Homework 1

1- OOP ensures that the written code can be reused. Extensibility can be achieved by adding new features and functions to the code we have written.

The program written in OOP is more suitable for teamwork, thus increasing the performance.

An error can be detected more easily.

Complex projects are easier to manufacture and maintain.

Some of major OOP languages : Java , C ++, Phyton, Swift

2-Abstract vs Interface

Abstract class can contain static methods but Interface cannot contain overridable static methods.

An abstract class can have implementations of methods. In Interface, methods can only have signatures.

Abstract class can contain constructor and destructor. Interface cannot contain constructor or destructor. However, it may contain their signatures.

Abstract class can have a method with each access modifier (such as private, protected, public). Interface's method signatures can only have a 'public' access modifier.

An abstract class can inherit only from one class. Or, a class can only inherit from an abstract class in general. Interface can inherit from more than one interface.

Abstract class can express what the class derives from.Interface can express what capabilities the class has.

Abstract class should be used to gain common class behavior. Interface should be used for gaining common capability method.

The abstract class can specify what the object should do as well as how it should do it. Interface determines what the object should do, but not how it should do.

3- Uses of hashCode() and equals() Methods. equals(Object otherObject) – verifies the equality of two objects. Its default implementation simply checks the object references of two objects to verify their equality. Whenever we must override hashcode is invoked on the same object more than once during an execution of a Java application, the hashCode method must consistently return the same integer, provided no information used in equals comparisons on the object is modified.

4- The diamond problem is a common problem in Java when it comes to inheritance. There are different types of inheritance such as, single, multiple, multi-level, and hybrid inheritance. But Java does not support the multiple inheritance because of the diamond problem. Where one class can inherit properties from more than one class. It is known as the diamond problem. The solution to the diamond problem is default methods and interfaces. We can achieve multiple inheritance by using these two things.

5- 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- We can make a variable or method belonging to the class static, the variable we set as static will now be the common property of all objects created from that class. Non-static variables are created with the creation of the object and are located in the memory area of the object, while static variables take their place in memory with the creation of the class, so every time an object is created, memory is not repeatedly referenced for this variable to take up memory because the location of the static variable was opened when the class was created.

7- Immutable objects don't change their internal state in time, they are thread-safe and side-effects free. Because of those properties, immutable objects are also especially useful when dealing with multi-thread environments.

8- Composition and aggregation are two types of association which is used to represent relationships between two classes. In Aggregation , parent and child entity maintain Has-A relationship but both can also exist independently. We can use parent and child entity independently. In Composition, parent owns child entity so child entity can't exist without parent entity. We can't directly or independently access child entity.

9- Cohesion is the density of relevance between modules. Each module should have a single focus. Single focus generally means full cohesion. Good software design has high cohesion and low coupling.

10. 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. If a immutable value with declared dimensions is used during the program, we should use stack, if we use a mutable value, we should use heap.

11- Exceptions are exceptions that occur during program runtime .

Type of exceptions :

ArithmeticException

ArrayIndexOutOfBoundsException

ClassNotFoundException

FileNotFoundException

IOException

InterruptedException

NoSuchFieldException

NoSuchMethodException

NullPointerException

NumberFormatException

RuntimeException

StringIndexOutOfBoundsException

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

13- 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. It happens because static methods are resolved at compile time.

14- 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.

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.

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.