Basic Object-Oriented Programming Questions.

What is OOP?

Object-Oriented Programming (OOP) is a programming paradigm where the complete software operates as a bunch of objects talking to each other. An object is a collection of data and methods that operate on its data.

Why OOP?

The main advantage of OOP is better manageable code that covers the following:

a) The overall understanding of the software is increased as the distance between the language spoken by developers and the spoken by users.
b) Object orientation eases maintenance by the user of encapsulation. One can easily change the underlying representation by keeping the methods the same.

What are the main features of OOP?

The main feature or the principles of OOP are:

a) Inheritance
b) Encapsulation
c) Polymorphism
d) Abstraction

What is Inheritance?

The idea of inheritance is- that a class is based on another class and uses data and implementation of the other class. The purpose of inheritance is Code Reuse.

In other words- A subclass can inherit the states and behaviors of its superclass is known as inheritance.

There are mainly five types of Inheritance:

a) Single: The inheritance in which a single derived class is inherited from a single base class is known as the Single Inheritance.
b) Multiple: A child class inheriting states and behaviors from multiple parent classes is known as multiple inheritances. Java does not support multiple inheritances.
c) Multi-Level: A class can also be derived from one class, which is already derived from another class known as multi-level inheritance.
d) Hierarchical: When two or more classes get derived from a single base class, it is known as hierarchical inheritance. This gives us the freedom to use the same code with different scopes and flexibility in different classes.

e) Hybrid: A hybrid inheritance is a combination of more than one type of inheritance. For example when class A and B extend class C & another class D extends class A then this is a hybrid inheritance because it is a combination of single and hierarchical inheritance.

Types of classes in inheritance:

a) Parent class Superclass, Base class.
b) Child class Subclass, Derived class.

What is Encapsulation?

Encapsulation is an OOPS (Object-Oriented Programming System) concept to create and define the permission and restrictions of an object and its member variables and methods. A simple example to explain the concept is to make the member variables of a class private and provide public getter and setter methods. In a common word, encapsulation provides security to code.

Encapsulation refers to following two notions:

a) Data Hiding:

A language feature to restrict access to members of an object. For example, private and protected members in C++.

b) Bundling of data and methods together:

Data and methods that operate on that data and bundle together.

Types of encapsulations:

Java provides four types of access modifiers: Public, Private, Protected, and no Modifiers.

C# provides five types of access modifiers: Public, Private, Protected, Internal, Protected Internal.

What is Polymorphism?

Polymorphism means that some code or operations or objects behave differently in different contexts. In easy word, polymorphism is the occurrence of something in various forms. There are two types of polymorphism mostly known: Compile time polymorphism method overloading, Run time polymorphism method over ridding.

Run-time or dynamic polymorphism:

> Dynamic or Run time polymorphism is also known as method overriding in which a call to an overridden function is resolved during run time, not at the compile time. It means having two or more methods with the same name, and same signature but with different implementations.

a) Method over-loading: Method Overloading is a feature that allows a class to have more than one method having the same name if their argument lists are different. It is similar to constructor overloading in Java, which allows a class to have more than one constructor having different argument lists.
b) Method over-riding: Method overriding, in object-oriented programming, is a language feature that allows a subclass or child class to provide a specific implementation of a method that is already provided by one of its super-classes or parent classes.
c) Constructor over-loading: Constructors can be overloaded in a similar way as function overloading. Overloaded constructors have the same name (name of the class) but a different number of arguments. Depending upon the number and type of arguments passed, the corresponding constructor is called.
d) Operator overloading: Operator overloading is a function where different operators are applied and depend on the arguments. The operator can be used to pass through the function and has its own precedence to execute.
e) Polymorphic behavior of an object.

What is Abstraction?

Abstraction means “Hiding details”. Its main goal is to handle complexity by hiding unnecessary details from the user. Two types of abstraction are possible in OOP. Full Abstraction, Partial Abstraction. In abstraction, there is the Concrete class and an Abstract class. A concrete class can create both objects and references on the other hand Abstract class can only create references.

What is Exception handling?

An exception is an event that occurs during the execution of a program. Exceptions can be of any type — Run time exceptions, Error exceptions. Those exceptions are handled properly through exception handling mechanisms like try, catch and throw keywords.

What is the difference between Abstraction and Encapsulation?

“Program to interfaces, not implementations” is the principle for Abstraction, and “Encapsulate what varies” is the OO principle for Encapsulation.

Abstraction provides a general structure of a class and leaves the details for the implementers. Encapsulation is to create and define the permissions and restrictions of an object and its member variables and methods.

Abstraction is implemented in Java using the interface and abstract class while Encapsulation is implemented using four types of access level modifiers: public, protected, no modifier, and private.

What is the diamond problem in inheritance?

In case of multiple inheritances, suppose class A has two subclasses B and C, and class D has two super-classes B and C. If a method present in A is overridden by both B and C but not by D then from which class D will inherit that method B or C? This problem is known as a diamond problem.

What is the meaning of the “IS_A” and “HAS_A” relationship?

