Sachin Rao

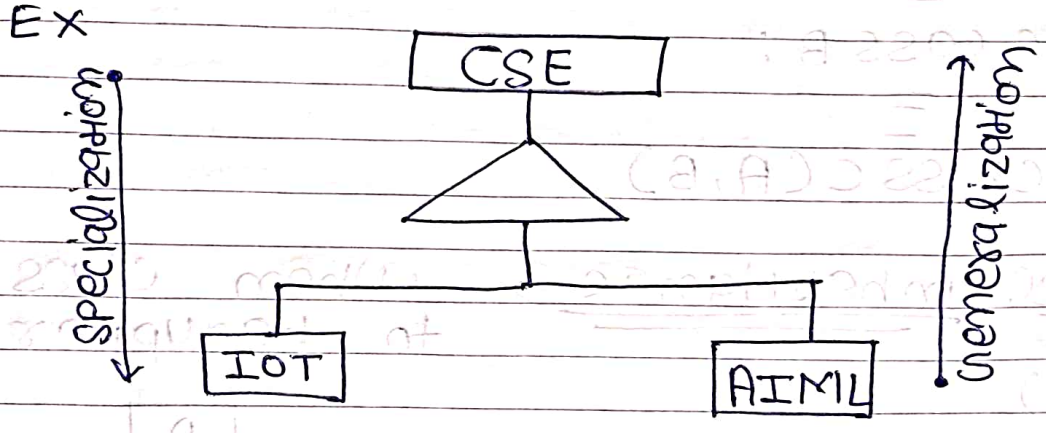
Date

Q1 Differentiate between
a) Generalization and specialization
b) Abstract method and concreate method
c) multiple and multi level inheritance

Ans a) → Generalization → The process of extracting common charcter from two or more classes and combining them into a generalized super class is called Generalization.
EX → CSE

Specialization → is the reverse process of Generalization. EX AIML, DS

EX



Abstract method → which happens inside the method class. and If you don't do anything there, It is called abstract method

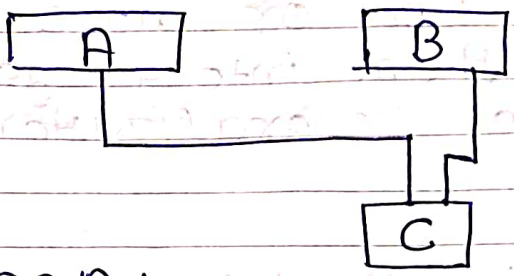
EX
```
class A:
    def funcself):
        pass → implemented
```
OR
that is declared but not implement-

Date __/__/__

Concrete method -: A method that is implemented in Abstract class itself is called Concrete method.

multiple inheritance -: when two class different, but third class inside the diffrent class

EX →

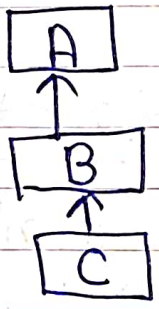


code

```
class A:
    ==
class B:
    ==
class C(A,B)
```

multilevel inheritance → when class connect to beaupper class (only one)

EX




```
class A:
    ==
classB(A):
    ==
class C(B):
    ==
```

Q.2 What is inheritance? what are it's adv

Ans The method of inheriting the properties of parent class into a child class is known as inheretance.
It is transitive in nature. If a child inherits properties from a parent class, then, all other sub-classes of the child class will also inherent the properties of the parent class.

Q.3
Solve

```
Class Theory :
    marks = [ ]
    def getdata (self, name, m1, m2, m3)
        Theory. name = name
        Theory. marks. append (m1)
        Theory. marks. append (m2)
        Theory. marks. append (m3)
    def display (self) :
        print ("Student details")
        print ("name:", Theory.name)
        print ("marks:, self. total())
        print ("marks:, self.averag()
    def total (self):
        m = theory.marks
        return m[0] + m[1] + m[2]
    def averag (self),
        m = (theory. marks) /3
        return m[0], + m[1] + m[2]
```

```
class Lab:
    Labmarks = [ ]
    def getdata(self, P1, P2)
        Lab.Labmarks.append(P1)
        Lab.Labmarks.append(P2)
    def display(self):
        Print ("Lab details")
        Print ("marks", self.total())
    def total (self)
        m = Lab.marks
        return m[0]+m1[1]
name = input ("Enter the name:")
m1 = ''
m2 = ''
```

Q.4 write short notes on:

(a) super() function

(b) polymorphism

Ans **super** :- Python super() function provide us the facility to refer to the parent class explicity. It is basicaly useful where we have to call super-class functions it returns the proxy object that allows us to refer parent class by "super".

Polymorphism :- one name many form.

Polymorphism is taken from the creek words poly (many) and morphism (forms) It means that the same function name can be used for different types. This makes programming more intuitive and easier.

Q5 what is method overriding ? Explain with an example ?

Ans we can define functions in the derived class that has the same name as the

Date / /

function in the base class. Here, we re-implement the functions is the derived class. The phenomenon of re-implementing a function in the derived class is known as method overiding.

Ex→

```
class sachin :
    def fun(self):
        print("fun of class A")

class RaO:
    def fun(self):
        print("fun of class B")

obj = B()
obj.fun()
```

output

fun of class B

Q.5 What is ~~method~~ abstract class ~~class~~? How do you write an abstract class in python?

Ans A class is called Abstract class if it contains one or more abstract methods when → abstract method → Pass 3

An Abstract method is a method that is declared but contains no implementation Abstract classes cannot be instantiated and its abstract method must be implemented by its sub cla

```
from abc import ABC, abstractmethod
class Jay(ABC):
    @abstract method
    def fun(self):
        pass
class try(Jay):
    def fun(self):
        print("I am try")

class Re(Jay):
    def fun(self):
        print("I am Re")

class He(Jay):
    def fun(self):
        print("I am He")
```

Q.7 What is method resalution order (MRO)? Explain the principles followe by MRO with Example?

A) MRO is a concept used in inheritance it is the order in which python looks for a method in a hierarchy of classes it is especially useful in python because python support mutiple inheritance.

EX
```
class A:
    def fun(self):
        print('fun of class A')
```

Date __/__/__

```
class B:
    pass

obj = C()
obj.fun()
print(C.mro())
```

output fun of class A

MRO → C → A → B → object

Q. What is introspection in python? Explain five function each build-in and inspect module.?

Ans — Build - In functions that are used for code introspection python provide some build-in function that are used for code introspection These are following.

① Type obj : Return the type of an object

② dir (obj) : Return list of methods an attributes associated with that o

③ Str (obj) : Convert everything into a st

④ id (obj) : Returns a unique id of an

⑤ locals() : return you a dictonnary o variables decared is the local

Q.9

Solve X class time:

```
def print24(S):
    h1=ord(S[1]) - ord('0')
    h2=ord(S[0]) - or('0')
    hh = (h2*10 + h1%10)

If (S[8] == 'A'):
    If (hh = 12):
        print ('00', end = ' ')
        for i in range(2,8)
            print ('00', end = ' ')
    else:
        for i in range(0,8):
            print(S[i] ,end = '')
```

Q.10 X