- **What is object-oriented programming in general terms?**

  Object-oriented programming (OOP) is a programming paradigm based on the concept of "objects", which can contain data and code: data in the form of fields (often known as attributes or properties), and code, in the form of procedures (often known as methods).

- **What is a class?**

  In object-oriented programming, a class is a blueprint for creating objects (a particular data structure), providing initial values for state (member variables or attributes), and implementations of behavior (member functions or methods).

- **What is an object?**

  In object-oriented programming (OOP), objects are the things you think about first in designing a program and they are also the units of code that are eventually derived from the process. ... Each object is an instance of a particular class or subclass with the class's own methods or procedures and data variables.

- **What is an instance?**

  In object-oriented programming (OOP), an instance is a concrete occurrence of any object, existing usually during the runtime of a computer program. ... Each realized variation of that object is an instance of its class. That is, it is a member of a given class that has specified values rather than variables.

- **What is a property?**

  A property, in some object-oriented programming languages, is a special sort of class member, intermediate in functionality between a field (or data member) and a method.

- **What is a method?**

  A method in object-oriented programming (OOP) is a procedure associated with a message and an object. ... This allows the sending objects to invoke behaviors and to delegate the implementation of those behaviors to the receiving object. A method in Java programming sets the behavior of a class object.

- **What is the difference between a function and a method?**

  A method, like a function, is a set of instructions that perform a task. The difference is that a method is associated with an object, while a function is not.

- **What is a constructor?**

  In class-based object-oriented programming, a constructor (abbreviation: ctor) is a special type of subroutine called to create an object. It prepares the new object for use, often accepting arguments that the constructor uses to set required member variables.

- **What is the difference between a class, an object and an instance?**

  A class is a blueprint which you use to create objects. An object is an instance of a class - it's a concrete 'thing' that you made using a specific class. So, 'object' and 'instance' are the same thing, but the word 'instance' indicates the relationship of an object to its class.

- **What do we understand about the concept of encapsulation?**

  Encapsulation in OOP Meaning: In object-oriented computer programming languages, the notion of encapsulation (or OOP Encapsulation) refers to the bundling of data, along with the methods that operate on that data, into a single unit. Many programming languages use encapsulation frequently in the form of classes.

- **What do we understand about the concept of abstraction?**

  In object-oriented programming, abstraction is one of three central principles (along with encapsulation and inheritance). Through the process of abstraction, a programmer hides all but the relevant data about an object in order to reduce complexity and increase efficiency.

- **What do we understand about the concept of inheritance?**

  Inheritance is the procedure in which one class inherits the attributes and methods of another class. The class whose properties and methods are inherited is known as the Parent class.

- **What do we understand about the concept of polymorphism?**

  “In programming languages and type theory, polymorphism is the provision of a single interface to entities of different types, or the use of a single symbol to represent multiple different types.”

- **What do we understand about the concept of Overload?**

  Overloading refers to the ability to use a single identifier to define multiple methods of a class that differ in their input and output parameters. Overloaded methods are generally used when they conceptually execute the same task but with a slightly different set of parameters.

- **What do we understand about the concept of Override?**

  Overriding is an object-oriented programming feature that enables a child class to provide different implementation for a method that is already defined and/or implemented in its parent class or one of its parent classes. ... Overriding enables handling different data types through a uniform interface.

- **What differences exist between the concept of Overload and Override?**

  In the method overloading, methods or functions must have the same name and different signatures. Whereas in the method overriding, methods or functions must have the same name and same signatures.

- **What is a static class?**

  A static class is basically the same as a non-static class, but there is one difference: a static class cannot be instantiated. In other words, you cannot use the new operator to create a variable of the class type.

- **Look for 3 advantages over object-oriented programming compared to other programming paradigms**
  - We can build the programs from standard working modules that communicate with one another, rather than having to start writing the code from scratch which leads to saving of development time and higher productivity,

    - OOP language allows to break the program into the bit-sized problems that can be solved easily (one object at a time).
    - The new technology promises greater programmer productivity, better quality of software and lesser maintenance cost.

- **Look for disadvantages of this paradigm.**
    - The length of the programmes developed using OOP language is much larger than the procedural approach. Since the programme becomes larger in size, it requires more time to be executed that leads to slower execution of the programme.
    - We can not apply OOP everywhere as it is not a universal language. It is applied only when it is required. It is not suitable for all types of problems.
    - Programmers need to have brilliant designing skill and programming skill along with proper planning because using OOP is little bit tricky.
    - OOPs take time to get used to it. The thought process involved in object-oriented programming may not be natural for some people.
    - Everything is treated as object in OOP so before applying it we need to have excellent thinking in terms of objects.