1. What Is Object-Oriented Programming?

In [ ]:
```
1>> OOPS full form is object oriented progtamming.

2>>OOP is pragramming paradigm uses class Classes are like BLUE PRINT of an obje


3>> Main Concepts of Object-Oriented Programming (OOPs) :- Class Objects Polymor
Inheritance

4>> Object oriented programming language aims is to used the real world
entities such as inheritance,Encapsulation and
 Polymorphism etc.

5>> 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
```

2. Difference between Procedural programming and OOPs?

In [ ]:
```
1)Procedural programming :-
    1.)In procedural programming, program is  divided into small parts called fu
    2.)Procedural programming follows top to down approach.
    3.)In procedural programming there is no such access modifiers.

    4. Adding new data and function is not easy.
    5. It is less secure than object oriented programming language

2)OOPs :-
    1.In object oriented programming, program is divided into small parts called
```

In [ ]:
```
      Procedural Oriented Programming                                    Object Oriented Prog

1)In procedural programming, program is           1)In object oriented programmin
 divided into small parts called functions.          divided into small parts calle

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

3)There is no access specifier in                 3)Object oriented programming ha
procedural programming.                              specifiers like private, publi

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

5)Procedural programming does not have any        5)Object oriented programming pr
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 obj
is not possible.                                      programming.

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

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

9)Examples: procedual langauge are used in        9)Examples: oops used in C++, Ja
   C, FORTRAN, Pascal, Basic etc.
```

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

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

4. What is an object?

In [ ]:
```
1.object is a real world entities i,e tablet ,car, house, mobile price ar real wo
2.object is a instance of class.
3.everything in python is object . string ,list ,dict are bject.
4.An object consists of :

    State: It is represented by the attributes of an object. It also reflects the

    Behavior: It is represented by the methods of an object. It also reflects the

    Identity: It gives a unique name to an object and enables one object to inter

        -e.g. dog has states asn like name, clor etc. and behavour like barking
```

5. What is a class?

In [ ]:
```
1.Class is the blue print of an object
2. Class contain variable and methods
3.blueprint  which is consisting of method and varible.
4.class is collection of multiple object.
```

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

In [ ]:
```
                Class                                              Object

1)Class is used as a template for                 1)An object is an instance of a
declaring and creating the objects.

2)When a class is created, no memory              2)Objects are allocated memory
is allocated.                                       whenever they are created.

3)The class has to be declared only once.         3)An object is created many ti
                                                     requirement.

4)A class cannot be manipulated as they are not    4) Objects can be manipulated
available in the memory.

5)A class is a logical entity.                      5)An object is a physical ent

6)It is declared with the class keyword             6)It is created with a , obj
                                                     with the new keywords in Jav

7)Class does not contain any values which Each      7)object has its own values,
can be associated with the field.                      associated with it.

8)A class is used to bind data as well as methods   8) Objects are like a variabl
 together as a single unit.
```

```
#   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
```

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

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



yes. a base class method can be called wityhot using instance.
the way is by using inheritance.
in inheritace other class which can be called child class inherits all the varial
and can be called using instance of child class.
```

8. What is inheritance?

In [ ]:
```
1>> Inheritance is the one of the fundamental features of object oriented
programming language

2>> Inheritnace meaning we can inherate one class properties in other class

3>>Inheritance is used to access the variable and methods from the base class
into the derived class

4>> with the help of inheritance we can reduse or reuse the block of code.
```

9. What are the different types of inheritance?

In [ ]:
```
1)Normal or Single Inheritance:In Normal Inheritance there is only one Base and
  Derived class

2)multilevel inheretance :-  features of the base class and the derived class ar
     inherited into the new derived class. This is similar to a relationship repr
    a child and grandfather. eg, grandfather >> father >> child

3)Multiple inheritance:   There is one Derived class with multiple Base class
                          e.g father >>child  and mother >> child

Hierarchical Inheritance: When more than one derived classes are created from a 
                          this type of inheritance is called hierarchical inheri
                    In this program, we have a parent (base) class and two child
```

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

In [ ]:
```
1)  multilevel inheretance :-  features of the base class and the derived class a
     inherited into the new derived class. This is similar to a relationship repr
    a child and grandfather. eg, grandfather >> father >> child

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

 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")


2)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")
```

11. What are the limitations of inheritance?

In [ ]:
```
Disadvantages:-

    1.Inherited functions work slower than normal function as there is indirecti

    2.Improper use of inheritance may lead to wrong solutions.
      Often, data members in the base class are left unused which may lead to mer


    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

    2.it can access all the functionality of inherited class.

