****

**FACULTY OF TECHNOLOGY AND ENGINEERING**

**DEVANG PATEL INSTITUTE OF ADVANCE TECHNOLOGY AND RESEARCH**

**DEPARTMENT OF COMPUTER ENGINEERING**

**A.Y. 2023-24 [EVEN]**

**LAB MANUAL**

**CE259: PROGRAMMING IN PYTHON**

****

|  |  |
| --- | --- |
| **Semester:IV** | **Academic Year:2023-2024** |
| **Subject Code:CE259** | **Subject Name:PROGRAMMING IN PYTHON** |
| **Student Id:22DCE006** | **Student Name:PROBIN BHAGCHANDANI** |

**PRACTICAL INDEX**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Sr.**  **No.** | **AIM** | **Assigned Date** | **Completion**  **Date** | **Grade** | **Assessment**  **Date** | **Signature** |
| 1 | A) Introduction to Python Programming. Installation &amp; Configuration of Python. Along  with its all-major editors, IDLE, Pycharm, Anaconda, Jupyter, Interpreter etc.  B) Write a python program to calculate simple interest. | 21/12/2023 | 21/12/2023 |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

**PRACTICAL-1**

**AIM: A) Introduction to Python Programming. Installation &amp; Configuration of Python. Along with its all-major editors, IDLE, Pycharm, Anaconda, Jupyter, Interpreter etc.**

**B) Write a python program to calculate simple interest.**

**IDLE**

IDLE (short for Integrated Development and Learning Environment) is an integrated

development environment for Python, which has been bundled with the default implementation of the language since 1.5.2b1. It is packaged as an optional part of the Python packaging with many Linux distributions. It is completely written in Python and its GUI toolkit. IDLE is intended to be a simple IDE and suitable for beginners, especially in an educational environment. To that end, it is cross-platform, and avoids feature clutter. According to the included README, its main features are:

• Multi-window text editor with syntax highlighting, autocompletion, smart indent and

other.

• Python shell with syntax highlighting.

• Integrated debugger with stepping, persistent breakpoints, and call stack visibility

**PyCharm**

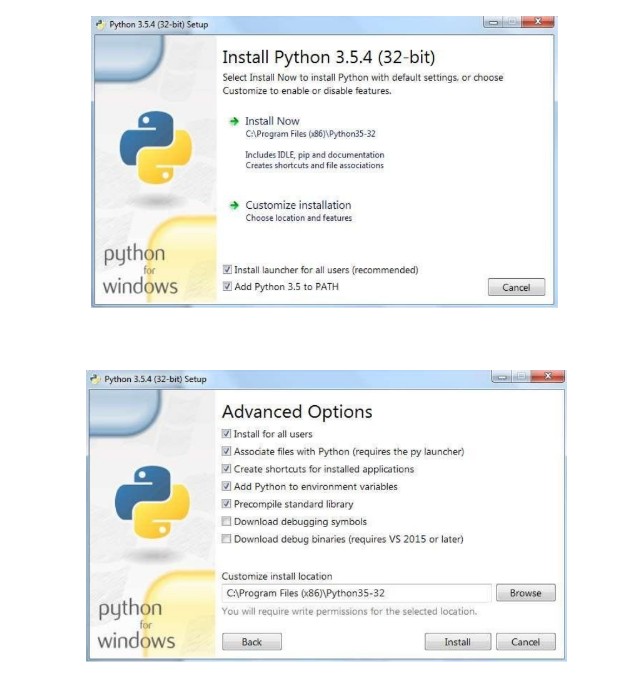
PyCharm is an integrated development environment (IDE) used for programming in Python. It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems, and supports web development with Django. PyCharm is developed by the Czech company JetBrains.

It is cross-platform, working on Microsoft Windows, macOS and Linux. PyCharm has a Professional Edition, released under a proprietary license and a Community Edition released under the Apache License. PyCharm Community Edition is less extensive than the Professional Edition.

**Anaconda**

Anaconda is a distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics,etc.), that aims to simplify package management and deployment. The distribution includes data-

science packages suitable for Windows, Linux, and macOS.



**B)**

**PROGRAM CODE:**

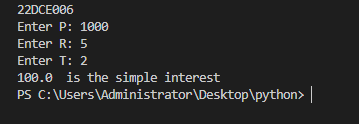
print("22DCE006")

p=input("Enter P: ")

r=input("Enter R: ")

t=input("Enter T: ")

print(int(int(p)\*int(r)\*int(t))/100, " is the simple interest")

**OUTPUT:**

**CONCLUSION:** From this practical, I learned about the python fundamentals, installation and configuration of its environment.

