work like overriding, because static methods are not polymorphic.

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

1) Abstraction allows a programmer to design software better by thinking in general terms rather than specific terms while Polymorphism allows a programmer to defer choosing the code you want to execute at runtime.

2) Another difference between Polymorphism and Abstraction is that Abstraction is implemented using abstract class and interface in Java while Polymorphism is supported by overloading and overriding in Java.

3) 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.