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

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

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

EXP NO: 1

Aim: Program using Basic Data types (Numeric (int, float), list, Tuple, set, dictionaries and string)

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

b) To swap two numbers

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

d) Perform (at least 5) operation on each

i) list, ii) Tuple, iii) set, iv) dictionaries, v) string

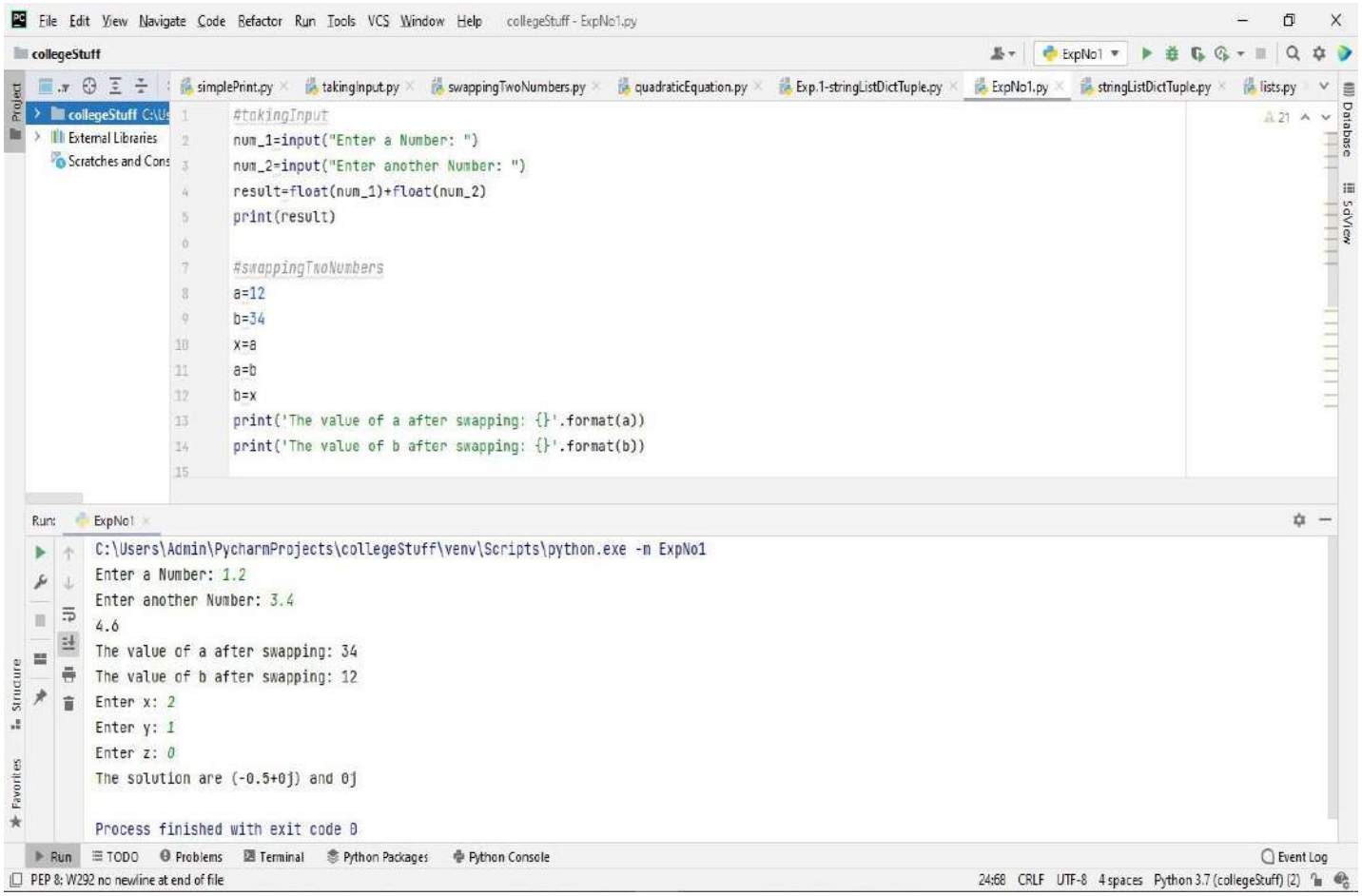
Theory:

- int (Signed integers) eg:- 10, -0X260
- float (floating point real values) eg:- 0.0, -32.54e100
- list → non-homogenous data structure that stores the elements in single row and multiple rows and columns.
  → list can be represented by []
- Tuple → non-homogenous data structure that stores the elements single row and multiple rows and columns.
  → Tuple can be represented by ()
- set → non-homogenous data structure but stores in single row.
  → Set can be represented by {}
- Dictionary → non-homogenous 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. " ".

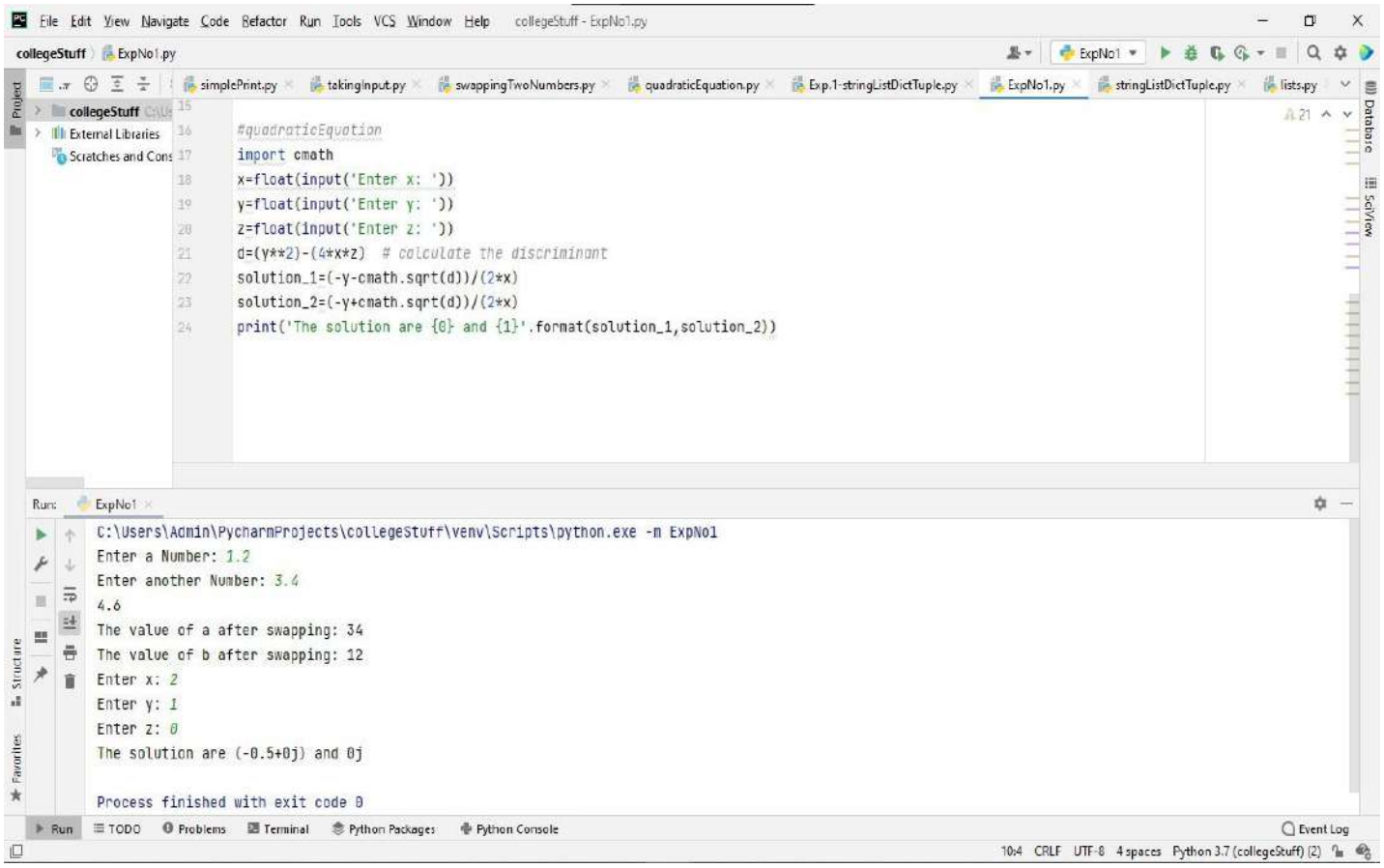
	list	tuple	set	Dictionary
→	l=[]	t=()	a=set()	d={}
			b=set(a)	
→	ordered	ordered	unordered	ordered (in python 3.7 & T)
→	nested	nested	nested	nested

