



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING  
(ARTIFICIAL INTELLIGENCE & MACHINE LEARNING)**

**S.E/SEM IV/CBCGS/AIML  
Academic Year: 2021-22**

<b>NAME</b>	PRATHAMESH CHIKANKAR
<b>BRANCH</b>	CSE-(AI&ML)
<b>ROLL NO.</b>	AIML08
<b>SUBJECT</b>	<b>PYTHON LAB</b>
<b>COURSE CODE</b>	<b>CSL405</b>
<b>PRACTICAL NO.</b>	01
<b>DOP</b>	<b>11/02/2022</b>
<b>DOS</b>	<b>24/03/2022</b>



## EXP NO: 1

Aim: Programming using Basic Data types (Numeric (int, float), List, Tuple, Set, dictionaries and string)

- Take two numbers as input (float) and print addition.
- To swap two numbers
- Solve the quadratic equation  $ax^2 + bx + c = 0$
- Perform (at least 5) operation on each
  - List
  - Tuple
  - Set
  - Dictionaries
  - String

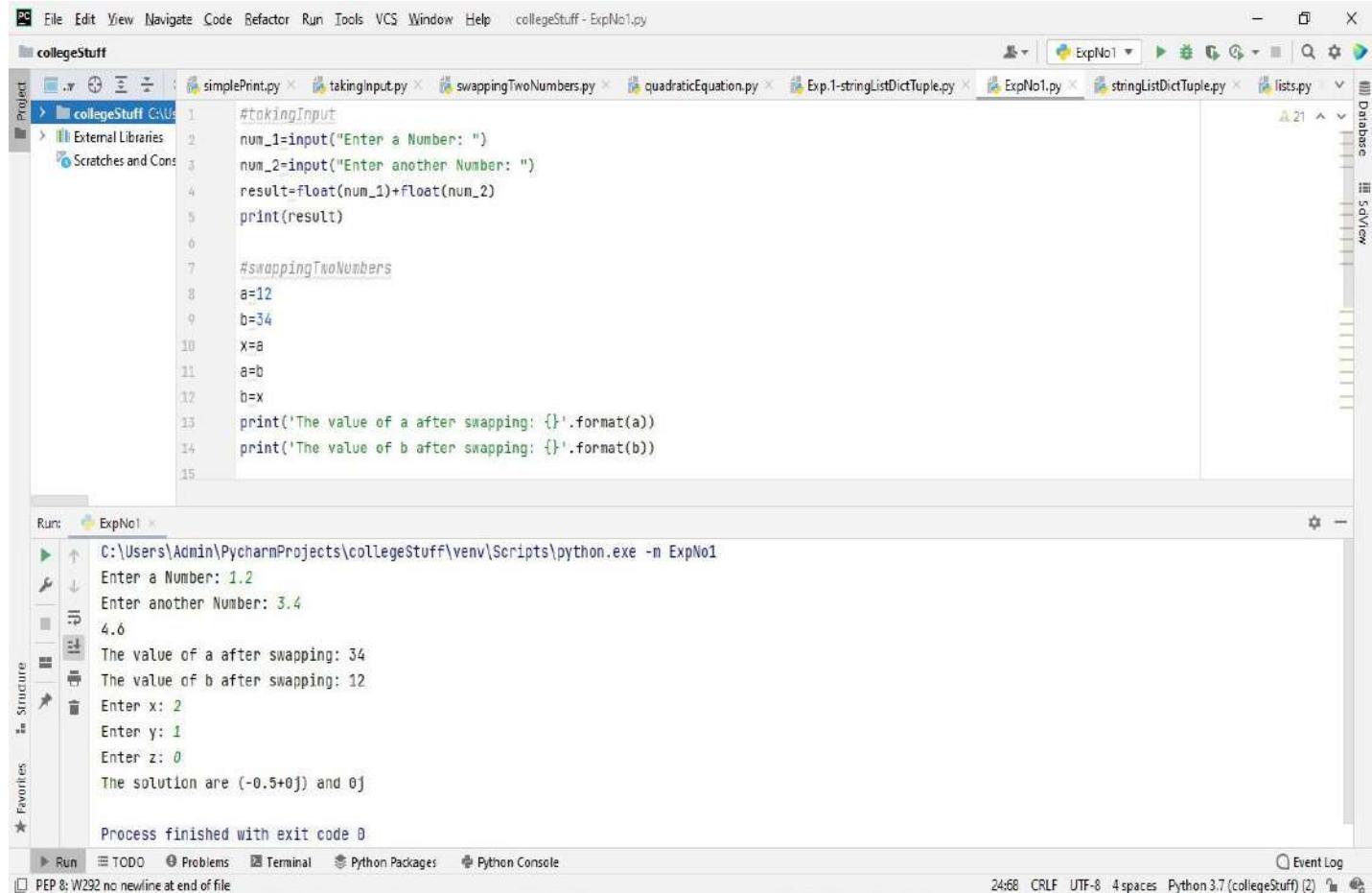
Theory :

- int (Signed integers) eg:- 10, -0x260
- float (floating point real values) eg:- 0.0, -32.54e100
- List → non-homogeneous data structure that stores the elements in single row and multiple rows and columns.  
→ List can be represented by []
- Tuple → non-homogeneous data structure that stores the elements in single row and multiple rows and columns.  
→ Tuple can be represented by ()
- Set → non-homogeneous data structure but stores in single row.  
→ Set can be represented by {}
- Dictionary → non-homogeneous data structure which stores key value pairs.  
→ Dictionary can be represented by {}
- String → arrays of bytes representing unicode characters. Square brackets can be used to access elements of the string.  
→ Created using single quote, double quote or even triple quote. " " .

	List	Tuple	Set	Dictionary
→	<code>l = []</code>	<code>t = ()</code>	<code>a = set()</code>	<code>d = {}</code>
			<code>b = set(a)</code>	
→	ordered	ordered	unordered	ordered (in python 3.7 & T)
→	nested	nested	nested	nested

a)Take two numbers as input(float) and print addition

b)Two swap two numbers



The screenshot shows the PyCharm IDE interface with the following details:

- File Bar:** File Edit View Navigate Code Refactor Run Tools VCS Window Help collegeStuff - ExpNo1.py
- Project View:** collegeStuff contains simplePrint.py, takingInput.py, swappingTwoNumbers.py, quadraticEquation.py, Exp.1-stringListDictTuple.py, ExpNo1.py, stringListDictTuple.py, and lists.py.
- Code Editor:** The swappingTwoNumbers.py file is open, containing the following code:

```

1 #takingInput
2 num_1=input("Enter a Number: ")
3 num_2=input("Enter another Number: ")
4 result=float(num_1)+float(num_2)
5 print(result)
6
7 #swappingTwoNumbers
8 a=12
9 b=34
10 x=a
11 a=b
12 b=x
13 print('The value of a after swapping: {}'.format(a))
14 print('The value of b after swapping: {}'.format(b))
15

```
- Run Tab:** Run configuration ExpNo1 is selected. The output shows:

```

C:\Users\Admin\PycharmProjects\collegeStuff\venv\Scripts\python.exe -m ExpNo1
Enter a Number: 1.2
Enter another Number: 3.4
4.6
The value of a after swapping: 34
The value of b after swapping: 12
Enter x: 2
Enter y: 1
Enter z: 0
The solution are (-0.5+0j) and 0j

Process finished with exit code 0

```
- Bottom Status Bar:** PEP 8: W292 no newline at end of file, 2468 CRLF, UTF-8, 4 spaces, Python 3.7 (collegeStuff) (2)

c) Solve the quadratic equation  $ax^{**2} + bx + c = 0$

The screenshot shows the PyCharm IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help, and collegeStuff - ExpNo1.py. The left sidebar shows a Project tree with 'collegeStuff' selected, containing 'External Libraries' and 'Scratches and Cons'. The main code editor window displays the following Python script:

```

15 #quadraticEquation
16 import cmath
17
18 x=float(input('Enter x: '))
19 y=float(input('Enter y: '))
20 z=float(input('Enter z: '))
21 d=(y**2)-(4*x*z) # calculate the discriminant
22 solution_1=(-y-cmath.sqrt(d))/(2*x)
23 solution_2=(-y+cmath.sqrt(d))/(2*x)
24 print('The solution are {0} and {1}'.format(solution_1,solution_2))

```

The 'Run' tab at the bottom shows the output of the script:

```

Run: ExpNo1
C:\Users\Admin\PycharmProjects\collegestuff\venv\Scripts\python.exe -m ExpNo1
Enter a Number: 1.2
Enter another Number: 3.4
4.6
The value of a after swapping: 34
The value of b after swapping: 12
Enter x: 2
Enter y: 1
Enter z: 0
The solution are (-0.5+0j) and 0j

Process finished with exit code 0

```

The status bar at the bottom right indicates the file is saved with CRLF, UTF-8, 4 spaces, Python 3.7 (collegeStuff) (2), and a timestamp of 10:4.

d) Perform (at least 5) operations on each.

i) list

```

File Edit View Navigate Code Refactor Run Tools VCS Window Help collegeStuff - ExpNo1.py
collegeStuff > ExpNo1.py simplePrint.py takingInput.py swappingTwoNumbers.py quadraticEquation.py Exp.1-stringListDictTuple.py ExpNo1.py stringListDictTuple.py lists.py
Project collegeStuff External Libraries Scratches and Cons
26 #list
27 lucky_numbers=[4,8,15,16,23,42]
28 friends=["Dhruv","Aayan","Prathamesh","Rihan","Ashfan","Sohail"]
29 print(lucky_numbers)
30 print(friends)
31 friends.append("Zubair")
32 lucky_numbers.append(51)
33 friends.extend(lucky_numbers)
34 friends.insert(0,"Eray")
35 friends.remove("Sohail")
36 print(friends)
37 friends=["Dhruv","Aayan","Prathamesh","Rihan","Ashfan","Sohail"]
38 friends.pop()
39 print(friends)
40 print(friends.index("Aayan"))
41 friends=["Dhruv","Dhruv","Aayan","Prathamesh","Rihan","Ashfan","Sohail"]
42 friends.sort()
43 print(friends)

Run: ExpNo1
[4, 8, 15, 16, 23, 42]
['Dhruv', 'Aayan', 'Prathamesh', 'Rihan', 'Ashfan', 'Sohail']
['Eray', 'Dhruv', 'Aayan', 'Prathamesh', 'Rihan', 'Ashfan', 'Zubair', 4, 8, 15, 16, 23, 42, 51]
['Dhruv', 'Aayan', 'Prathamesh', 'Rihan', 'Ashfan']
1
['Aayan', 'Ashfan', 'Dhruv', 'Dhruv', 'Prathamesh', 'Rihan', 'Sohail']

Process finished with exit code 0