**Staff Signature:**

**Grade:**

**Remarks by the Staff:**

**PRACTICAL-2**

**AIM:** A) Create a list and apply methods (append, extend, remove, reverse), arrange created list in ascending and descending order.

B) List1 = [1, 2, 3, 4, ["python", "java", "c++", [10,20,30]], 5, 6, 7, ["apple", "banana","orange"]] . From above list get word “orange” and “Python” &amp; repeat this list five times without using loops.

C) Create a list and copy it using slice function

D) Create a tuple and apply different type of mathematical operation on it (Sum,

Maximum, minimum etc.).

**A)**

**PROGRAM CODE:**

print("22DCE006")

list=[10,20,40,80]

print(list)

print("Append Function")

list.append(77)

print(list)

list2=[11,22,33]

print("Extend Function")

list.extend(list2)

print(list)

print("Remove Function")

list.remove(77)

print(list)

print("Reverse Function")

list.reverse()

print(list)

print("Ascending Sorting")

list.sort()

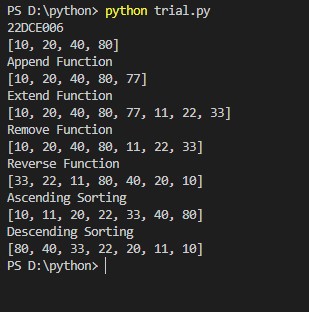
print(list)

print("Descending Sorting")

list.sort(reverse=True)

print(list)

**OUTPUT:**



**B)**

**PROGRAM CODE:**

print("22DCE006")

List1 = [1, 2, 3, 4, ["python", "java", "c++", [10,20,30] ], 5, 6, 7, [ "apple" , "banana" , "orange" ] ]

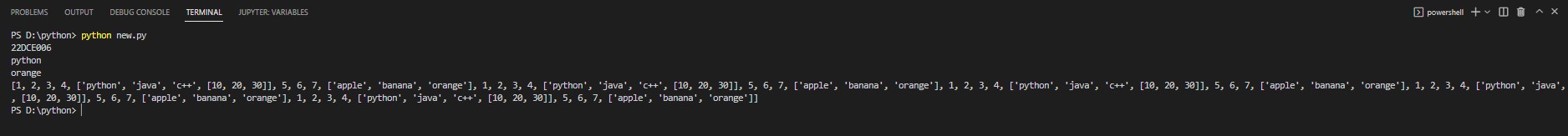
print(List1[4][0])

print(List1[8][2])

repeatedlist=List1\*5

print(repeatedlist)

**OUTPUT:**



**C)**

**PROGRAM CODE:**

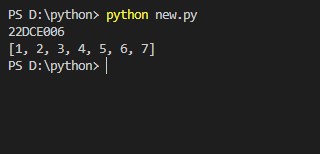
print("22DCE006")

list=[1,2,3,4,5,6,7]

x=slice(0,7)

list2=list[x]

print(list2)

**OUTPUT:**

**D)**

**PROGRAM CODE:**

print("22DCE006")

a=(1,2,3,4,5,6,7)

print(a)

print("Sum Operation:")

print(sum(a))

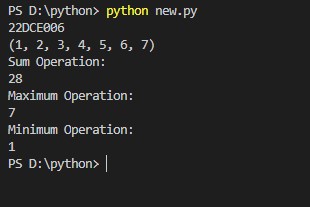
print("Maximum Operation:")

print(max(a))

print("Minimum Operation:")

print(min(a))

**OUTPUT:**



**CONCLUSION:** In this practical I learned various method like append, extend, remove, reverse, sort in python and how to use slice method and maximum and minimum method in python.

**Staff Signature:**

**Grade:**

**Remarks by the Staff:**

**PRACTICAL-3**

**AIM:**

A) String Operations:

* Reverse a string, replace string with other string, merge two strings
* Find character is in string or not without using loops
* Split string into multiple words & characters

B) Dictionaries Operations:

* Apply “Update, Delete, clear, popitem, pop, get, keys and values” operation in dictionary.
* Create 3 dictionaries and merge them into 1 dictionary

