Q.1)What Is Object-Oriented Programming?

```
1>> object oriented programming language is a programming language that uses
classes and objects in the programing
2>> Object oriented programming language aims is to used the real world
entities such as inheritance,Encapsulation and
    Polymorphism etc.
3>> The main concept of object oriented programming is to bind the data and
function all together in the single unit
    so that no other parts of the code can access this data
```

Q.2)Difference between Procedural programming and OOPs?

```
Procedural programming Vs OOPs programming:
1>> Procedural programming is a programming language which uses a step by step
approach to break down a task into collection of
    variable and routines and sub-routines through a sequence of instruction.
2>> In procedural programming the main programm is divided into small parts
based on the functions and is treated as separate
    program for individual smaller program.
3>> In procedural programming there is no such access modifiers.
4>> It is less secure than object oriented programming language



OOPs programming:
1>> Object-oriented Programming is a programming language that uses classes and
objects to create models based on the real
    world environment.
2>> In object-oriented Programming language the program is divided into class
and object and object is the instance of the class.
3>> In object-oriented Programming language there is access modifiers such as
public and private variable or methods.
4>> It is more secure than Procedural programming language
```

Q.3)What are the fundamental principles/features of Object-Oriented Programming?

```
There are the four main funcdamental principle of object oriented programming
language:
1>> Inheritance
2>> Encapsulation
3>> Polymorphism
4>> Abstraction
```

Q.4)What is an object?

```
1>> Objects are the real world entity
2>> Objects is the instance of the class
```

Q.5)What is a class?

```
1>> Class is the blue print of an object
2>> Class contain variable and methods
```

Q.6)What is the difference between a class and an object?

```
   Class:
1>> A class is a blueprint from which we can create an object
2>> A class is used to bind the variable and method in a single unit
3>> Class have a logical existence
4>> Class doesn't take any memory space while creating a class
5>> Class has to be declared only onces

   Objects:
1>> An object is the instance of class by which we can access the variable and
method of class
2>> Object act's like a variable of the class
3>> Object have a physical existence
4>> An object take memory when we create an object of the class
5>> Object can be declared multiple times depending on the requirement
```

Q.7) Can you call the base class method without creating an instance?

```
Yes, We can do that but in that case we need to inherit the base class in the
child class.
```

Q.8)What is inheritance?

```
1>> Inheritance is the one of the fundamental features of object oriented
programming language
2>> Inheritance is used to access the variable and methods from the base class
into the derived class
3>> with the help of inheritance we can reduse or reuse the block of code
```

Q.9)What are the different types of inheritance?

```
The Inheritance is divided into three types:
1)Normal or Single Inheritance:In Normal Inheritance there is only one Base and
Derived class
```

```
2)Multy Inheritance: In Multy Inheritance there is One Base class and can be
multiple Derived class
3)Multiple Inheritance: In Multiple Inheritance There is one Derived class with
multiple Base class
```

Q.10)What is the difference between multiple and multilevel inheritances?

```
Multy Inheritance: In Multy Inheritance there is One Base class and can be
multiple Derived class
Syntax is:
    class GrandFather(): # Base Class
        def __init__(self):
            print("we are in grandfather class")
    class Father(GrandFather): # Derived Class of above class and it is a base
class for the below class
        def __init__(self):
            print("we are in father class")
    class Son(Father):  # Derived Class of the above class
        def __init__(self):
            print("we are in Son class")

Multiple Inheritance: In Multiple Inheritance There is one Derived class with
multiple Base class
Syntax is:
    class Father(): # Base Class
        def __init__(self):
            print("we are in Father class")
    class Mother(): # Base Class
        def __init__(self):
            print("we are in Mother class")
    class Son(Father,Mother):  # Derived Class of all the above class
        def __init__(self):
            print("we are in Son class")
```

Q.11)What are the limitations of inheritance?

```
1>>Inherited functions work slower than normal function as there is
indirection.
2>>Improper use of inheritance may lead to wrong solutions.
3>>Often, data members in the base class are left unused which may lead to
memory wastage.
4>>Inheritance increases the coupling between base class and derived class.
```

Q.12)What are the superclass and subclass?

Superclass 1>>Superclass is also known as parent class and base class 2>>super class is the parent or base class for the derived class

Subclass 1>>Subclass is also known as child class and derived class 2>>Subclass is the derived class where we can access all the properties of base class or super class

Q.13)What is the super keyword?

Super keyword is used to call the **init** method of base class into child class syntax is : super.**init**()

Q.14)What is encapsulation?

```
1>> Encapsulation is one of the fundamental concepts in object-oriented
programming.
2>> It describes the idea of wrapping data and the methods that work on data
within one unit.
3>> This puts restrictions on accessing variables and methods directly and can
prevent the accidental modification of data.
```

Q.15)What is the name mangling and how does it work?

```
Name mangling: It is used to access or modify the private variable or private
method from outside of the class
Syntax is:
    obj._ClassName__Variable/method()
```

Q.16)What is the difference between public and private access modifiers?

```
Public access modifier:
    By default all the variable or method in a class is of public type we can
access as well as modified the public variable
    and public method of class
    e.g x = 10  # public variable
        def add() #public method

Private access modifier:
    By default all the variable or method in a class is of public type we can
convert the public variable and public method
    of class into private by puting (__) at the start of variable or
method,This will restict the access to that variable
    or method outside of the class
    e.g __x = 10  #Private variable
        def __add() #Private method
```

Q.17)Is Python 100 percent object-oriented?

```
Python supports all the concept of "object oriented programming" but it is NOT
fully object oriented because
The code in Python can also be written without creating classes
```

Q.18)What is data abstraction?

```
1>> Abstraction is used to hide the internal functionality of the function from
the users.
2>>The users only interact with the basic implementation of the function, but
inner working is hidden.
3>>User is familiar with that "what function does" but they don't know "how it
does."
```

Q.19)How to achieve data abstraction?

```
1>>In Python, abstraction can be achieved by using abstract classes and
interfaces.
2>>A class that consists of one or more abstract method is called the abstract
class.
3>>Abstract methods do not contain their implementation.
4>>Abstract class can be inherited by the subclass and abstract method gets its
definition in the subclass.
```

Q.20)What is an abstract class?

```
1>> A class containing one or more abstract methods is called an abstract
class.
2>> Abstract methods do not contain any implementation. Instead, all the
implementations can be defined in the methods
of sub-classes that inherit the abstract class.
3>> An abstract class is created by importing a class named 'ABC' from the
'abc' module and inheriting the 'ABC' class.

Syntax is:
from abc import ABC
Class ClassName(ABC):
```

Q.21)Can you create an object of an abstract class?

```
No, We can not create an object of an abstract class
```

Q.22)Differentiate between data abstraction and encapsulation

```
Abstraction:
1. Abstraction works on the design level.
```

```
2. Abstraction is implemented to hide unnecessary data and withdrawing relevant
data.
3. It highlights what the work of an object instead of how the object works is.
4. Abstraction focuses on outside viewing, for example, shifting the car.
5. Abstraction is supported in Python with the help of abc module.
Encapsulation:
1. Encapsulation works on the application level.
2. Encapsulation is the mechanism of hiding the code and the data together from
the outside world or misuse.
3. It focuses on the inner details of how the object works. Modifications can
be done later to the settings.
4. Encapsulation focuses on internal working or inner viewing, for example, the
production of the car.
5. Encapsulation is supported using, e.g. public, private and secure access
modification systems.
```

Q.23)What is polymorphism?

```
1>> The word Polymorphism stands for many forms
2>> When the methods have the identical name but the functionality is different
then we can say it is a polymorphism.
```

Q.24)What is the overloading method?

```
1>> When the method of base class is present in the child class with same
method name but with differnet functionality
    then it is called the method overloading
2>> while calling the same method by creating the object of child class will
over load the methods of base class with
    the method present in the child class

E.g
class Father():
    def gender(self):
        print("MALE")

class Child(Father):
    def gender(self):
        print("FEMALE")

Obj = Child()
Obj.gender()  # here we do the method over loading the output will be (FEMALE)
```

Q.25)What are the limitations of OOPs?

```
Following are the Limitation of OOPs:
1>>he length of the programmes developed using OOP language is much larger than
the procedural approach.
2>>The programme becomes larger in size, it requires more time to be executed
that leads to slower execution of the programme.
```

```
3>>We can not apply OOP everywhere as it is not a universal language. It is
applied only when it is required.
4>>Programmers need to have brilliant and programming skill along with proper
planning because using OOP is little bit tricky.
5>>OOPs take time to get used to it.
6>>The thought process involved in object-oriented programming may not be
natural for some people.
7>>Everything is treated as object in OOP so before applying it we need to have
excellent thinking in terms of objects.
```