Q1 What Is Object-Oriented Programming?

```
1)object-oriented Programming (OOPs) is a programming paradigm that uses objects and classes in
programming.
2)It aims to implement real-world entities like inheritance, polymorphisms, encapsulation, etc. in
the programming.
3)The main concept of OOPs is to bind the data and the functions that work on that together
```

Q2 Difference between Procedural programming and OOPs?

```
  Procedural Oriented Programming                        Object Oriented Programming
1)In procedural programming, program is      1)In object oriented programming, program is
  divided into small parts called functions.   divided into small parts called objects.

2)Procedural programming follows top         2)Object oriented programming follows bottom
  down approach.                               up approach.

3)There is no access specifier in            3)Object oriented programming have access
procedural programming.                       specifiers like private, public, protected
                                               etc.

4)Adding new data and function is not easy.    4)Adding new data and function is easy.

5)Procedural programming does not have any     5)Object oriented programming provides data
proper way for hiding data so it is less secure. hiding so it is more secure.

6)In procedural programming, overloading       6)Overloading is possible in object oriented
is not possible.                                 programming.

7)In procedural programming, function is       7)In object oriented programming, data is
more important than data.                       more important than function.

8)Procedural programming is based on           8)bject oriented programming is based on real world
unreal world.

9)Examples: C, FORTRAN, Pascal, Basic etc.     9)Examples: C++, Java, Python, C# etc.
```

Q3 What are the fundamental principles/features of Object-Oriented Programming?

In [ ]:

```
There are 4 major principles that make an language Object Oriented.
These are Encapsulation, Data Abstraction, Polymorphism and Inheritance.
These are also called as four pillars of Object Oriented Programming.
1)Inheritance:-
    Inheritances expresses “is-a” and/or “has-a” relationship between two objects.
    Using Inheritance, In derived classes we can reuse the code of existing super classes.
    In Java, concept of “is-a” is based on class inheritance (using extends) or interface
    implementation (using implements).
2)Polymorphism
    It means one name many forms. It is further of two types — static and dynamic.
    Static polymorphism is achieved using method overloading and dynamic polymorphism
    using method overriding. It is closely related to inheritance. We can write a code
    that works on the superclass, and it will work with any subclass type as well.
3)Encapsulation
    Encapsulation is the mechanism of hiding of data implementation by restricting access
    to public methods. Instance variables are kept private and accessor methods are made
    public to achieve this.
4)Abstraction
    Abstract means a concept or an Idea which is not associated with any particular
    instance.Abstract means a concept or an Idea which is not associated with any particular
```

Q4 What is an object?

```
    Consider your mobile phone as an object. There can be different properties for your mobile phone like its model, software version, and memory in it. This object can also have
functions like switch on the camera, turn off Bluetooth, restart, etc. In simple words,
```

Q5 What is a class?

```
This is another important term in object-oriented programming. A class is like a template
from which new objects are created. Any class you create will always have a head and a
```

```
body. A head typically includes modifiers and the keyword of the class while the body
includes data members and member functions.

Here are the different components of a class -

    1)Public - The class members can be accessed from everywhere.
    2)Private - The class members can only be accessed by the defining class
    3)Protected - the class members can only be accessed by parent and inherited classes

```

Q6 What is the difference between a class and an object?

In [ ]:

```
            Class                                                  Object
Class is used as a template for                  An object is an instance of a class.
declaring and creating the objects.

When a class is created, no memory               Objects are allocated memory space
is allocated.                                    whenever they are created.

The class has to be declared only once.          An object is created many times as per
                                                  requirement.
A class cannot be manipulated as they are not    Objects can be manipulated.
available in the memory.

A class is a logical entity.                      An object is a physical entity.

It is declared with the class keyword            It is created with a class name in C++ and
                                                 with the new keywords in Java.

Class does not contain any values which          Each object has its own values, which are
can be associated with the field.                associated with it.

A class is used to bind data as well as          Objects are like a variable of the class.
```

Q7 Can you call the base class method without creating an instance?

```
Yes,it is possible
```

```
1)If it is a static Method
2)By inheriting from that class
3)From derived class using base Key word
```

Q8 What is inheritance?

```
Inheritance is the capability of one class to derive or inherit the properties from
another class. The benefits of inheritance are:
    1)It represents real-world relationships well.
    2)It provides reusability of a code. We don’t have to write the same code again and
      again. Also, it allows us to add more features to a class without modifying it.
    3)It is transitive in nature, which means that if class B inherits from another
```

Q9 What are the different types of inheritance?