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

b)Two swap two numbers

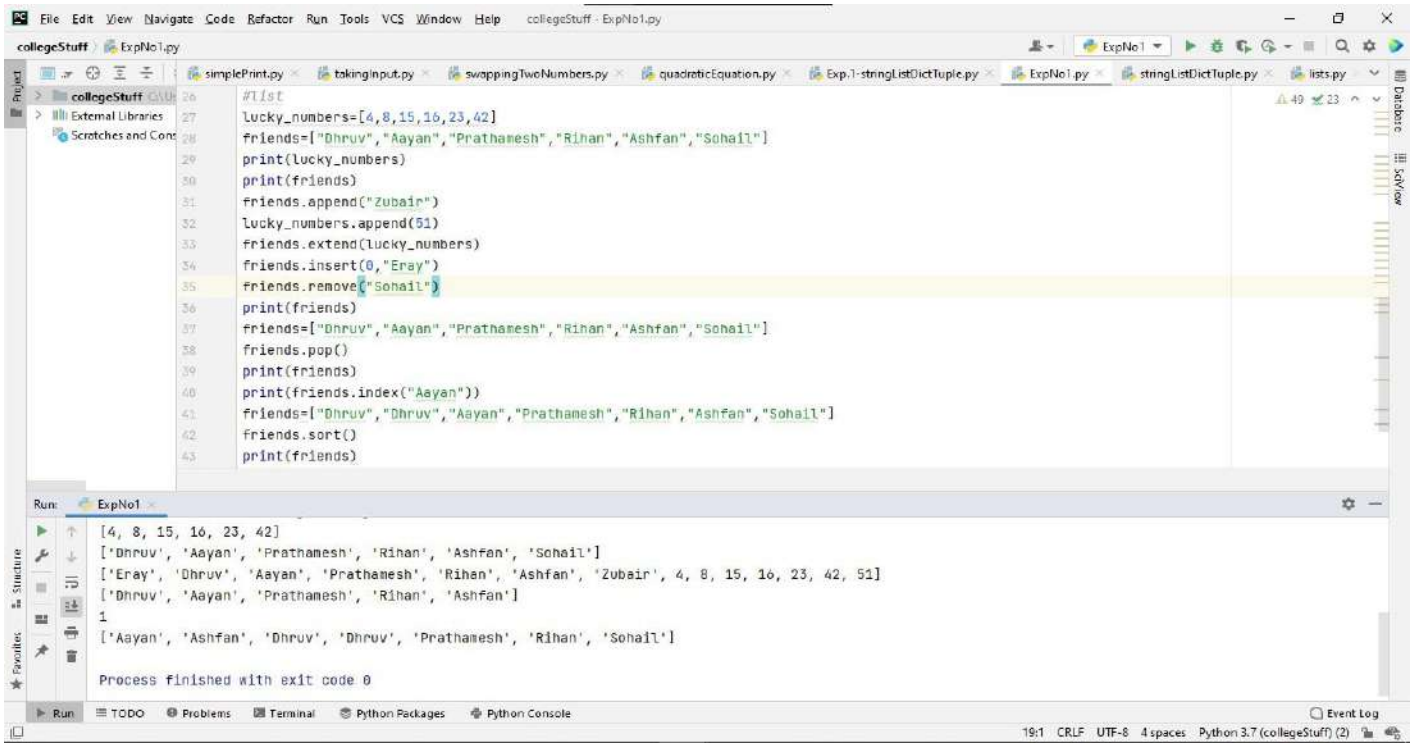


c)Solve the quadratic equation ax**2 + bx + c = 0

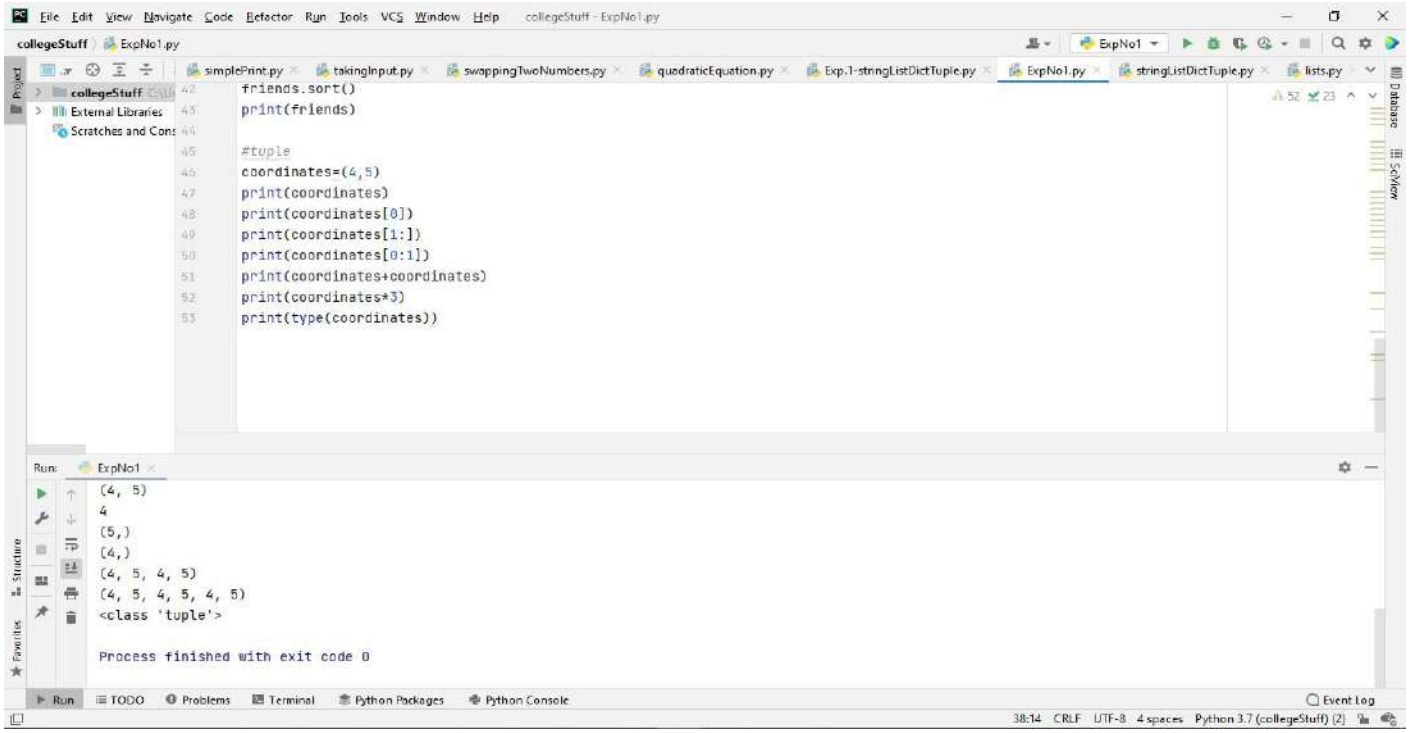


d)Perform (at least 5) operations on each.

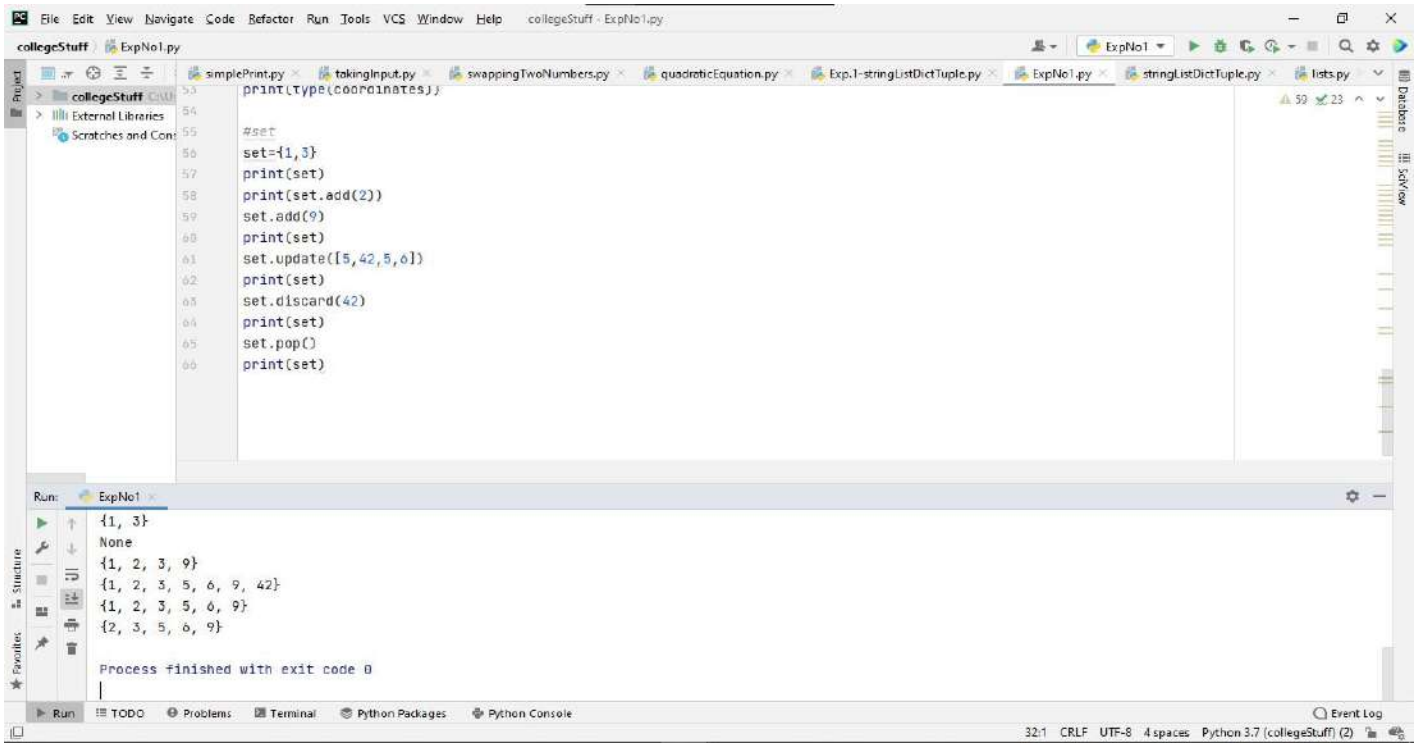
i)list



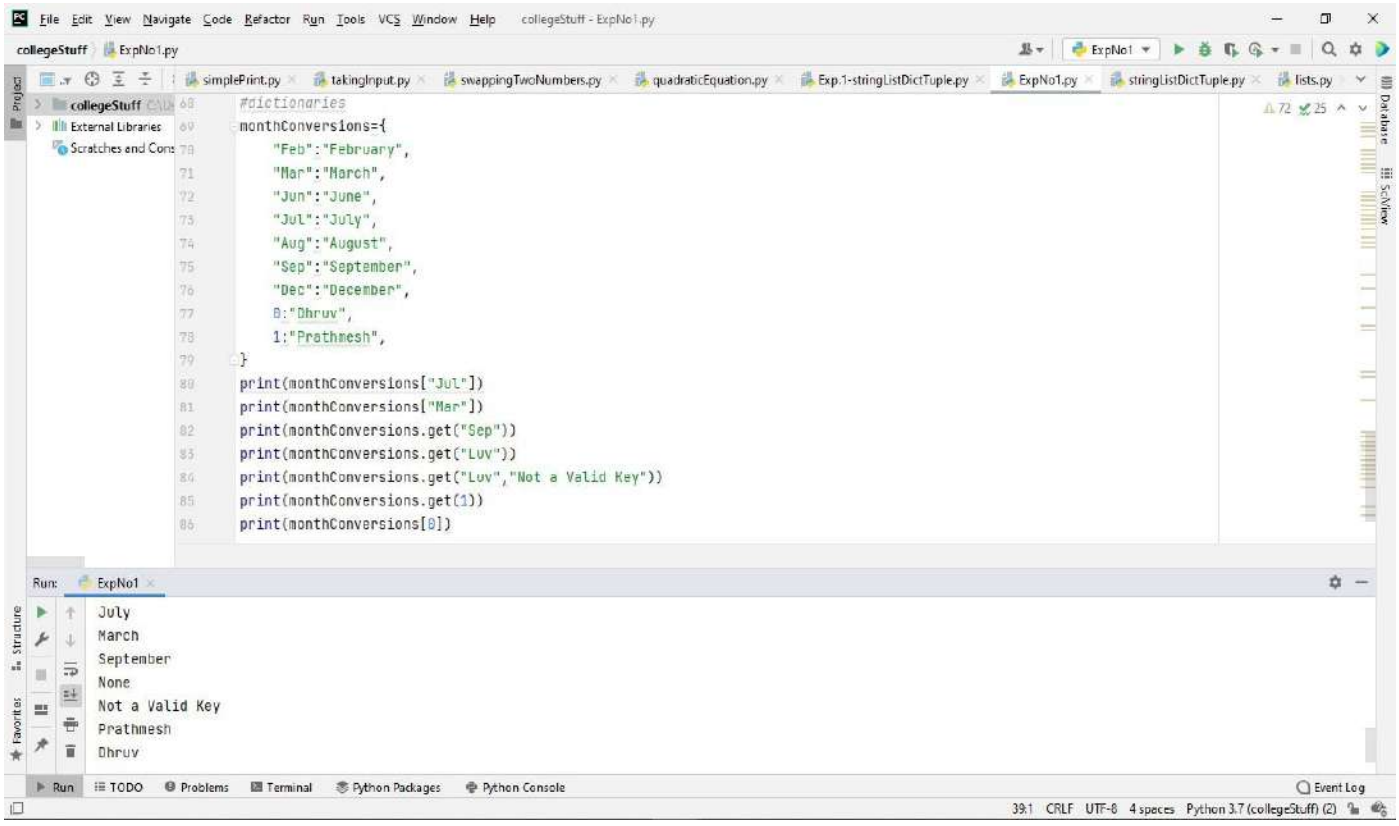
ii)tuple



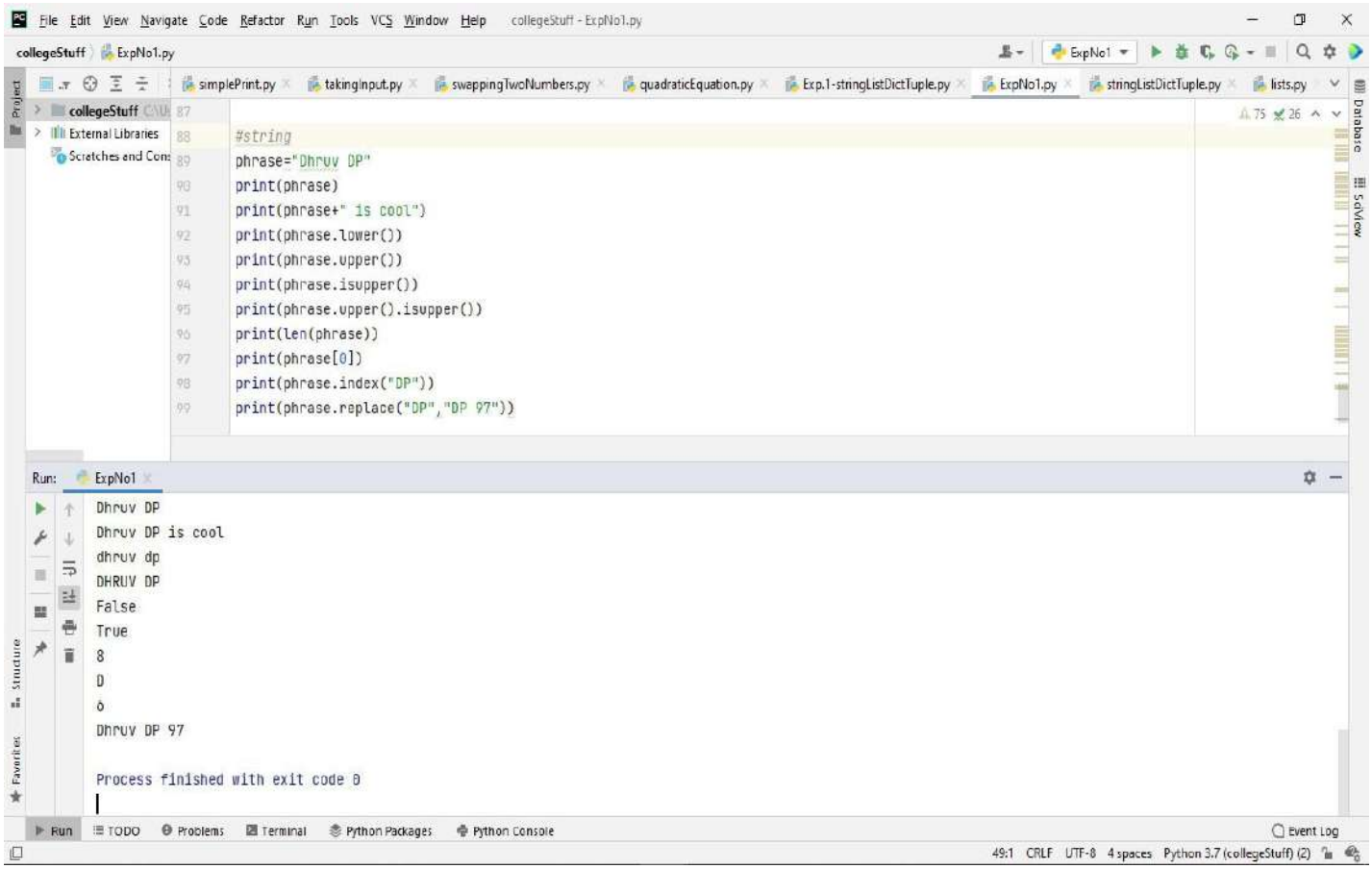
iii)set



iv)dictionaries



v)string


File Edit View Navigate Code Refactor Run Tools VCS Window Help collegeStuff - ExpNo1.py
collegeStuff ExpNo1.py
ExpNo1
simplePrint.py takingInput.py swappingTwoNumbers.py quadraticEquation.py Exp.1-stringListDictTuple.py ExpNo1.py stringListDictTuple.py lists.py
collegeStuff
External Libraries
Scratches and Cons
#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
6
Dhruv DP 97
Process finished with exit code 0
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 on the compiler (thonny).

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

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

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

EXP NO : 2

Aim : program using- Different 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 (condition2):
```

- for loop - A for loop is used for iterating over a sequence. We can execute a set of statements.

eg:-