**PROGRAM CODE:**

import re

print("22DCE006\n")

str1="hello" [::-1]

print("Reverse String="+str1)

str2 ="Hello World"

str3 = str2.replace("Hello", "Bye")

print("Replacing the String = "+str3)

str4="Charusat"

str5="University"

str6 = str4 + ' ' + str5

print("Merging two Strings = "+str6)

str7="Python Lab"

print("The 'o' character is present in the string:", 'o' in str7)

str8="Charsuat University"

result = re.findall(r"[\w']+", str8)

print("After splitting the string: ", result)

Dict= { "movie" : "Hero", "theme" : "fiction", "year" :"2021" }

print(Dict)

Dict.update({"bg":"dark"})

print("\nUpdating the dictionary: "+str(Dict))

del Dict["bg"]

print("After deleting: "+str(Dict))

print("Using keys() method: "+str(Dict.keys()))

print("Using values() method: "+str(Dict.values()))

Dict.popitem()

print("Using popitem: "+str(Dict))

print("Using get() method: "+str(Dict.get("movie")))

Dict.pop("theme")

print("Dictionary after using pop: "+str(Dict))

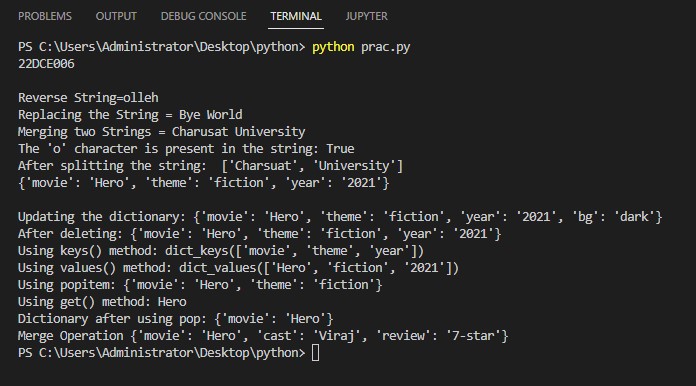
Dict2 = { "cast" : "Viraj" }

Dict3 = { "review" : "7-star" }

Dict.update(Dict2)

Dict.update(Dict3)

print("Merge Operation "+str(Dict))

**OUTPUT:**

**CONCLUSION:** In this practical I learned various operations that can be performed on strings and also about dictionary operations.

**Staff Signature:**

**Grade:**

**Remarks by the Staff:**

**PRACTICAL-4**

**AIM: Write python programs by declaring a function**

**A) Found which grade student will get based on SGPA.**

**B) Find max from three numbers**

**C) Calculate number of Uppercase and lowercase letters of string given by user**

**D) Find a Square of a given list using lambda function**

**E) Enter value from user and print multiplication table**

**F) Create a list by user given value and make sum of it using loop**

**G) Use comprehension method**

* **Create a two separate list of even and odd numbers from 1 to 50**
* **Get value which are divided by 5 from 1 to 100**

**A)**

**Program Code:**

print("22DCE006\n")

maths=float(input("Enter the marks of maths:"));

science=float(input('enter the marks of science:'));

English=float(input('enter the marks of English:'));

Hindi=float(input('Enter the marks of Hindi:'));

grade='\0'

total=maths+science+English+Hindi

avg=total/4

if(avg>=90):

     grade ='A'

elif((avg>=80) and (avg<90)):

     grade='B'

elif((avg>=70) and (avg<80)):

     grade='C'

elif((avg>=60) and (avg<70)):

     grade='D'

elif((avg>=50) and (avg<60)):

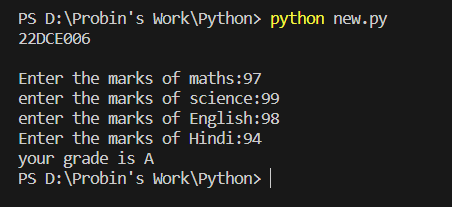
     grade='E'

else:

     grade='F'

print("your grade is",grade)

**Output:**

****

**B)**

**Program Code:**

print("22DCE006\n")

a = int(input('Enter first number : '))

b = int(input('Enter second number : '))

c = int(input('Enter third number : '))

max = 0

if a > b and a > c :

     max = a

elif b > c :

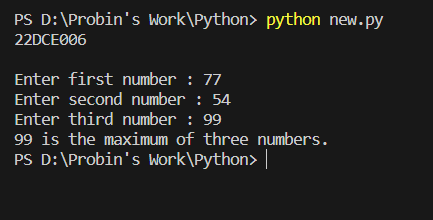
     max = b

else :

     max = c

print(max, "is the maximum of three numbers.")

**Output:**

****

**C)**

**Program Code:**

print("22DCE006\n")

def count\_letters(string):

    uppercase\_count = 0

    lowercase\_count = 0

    for char in string:

        if char.isupper():

            uppercase\_count += 1

        elif char.islower():

            lowercase\_count += 1

    return uppercase\_count, lowercase\_count

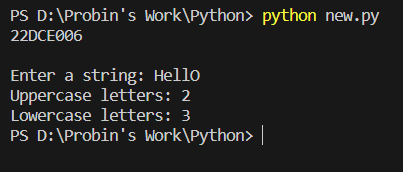
string = input("Enter a string: ")

uppercase, lowercase = count\_letters(string)

print(f"Uppercase letters: {uppercase}")

print(f"Lowercase letters: {lowercase}")

**Output:**

****

**D)**

**Program Code:**

print("22DCE006\n")

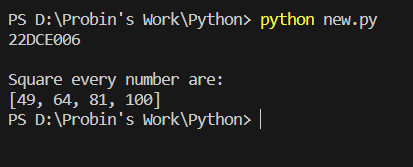
number = [7,8,9,10]

print("Square every number are:")

square\_number = list(map(lambda x: x \*\* 2, number))

print(square\_number)

**Output:**

****

**E)**

**Program Code:**

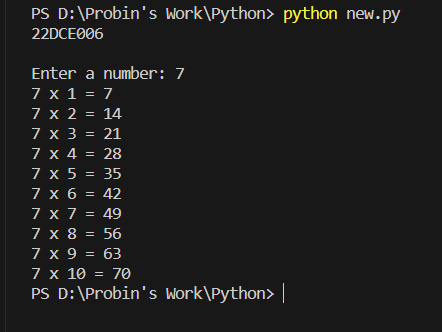
print("22DCE006\n")

n = int(input("Enter a number: "))

for i in range(1,11):

     print(n,'x',i,'=',n\*i)

**Output:**

****

**F)**

**Program Code:**

print("22DCE006\n")

LIST=[]

n=int(input("ENter the number of element you want to insert in the list: "))

for i in range(0,n):

     e=int(input())

     LIST.append(e)

print("LIST IS :",LIST)

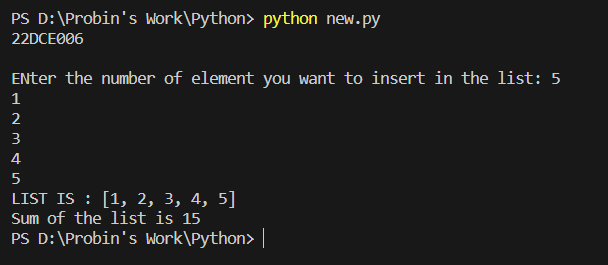
sum=0

for i in range(0,len(LIST)):

     sum=sum+LIST[i]

print("Sum of the list is",sum)

**Output:**

****

**G)**

**Program Code:**

print("22DCE006\n")

LIST1=[]

LIST2=[]

for i in range(1,51):

     if i&1:

          LIST1.append(i)

     else:

          LIST2.append(i)

print("The list of the even number is",LIST2)

print("The list of the odd number is",LIST1)

list3=[]

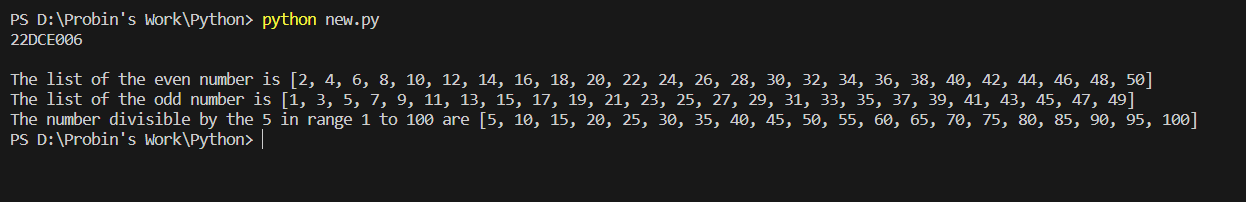
for i in range(1,101):

     if i%5==0:

          list3.append(i)

print("The number divisible by the 5 in range 1 to 100 are",list3)

**Output:**

****

**Conclusion:** In this practical we learnt about the different type of the fuction like the lambda fucntion, append and how to create the list in which user can enter the data,if else statement in the python and its usage for manipulating the list .

**Staff Signature:**

**Grade:**

**Remarks by the Staff:**

**PRACTICAL-5**

**AIM:**

**A) Create a class employee and display employee details**

**B) From above create class and count number of employee and display a salary**

**amount, if the salary is raised to 1.04%.**

**C) Fetch children class details using different types of inheritance (Single, Multilevel,**

**and Multiple) With constructor**

**D) Find who will be first among two students using polymorphism.**

**A)**

**Program Code:**

print("22DCE006\n")

class emp:

     c=0

     def \_\_init\_\_(self,name,e\_id,salary):

          self.name=name

          self.e\_id=e\_id

          self.salary=salary

          emp.c+=1

     def disp(self):

          print("The name of the employee is : ",self.name)

          print("The Employee id is : ",self.e\_id)

          print("The salary is : ",self.salary)

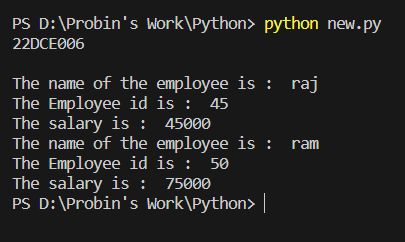
p1=emp("raj",45,45000)

p1.disp()

p2=emp("ram",50,75000)

p2.disp()

**Output:**

****

**B)**

**Program Code:**

print("22DCE006\n")

class emp:

     c=0

     def \_\_init\_\_(self,name,e\_id,salary):

          self.name=name

          self.e\_id=e\_id

          self.salary=salary

          emp.c+=1

     def disp(self):

          print("The name of the employee is : ",self.name)

          print("The Employee id is : ",self.e\_id)

          print("The salary is : ",self.salary)

     def inc(self):

          inc=(self.salary\*0.0104)+self.salary

          print("Acutal salary is : ",self.salary)

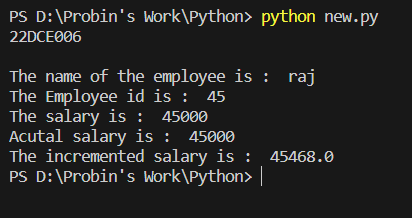
          print("The incremented salary is : ",inc)

p1=emp("raj",45,45000)

p1.disp()

p1.inc()

**Output:**

****

**C)**

**Program Code:**

print("22DCE006\n")

# Single Inheritance

class Parent:

    def \_\_init\_\_(self, name):

        self.name = name

    def display(self):

        print("Parent Class")

class Child(Parent):

    def \_\_init\_\_(self, name, age):

        super().\_\_init\_\_(name)

        self.age = age

    def display\_child(self):

        print("Child Class")

        print(f"Name: {self.name}")

        print(f"Age: {self.age}")

# Multilevel Inheritance

class GrandParent:

    def \_\_init\_\_(self, name):

        self.name = name

    def display(self):

        print("GrandParent Class")

class Parent(GrandParent):

    def \_\_init\_\_(self, name, age):

        super().\_\_init\_\_(name)

        self.age = age

    def display\_parent(self):

        print("Parent Class")

        print(f"Name: {self.name}")

        print(f"Age: {self.age}")

class Child(Parent):

    def \_\_init\_\_(self, name, age, grade):

        super().\_\_init\_\_(name, age)

        self.grade = grade

    def display\_child(self):

        print("Child Class")

        print(f"Name: {self.name}")

        print(f"Age: {self.age}")

        print(f"Grade: {self.grade}")

# Multiple Inheritance

class Father:

    def \_\_init\_\_(self, name):

        self.name = name

    def display\_father(self):

        print("Father Class")

        print(f"Name: {self.name}")

class Mother:

    def \_\_init\_\_(self, age):

        self.age = age

    def display\_mother(self):

        print("Mother Class")

        print(f"Age: {self.age}")

class Child(Father, Mother):

    def \_\_init\_\_(self, name, age, grade):

        super().\_\_init\_\_(name)

        Mother.\_\_init\_\_(self, age)

        self.grade = grade

    def display\_child(self):

        print("Child Class")

        print(f"Name: {self.name}")

        print(f"Age: {self.age}")

        print(f"Grade: {self.grade}")

# Single Inheritance Example

child1 = Child("Rajesh", 10 , 7)

child1.display\_child()

# Multilevel Inheritance Example

child2 = Child("Suresh", 12, 6)

child2.display\_child()

# Multiple Inheritance Example

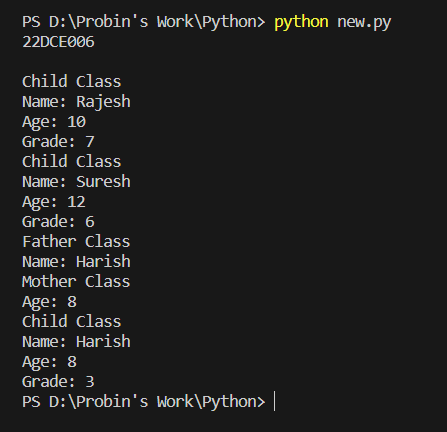
child3 = Child("Harish", 8, 3)

child3.display\_father()

child3.display\_mother()

child3.display\_child()

**Output:**

****

**D)**

**Program Code:**

print("22DCE006\n")

class Student:

    def \_\_init\_\_(self, name, id):

        self.name = name

        self.id = id

    def \_\_lt\_\_(self, other):

        return self.id < other.id

student1 = Student("Rajesh", 1)

student2 = Student("Neeraj", 2)

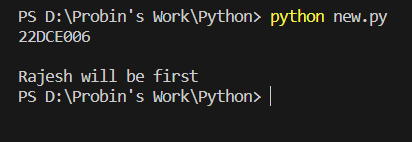
if student1 < student2:

    print(f"{student1.name} will be first")

else:

    print(f"{student2.name} will be first")

**Output:**

****

**Conclusion:** From this practical we learned the concept of class and constructors. Also learned about the different types of inheritance , its usage and polymorphism.

**Signature:**

**Grade:**

**Remarks by the Staff:**

**PRACTICAL-6**

**AIM:**

**Consider an example of declaring the examination result. Design three classes:**

**Student, Exam, and Result. The Student class has data members such as those**

**representing rollNumber, Name, etc. Create the class Exam by inheriting Student class. The Exam class adds fields representing the marks scored in six subjects. Derive Result from the Exam class, and it has its own fields such as total marks. Write an interactive program to model this relationship.**

**Program Code:**

print("22DCE006\n")

class Student:

    def \_\_init\_\_(self, roll\_number, name):

        self.roll\_number = roll\_number

        self.name = name

    def display(self):

        print(f"Roll Number: {self.roll\_number}")

        print(f"Name: {self.name}")

class Exam(Student):

    def \_\_init\_\_(self, roll\_number, name, marks):

        super().\_\_init\_\_(roll\_number, name)

        self.marks = marks

    def display\_marks(self):

        print("Marks:")

        for subject, marks in self.marks.items():

            print(f"{subject}: {marks}")

class Result(Exam):

    def \_\_init\_\_(self, roll\_number, name, marks, total\_marks):

        super().\_\_init\_\_(roll\_number, name, marks)

        self.total\_marks = total\_marks

    def display\_result(self):

        self.display()

        self.display\_marks()

        print(f"Total Marks: {self.total\_marks}")

        print("Percentage obtained by the student is : " + str(self.total\_marks/6))

# Interactive program

roll\_number = input("Enter Roll Number: ")

name = input("Enter Name: ")

marks = {}

subjects = ["Subject1", "Subject2", "Subject3", "Subject4", "Subject5", "Subject6"]

for subject in subjects:

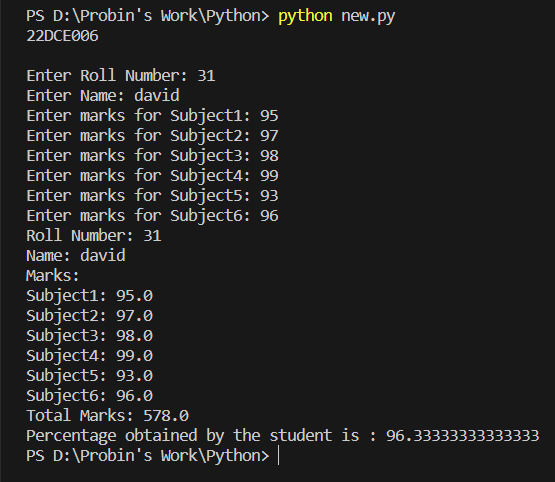
    marks[subject] = float(input(f"Enter marks for {subject}: "))

total\_marks = sum(marks.values())

result = Result(roll\_number, name, marks, total\_marks)

result.display\_result()

**Output:**

****

**Conclusion:** From this practical we learned the key concepts such inheritance, polymorphism, encapsulation, and the use of interactive programming techniques. With these concepts we can help in designing and implementing effective code structures.