In [ ]:

```
1. Single inheritance:
        When a child class inherits from only one parent class, it is called single
        inheritance. We saw an example above.
2. Multiple inheritance:
        When a child class inherits from multiple parent classes,it is called multiple
        inheritance. Unlike Java and like C++, Python supports multiple inheritance.
        We specify all parent classes as a comma-separated list in the bracket.
3.Multilevel inheritance:
    When we have a child and grandchild relationship.
```

Q10 What is the difference between multiple and multilevel inheritances?

In [ ]:

```
Multiple Inheritance                       Multilevel Inheritance
Multiple Inheritance is an Inheritance     Multilevel Inheritance is an Inheritance type
type where a class inherits from more      that inherits from a derived class, making that
than one base class.                       derived class a base class for a new class.
                                      Usage

Multiple Inheritance is not widely used    Multilevel Inheritance is widely used.
```

```
because it makes the system more complex.
                                    Class Levels

Multiple Inheritance has two class levels    Multilevel Inheritance has three class levels namely,
```

Q11 What are the limitations of inheritance?

In [ ]:

```
Disadvantages:-

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

Advantages:

    1.Inheritance promotes reusability. When a class inherits or derives another class,
    2.it can access all the functionality of inherited class.
    3.Reusability enhanced reliability. The base class code will be already tested
      and debugged.
    4.As the existing code is reused, it leads to less development and maintenance costs.
    5.Inheritance makes the sub classes follow a standard interface.
    6.Inheritance helps to reduce code redundancy and supports code extensibility.
    7.Inheritance facilitates creation of class libraries.
```

Q12 What are the superclass and subclass?

```
In object-oriented programming languages, classes can be derived from other classes.
The derived class (the class that is derived from another class) is called a subclass.
```

Q13 What is the super keyword?

```
# In an inherited subclass, a parent class can be referred to with the use of the super()
function. The super function returns a temporary object of the superclass that allows
```

```
access to all of its methods to its child class.
****The benefits of using a super function are:-
    1.Need not remember or specify the parent class name to access its methods.
    2.This function can be used both in single and multiple inheritances.
    3.This implements modularity (isolating changes) and code reusability as there is no
      need to rewrite the entire function.
    4.Super function in Python is called dynamically because Python is a dynamic
      language unlike other languages.
```

Q14 What is encapsulation?

In [ ]:

```
Encapsulation
    Encapsulation is the mechanism of hiding of data implementation by restricting access
    to public methods. Instance variables are kept private and accessor methods are made
```

Q15 What is the name mangling and how does it work?

In [ ]:

```
In name mangling process any identifier with two leading underscore and one trailing
underscore is textually replaced with _classname__identifier where classname is the
name of the current class. It means that any identifier of the form __geek (at least
two leading underscores or at most one trailing underscore) is replaced with
_classname__geek, where classname is the current class name with leading underscore(s)
stripped.
Name mangling process
    With the help of dir() method, we can see the name mangling process that is done to
    the class variable. The name mangling process was done by the Interpreter. The dir()
    method is used by passing the class object and it will return all valid attributes
    that belong to that object.
Accessing Name Mangled variables
    The name mangling process helps to access the class variables from outside the class.
    The class variables can be accessed by adding _classname to it. The name mangling is
    closest to private not exactly private.
```

Q16 What is the difference between public and private access modifiers?