```
fruits = ["apple", "banana", "cherry"]
for x in fruits
    print(x)
```

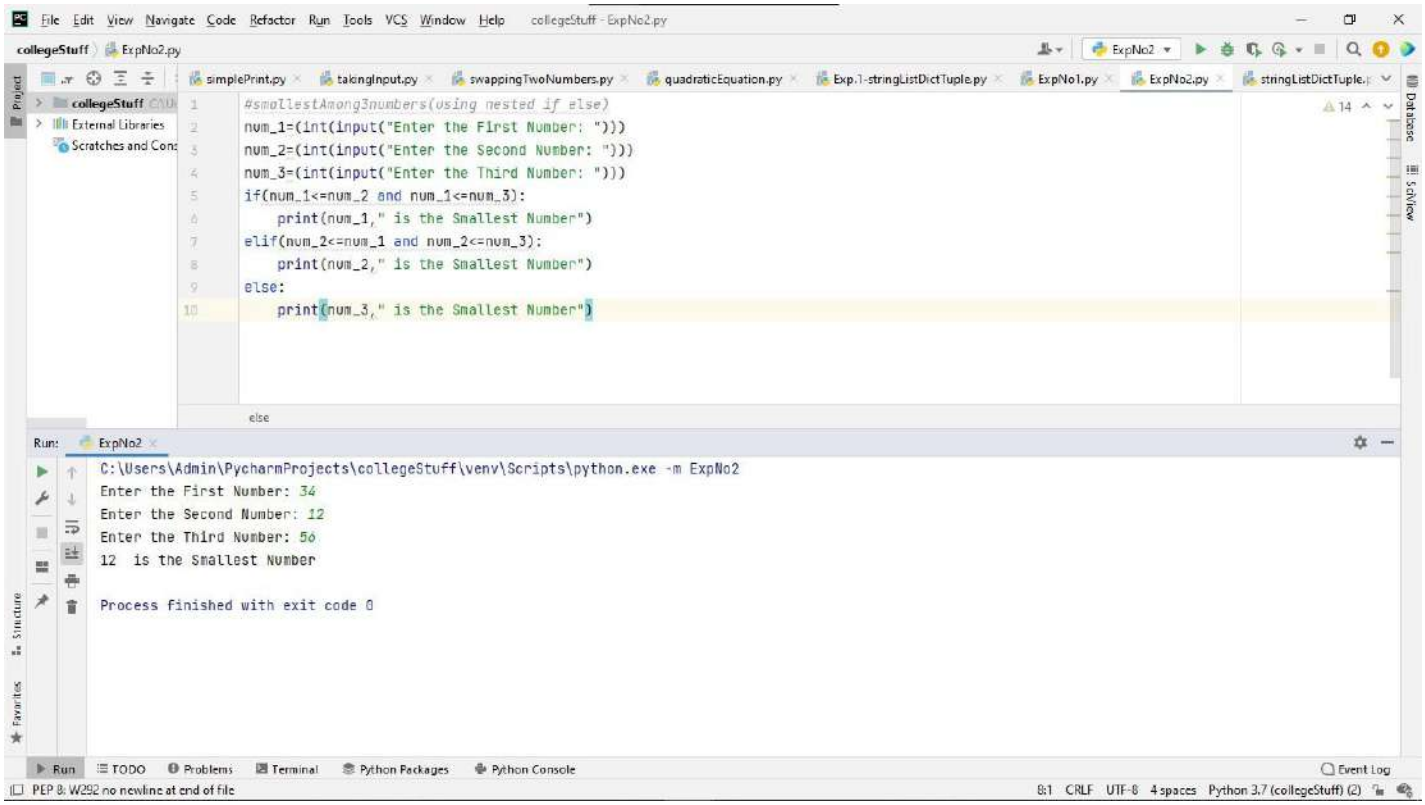
- while loop - With the while we can execute a set of statements as long as a condition is true.

eg:-

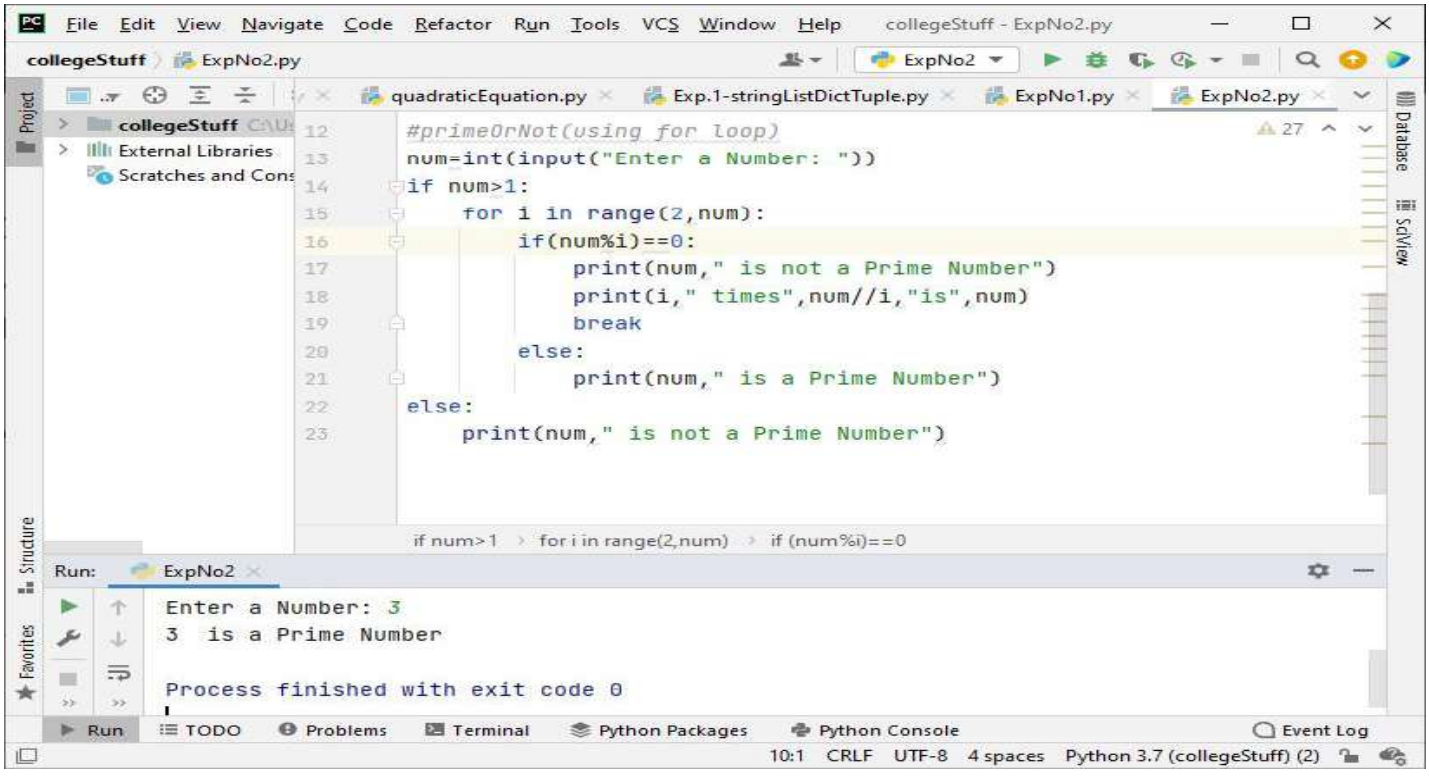
```
i = 1
while i<6:
    print(i)
    i += 1
```

Remember to increment i, or else the loop will continue forever.

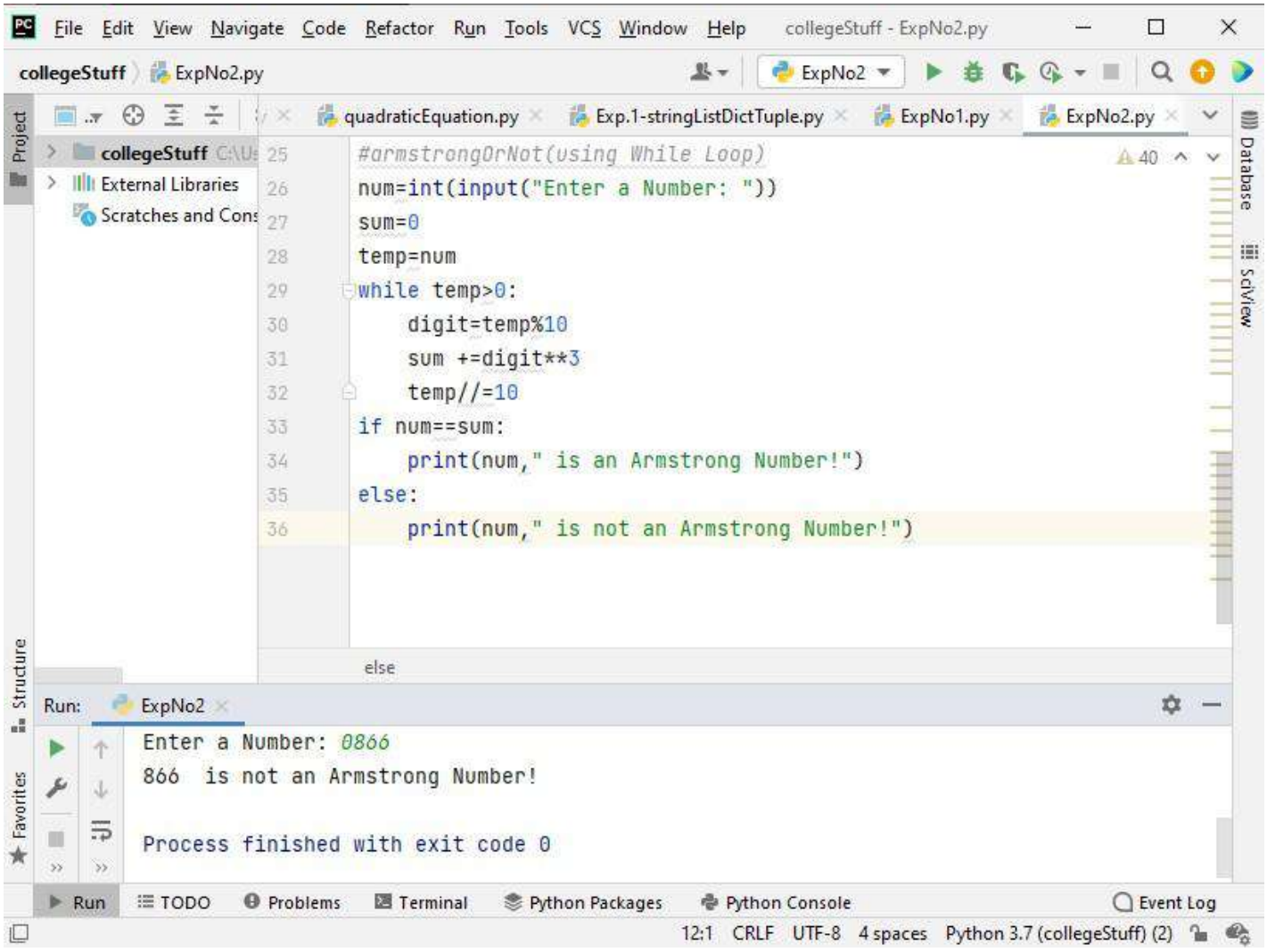
a) Smallest of 3 numbers (using nested if else)



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



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



Conclusion: we learned the different control statements and succesfully 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

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

EXP NO : 3

Aim : program using concept 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 - 1) using normal functn
2) using lambda functn

ii - a) write a program to print Employee information (usr object variable 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()
- Uses-defined functn : users create.
- Anonymous functn : also called as lambda functn

Classes :- A class is like an object constructor, or a "blue print" 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

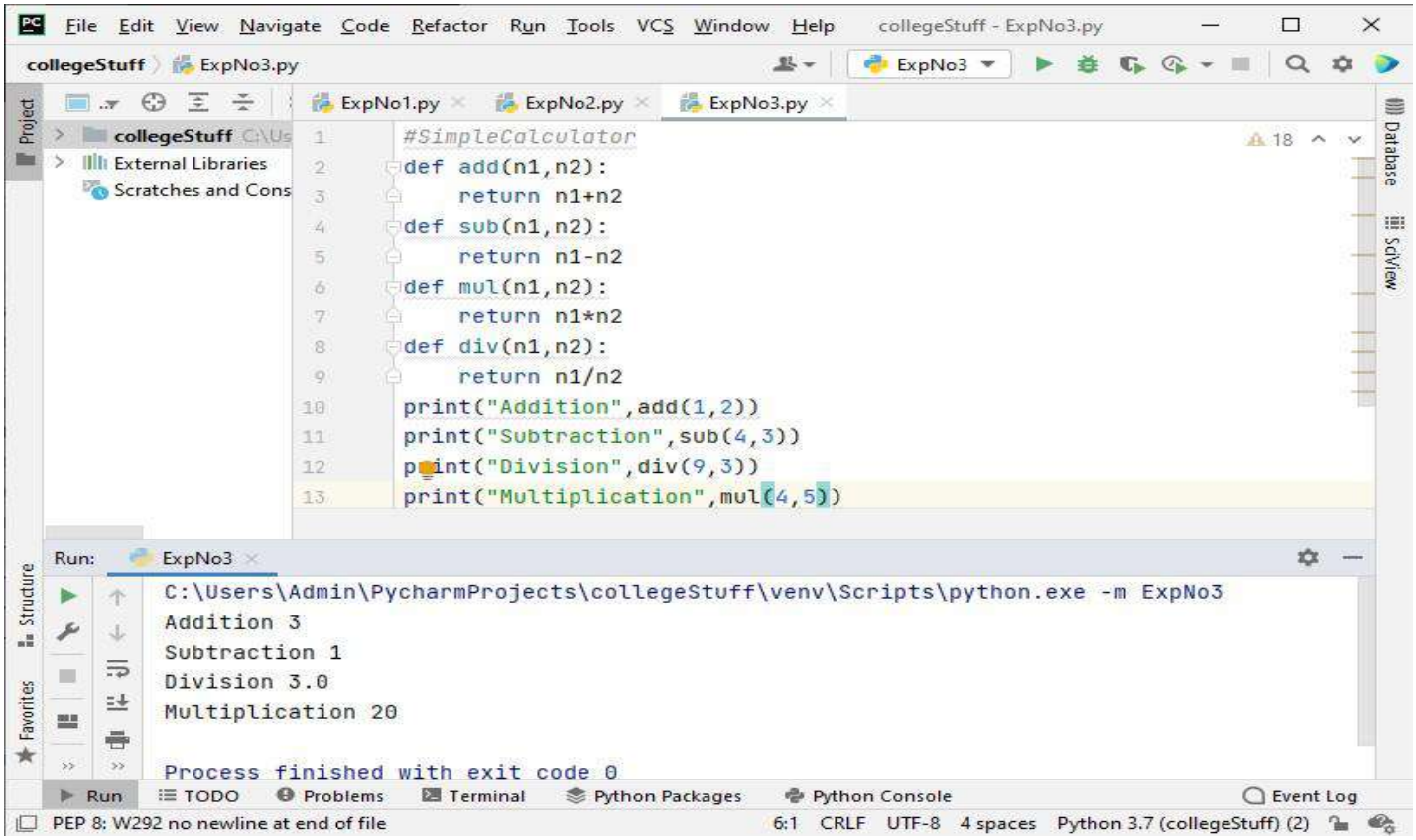
eg :-
```
p1 = Myclass()
print(p1.x)
```

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

Self parameter

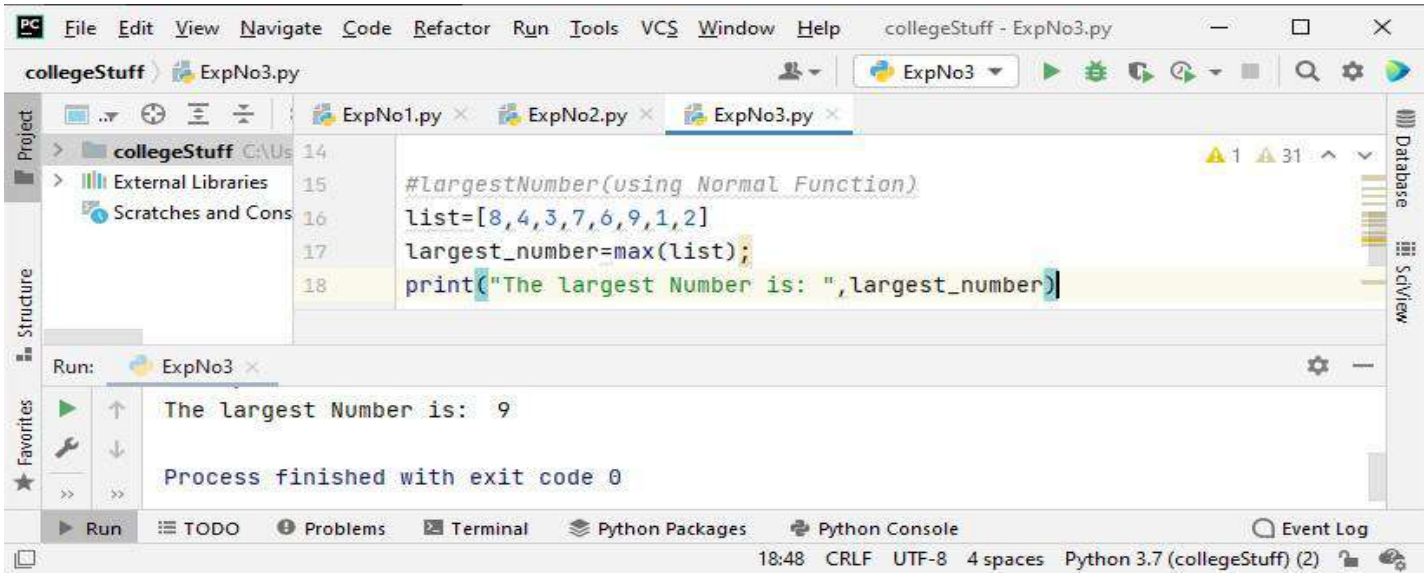
Program -

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