“IS-A” relationship implies inheritance. A sub-class object is said to have an “IS-A” relationship with the superclass or interface. If class A extends B then A “IS-A” B. It is transitive, that is, if class A extends B and class B extends C then A “IS-A” C. The “instance of” operator in java determines the “IS-A” relationship.

When class A has a member reference variable of type B then A “HAS-A” B. It is also known as Aggregation.

What is Static binding?

Static or early binding is resolved at compile time. Method overloading is an example of static binding. Binding is nothing but the association of a name with the class. Static binding is a binding in which a name can be associated with the class during compilation time, and it is also called an early Binding.

What is Dynamic binding?

Dynamic binding is a binding in which a name can be associated with the class during execution time, and it is also called Late Binding. Dynamic or late or virtual binding is resolved at run time. Method overriding is an example of dynamic binding.

What is a Class?

A class is the specification or template of an object.

What is an Object?

An object is an instance of a class.

What is the difference between a class and an object?

An object is an instance of a class. Objects hold any information, but classes don’t have any information. Definition of properties and functions can be done in class and can be used by the object. Classes can have sub-classes, and an object doesn’t have sub-objects.

What is the difference between structure and a class?

Structure default access type is public, but the class access type is private. A structure is used for grouping data whereas a class can be used for grouping data and methods. Structures are exclusively used for data and it doesn't require strict validation, but classes are used to encapsulate and inherit data that requires strict validation.

What is an abstract class?

An abstract class is a class that cannot be instantiated. The creation of an object is not possible with an abstract class, but it can be inherited. An abstract class can contain only the Abstract method. Java allows only abstract methods in abstract class while other languages allow non-abstract methods as well.

What is the Inline function?

Inline function is a technique used by the compilers and instructs to insert the complete body of the function wherever that function is used in the program source code.

What is an interface?

An interface is a collection of the abstract method. If the class implements an inheritance, and then thereby inherits all the abstract methods of an interface.

What is Association?

Association is a relationship between two objects with multiplicity.

What is Aggregation?

Aggregation is also known as the "HAS-A" relationship. When class Car has a member reference variable of type Wheel then the relationship between the classes Car and Wheel is known as Aggregation. Aggregation can be understood as a "whole to its parts" relationship.

What is Composition?

A composition is a special form of Aggregation where the part cannot exist without the whole. Composition is a strong Association. The composition relationship is represented as aggregation with one difference that the diamond shape is filled.

What is Dependency?

When one class depends on another because it uses that at some point in time then this relationship is known as Dependency. One class depends on another if the independent class is a parameter variable or local variable of a method of the dependent class. A Dependency is drawn as a dotted line from the dependent class to the independent class with an open arrowhead pointing to the independent class.

What is the difference between Association and Dependency?

The main difference between Association and Dependency is in the case of Association one class has an attribute or member variable of the other class type but in the case of Dependency a method takes an argument of the other class type or a method has a local variable of the other class type.

What is a constructor?

Constructor is a method used to initialize the state of an object, and it gets invoked at the time of object creation. Rules for constructors are: The constructors' name should be the same as the class name. The constructor must have no return type.

What is a destructor?

Destructor is a method that is automatically called when the object is made of scope or destroyed. Destructor name is also the same as the class name but with the tilde symbol before the name.

What is a copy constructor?

This is a special constructor for creating a new object as a copy of an existing object. There will be always only one copy constructor that can be either defined by the user or the system.

What is OOPS?

An Object-Oriented Programming System is the programming technique to write programs based on real-world objects. The stated behaviors of an object are represented as the member variables and methods. In OOPS programming, programs are organized around objects and data rather than actions and logic. The core concepts of OOPS are Abstraction, Encapsulation, Polymorphism, Inheritance, Composition, Association, Aggregation.

Advantages of OOPS:

a) Simplicity: OOPS programming objects model real-world objects, so the complexity is reduced and the program structure is clear.

b) Modularity: Each object forms a separate entity whose internal workings are decoupled from other parts of the system.
c) Modifiability: It is easy to make minor changes in the data representation or the procedures in an OO program. Changes inside a class do not affect any other part of a program, since the only public interface that the external world has to a class is through the use of methods.
d) Extensibility: adding new features or responding to changing operating environments can be solved by introducing a few new objects and modifying some existing ones.
e) Maintainability: Objects can be maintained separately, making locating and fixing problems easier.
f) Reusability: Objects can be reused in different programs.

What is the difference between Procedural programming and OOPS?

a) A procedural language is based on functions but object-oriented language is based on real-world objects.
b) Procedural language gives importance to the sequence of function execution but object-oriented language gives importance to the states and behaviors of the objects.
c) Procedural language exposes the data to the entire program but object-oriented language encapsulates the data.
d) Procedural language follows a top-down programming paradigm but object-oriented language follows a bottom-up programming paradigm.
e) A procedural language is complex in nature so it is difficult to modify, extend and maintain but object-oriented language is less complex in nature so it is easier to modify, extend and maintain.
f) Procedural language provides less scope for code reuse but object-oriented language provides more scope for code reuse.