```

ii) tuple

```

File Edit View Navigate Code Refactor Run Tools VCS Window Help collegeStuff - ExpNo1.py
collegeStuff > ExpNo1.py simplePrint.py takingInput.py swappingTwoNumbers.py quadraticEquation.py Exp.1-stringListDictTuple.py ExpNo1.py stringListDictTuple.py lists.py
Project collegeStuff External Libraries Scratches and Cons
42 friends.sort()
43 print(friends)

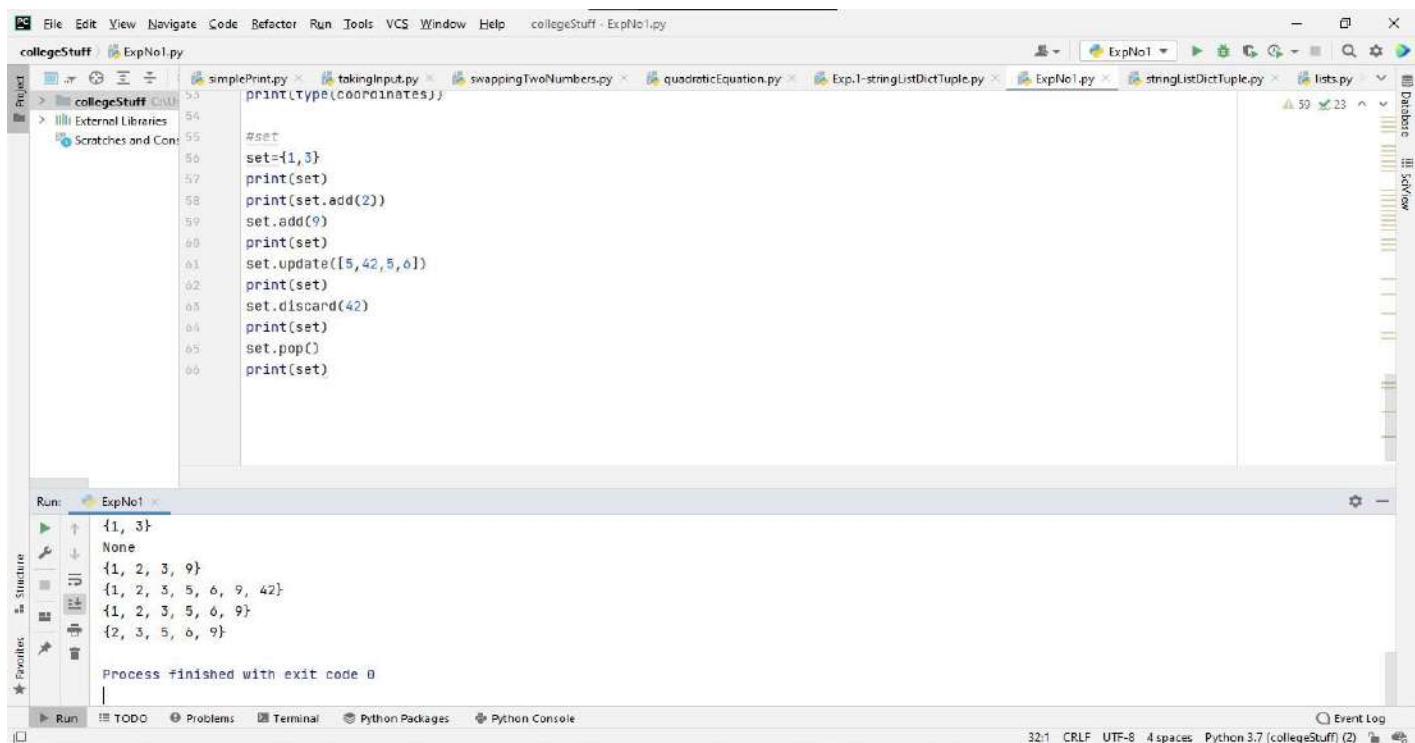
44 #tuple
45 coordinates=(4,5)
46 print(coordinates)
47 print(coordinates[0])
48 print(coordinates[1:])
49 print(coordinates[0:1])
50 print(coordinates+coordinates)
51 print(coordinates*3)
52 print(type(coordinates))

Run: ExpNo1
(4, 5)
4
(5,)
(4,)
(4, 5, 4, 5)
(4, 5, 4, 5, 4, 5)
<class 'tuple'>

Process finished with exit code 0

```

### iii) set



```

collegeStuff > ExpNo1.py
  53 simplePrint.py × takingInput.py × swappingTwoNumbers.py × quadraticEquation.py × Exp.1-stringListDictTuple.py × ExpNo1.py × stringListDictTuple.py × lists.py ×
  54
  55 print(type(coordinates))
  56
  57 #set
  58 set={1,3}
  59 print(set)
  60 set.add(2)
  61 print(set)
  62 set.update([5,42,5,6])
  63 print(set)
  64 set.discard(42)
  65 print(set)
  66 set.pop()
  67 print(set)

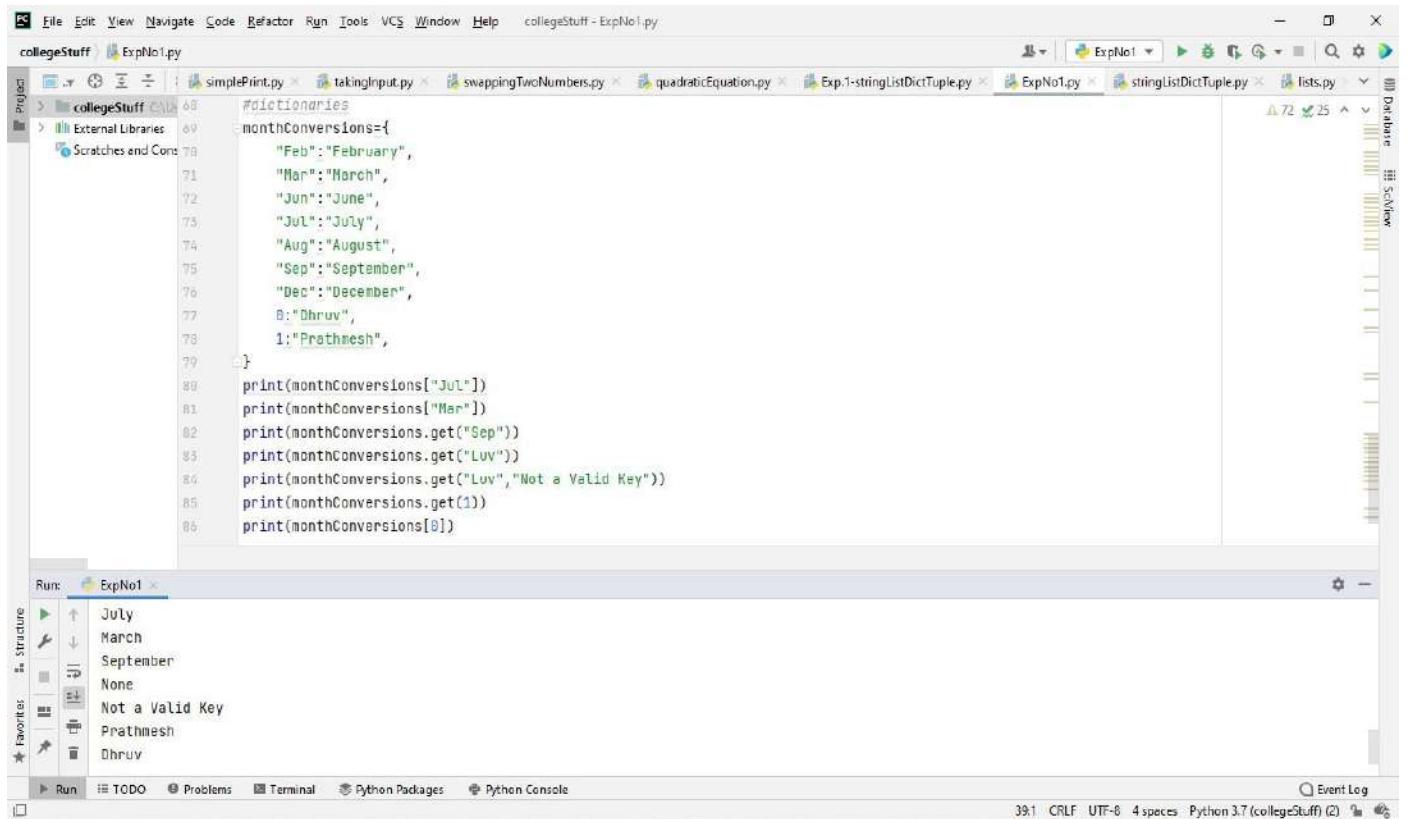
Run: ExpNo1 ×
  68 {1, 3}
  69 None
  70 {1, 2, 3, 9}
  71 {1, 2, 3, 5, 6, 9, 42}
  72 {1, 2, 3, 5, 6, 9}
  73 {2, 3, 5, 6, 9}

Process finished with exit code 0

```

Event Log: 32:1 CRLF UTF-8 4 spaces Python 3.7 (collegeStuff) (2)

### iv) dictionaries



```

collegeStuff > ExpNo1.py
  68 #dictionaries
  69 monthConversions={}
  70
  71     "Feb":"February",
  72     "Mar":"March",
  73     "Jun":"June",
  74     "Jul":"July",
  75     "Aug":"August",
  76     "Sep":"September",
  77     "Dec":"December",
  78     0:"Dhruv",
  79     1:"Prathmesh",
  80
  81 print(monthConversions["JUL"])
  82 print(monthConversions["Mar"])
  83 print(monthConversions.get("Sep"))
  84 print(monthConversions.get("Luv"))
  85 print(monthConversions.get("Luv","Not a Valid Key"))
  86 print(monthConversions.get(1))
  87 print(monthConversions[0])

Run: ExpNo1 ×
  88 July
  89 March
  90 September
  91 None
  92 Not a Valid Key
  93 Prathmesh
  94 Dhruv

Event Log: 39:1 CRLF UTF-8 4 spaces Python 3.7 (collegeStuff) (2)

```

## v)string

```
#string
phrase="Dhruv DP"
print(phrase)
print(phrase+" is cool")
print(phrase.lower())
print(phrase.upper())
print(phrase.isupper())
print(phrase.upper().isupper())
print(len(phrase))
print(phrase[0])
print(phrase.index("DP"))
print(phrase.replace("DP", "DP 97"))
```

Run: ExpNo1

Dhruv DP  
Dhruv DP is cool  
dhruv dp  
DHRUV DP  
False  
True  
8  
D  
0  
Dhruv DP 97

Process finished with exit code 0

File Edit View Navigate Code Refactor Run Tools VCS Window Help collegeStuff - ExpNo1.py collegeStuff ExpNo1.py

collegeStuff ExpNo1.py

simplePrint.py takingInput.py swappingTwoNumbers.py quadraticEquation.py Exp.1-stringListDictTuple.py ExpNo1.py stringListDictTuple.py lists.py

75 26

Database SQL Server

Run TODO Problems Terminal Python Packages Python Console Event Log

49:1 CRLF UTF-8 4 spaces Python 3.7 (collegeStuff) (2)



Conclusion : we learned the basic data types in python and executed them (numeric (int, float), list, tuple, set, dictionaries and strings) successfully in the compiler (through) :-



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING  
(ARTIFICIAL INTELLIGENCE & MACHINE LEARNING)**

**S.E/SEM IV/CBCGS/AIML  
Academic Year: 2021-22**

<b>NAME</b>	PRATHAMESH CHIKANKAR
<b>BRANCH</b>	CSE-(AI&ML)
<b>ROLL NO.</b>	AIML08
<b>SUBJECT</b>	<b>PYTHON LAB</b>
<b>COURSE CODE</b>	<b>CSL405</b>
<b>PRACTICAL NO.</b>	<b>02</b>
<b>DOP</b>	<b>17/02/2022</b>
<b>DOS</b>	<b>24/03/2022</b>

Exp No : 2

Aim: program using- decision control statements

- a) Smallest of 3 numbers (using nested if else)
  - b) To check if the input numbers is prime or not (using for loop)
  - c) To check if number provided by the user is Armstrong number or not (using while loop)

## Theory:

All the statements indented by the same number of character spaces, after a programming construct are considered to be part of a single block of code. Python uses indentation as its method of grouping statements.

- Nested if else - we can have an if...elif...else statement inside another if...elif...else statement. This is called nesting in computer programming. Any number of these statements can be nested inside one another. Indentation is the only way to figure out the level of nesting.

Syntax : if (condition1) :

if (cond1 || cond2);

- for loop - A for loop is used for iterating over a sequence.

we can execute a set of statements.

eg:- fruits = ["apple", "banana", "chessey"]  
for x in fruits  
print(x)

- while loop - with the while we can execute a set of statements as long as a condtn is true.

eg:-  $i = 1$

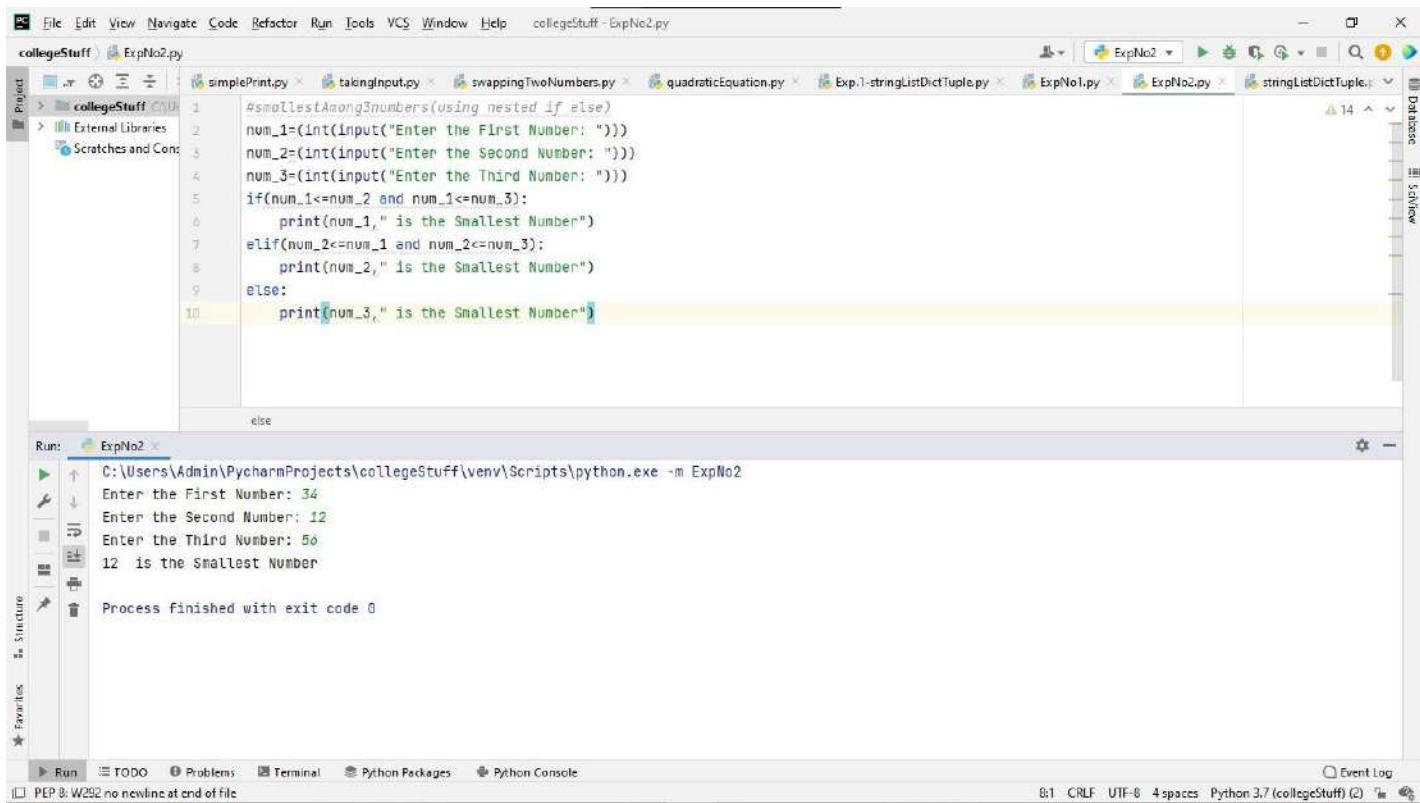
while i < 6:

peunt (i)

i + 1

so it will try to increment i, or else the loop will continue forever

### a) Smallest of 3 numbers (using nested if else)



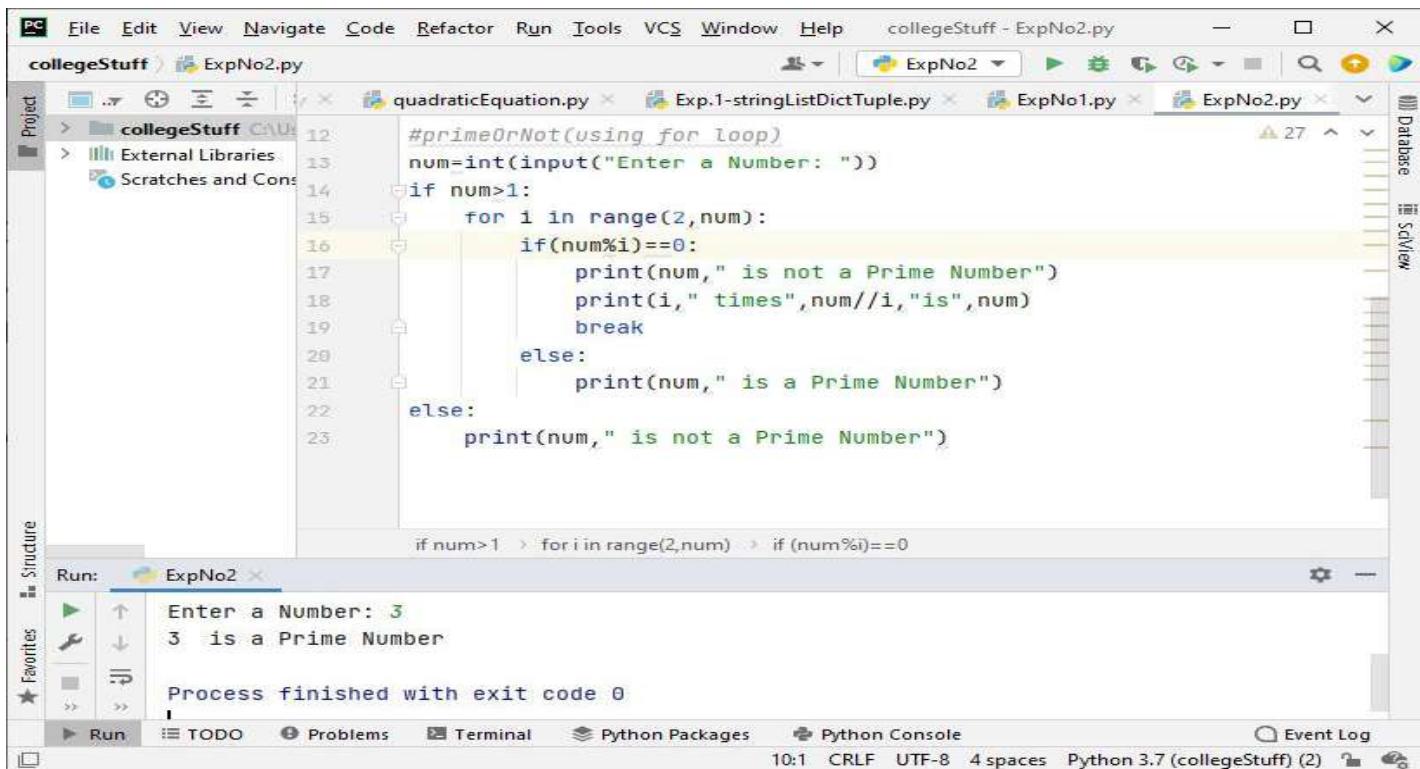
```
#smallestAmong3numbers(using nested if else)
num_1=int(input("Enter the First Number: "))
num_2=int(input("Enter the Second Number: "))
num_3=int(input("Enter the Third Number: "))
if(num_1<=num_2 and num_1<=num_3):
    print(num_1," is the Smallest Number")
elif(num_2<=num_1 and num_2<=num_3):
    print(num_2," is the Smallest Number")
else:
    print(num_3," is the Smallest Number")
```

The Run tab shows the output of the program:

```
C:\Users\Admin\PycharmProjects\collegeStuff\venv\Scripts\python.exe -m ExpNo2
Enter the First Number: 34
Enter the Second Number: 12
Enter the Third Number: 56
12 is the Smallest Number

Process finished with exit code 0
```

### b) To check if the input number is prime or not (using for loop)



```
#primeOrNot(using for loop)
num=int(input("Enter a Number: "))
if num>1:
    for i in range(2,num):
        if(num%i)==0:
            print(num," is not a Prime Number")
            print(i," times",num//i,"is",num)
            break
        else:
            print(num," is a Prime Number")
else:
    print(num," is not a Prime Number")
```

The Run tab shows the output of the program:

```
Enter a Number: 3
3 is a Prime Number

Process finished with exit code 0
```

c) To check if number provided by the user is Armstrong number or not(using while loop)

The screenshot shows the PyCharm IDE interface. The top menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help, and collegeStuff - ExpNo2.py. The toolbar has icons for file operations like Open, Save, and Run. The Project tool window on the left shows a folder named 'collegeStuff' containing 'ExpNo2.py'. The code editor displays Python code for checking if a number is Armstrong. The run tool window at the bottom shows the output of running the code with the input '866', which prints '866 is not an Armstrong Number!'. The status bar at the bottom right shows the file path 'collegeStuff (2)' and other details.

```
#armstrongOrNot(using While Loop)
num=int(input("Enter a Number: "))
sum=0
temp=num
while temp>0:
    digit=temp%10
    sum +=digit**3
    temp/=10
if num==sum:
    print(num," is an Armstrong Number!")
else:
    print(num," is not an Armstrong Number!")

else
```

Run: ExpNo2

Enter a Number: 866  
866 is not an Armstrong Number!

Process finished with exit code 0

Run TODO Problems Terminal Python Packages Python Console Event Log

12:1 CRLF UTF-8 4 spaces Python 3.7 (collegeStuff) (2)



Conclusion: we learned the different control statements and successfully executed the programs using the nested if else, for loop and the while loop in python.



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING  
(ARTIFICIAL INTELLIGENCE & MACHINE LEARNING)**

**S.E/SEM IV/CBCGS/AIML**  
**Academic Year: 2021-22**

<b>NAME</b>	PRATHAMESH CHIKANKAR
<b>BRANCH</b>	CSE-(AI&ML)
<b>ROLL NO.</b>	AIML08
<b>SUBJECT</b>	<b>PYTHON LAB</b>
<b>COURSE CODE</b>	<b>CSL405</b>
<b>PRACTICAL NO.</b>	03
<b>DOP</b>	<b>24/02/2022</b>
<b>DOS</b>	<b>24/03/2022</b>

### EXP NO : 3

Aim: program using concepts of functions, classes and objects..

i - a) make a simple calculator that can add, subtract, multiply and divide using function.

b) To find largest number from the list - i) using normal functn  
ii) using lambda functn

ii - a) write a program to print Employee information (use object variables and class variables).

#### Theory :

functions:- A functn is a block of code which only runs when it is called. You can pass data, known as parameters, into a functn. A functn can return data as a result.

In python a functn is defined using the def Keyword.

There are three types of functions in python.

- Built-in functn : help(), min(), print()

- User-defined functn: uses create.

- Anonymous functn: also called as lambda functn

Classes:- A class is like an object constructor, or a "blueprint" for creating objects.

To Create a class, use the Keyword class:

Eg:- Create a class name MyClass, with a property named x:

class MyClass:

x=5

- The \_\_init\_\_ () functn or del, pass, .. etc..

Objects:- To Create an object

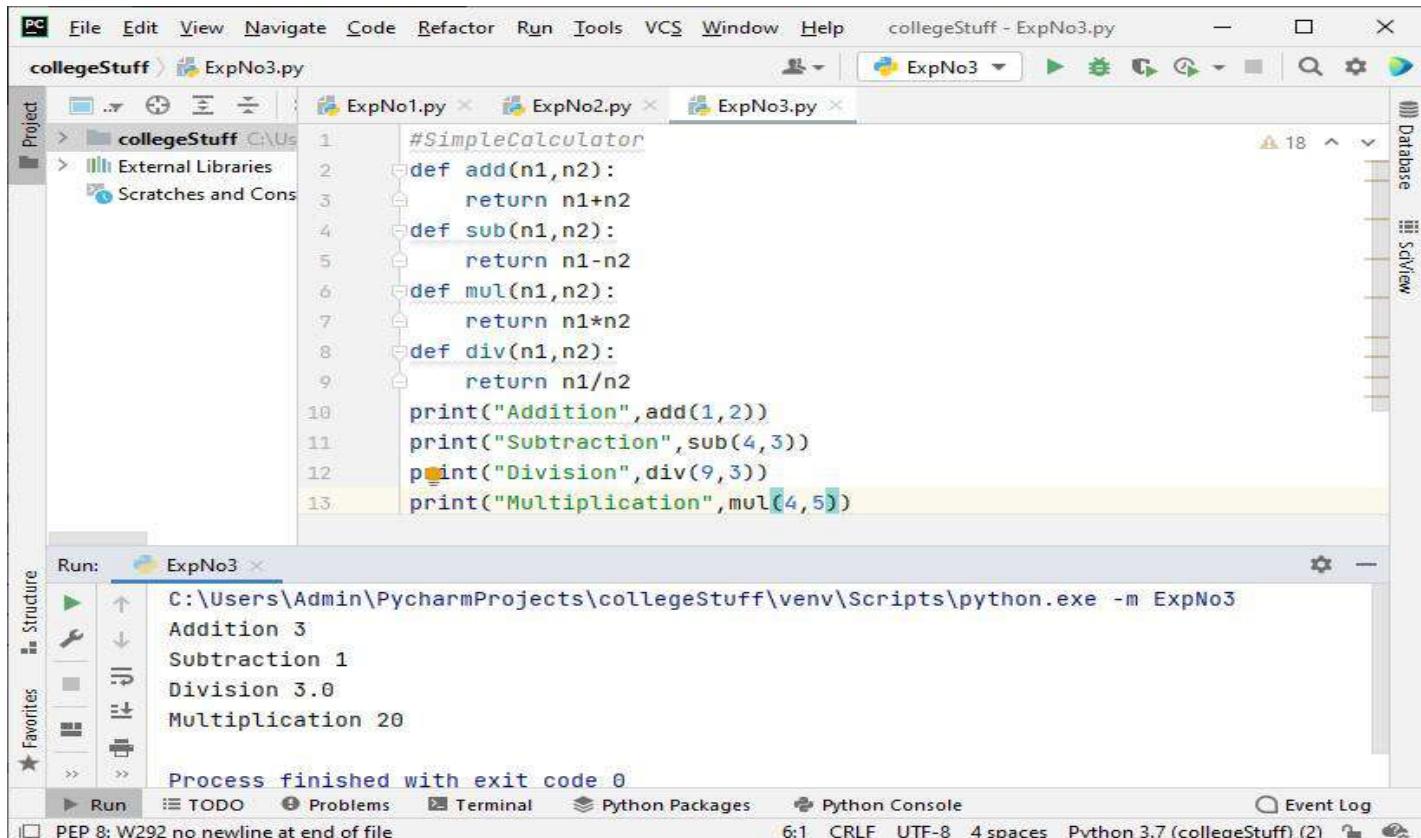
Eg:- p1 = MyClass()  
print(p1.x)

- Objects can also contain methods. Method in objects are functn that belong to the object.

Self parameter

## Program -

- > i-a) Make a simple calculator that can add, subtract, multiply and divide using function.



The screenshot shows the PyCharm IDE interface with the following details:

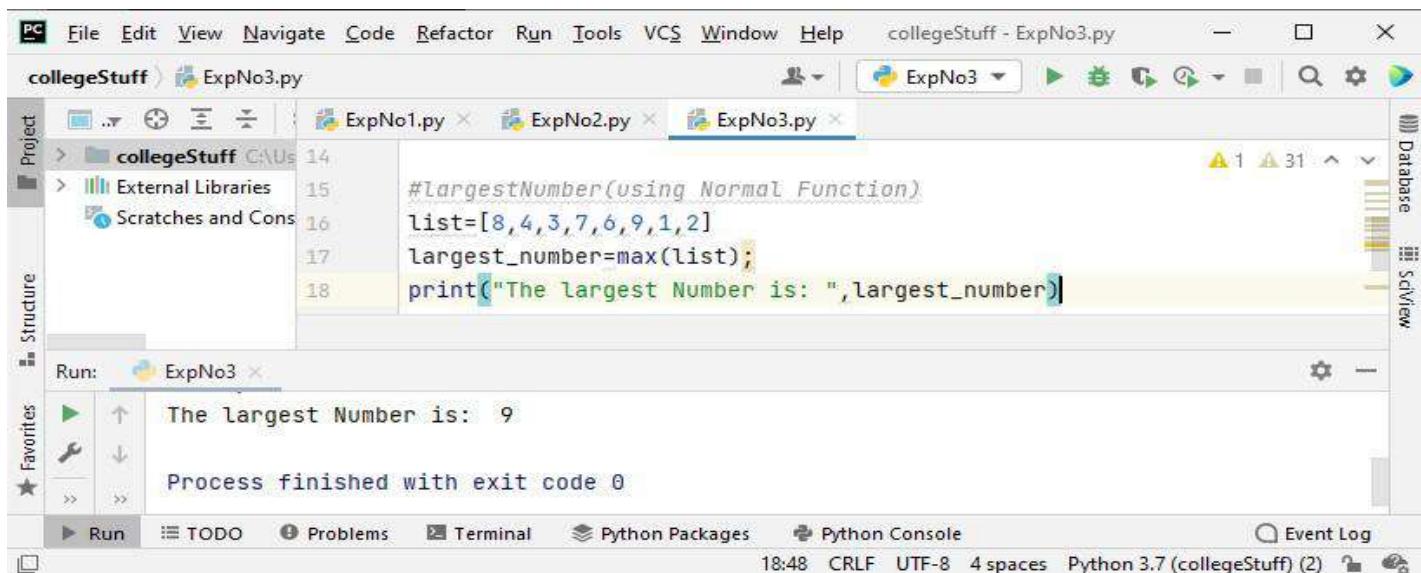
- File Menu:** PC, File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help.
- Title Bar:** collegeStuff - ExpNo3.py
- Toolbar:** Standard PyCharm icons for file operations.
- Project View:** collegeStuff (C:\Users\...).
- Code Editor:** Content of ExpNo3.py:
 

```
#SimpleCalculator
def add(n1,n2):
    return n1+n2
def sub(n1,n2):
    return n1-n2
def mul(n1,n2):
    return n1*n2
def div(n1,n2):
    return n1/n2
print("Addition",add(1,2))
print("Subtraction",sub(4,3))
print("Division",div(9,3))
print("Multiplication",mul(4,5))
```
- Run Tab:** Run configuration for ExpNo3, showing output:
 

```
C:\Users\Admin\PycharmProjects\collegeStuff\venv\Scripts\python.exe -m ExpNo3
Addition 3
Subtraction 1
Division 3.0
Multiplication 20
```
- Python Console:** Shows the process finished with exit code 0.
- Status Bar:** PEP 8: W292 no newline at end of file, 6:1 CRLF, UTF-8 4 spaces, Python 3.7 (collegeStuff) (2).

- b) To find largest number from the list

- i) using normal function



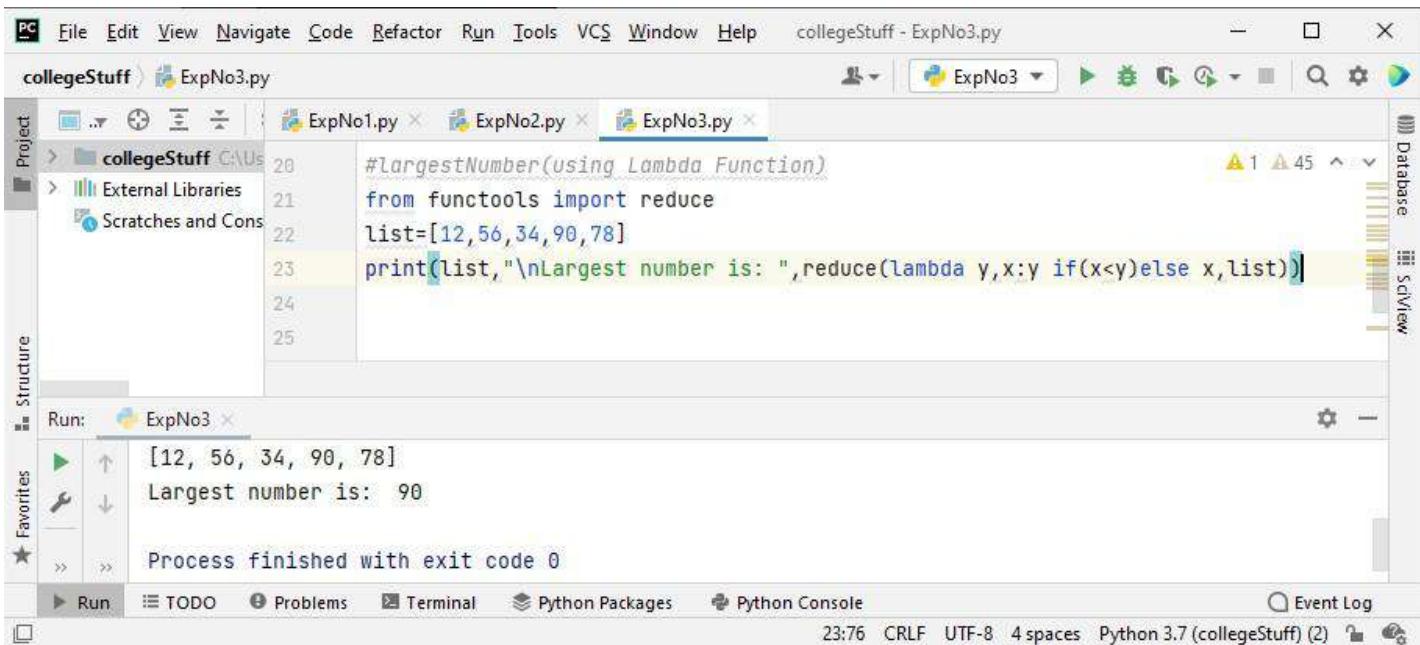
The screenshot shows the PyCharm IDE interface with the following details:

- File Menu:** PC, File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help.
- Title Bar:** collegeStuff - ExpNo3.py
- Toolbar:** Standard PyCharm icons for file operations.
- Project View:** collegeStuff (C:\Users\...).
- Code Editor:** Content of ExpNo3.py:
 

```
#largestNumber(using Normal Function)
list=[8,4,3,7,6,9,1,2]
largest_number=max(list);
print("The largest Number is: ",largest_number)
```
- Run Tab:** Run configuration for ExpNo3, showing output:
 

```
The largest Number is: 9
```
- Python Console:** Shows the process finished with exit code 0.
- Status Bar:** 18:48 CRLF, UTF-8 4 spaces, Python 3.7 (collegeStuff) (2).