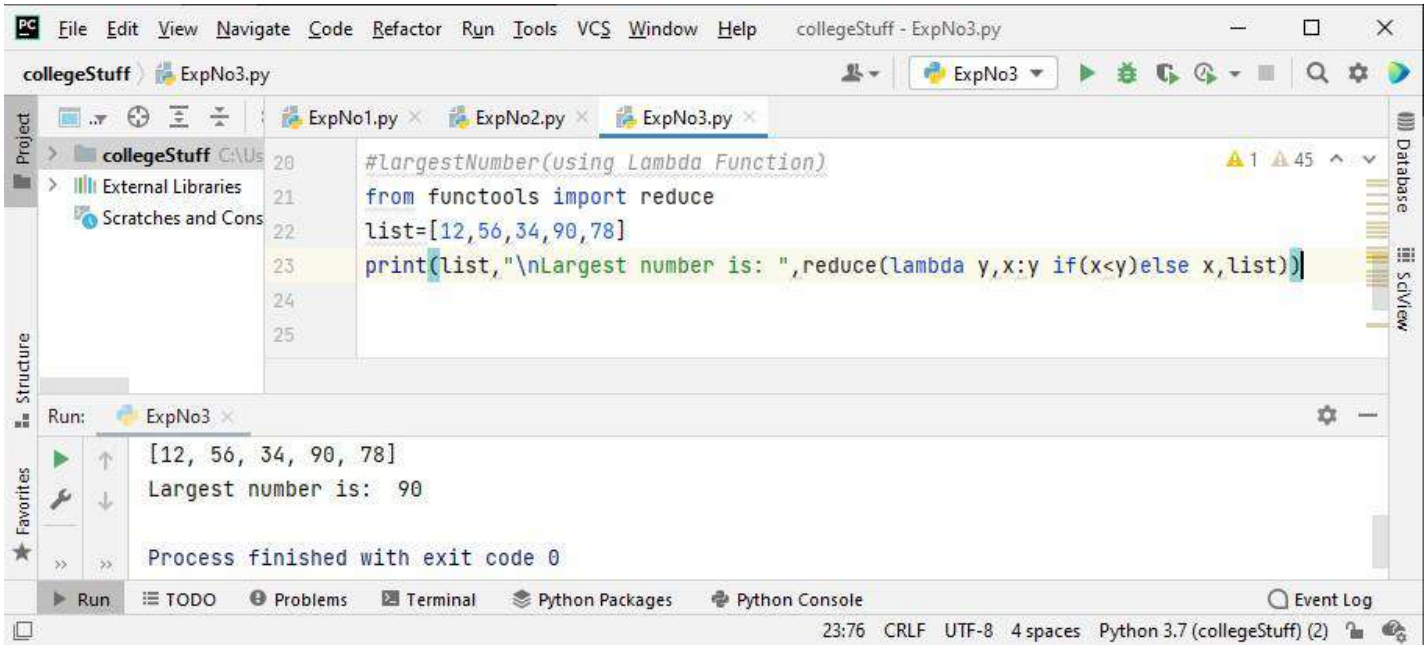


b) To find largest number from the list

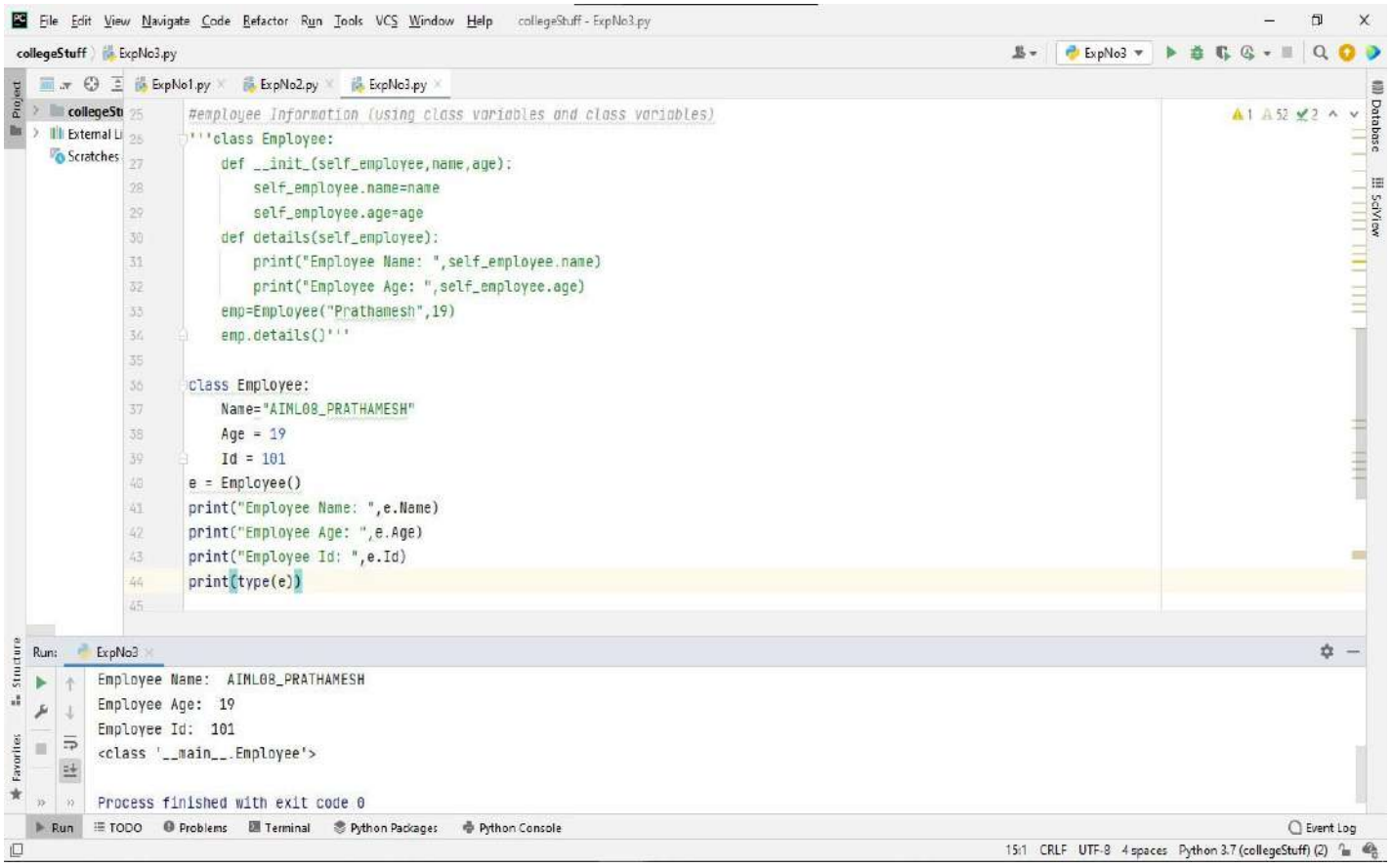
i) using normal function



ii) using Lambda function



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



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

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

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

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

EXP NO: 4

Aim: Program for file handeling and directories...

Theory: python too supports file handeling and allows users to handle files i.e., to read and write files, along with many other file handeling options, to operate on files.

a) A program that count 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 line 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, wrote and append

r: read file (existing), wrote: wrote 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 renaming use rename()

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

Text file-

test1.txt

Hey there

i'm prathamesh

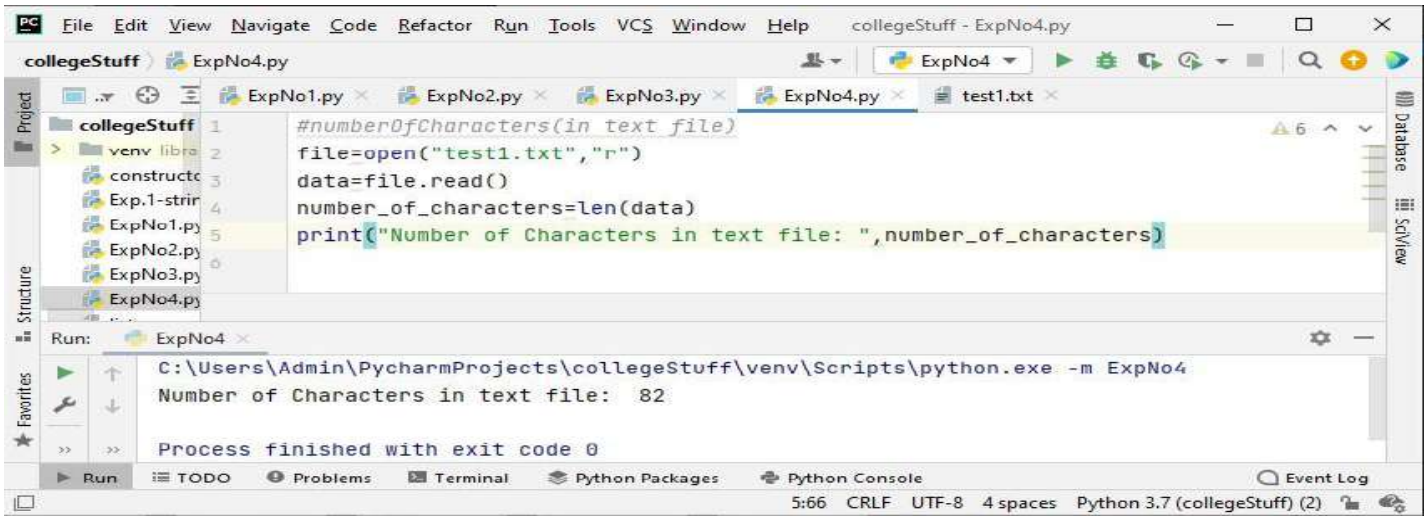
from cse-(ai&ml) branch

from ltcoe

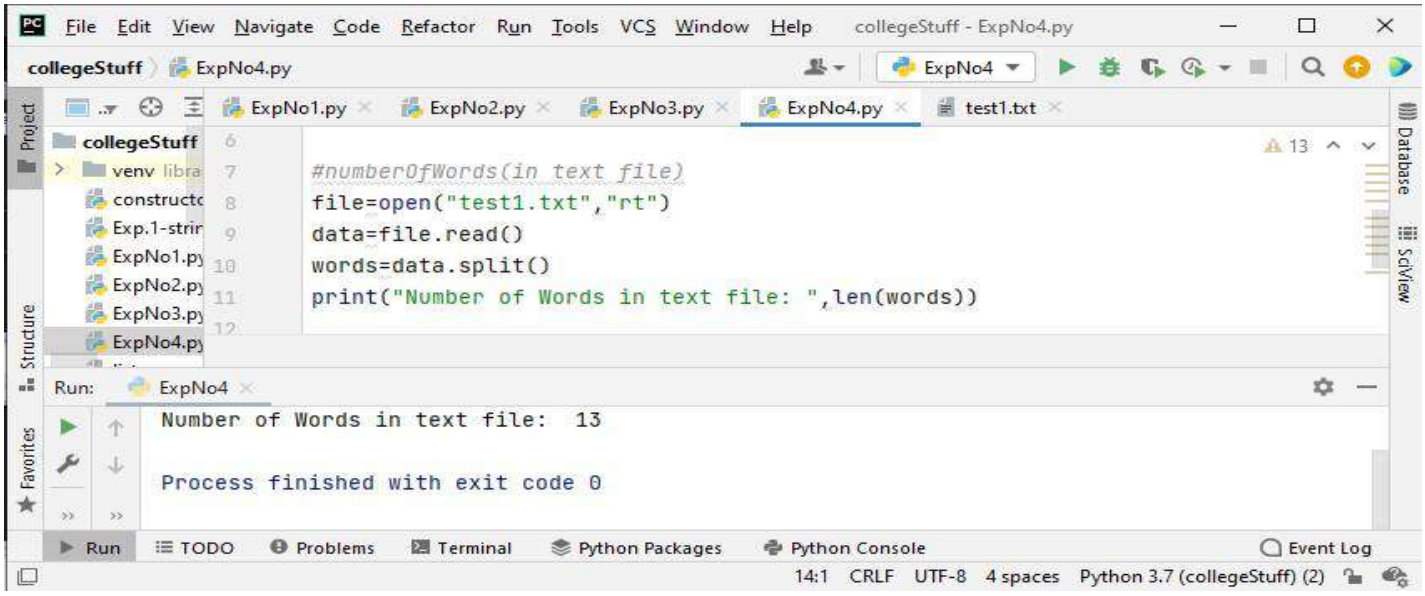
located in navi mumbai

Program -

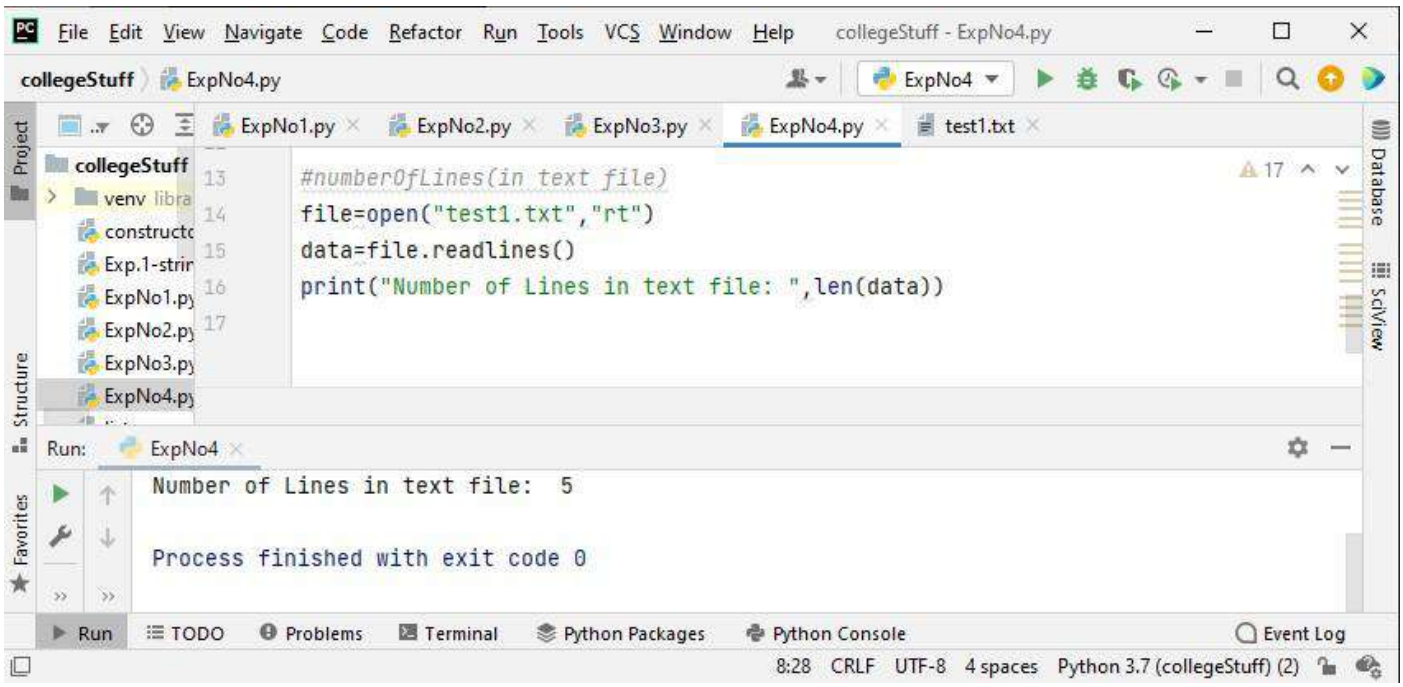
a) A program that count the number of characters in a text file



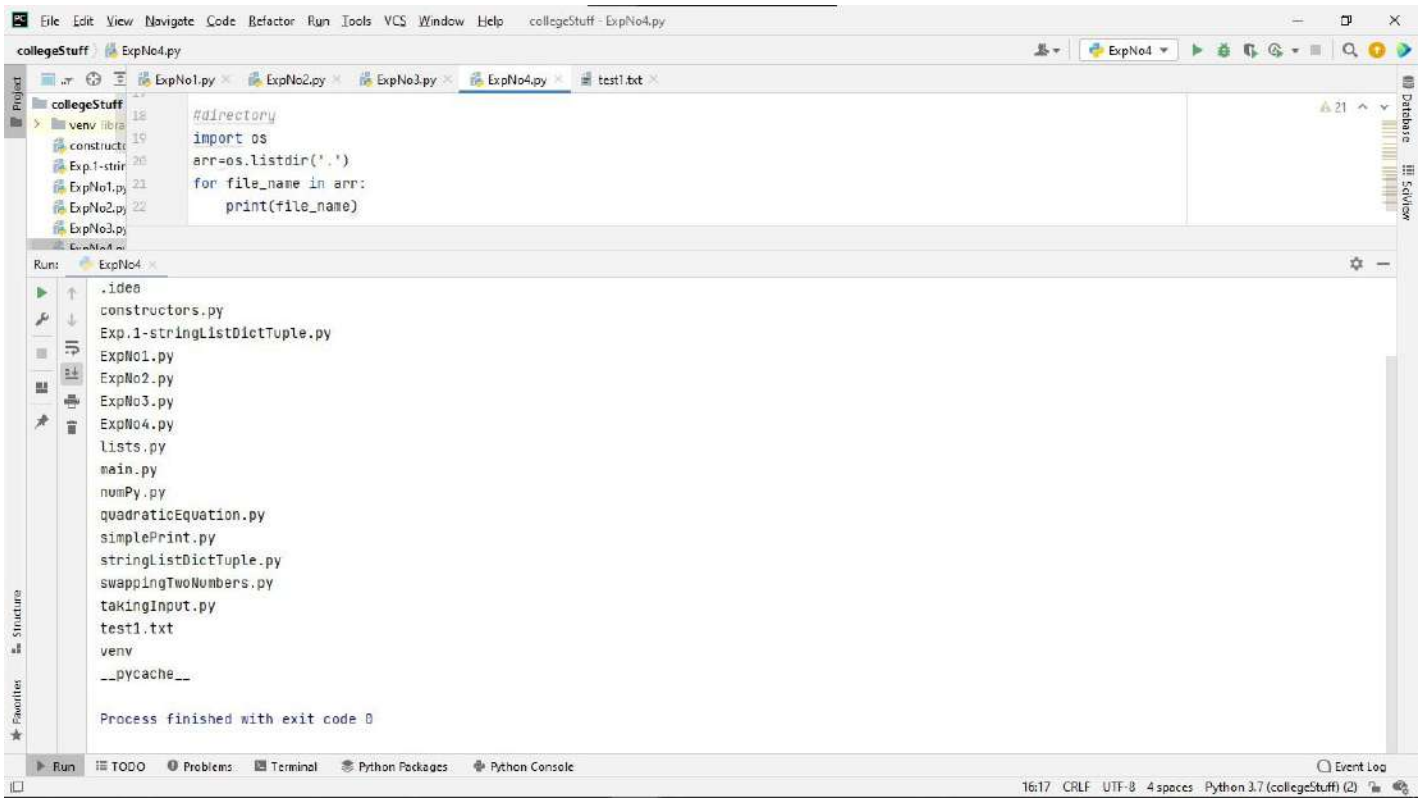
b) To count number of words in a text file



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



d) To display file available in current directory