    3.Reusability enhanced reliability. The base class code will be already teste
      and debugged.

    4.As the existing code is reused, it leads to less development and maintenan

    5.Inheritance makes the sub classes follow a standard interface.

    6.Inheritance helps to reduce code redundancy and support
```

12. What are the superclass and subclass?

In [ ]:
```
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 
```

13. What is the super keyword?

In [ ]:
```
1.Super keyword is used to call the init method of base class into child class
2.syntax is : super.__init__()
3.a parent class can be referred to with the use of the super() function.
4.Super keyword is used to call the  method of base class into 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
 need to rewrite the entire function.
 4.Super function in Python is called dynamically because Python is a dynamic
 language unlike other languages.
```

14. What is encapsulation?

In [ ]:
```
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.

4>>Encapulation is use to make methods and variables private to avoid modificati
methods cant be accesed from outside the class'

5>>Protecting data from modification of private variable/methods

6>>Syntax >> __variablename   for private varible
             __methodename    for private method

7>>if varible is privateor method is private then we can't accssed ya modified i
    outside the class
```

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

In [ ]:
```
1.Name mangling: It is used to access or modify the private variable or private
                 method from outside of the class

2.Syntax is:
 obj._ClassName__Variable/method()
 Object._ClassName__privatevariable
 Object._ClassName__privatemethod


3.e.g
class Employee():
    def _init_(self,name):
        self.name=name

    def emp_det(self,empname):
        __empname="sagar"
        print("empname", __empname)

     obj=Employee()

obj._Employee.__empname()        >> o/p >> "sagar"
```

16. 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
default.

Protected Access Modifier:
The members of a class that are declared protected are only accessible to a clas
derived from it. Data members of a class are declared protected by adding a singl
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 o
class are declared private by adding a double underscore '__' symbol before the
member of that class.
```

17. Is Python 100 percent object-oriented?

In [ ]:
```
1.Python supports all the concept of "object oriented programming" but it is NOT
fully object oriented because

2.The code in Python can also be written without creating classes

3.Why Python is not pure object oriented language?
 Python supports most of the terms associated with OOP language except strong en
 It is not completely Object oriented because Guido never believed in hiding thi
```

18. What is data abstraction?

In [ ]:
```
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."

4>> Abstraction means displaying only essential information
and hiding the details.

5>>' Abstraction is use to defined specific methods those are going to used in e
Abstraction cant be used for implemenation only use for declaration
```

19. How to achieve data abstraction?

In [ ]:
```
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.

5>>In Python, we can achieve abstraction using ABC (abstraction class) or abstra
ABC is a class from the abc module in Python. ...

6>>When we annotate any method with an abstractmethod keyword, then it is an abs
```

20. What is an abstract class?

In [ ]:
```
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.

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

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

In [ ]:
```
No, We can not create an object of an abstract class
```

In [ ]:
```
1.Abstract classes are incomplete because they have methods that have nobody.

2.If python allows creating an object for abstract classes then using that objec
anyone calls the abstract method,

3.but there is no actual implementation to invoke.

4.So we use an abstract class as a template and according to the need, we extend
and build on it before we can use it.

5.Due to the fact, an abstract class is not a concrete class
```

22. Differentiate between data abstraction and encapsulation

In [ ]:
```
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.
```

23. What is polymorphism?

In [ ]:
```
1>> The word Polymorphism stands for many forms or It means one name many forms.

2>> When the methods have the identical name but the functionality is different
then we can say it is a polymorphism.
```

24. What is the overloading method?

In [ ]:
```
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

3>>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 (FEMA
```

25. What are the limitations of OOPs?

In [ ]:
```
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.
```

In [ ]: