OBJECT ORIENTED PROGRAMMING

1.What is OOPs? Explain its key features. Object-Oriented Programming(OOPs) is a type of programming that is based on objects rather than just functions and procedures. Individual objects are grouped into classes. OOPs implements real-world entities like inheritance, polymorphism, hiding, etc into programming. It also allows binding data and code together. Object-Oriented Programming (OOP) is a programming model that uses classes and objects. It's utilized to break down a software program into reusable code blueprints (called classes) that you may use to build specific instances of things. Its Key features are: Inheritance,Abstraction,Polymorphism,Encapsulation.

2.What is a class and how is it different from an object? Class is a logical entity that has some specific attributes and methods.It is defined as the collection of objects. Class is a user-defined datatype that has its own data members and member functions whereas an object is an instance of class by which we can access the methods of the class,also it is a real world entity.

3.What is a purpose of an init method in a class, and how is it used? init () is a special method, which is called class constructor or initialization method that Python calls when you create a new instance of that particular class The ___init___ class is used to create instances and set initial values for its attributes. This method is called automatically whenever a new object of a class is created. This type of function is also known as the constructor function. A constructor function is a method that is called every time a new class object is created.

4.What is inheritance and how is it implemented? Inheritance allows us to define a class that inherits all the methods and properties from another class. The Parent class is the class being inherited from, also called base class. The Child class is the class that inherits from another class, also called derived class. Inheritance is a process of obtaining properties and characteristics(variables and methods) of another class. By using Inheritance, we can define a new class with a little or no changes to the existing class. It provides reusability of the code. To Implement it, we have to create a parent class and with constructor and method and then we have to inherits the the properties of parent class to a new class which is a child class.

5.What is polymorphism, and how is it implemented? The ability of a class to provide different implementations of a method, depending on the type of object that is passed to the method.It defines that one task can be performed in different ways.

6.What is encapsulation, and how is it achieved? Encapsulation is one of the feature of Object Oriented Programmings. It is used to restrict access to methods and variables.It provides well defined and readable code and provides security. It is achieved through the use of access modifier such as Provate,Protector and Public.

7.What is the difference between a private and a protected attribute or method

in a class? Private means that the attribute/method can only be used inside the class where it is defined. Protected means that the attribute/method can only be used in the class where it is defined or its subclasses.

8.What is method overriding and how is it implemented? Method overriding is a feature of object-oriented programming languages where the subclass or child class can provide the program with specific characteristics or a specific implementation process of data provided that are already defined in the parent class or superclass. We can achieve this as the "+" operator is overloaded by the "int" class and "str" class.

9.How does Python support multiple inheritance and what are its benefits and drawbacks? Python allows you to inherit from two different classes by specifying them between parenthesis in the class declaration. Multiple Inheritance is an object or class can inherit characteristics and features from more than one parent object or the parent class. Pros: It allows a class to inherit the functionality of more than one base class; thus allowing for modeling of complex relationships. You categorize classes in many different ways. Multiple inheritance is a way of showing our natural tendency to organize the world. During analysis, for example, we use multiple inheritance to capture the way users classify objects. By having multiple superclasses, your subclass has more opportunities to reuse the inherited attributes and operations of the superclasses. Application development time is less and application takes less memory.

Cons: It can lead to a lot of confusion when two base classes implement a method with the same name. The more superclasses your subclass inherits from, the more maintenance you are likely to perform. If one of the superclasses happens to change, the subclass may have to change as well.

10.What are abstract classes and methods in Python OOPs, and how are they used? An abstract class in Python is typically created to declare a set of methods that must be created in any child class built on top of this abstract class. Similarly, an abstract method is one that doesn't have any implementation.

11. What is the difference between a static method and a class method, and when would you use each? Class methods don't need a class instance. They can't access the instance (self) but they have access to the class itself via cls . Static methods don't have access to cls or self . They work like regular functions but belong to the class's namespace. classmethods can serve as alternative constructors.Class methods are typically useful when we need to access the class itself.We call a static method by preceding it with the class name and using dot-notation.