EXPERIMENT NO-02

NAME-YASH WAGH|SE-D|122A1125

AIM:-

Write a program to visualize the usage of following :

i)Methods and Lambda methods

ii).Class and objects

iii).Inheritance (OOP)

iv).try…except (Exception Handling)

THEORY:-

A function is a block of code which only runs when it is called.

You can pass data, known as parameters, into a function.

A function can return data as a result.

Syntax – def method_name(self):

Lambda method - A lambda method is a small anonymous method.

A lambda method can take any number of arguments, but can only have one expression.

Syntax : lambda arguments : expression.

PROGRAM:-

A]

```
class experiment:
    def sum():
        a=int(input("enter first number:-"))
        b=int(input("enter second nmuber:-"))
        print("sum of",a,"and",b,"is",a+b)
obj=experiment
obj.sum()
```

B]

```
a=int(input("enter first number:-"))
```

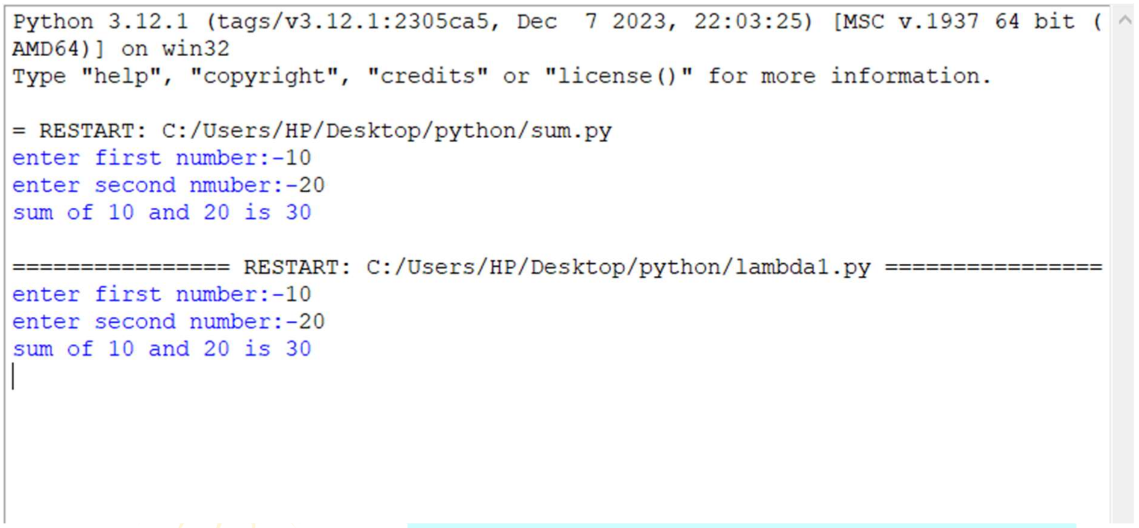
b=int(input("enter second number:-"))

c=lambda a,b:a+b

d=c(a,b)

print("sum of",a,"and",b,"is",d)

OUTPUT:-



```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

= RESTART: C:/Users/HP/Desktop/python/sum.py
enter first number:-10
enter second nmuber:-20
sum of 10 and 20 is 30

================ RESTART: C:/Users/HP/Desktop/python/lambda1.py ================
enter first number:-10
enter second number:-20
sum of 10 and 20 is 30
```

THEORY:-

Python is an object oriented programming language.

Almost everything in Python is an object, with its properties and methods.

A Class is like an object constructor, or a "blueprint" for creating objects.

Syntax :

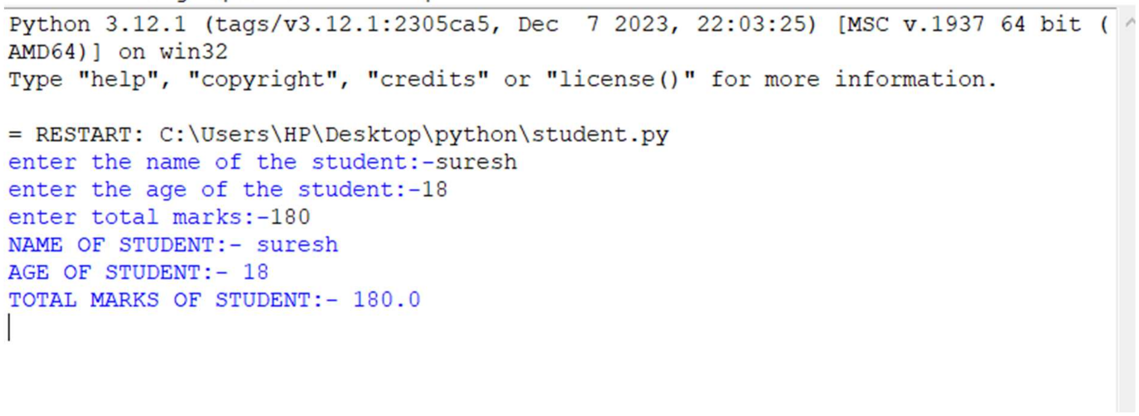
class class_name:

obj1 = class_name()