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

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

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 begning of the queue

Operation on stack and queue => 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 - > Simple linked list
> Doubly linked list - one is previous pointer and other is next.
> 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]," poped out from the stack")
            poped=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
```

```
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  poped 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 embty")
            return
        self.front+=1
    def display(self):
        if self.front==-1:
            print("empyt")
            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()
```

```
    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

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

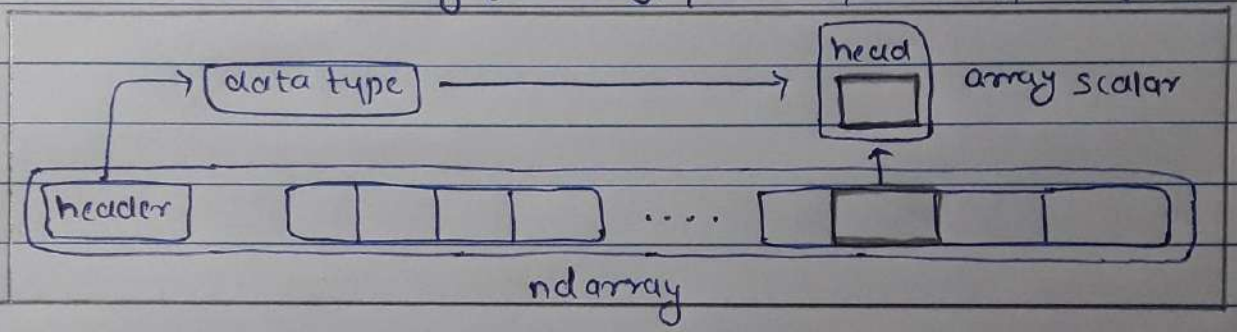
EXP NO 6

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)



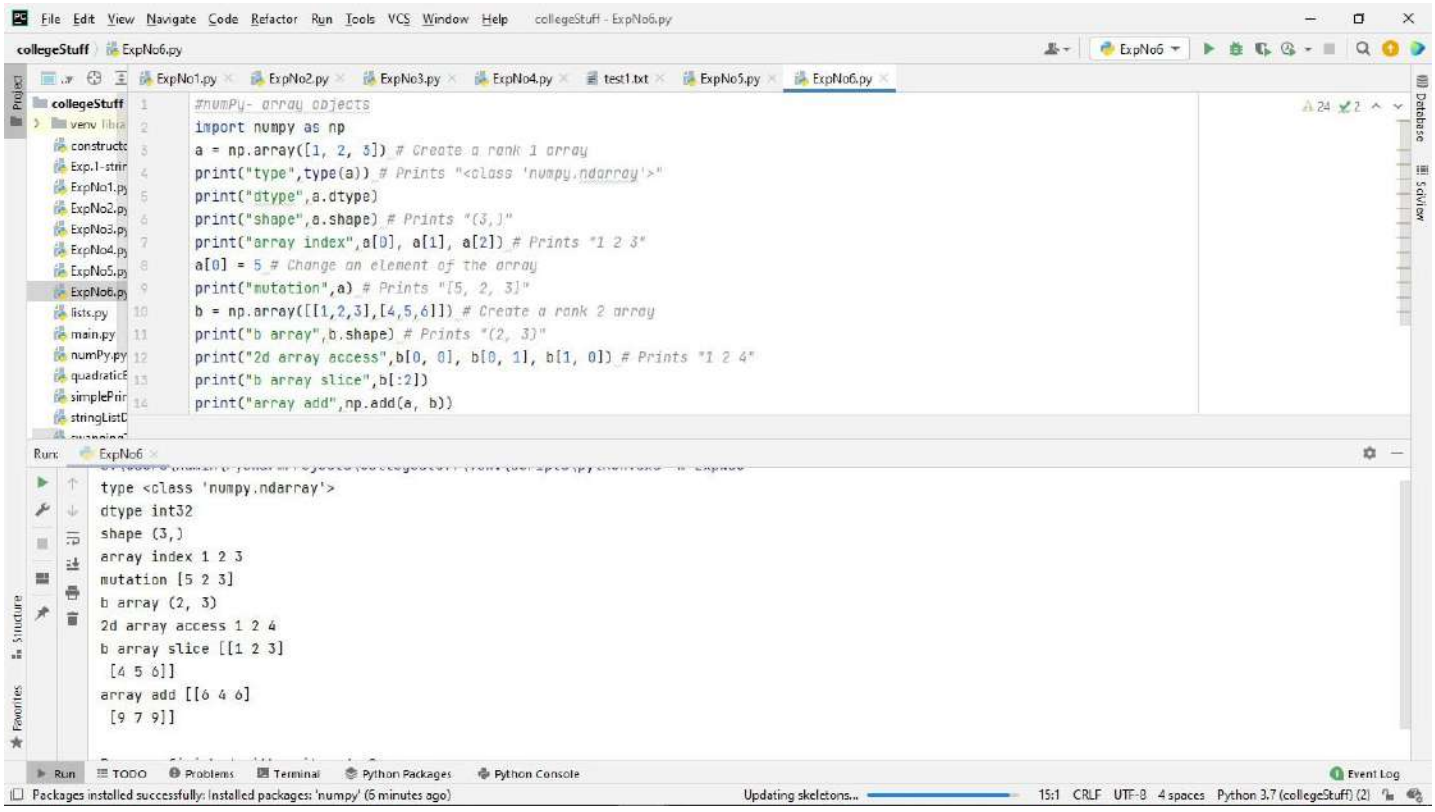
Numpy methods -

numpy.ndarray => class numpy.ndarray(shape, dtype=float, buffer=None, offset=0, strides=None, order=None)

* Array creation, Array Indexing, Basic operation, sorting array.

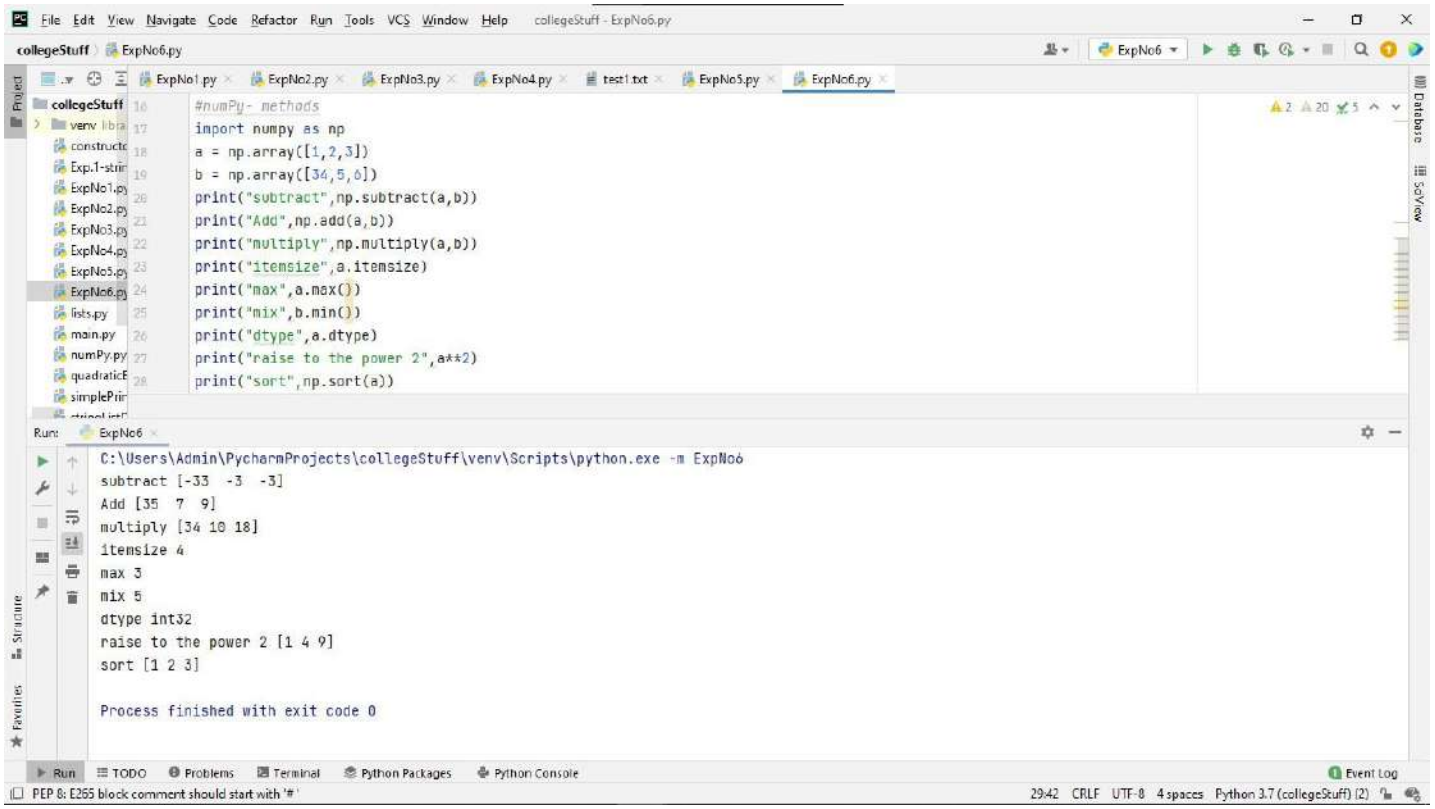
- np.array	np.copy(arr)	np.delete
arr.size	dtype arr.sort()	arr np.insert
arr.datatype	arr.reshape(3,4)	np.eye
arr.tolist()	np.append	

Program with Output -