In [ ]:

```
Public Access Modifier:
        The members of a class that are declared public are easily accessible from any
    part of the program. All data members and member functions of a class are public by
    default.

Protected Access Modifier:
    The members of a class that are declared protected are only accessible to a class
    derived from it. Data members of a class are declared protected by adding a single
    underscore '_' symbol before the data member of that class.
Private Access Modifier:
        The members of a class that are declared private are accessible within the class
    only, private access modifier is the most secure access modifier. Data members of a
    class are declared private by adding a double underscore '__' symbol before the data
    member of that class.
```

Q17 Is Python 100 percent object-oriented?

```
Yes. Python is an object-oriented programming language.
You can write programs in Python either in a procedural way or in an object-oriented way.

Why Python is not pure object oriented language?
        Python supports most of the terms associated with
    "objected-oriented" programming language except strong encapsulation.
    It is not completely Object oriented because Guido never believed in hiding things
```

Q18 What is data abstraction?

```
    Data abstraction is one of the most essential and important feature of object
oriented programming in python. Abstraction means displaying only essential information
and hiding the details. Data abstraction refers to providing only essential information
```

Q19 How to achieve data abstraction?

In [ ]:

```
In Python, abstraction can be achieved by using abstract classes and interfaces.
```

```
    A class that consists of one or more abstract method is called the abstract class.
    Abstract methods do not contain their implementation. Abstract class can be inherited
    by the subclass and abstract method gets its definition in the subclass. Abstraction
    classes are meant to be the blueprint of the other class. An abstract class can be
    useful when we are designing large functions. An abstract class is also helpful to
    provide the standard interface for different implementations of components. Python
    provides the abc module to use the abstraction in the Python program. Let's see the
    following syntax.
            from abc import ABC
            class ClassName(ABC):
        Python doesn't provide the abstract class itself. We need to import the abc module,
    which provides the base for defining Abstract Base classes (ABC). The ABC works by
    decorating methods of the base class as abstract. It registers concrete classes as
    the implementation of the abstract base. We use the @abstractmethod decorator to
    define an abstract method or if we don't provide the definition to the method, it
```

Q20 What is an abstract class?

In [ ]:

```
        In Python, abstraction can be achieved by using abstract classes and interfaces.
    A class that consists of one or more abstract method is called the abstract class.
    Abstract methods do not contain their implementation. Abstract class can be inherited
    by the subclass and abstract method gets its definition in the subclass. Abstraction
    classes are meant to be the blueprint of the other class. An abstract class can be
    useful when we are designing large functions. An abstract class is also helpful to
    provide the standard interface for different implementations of components. Python
    provides the abc module to use the abstraction in the Python program. Let's see the
    following syntax.
            from abc import ABC
```

Q21 Can you create an object of an abstract class?

In [ ]:

```
Abstract classes are incomplete because they have methods that have nobody.
If python allows creating an object for abstract classes then using that object if
anyone calls the abstract method, but there is no actual implementation to invoke.
So we use an abstract class as a template and according to the need, we extend it
and build on it before we can use it. Due to the fact, an abstract class is not a concrete class,
```

Q22 Differentiate between data abstraction and encapsulation

In [ ]:

```
Encapsulation:
Encapsulation is the mechanism of hiding of data implementation by restricting access to public
methods. Instance variables are kept private and accessor methods are made public to achieve this.

Abstraction :
Abstract means a concept or an Idea which is not associated with any particular instance. Using
abstract class/Interface we express the intent of the class rather than the actual implementation.
a way, one class should not know the inner details of another in order to use it, just knowing the
```

Q23 What is polymorphism?

In [ ]:

```
It means one name many forms.
In Python, Polymorphism lets us define methods in the child class that have the same name as the
methods in the parent class. In inheritance, the child class inherits the methods from the parent
```

Q24 What is the overloading method?

In [ ]:

```
In Python, Polymorphism lets us define methods in the child class that have the same name as
the methods in the parent class. In inheritance, the child class inherits the methods from the par
class. However, it is possible to modify a method in a child class that it has inherited from the
class. This is particularly useful in cases where the method inherited from the parent class doesn

quite fit the child class. In such cases, we re-implement the method in the child class. This proc
```

Q25 What are the limitations of OOPs

In [ ]:

```
1.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.
2.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.
```

```
3.Programmers need to have brilliant designing skill and programming skill along with proper
planning because using OOP is little bit tricky.
4.OOPs take time to get used to it. The thought process involved in object-oriented programming
may not be natural for some people.
5.Everything is treated as object in OOP so before applying it we need to have excellent thinking
```