**Signature:**

**Grade:**

**Remarks by the Staff:**

**Practical – 8**

* **Pre-process row data using pandas and create different graph using matplotlib and seaborn.**

**PROGRAM CODE:**

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sb

data =pd.read\_csv('simple.csv')

# data.dropna(implace=True)

plt.figure(figsize=(8,6))

plt.scatter(data['SepalLengthCm'],data['SepalWidthCm'])

plt.title('Scatter Plot')

plt.xlabel('SepalLengthCm')

plt.ylabel('SepalWidthCm')

plt.grid(True)

plt.show()

# histogram

plt.figure(figsize=(8,6))

sb.histplot(data['PetalWidthCm'], bins=20,kde=True)

plt.title('Histogram')

plt.xlabel('Width')

plt.ylabel('Cm')

plt.show()

# boxplot

plt.figure(figsize=(8,6))

sb.boxplot(x='Id',y='SepalLengthCm',data=data)

plt.title('Boxplot')

plt.xlabel('Id')

plt.ylabel('Cm')

plt.show()

# bar plot

plt.figure(figsize=(8,6))

sb.barplot(x='Id',y='SepalLengthCm',data=data)

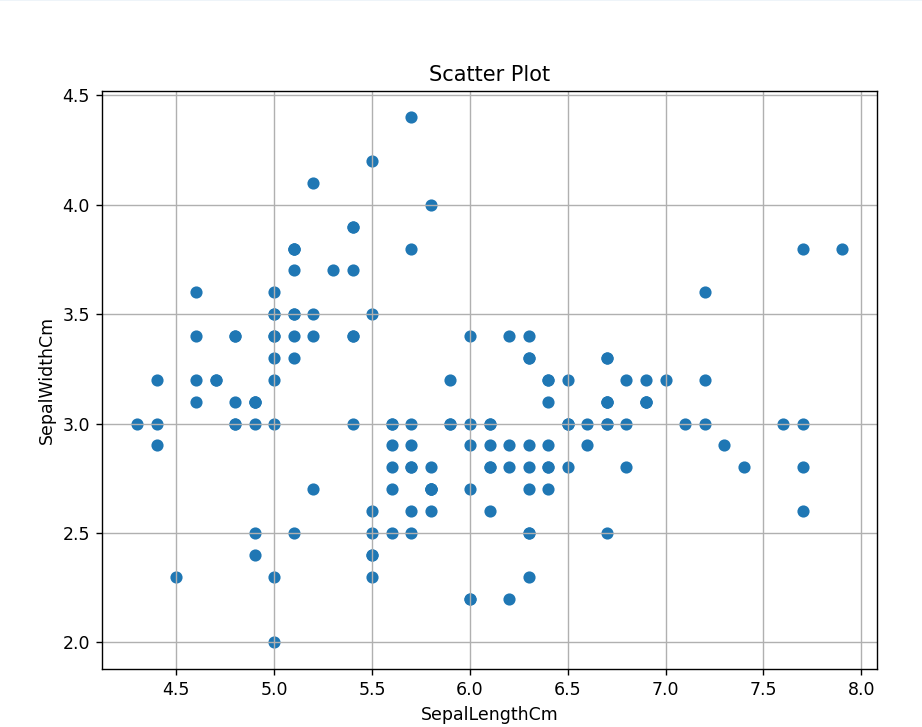
plt.title('Barplot')