```
#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))
```

```
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]]
```



```
#numPy- methods
import numpy as np
a = np.array([1,2,3])
b = np.array([34,5,6])
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("mix",b.min())
print("dtype",a.dtype)
print("raise to the power 2",a**2)
print("sort",np.sort(a))
```

```
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
mix 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

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

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 nd array, dictionaries and scalar values.

- random.randint() } 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.

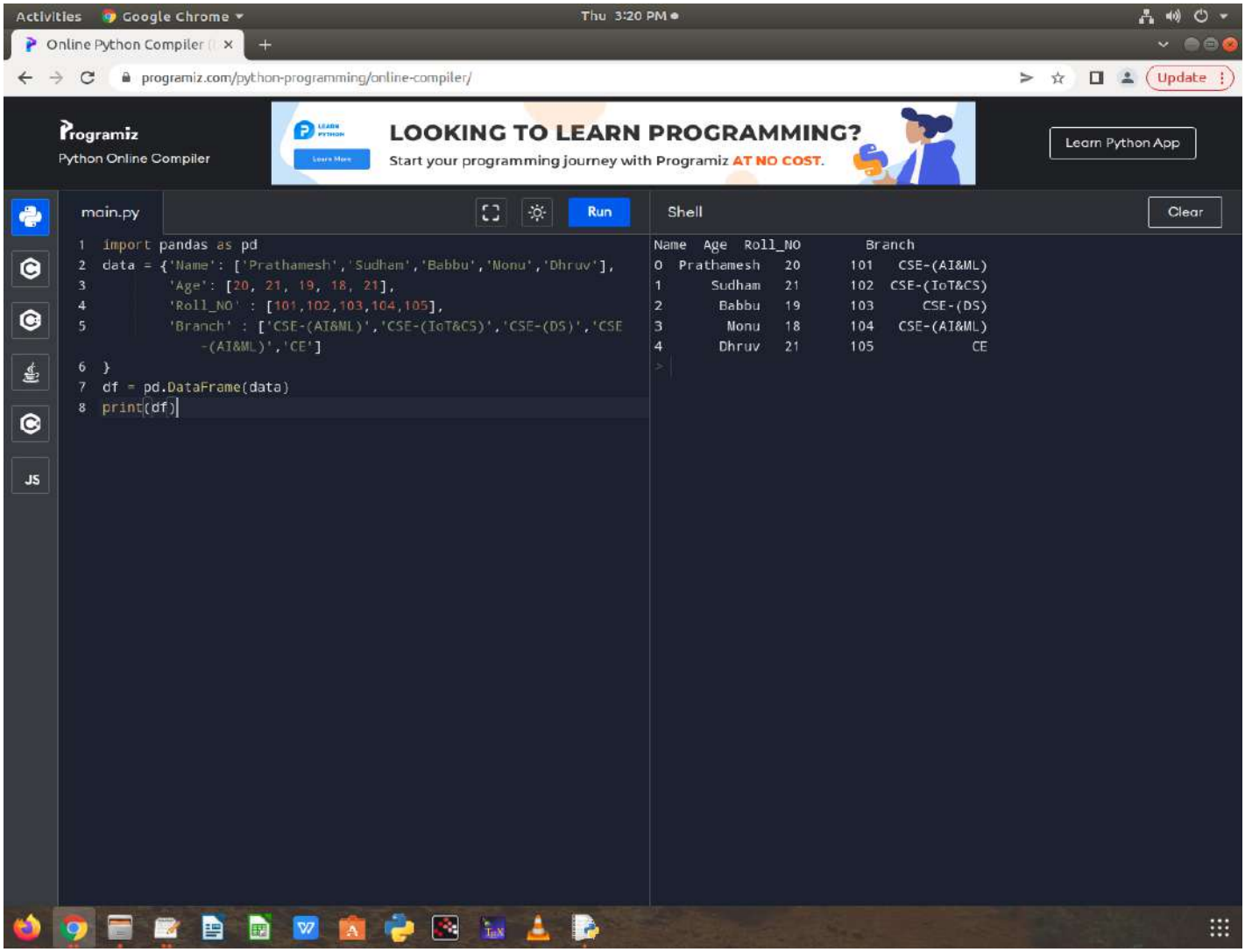
The data parameter similar to series can accept a broad range of data types such as series, a dictionaries of series, structured arrays and Numpy arrays.

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

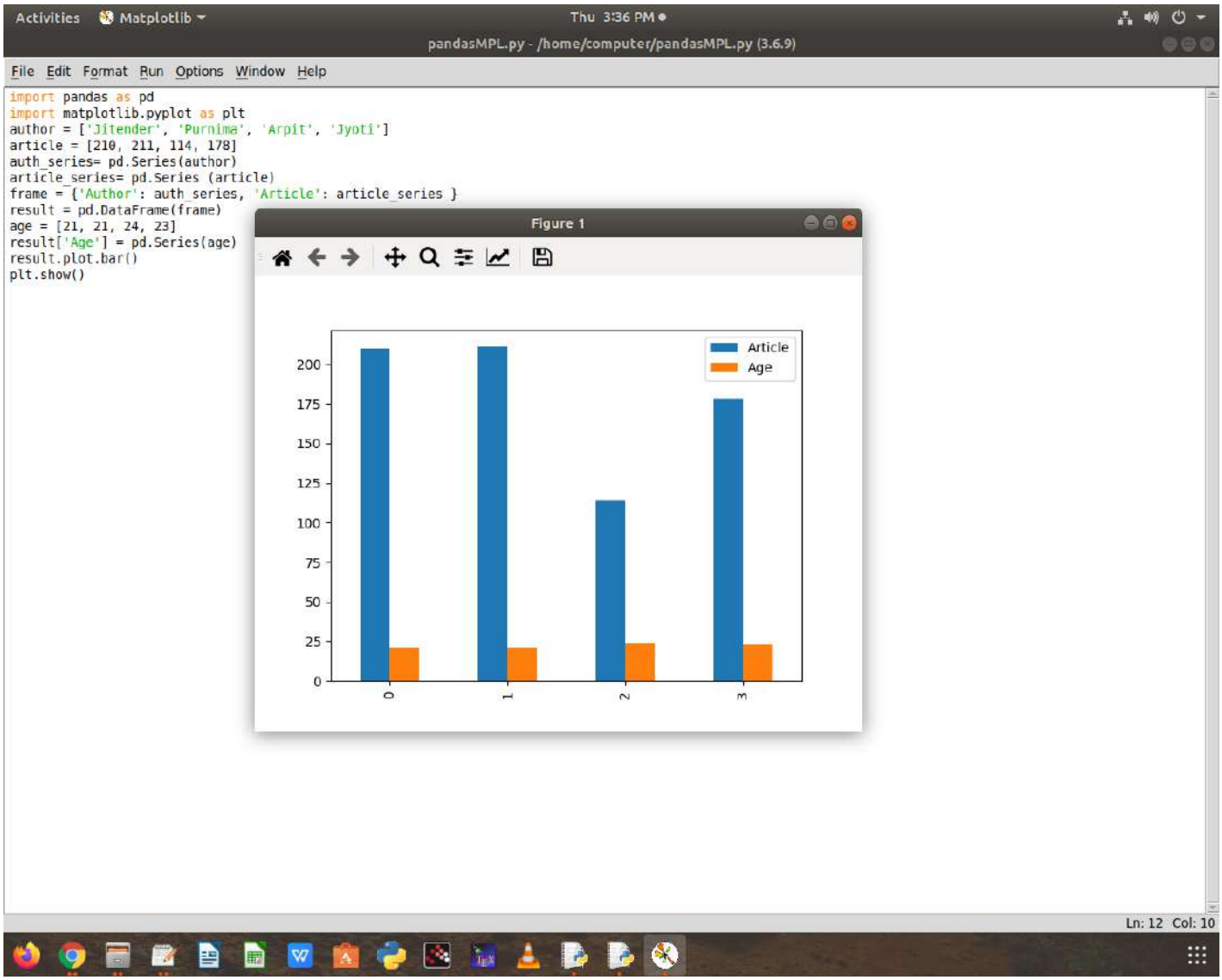
Program with Output -

DataSeries -

DataFrame -



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
Article
Age
200
175
150
125
100
75
50
25
0
0
1
2
3
Ln: 12 Col: 10

Conclusion :- Successfully demonstrated the data series and data frames using panday package in python...