PROGRAM:-

```
class student:
    def information():
        name=input("enter the name of the student:-")
        age=int(input("enter the age of the student:-"))
        marks=float(input("enter total marks:-"))
        print("NAME OF STUDENT:-",name)
        print("AGE OF STUDENT:-",age)
        print("TOTAL MARKS OF STUDENT:-",marks)
obj=student
obj.information()
```

OUTPUT:-



```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

= RESTART: C:\Users\HP\Desktop\python\student.py
enter the name of the student:-suresh
enter the age of the student:-18
enter total marks:-180
NAME OF STUDENT:- suresh
AGE OF STUDENT:- 18
TOTAL MARKS OF STUDENT:- 180.0
```

THEORY:-

Inheritance allows us to define a class that inherits all the methods and properties from another class.

Parent class is the class being inherited from, also called base class.

Child class is the class that inherits from another class, also called derived class.

PROGRAM:-

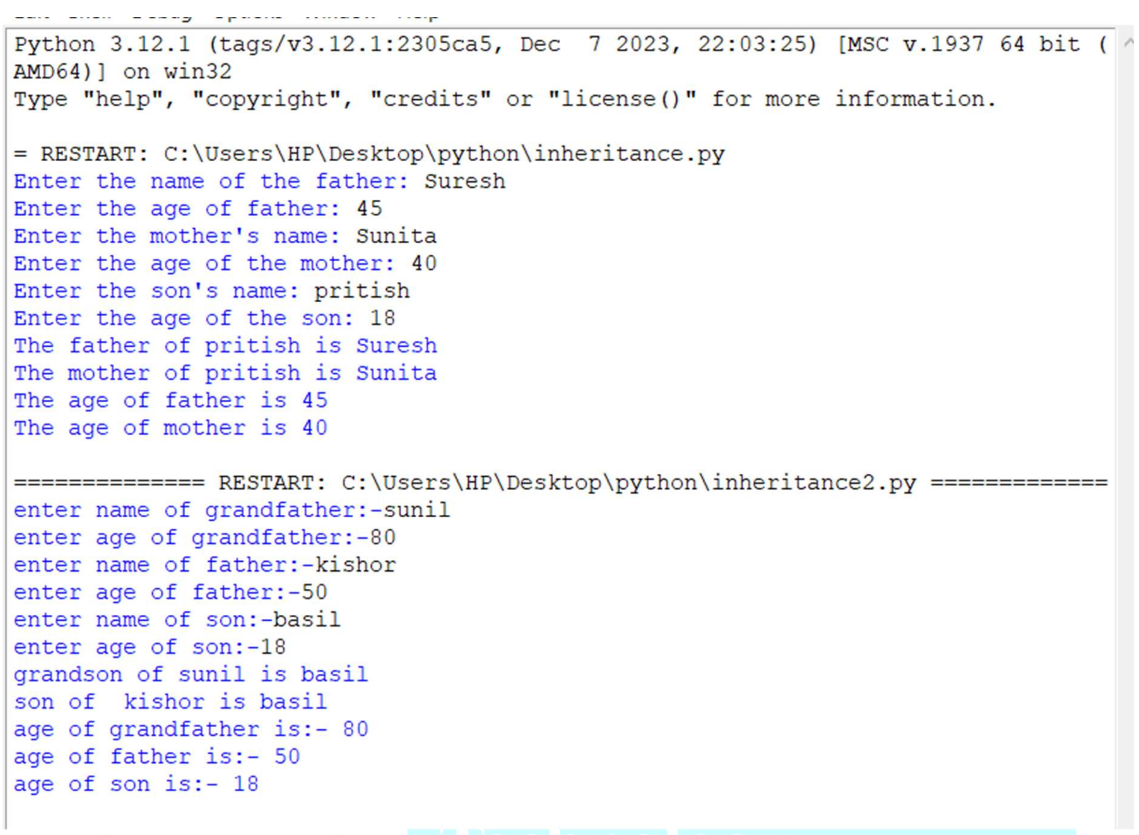
A]

```
class Father():
    def father_info(self):
        self.father_name = str(input("Enter the name of the father: "))
        self.father_age = int(input("Enter the age of father: "))

class Mother():
    def mother_info(self):
        self.mother_name = str(input("Enter the mother's name: "))
        self.mother_age = int(input("Enter the age of the mother: "))

class Child(Father, Mother):
    def son_info(self):
        son_name = str(input("Enter the son's name: "))
        son_age = int(input("Enter the age of the son: "))
        print("The father of", son_name, "is", self.father_name)
        print("The mother of", son_name, "is", self.mother_name)
        print("The age of father is", self.father_age)
        print("The age of mother is", self.mother_age)

obj1 = Child()
obj1.father_info()
obj1.mother_info()
obj1.son_info()
```