## ii) using Lambda function

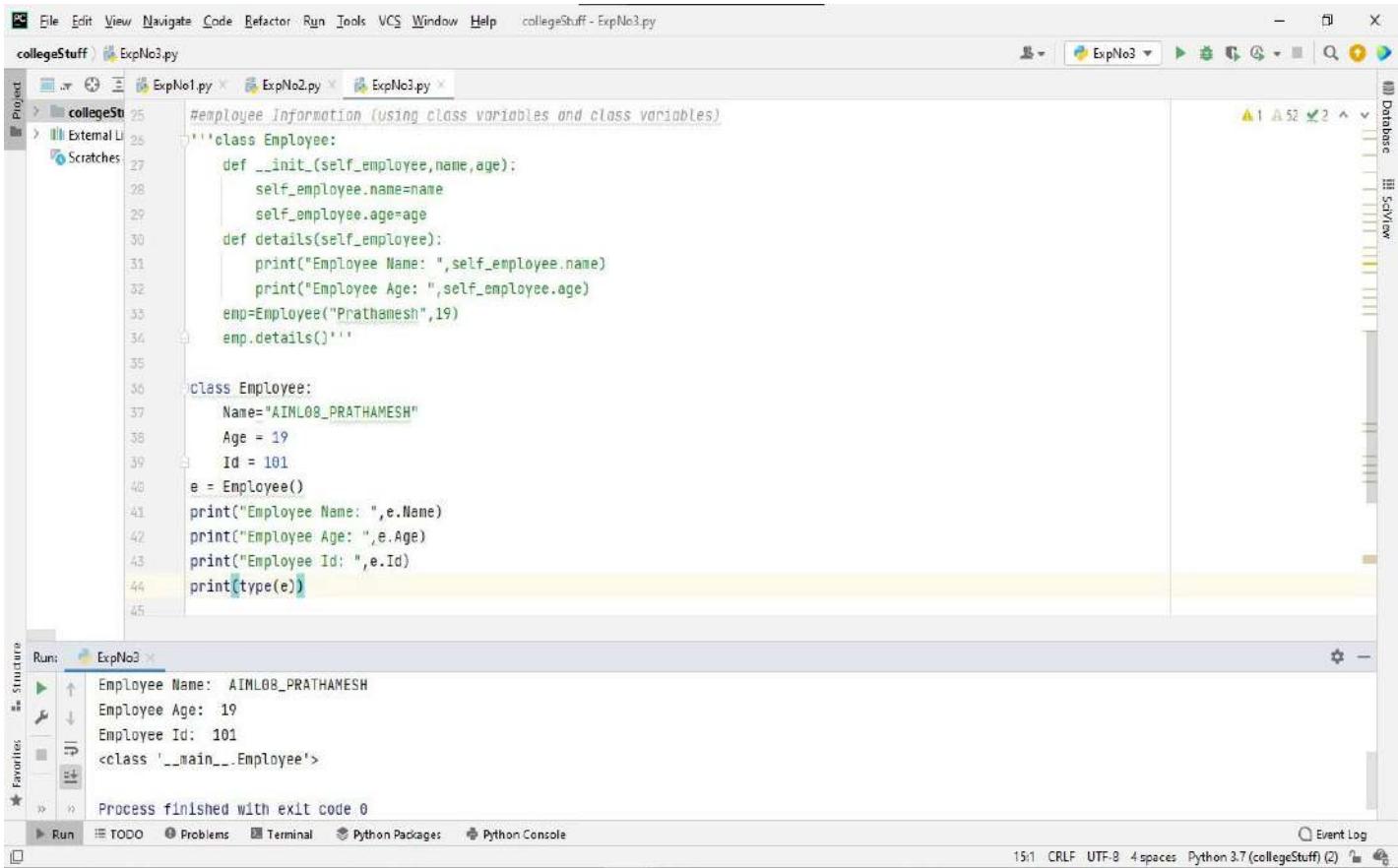


The screenshot shows the PyCharm IDE interface with the following details:

- File Menu:** File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help.
- Project:** collegeStuff - ExpNo3.py
- Code Editor:** Contains the following Python code:
 

```
#largestNumber(using Lambda Function)
from functools import reduce
list=[12, 56, 34, 90, 78]
print(list,"\nLargest number is: ",reduce(lambda y,x:y if(x<y)else x,list))
```
- Run Tab:** Shows the output of the run command: [12, 56, 34, 90, 78] and Largest number is: 90.
- Bottom Status Bar:** 23:76 CRLF UTF-8 4 spaces Python 3.7 (collegeStuff) (2)

> ii-a) Write a program to print Employee information (use object variables and class variables)



The screenshot shows the PyCharm IDE interface with the following details:

- File Menu:** File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, Help.
- Project:** collegeStuff - ExpNo3.py
- Code Editor:** Contains the following Python code:
 

```
#employee Information (using class variables and class variables)
'''class Employee:
    def __init__(self, name, age):
        self.employee.name=name
        self.employee.age=age
    def details(self):
        print("Employee Name: ",self.employee.name)
        print("Employee Age: ",self.employee.age)
emp=Employee("Prathamesh",19)
emp.details()'''

class Employee:
    Name="AIML08_PRATHAMESH"
    Age = 19
    Id = 101
e = Employee()
print("Employee Name: ",e.Name)
print("Employee Age: ",e.Age)
print("Employee Id: ",e.Id)
print(type(e))
```
- Run Tab:** Shows the output of the run command: Employee Name: AIML08\_PRATHAMESH, Employee Age: 19, Employee Id: 101, <class '\_\_main\_\_.Employee'>.
- Bottom Status Bar:** 151 CRLF UTF-8 4 spaces Python 3.7 (collegeStuff) (2)



Conclusion: we successfully executed programs on functions, classes and objects in python. also the object variable and class variables, function including Normal and lambda...



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING  
(ARTIFICIAL INTELLIGENCE & MACHINE LEARNING)**

**S.E/SEM IV/CBCGS/AIML  
Academic Year: 2021-22**

<b>NAME</b>	PRATHAMESH CHIKANKAR
<b>BRANCH</b>	CSE-(AI&ML)
<b>ROLL NO.</b>	AIML08
<b>SUBJECT</b>	<b>PYTHON LAB</b>
<b>COURSE CODE</b>	<b>CSL405</b>
<b>PRACTICAL NO.</b>	<b>04</b>
<b>DOP</b>	<b>10/03/2022</b>
<b>DOS</b>	<b>24/03/2022</b>



## EXP NO: 4

Aim: Program for file handling and directories...

Theory: Python 2.0 supports file handling and allows users to handle files i.e., to read and write files, along with many other file handling options, to operate on files.

- a) A program that counts the no. of characters in a text file.
- b) To count no. of words in a text file
- c) No. of lines in text file
- d) To display file available in current directory.

Python treats file differently as text or binary and it's important. Each line of code includes sequence of characters and they form text file.

Each one of file is terminated with a special character, called the EOL or End of Line characters like comma ',', ; or newline character.

r! w! a! => read, write and append

r: read file (existing), w: write file (if existing then overridden)

a: for append operation, it won't override existing data.

A directory or folder is a collection of files and subdirectories.

Python has the os module that provides us with many useful methods to work with directories (and files as well).

To get present working directory use getcwd(), for changing directory use chdir, list of subdirectories use listdir(), for making a new directory use mkdir(), to remove use rmdir().

for renaming use rename()

- getcwd()	- rename()
- chdir()	- rmdir()
- listdir()	- remove()
- mkdir()	