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

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

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

Exp No 08

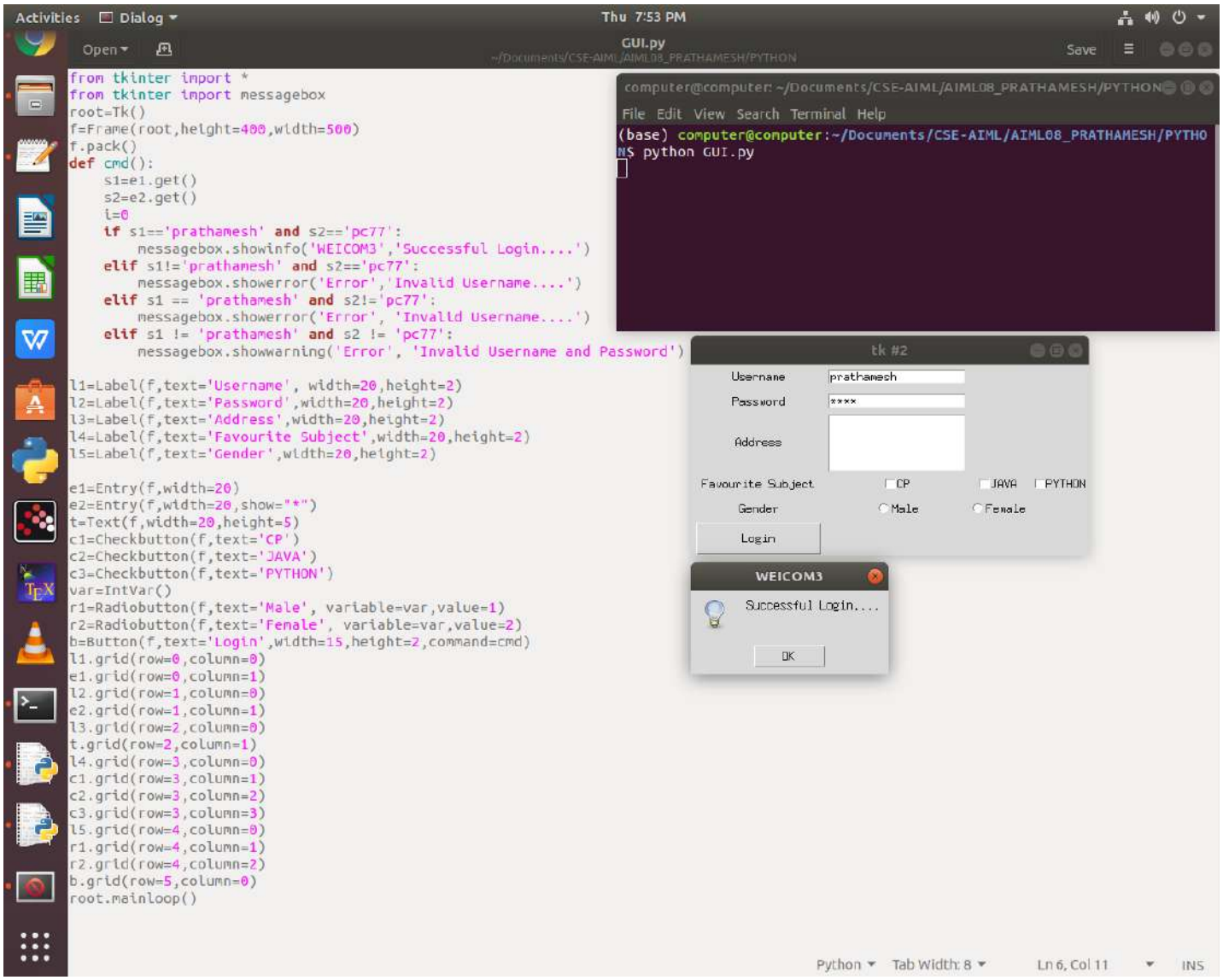
Aim: Creating GUI with python containing widgets such as labels, textbox, radio, checkboxes and custom dialog boxes.

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 program. There are a number of widgets which you can use in your tkinter program. Some of them are:

1) Button: To add a button in any program, this widget is used.
2) Check button: 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 grouping and organizing widget.
5) Radio button: It is used to offer multi-choice option to the user. It offers several options to the users and the user has to choose one option.

Program with output -



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

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

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 logging applications. The most common use for mysql however is for the purpose of a web database.

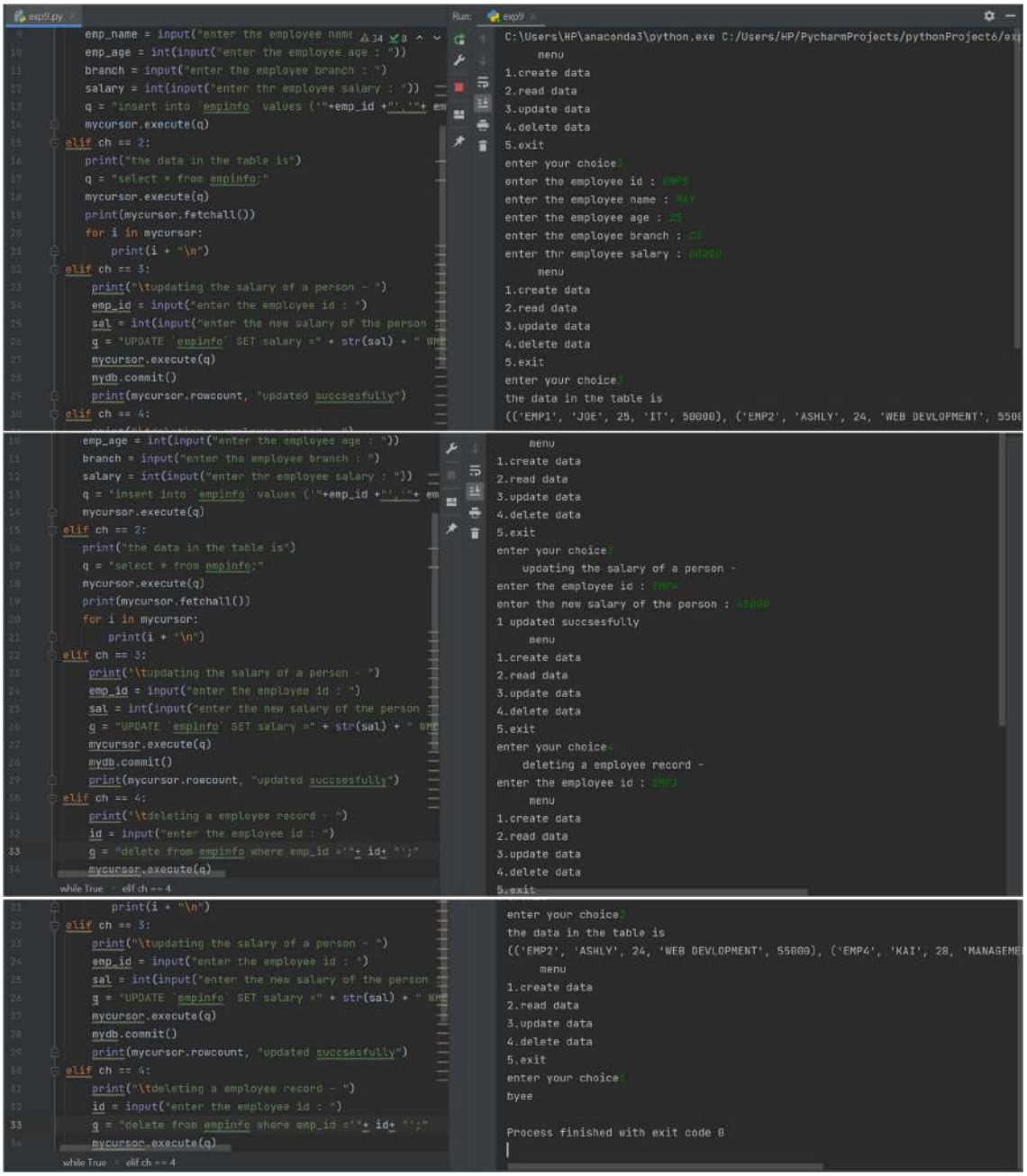
Connecting MySQL database in python:
We can connect MySQL database with a python program which enables us to manipulate the SQL database using python commands.

- 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 thr 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 succsesfully")
 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 –


exp9.py
Run: exp9
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 thr employee salary : "))
q = "insert into `empinfo` values ('"+emp_id +"','"+ em
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) + " WH
mycursor.execute(q)
mydb.commit()
print(mycursor.rowcount, "updated succsesfully")
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)
while True
elif ch == 4
C:\Users\HP\anaconda3\python.exe C:/Users/HP/PycharmProjects/pythonProject6/ex
menu
1.create data
2.read data
3.update data
4.delete data
5.exit
enter your choice:
enter the employee id : EMP4
enter the employee name : KAI
enter the employee age : 28
enter the employee branch : CS
enter thr employee salary : 60000
menu
1.create data
2.read data
3.update data
4.delete data
5.exit
enter your choice:2
the data in the table is
(('EMP1', 'JOE', 25, 'IT', 50000), ('EMP2', 'ASHLY', 24, 'WEB DEVLOPMENT', 550
menu
1.create data
2.read data
3.update data
4.delete data
5.exit
enter your choice:3
updating the salary of a person -
enter the employee id : EMP4
enter the new salary of the person : 45000
1 updated succsesfully
menu
1.create data
2.read data
3.update data
4.delete data
5.exit
enter your choice:4
deleting a employee record -
enter the employee id : EMP1
menu
1.create data
2.read data
3.update data
4.delete data
5.exit
enter your choice:2
the data in the table is
(('EMP2', 'ASHLY', 24, 'WEB DEVLOPMENT', 55000), ('EMP4', 'KAI', 28, 'MANAGEME
menu
1.create data
2.read data
3.update data
4.delete data
5.exit
enter your choice:5
byee
Process finished with exit code 0

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