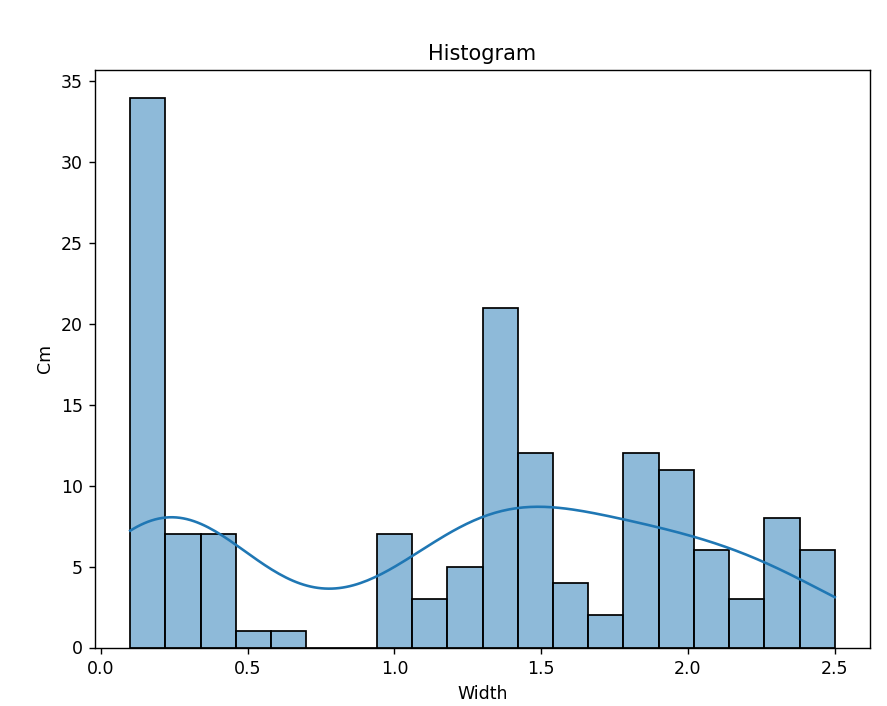
plt.xlabel('Id')

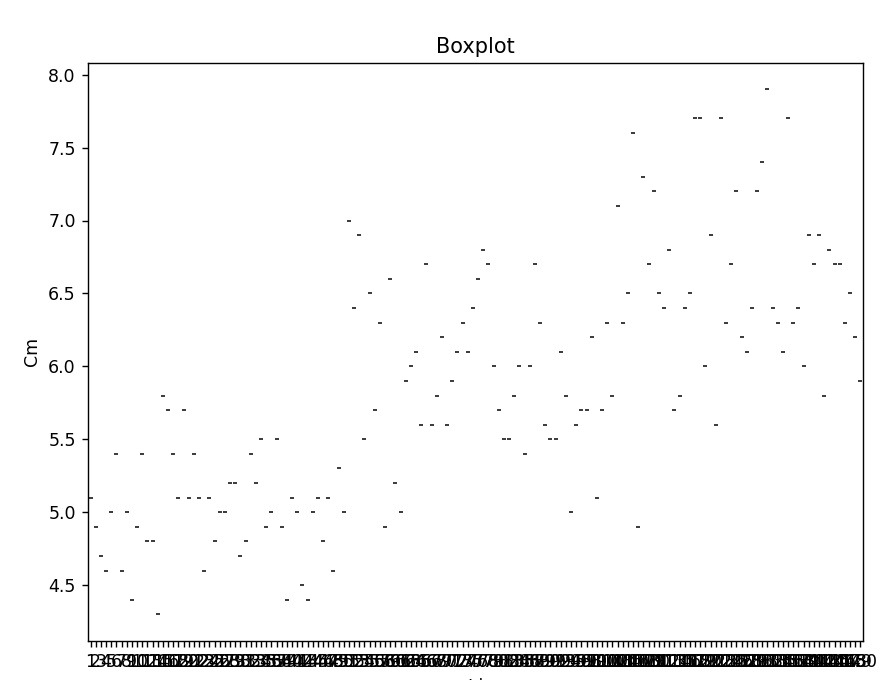
plt.ylabel('Cm')

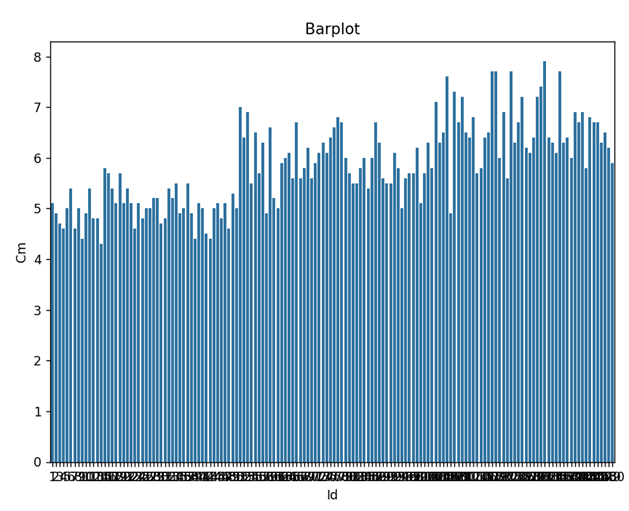
plt.show()

**OUTPUT:**









**CONCLUSION:**

In this practical we learnt about many graph in seaborn,metplob etc.

**Staff Signature:**

**Grade:**

**Remarks by the Staff:**