**Text file-**

test1.txt

Hey there

i'm prathamesh

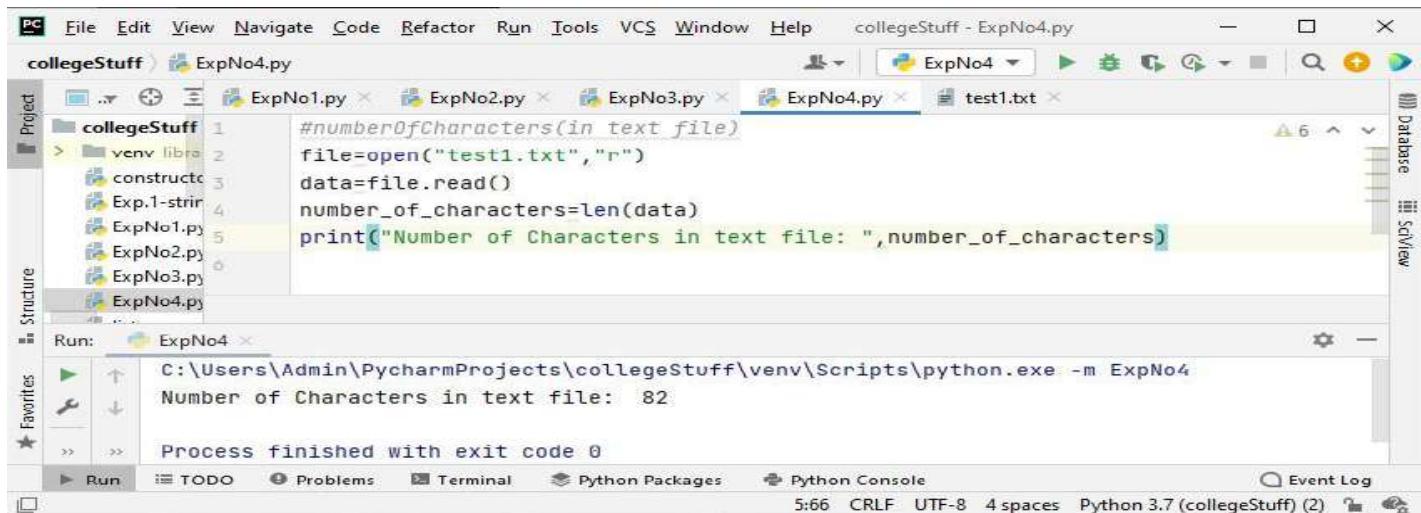
from cse-(ai&ml) branch

from ltcoe

located in navi mumbai

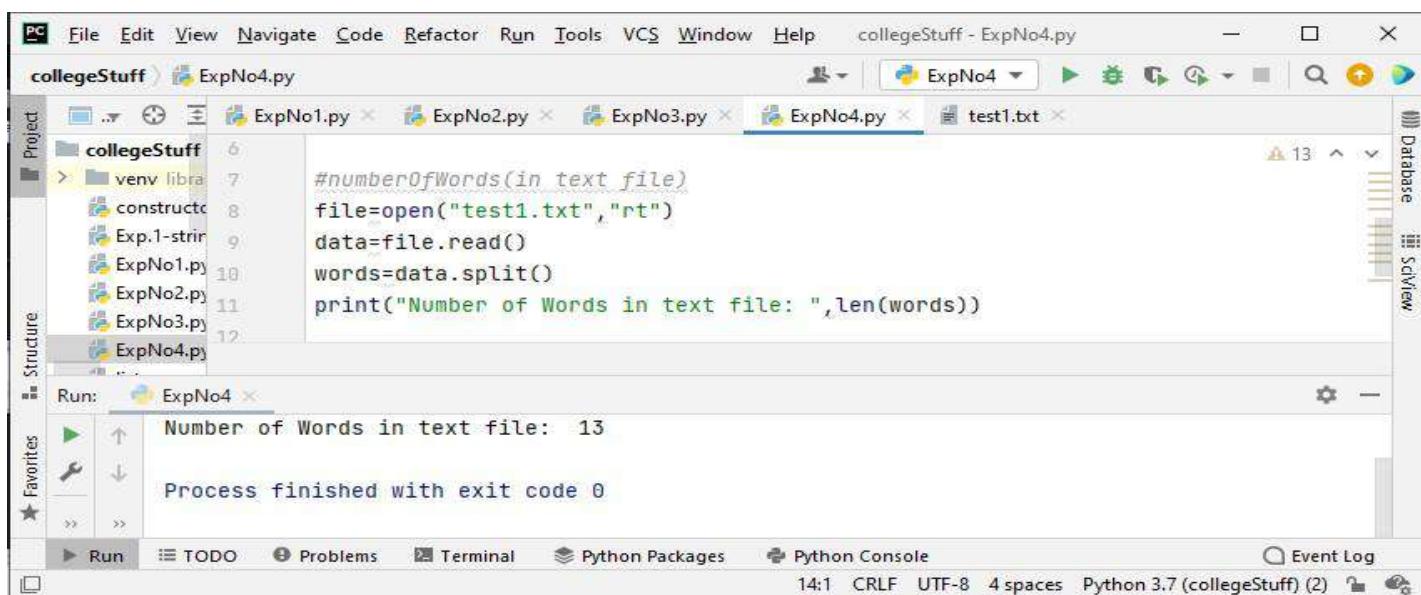
**Program -**

- a) A program that count the number of characters in a text file



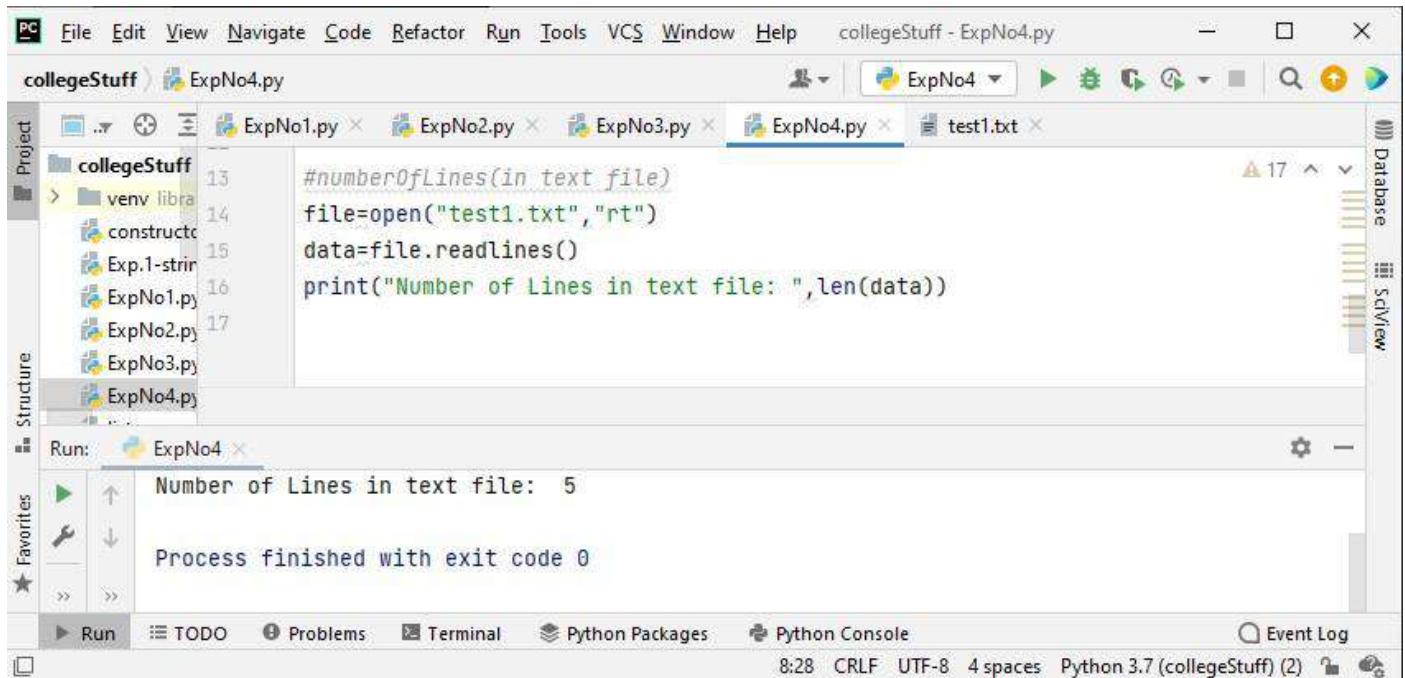
```
#numberOfCharacters(in text file)
file=open("test1.txt","r")
data=file.read()
number_of_characters=len(data)
print("Number of Characters in text file: ",number_of_characters)
```

- b) To count number of words in a text file



```
#numberOfWords(in text file)
file=open("test1.txt","rt")
data=file.read()
words=data.split()
print("Number of Words in text file: ",len(words))
```

c) A program that count the number of lines in a text file



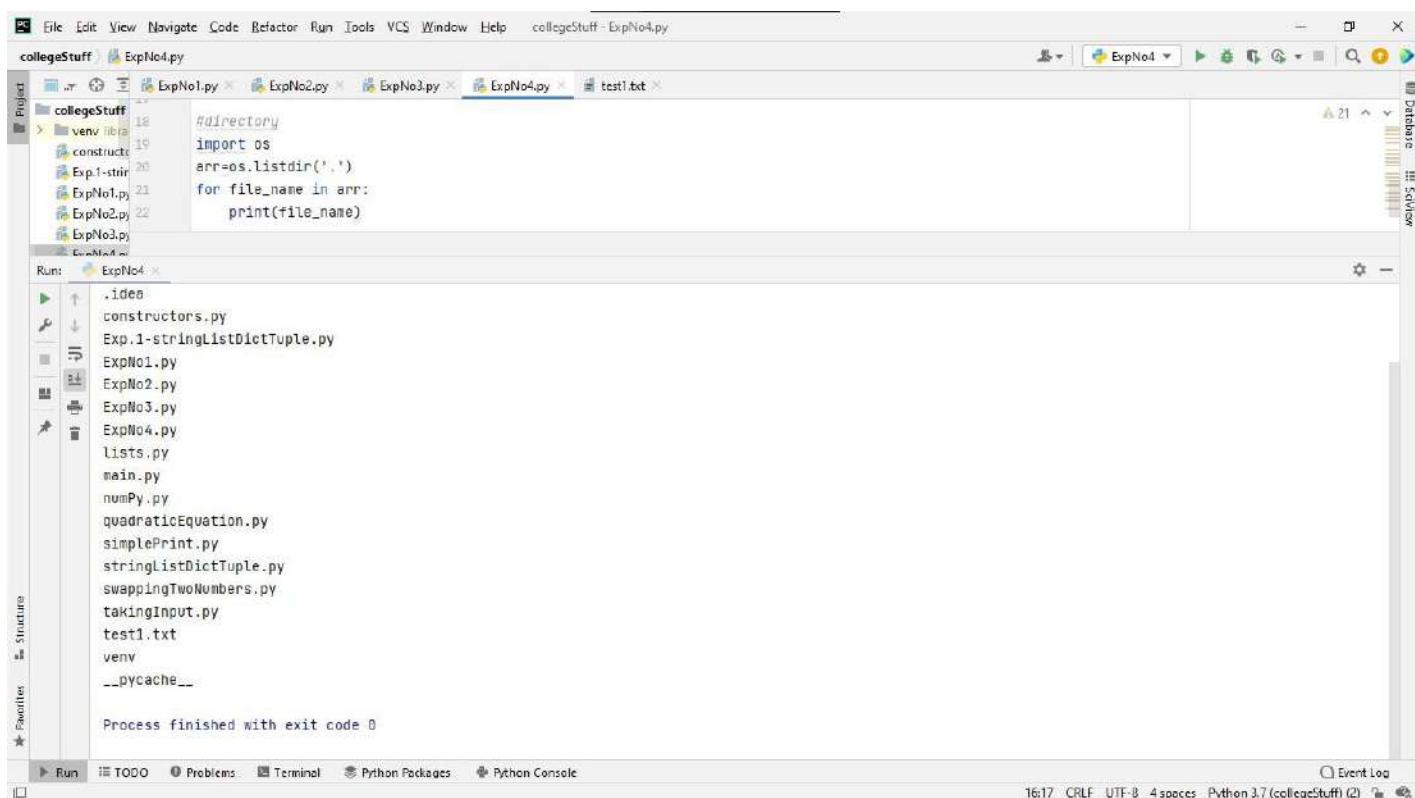
```

File Edit View Navigate Code Refactor Run Tools VCS Window Help collegeStuff - ExpNo4.py
collegeStuff > ExpNo4.py
Project collegeStuff
  venv libra
  constructors
  Exp.1-string
  ExpNo1.py
  ExpNo2.py
  ExpNo3.py
  ExpNo4.py
Structure
Run: ExpNo4
Number of Lines in text file: 5
Process finished with exit code 0
Run TODO Problems Terminal Python Packages Python Console Event Log
8:28 CRLF UTF-8 4 spaces Python 3.7 (collegeStuff) (2)

```

The screenshot shows the PyCharm IDE interface. The project name is 'collegeStuff'. The current file is 'ExpNo4.py'. The code in the editor counts the lines in 'test1.txt' and prints the result. The run output shows the program executed successfully with an exit code of 0.

d) To display file available in current directory



```

File Edit View Navigate Code Refactor Run Tools VCS Window Help collegeStuff - ExpNo4.py
collegeStuff > ExpNo4.py
Project collegeStuff
  venv libra
  constructors
  Exp.1-string
  ExpNo1.py
  ExpNo2.py
  ExpNo3.py
  ExpNo4.py
  lists.py
  main.py
  numPy.py
  quadraticEquation.py
  simplePrint.py
  stringListDictTuple.py
  swappingTwoNumbers.py
  takingInput.py
  test1.txt
  venv
  __pycache__
Structure
Run: ExpNo4
.idea
constructors.py
Exp.1-stringListDictTuple.py
ExpNo1.py
ExpNo2.py
ExpNo3.py
ExpNo4.py
lists.py
main.py
numPy.py
quadraticEquation.py
simplePrint.py
stringListDictTuple.py
swappingTwoNumbers.py
takingInput.py
test1.txt
venv
__pycache__
Process finished with exit code 0
Run TODO Problems Terminal Python Packages Python Console Event Log
16:17 CRLF UTF-8 4 spaces Python 3.7 (collegeStuff) (2)

```

The screenshot shows the PyCharm IDE interface. The project name is 'collegeStuff'. The current file is 'ExpNo4.py'. The code lists all files in the current directory. The run output shows the list of files present in the directory.



P : 04983

Conclusion: Executed the programs based on file handling and directories such as counting number of lines, words, characters and displayed file also.



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING  
(ARTIFICIAL INTELLIGENCE & MACHINE LEARNING)**

**S.E/SEM IV/CBCGS/AIML  
Academic Year: 2021-22**

<b>NAME</b>	PRATHAMESH CHIKANKAR
<b>BRANCH</b>	CSE-(AI&ML)
<b>ROLL NO.</b>	AIML08
<b>SUBJECT</b>	<b>PYTHON LAB</b>
<b>COURSE CODE</b>	<b>CSL405</b>
<b>PRACTICAL NO.</b>	<b>05</b>
<b>DOP</b>	<b>17/03/2022</b>
<b>DOS</b>	<b>24/03/2022</b>



## EXP NO: 5

Aim: program to demonstrate i) stack, ii) queue, iii) Linked List...

Theory:

Stack - A stack is a data structure that follows the LIFO (last in first out) principle. To implement a stack, we need two simple operations:

- push - it adds an element to the top of the stack.
- pop - it removes an element from the top of the stack.

Queue - A queue follows the first-in-first-out (FIFO) principle.

It is opened from both the ends hence we can easily add elements to the back and can remove elements from the front.

To implement a queue, we need two simple operations.

- enqueue - It adds an element to the end of the queue
- dequeue - It removes the element from the beginning of the queue.

Operations on stack and queue  $\Rightarrow$  Adding, Deletion, Traversing, update

Characteristics - insertion order of stack & queue is preserved.

- duplicacy is allowed & useful for passing CPU operation.

Linked List - Linked List is also a linear DS but it is slightly different from the other DS like stack, queue. Linked List have special structure called Node which have two fields Data and Next Node Address. So in Linked List all the node contains the address or points to the next node of the Linked List.

Types -  $\rightarrow$  Simple Linked List -

- $\triangleright$  Doubly Linked List - one is previous pointer and other is next.
- $\triangleright$  circular Linked List - last node contains points of first node.

## Program with Output -

### i) stack

```
#stack
class Stack:
    stack=[None]*5
    pos=-1
    def push(self,element):
        if self.pos==4:
            print("stack overflow")
            return
        self.pos+=1
        self.stack[self.pos]=element
    def pop(self):
        if self.pos== -1:
            print("stack underflow")
        else:
            print("pos",self.pos)
            print(self.stack[self.pos]," popped out from the stack")
            popped=self.stack[self.pos]
            self.stack[self.pos]=None
            self.pos-=1
    def display(self):
        for element in self.stack:
            print(element)
s=Stack()
while True:
    i=int(input("1.push \n2.pop \n3.display \n0.exit \n"))
    if i==1:
        element=int(input("Enter element to push in the stack: "))
        s.push(element)
    elif i==2:
        s.pop()
    elif i==3:
        s.display()
    elif i==0:
        break
```

## AIML08\_PRATHAMESH

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\Admin> python -u "f:\Prathamesh\Study-Time\SE\SEM4\PYTHON\stack.py"
1.push
2.pop
3.display
0.exit
1
Enter element to push in the stack: 12
1.push
2.pop
3.display
0.exit
1
Enter element to push in the stack: 34
1.push
2.pop
3.display
0.exit
3
12
34
None
None
None
1.push
2.pop
3.display
0.exit
2
pos 1
34  popped out from the stack
```

```
1.push
2.pop
3.display
0.exit
3
12
None
None
None
None
1.push
2.pop
3.display
0.exit
0
PS C:\Users\Admin>
```

**b) queue**

```

class Queue:
    def __init__(self):
        self.QueueLen=5
        self.queue=[None]*self.QueueLen
        self.front=-1
        self.rear=-1
    def insert(self,element):
        if self.rear==self.QueueLen-1:
            print("queue is full")
            return
        self.rear+=1
        if self.front==-1:
            self.front=0
        self.queue[self.rear]=element
    def delete(self):
        if self.front!=-1 and self.rear==self.front:
            self.front=self.rear=-1
            return
        if self.front== -1 and self.rear== -1:
            print("queue is empty")
            return
        self.front+=1
    def display(self):
        if self.front== -1:
            print("empty")
            return
        for i in range(self.front,self.rear+1):
            print(self.queue[i], " -> ", end = " ")
q=Queue()
while True:
    i=int(input("\n1.insert \n2.delete \n3.display \n0.exit \n"))
    if i==1:
        element=int(input("Enter element to insert in the queue: "))
        q.insert(element)
    elif i==2:
        q.delete()
    elif i==3:
        q.display()

```

## AIML08\_PRATHAMESH

```
    elif i==0:  
        break  
  
PS C:\Users\Admin> python -u "f:\Prathamesh\Study-Time\SE\SEM4\PYTHON\queue.py"  
  
1.insert  
2.delete  
3.display  
0.exit  
1  
Enter element to insert in the queue: 12  
  
1.insert  
2.delete  
3.display  
0.exit  
1  
Enter element to insert in the queue: 34  
  
1.insert  
2.delete  
3.display  
0.exit  
3  
12 -> 34 ->  
1.insert  
2.delete  
3.display  
0.exit  
2  
  
1.insert  
2.delete  
3.display  
0.exit  
3  
34 ->  
1.insert  
  
2.delete  
3.display  
0.exit  
0  
PS C:\Users\Admin>
```

### c)linked list

```
# Create a node
class Node:

    def __init__(self, item):
        self.item = item
        self.next = None

class LinkedList:

    def __init__(self):
        self.head = None

    # Insert at the beginning
    def insertAtBeginning(self, data):
        new_node = Node(data)
        new_node.next = self.head
        self.head = new_node

    # Insert at the end
    def insertAtEnd(self, data):
        new_node = Node(data)
        if self.head is None:
            self.head = new_node
            return
        last = self.head
        while (last.next):
            last = last.next
        last.next = new_node

    # Deleting a node
    def deleteNode(self, position):
        if self.head == None:
            return
        temp_node = self.head
        if position == 0:
            self.head = temp_node.next
            temp_node = None
            return
        # Find the key to be deleted
        for i in range(position - 1):
            temp_node = temp_node.next
            if temp_node is None:
                break
        # If the key is not present
```

```
if temp_node is None:
    return
if temp_node.next is None:
    return
next = temp_node.next.next
temp_node.next = None
temp_node.next = next
def printList(self):
    temp_node = self.head
    while (temp_node):
        print(str(temp_node.item) + "->", end="")
        temp_node = temp_node.next
if __name__ == '__main__':
    llist = LinkedList()
    llist.insertAtEnd(1)
    llist.insertAtBeginning(2)
    llist.insertAtBeginning(3)
    llist.insertAtEnd(4)
    print('Linked list: ')
    llist.printList()
    print("\nAfter deleting an element:")
    llist.deleteNode(3)
    llist.printList()
```

```
PS C:\Users\Admin> python -u "f:\Prathamesh\Study-Time\SE\SEM4\PYTHON\linkedList.py"
Linked list:
3->2->1->4->
After deleting an element:
3->2->1->
PS C:\Users\Admin> █
```

Conclusion: Demonstrated the programs based on stack, queue and linked list in python.



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING  
(ARTIFICIAL INTELLIGENCE & MACHINE LEARNING)**

**S.E/SEM IV/CBCGS/AIML  
Academic Year: 2021-22**

<b>NAME</b>	PRATHAMESH CHIKANKAR
<b>BRANCH</b>	CSE-(AI&ML)
<b>ROLL NO.</b>	AIML08
<b>SUBJECT</b>	<b>PYTHON LAB</b>
<b>COURSE CODE</b>	<b>CSL405</b>
<b>PRACTICAL NO.</b>	<b>06</b>
<b>DOP</b>	<b>10/03/2022</b>
<b>DOS</b>	<b>24/03/2022</b>



## EXP NO 6

Aim: program to demonstrate use of NumPy : array objects.

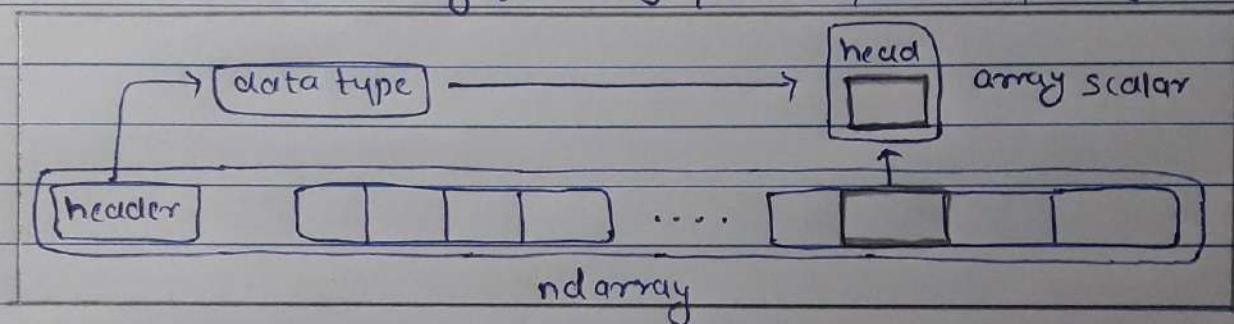
Exploring basics of NumPy methods.

Theory :

Array objects - Numpy provides an N-dimensional array type, the ndarray, which describes a collection of "items" of the same type. The items can be indexed using for example N integers.

All ndarrays are homogenous: every item takes up the same size block of memory, and all blocks are interpreted in exactly the same way.

The N-dimensional array (ndarray), scalars, data type objects (dtype)



NumPy Methods -

numpy.ndarray =

class numpy.ndarray (shape, dtype=float,

float buffer=None, offset=0, strides=None,  
order=None)

\* Array Creation , Array Indexing , Basic operations, sorting array.

- np.array

arr.size

arr.dtype

arr.toues()

np.copy (arr)

dtype arr.sort()

arr.reshape (3,4)

np.append

np.delete

arr np.insert

np.eye

## Program with Output -

The screenshot shows the PyCharm IDE interface with the project 'collegeStuff' open. The code editor displays `ExpNo6.py` containing Python code demonstrating various operations on NumPy arrays. The 'Run' tool window shows the execution results of the code.

```

#numpy- array objects
import numpy as np
a = np.array([1, 2, 3]) # Create a rank 1 array
print("type",type(a)) # Prints <class 'numpy.ndarray'>
print("dtype",a.dtype)
print("shape",a.shape) # Prints (3,)
print("array index",a[0], a[1], a[2]) # Prints 1 2 3
a[0] = 5 # Change an element of the array
print("mutation",a) # Prints [5, 2, 3]
b = np.array([[1,2,3],[4,5,6]]) # Create a rank 2 array
print("b array",b.shape) # Prints (2, 3)
print("2d array access",b[0, 0], b[0, 1], b[1, 0]) # Prints 1 2 4
print("b array slice",b[:,2])
print("array add",np.add(a, b))

```

The 'Run' tool window output is:

```

type <class 'numpy.ndarray'>
dtype int32
shape (3,)
array index 1 2 3
mutation [5 2 3]
b array (2, 3)
2d array access 1 2 4
b array slice [[1 2 3]
 [4 5 6]]
array add [[6 4 6]
 [9 7 9]]

```

The screenshot shows the PyCharm IDE interface with the project 'collegeStuff' open. The code editor displays `ExpNo6.py` containing Python code demonstrating various methods of NumPy arrays. The 'Run' tool window shows the execution results of the code.

```

#numpy- methods
import numpy as np
a = np.array([1,2,3])
b = np.array([34,5,0])
print("subtract",np.subtract(a,b))
print("add",np.add(a,b))
print("multiply",np.multiply(a,b))
print("itemsize",a.itemsize)
print("max",a.max())
print("min",b.min())
print("dtype",a.dtype)
print("raise to the power 2",a**2)
print("sort",np.sort(a))

```

The 'Run' tool window output is:

```

C:\Users\Admin\PycharmProjects\collegeStuff\venv\Scripts\python.exe -m ExpNo6
subtract [-33 -3 -3]
add [35 7 9]
multiply [34 10 18]
itemsize 4
max 3
min 5
dtype int32
raise to the power 2 [1 4 9]
sort [1 2 3]

Process finished with exit code 0

```

Conclusion:- Demonstrate various array objects and Basic methods in Numpy and the use of Numpy in Python...



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING  
(ARTIFICIAL INTELLIGENCE & MACHINE LEARNING)**

**S.E/SEM IV/CBCGS/AIML  
Academic Year: 2021-22**

<b>NAME</b>	PRATHAMESH CHIKANKAR
<b>BRANCH</b>	CSE-(AI&ML)
<b>ROLL NO.</b>	AIML08
<b>SUBJECT</b>	<b>PYTHON LAB</b>
<b>COURSE CODE</b>	<b>CSL405</b>
<b>PRACTICAL NO.</b>	<b>07</b>
<b>DOP</b>	<b>24/03/2022</b>
<b>DOS</b>	<b>31/03/2022</b>

EXP NO 07

Aim :- program to demonstrate Data Series and Data Frames using python, pandas...

Theory :- The pandas Series data structure is a one-dimensional labelled array. It is the primary building block for a DataFrame making up its rows and columns. You can view the constructor for the Series.

The data parameter can accept several different data types such as ndarray, dictionaries and scalar values.

- random.ranndint() } pd.Series  
series - from-dict

The pandas DataFrame is a two-dimensional data structure composed of columns and rows. You can think of the DataFrame as similar to a CSV or relational database table.

The data parameter similar to series can accept a broad range of data types such as series, a dictionaries of series, structured arrays and Numpy arrays.

- pip install pandas } pd.DataFrame  
- pip install Numpy

## Program with Output -

### DataSeries -

The screenshot shows a Jupyter Notebook interface with a code cell titled "main.py" and a "Shell" output area.

```
main.py
1 # AIML08_PRATHAMESH
2 import pandas as pd
3 if __name__ == '__main__':
4     data={'a':1.0,'b':2.0,'c':3.0,'d':4.0}
5     series=pd.Series(data=data,name='series_from_dict')
6     print(series)
```

**Shell**

```
a    1.0
b    2.0
c    3.0
d    4.0
Name: series_from_dict, dtype: float64
```

## DataFrame -

The screenshot shows an Online Python Compiler interface with a code cell titled "main.py" and a "Shell" output area.

```
main.py
1 import pandas as pd
2 data = {'Name': ['Prathamesh', 'Sudham', 'Babbu', 'Monu', 'Dhruv'],
3         'Age': [20, 21, 19, 18, 21],
4         'Roll_NO': [101,102,103,104,105],
5         'Branch': ['CSE-(AI&ML)', 'CSE-(IoT&CS)', 'CSE-(DS)', 'CSE-(AI&ML)', 'CE']}
6
7 df = pd.DataFrame(data)
8 print(df)
```

**Shell**

Name	Age	Roll_NO	Branch
0 Prathamesh	20	101	CSE-(AI&ML)
1 Sudham	21	102	CSE-(IoT&CS)
2 Babbu	19	103	CSE-(DS)
3 Monu	18	104	CSE-(AI&ML)
4 Dhruv	21	105	CE

## Matplotlib -

Activities Matplotlib • Thu 3:36 PM • pandasMPL.py - /home/computer/pandasMPL.py (3.6.9)

File Edit Format Run Options Window Help

```
import pandas as pd
import matplotlib.pyplot as plt
author = ['Jitender', 'Purnima', 'Arpit', 'Jyoti']
article = [210, 211, 114, 178]
auth_series= pd.Series(author)
article_series= pd.Series(article)
frame = {'Author': auth_series, 'Article': article_series }
result = pd.DataFrame(frame)
age = [21, 21, 24, 23]
result['Age'] = pd.Series(age)
result.plot.bar()
plt.show()
```

Figure 1

Category	Article	Age
o	210	25
i	211	25
n	114	25
m	178	25

Ln: 12 Col: 10

TO 014 953

Conclusion :- Successfully demonstrated the data series and data frame using pandas package in Python.



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING  
(ARTIFICIAL INTELLIGENCE & MACHINE LEARNING)**

**S.E/SEM IV/CBCGS/AIML**  
**Academic Year: 2021-22**

<b>NAME</b>	PRATHAMESH CHIKANKAR
<b>BRANCH</b>	CSE-(AI&ML)
<b>ROLL NO.</b>	AIML08
<b>SUBJECT</b>	<b>PYTHON LAB</b>
<b>COURSE CODE</b>	<b>CSL405</b>
<b>PRACTICAL NO.</b>	<b>08</b>
<b>DOP</b>	<b>31/03/2022</b>
<b>DOS</b>	<b>21/04/2022</b>

## Exp No 08

Aim: Creating GUI with python containing widgets such as labels, textbox, radio, checkboxes and custom dialog boxes.

Theory: Python offers multiple options for developing GUI (Graphical user interface). Out of all the GUI methods, Tkinter is the most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python.

Python with Tkinter is the fastest and easiest way to create a GUI using Tkinter is an easy task. Creating a GUI application. There are a number of widgets which you can use in your Tkinter application. Some of them are:

- 1) Button: To add a button in any application, this widget is used.
- 2) Checkbutton: To select any number of options by displaying a number of options to a user as toggle buttons.
- 3) Entry: It is used to input the single line text entry from the user, for multi-line text input, text widget is used.
- 4) Frame: It acts as a container to hold the widgets. It is used for growing and organizing widget.
- 5) Radio button: It is used to offer multi-choice option to the user. It offers several options to the user and the user has to choose one option.

## Program with output -

The screenshot shows a Linux desktop environment with a dark theme. On the left, there's a vertical dock with icons for various applications like a file manager, terminal, and system settings. The main window is a terminal window titled "Thu 7:53 PM" showing the command-line interface of a Python script named "GUI.py". The script uses the Tkinter library to create a login form. Below the terminal is a Python code editor with the same "GUI.py" file open, displaying the code. A modal dialog box titled "tk #2" is displayed, representing the running Tkinter application. It has fields for Username (prathamesh), Password (\*\*\*\*), and Address (empty). Under "Favourite Subject", there are three checkboxes: CP, JAVA, and PYTHON, with CP checked. Under "Gender", there are two radio buttons: Male (selected) and Female. A "Login" button is at the bottom. A message box titled "WEICOM3" is also visible, stating "Successful Login..." with an OK button.

```

from tkinter import *
from tkinter import messagebox
root=Tk()
f=Frame(root,height=400,width=500)
f.pack()
def cmd():
    s1=e1.get()
    s2=e2.get()
    i=0
    if s1=='prathamesh' and s2=='pc77':
        messagebox.showinfo('WEICOM3','Successful Login....')
    elif s1=='prathamesh' and s2=='pc77':
        messagebox.showerror('Error','Invalid Username....')
    elif s1 == 'prathamesh' and s2=='pc77':
        messagebox.showerror('Error','Invalid Username....')
    elif s1 != 'prathamesh' and s2 != 'pc77':
        messagebox.showwarning('Error','Invalid Username and Password')

l1=Label(f,text='Username', width=20,height=2)
l2=Label(f,text='Password',width=20,height=2)
l3=Label(f,text='Address',width=20,height=2)
l4=Label(f,text='Favourite Subject',width=20,height=2)
l5=Label(f,text='Gender',width=20,height=2)

e1=Entry(f,width=20)
e2=Entry(f,width=20,show="*")
t=Text(f,width=20,height=5)
c1=Radiobutton(f,text='CP')
c2=Radiobutton(f,text='JAVA')
c3=Radiobutton(f,text='PYTHON')
var=IntVar()
r1=Radiobutton(f,text='Male', variable=var,value=1)
r2=Radiobutton(f,text='Female', variable=var,value=2)
b=Button(f,text='Login',width=15,height=2,command=cmd)
l1.grid(row=0,column=0)
e1.grid(row=0,column=1)
l2.grid(row=1,column=0)
e2.grid(row=1,column=1)
l3.grid(row=2,column=0)
t.grid(row=2,column=1)
l4.grid(row=3,column=0)
c1.grid(row=3,column=1)
c2.grid(row=3,column=2)
c3.grid(row=3,column=3)
l5.grid(row=4,column=0)
r1.grid(row=4,column=1)
r2.grid(row=4,column=2)
b.grid(row=5,column=0)
root.mainloop()

```

Python ▾ Tab Width: 8 ▾ Ln 6, Col 11 ▾ INS

Conclusion: Created GUI with python containing widgets...



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING  
(ARTIFICIAL INTELLIGENCE & MACHINE LEARNING)**

**S.E/SEM IV/CBCGS/AIML  
Academic Year: 2021-22**

<b>NAME</b>	PRATHAMESH CHIKANKAR
<b>BRANCH</b>	CSE-(AI&ML)
<b>ROLL NO.</b>	AIML08
<b>SUBJECT</b>	<b>PYTHON LAB</b>
<b>COURSE CODE</b>	<b>CSL405</b>
<b>PRACTICAL NO.</b>	<b>09</b>
<b>DOP</b>	<b>31/03/2022</b>
<b>DOS</b>	<b>21/04/2022</b>

Exp. No. 09

Aim: write a program to demonstrate CRUD (Create, read, update and delete) operation on database (SQLite / MySQL)

Theory:

MySQL is a relational database management system based on SQL - Structured Query Language.

The application is used for a wide range of purposes, including data warehousing, e-commerce, and lagging application. The most common use for MySQL however is for the purpose of a web database.

Connecting MySQL database in python :

We can connect MySQL database with a python program which enables us to manipulate the SQL database using Python commands.

- connect()

The connect() constructor creates a connection to the MySQL server and returns a MySQL connection object.

**PROGRAM -**

```

import pymysql as sql
mydb = sql.connect(host='localhost', user='root', password='', database='emp')
mycursor = mydb.cursor()
while True:
    print("\t menu \n1.create data\n2.read data\n3.update data\n4.delete data\n5.exit")
    ch = int(input("enter your choice"))
    if ch == 1:
        emp_id = input("enter the employee id : ")
        emp_name = input("enter the employee name : ")
        emp_age = int(input("enter the employee age : "))
        branch = input("enter the employee branch : ")
        salary = int(input("enter the employee salary : "))
        q = "insert into `empinfo` values ('"+emp_id+"','"+emp_name+"','"+str(emp_age)+"','"+branch+"','"+str(salary)+"');"
        mycursor.execute(q)
    elif ch == 2:
        print("the data in the table is")
        q = "select * from empinfo;"
        mycursor.execute(q)
        print(mycursor.fetchall())
        for i in mycursor:
            print(i + "\n")
    elif ch == 3:
        print("\tupdating the salary of a person - ")
        emp_id = input("enter the employee id : ")
        sal = int(input("enter the new salary of the person : "))
        q = "UPDATE `empinfo` SET salary =" + str(sal) + " WHERE emp_id = '" + emp_id + "';"
        mycursor.execute(q)
        mydb.commit()
        print(mycursor.rowcount, "updated successfully")
    elif ch == 4:
        print("\tdeleting a employee record - ")
        id = input("enter the employee id : ")
        q = "delete from empinfo where emp_id ='" + id + "'"
        mycursor.execute(q)
        mydb.commit()
    elif ch == 5:
        print("byee")
        break
    else:
        print("invalid choice !!!")
        continue

```

## **OUTPUT -**

Conclusion: we have successfully connected the SQL database with python and also manipulated the data stored in the SQL database.