B]

```
class grandfather:
    def ginfo(self):
        self.gname=input("enter name of grandfather:-")
        self.gage=int(input("enter age of grandfather:-"))
class father(grandfather):
    def finfo(self):
        self.fname=input("enter name of father:-")
        self.fage=int(input("enter age of father:-"))
class son(father):
    def sinfo(self):
        self.sname=input("enter name of son:-")
        self.sage=int(input("enter age of son:-"))
        print("grandson of",self.gname,"is",self.sname)
        print("son of ",self.fname,"is",self.sname)
        print("age of grandfather is:-",self.gage)
        print("age of father is:-",self.fage)
        print("age of son is:-",self.sage)
obj=son()
obj.ginfo()
obj.finfo()
obj.sinfo()
```

OUTPUT:-



```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

= RESTART: C:\Users\HP\Desktop\python\inheritance.py
Enter the name of the father: Suresh
Enter the age of father: 45
Enter the mother's name: Sunita
Enter the age of the mother: 40
Enter the son's name: pritish
Enter the age of the son: 18
The father of pritish is Suresh
The mother of pritish is Sunita
The age of father is 45
The age of mother is 40

============== RESTART: C:\Users\HP\Desktop\python\inheritance2.py =============
enter name of grandfather:-sunil
enter age of grandfather:-80
enter name of father:-kishor
enter age of father:-50
enter name of son:-basil
enter age of son:-18
grandson of sunil is basil
son of  kishor is basil
age of grandfather is:- 80
age of father is:- 50
age of son is:- 18
```

THEORY:-

The `try` block lets you test a block of code for errors.

The `except` block lets you handle the error.

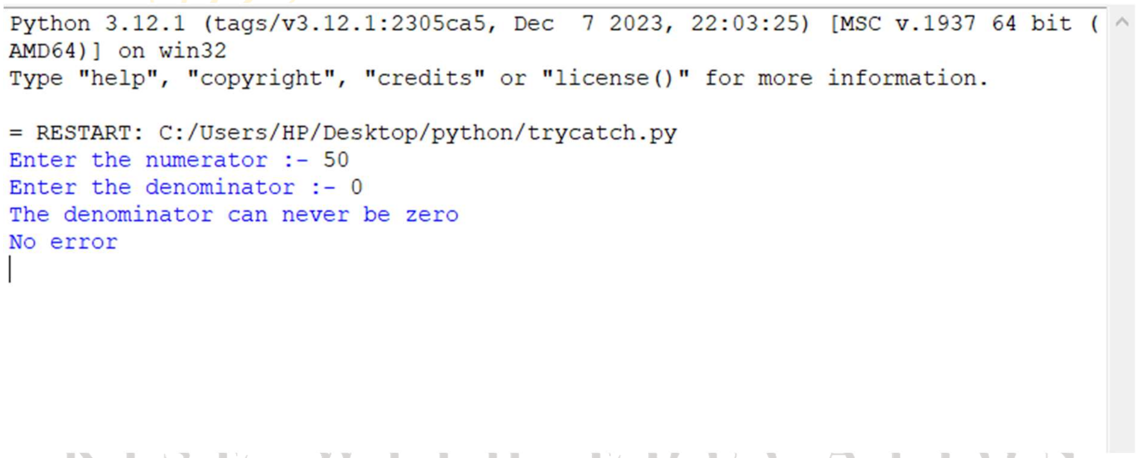
The `else` block lets you execute code when there is no error.

The `finally` block lets you execute code, regardless of the result of the try- and except blocks.

PROGRAM:-

```
try :
  num1 = int(input("Enter the numerator :- "))
  num2 = int(input("Enter the denominator :- "))
  quo = num1/num2
  print("The quotient after division is" , quo)
except ZeroDivisionError:
  print("The denominator can never be zero ")
else :
  print("There is no error ")
finally :
  print("No error ")
```

OUTPUT:-



```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.

= RESTART: C:/Users/HP/Desktop/python/trycatch.py
Enter the numerator :- 50
Enter the denominator :- 0
The denominator can never be zero
No error
```

CONCLUSION:-

We executed methods , classes , objects , multiple inheritance , multi-level inheritance and try…except blocks.