Practical-1

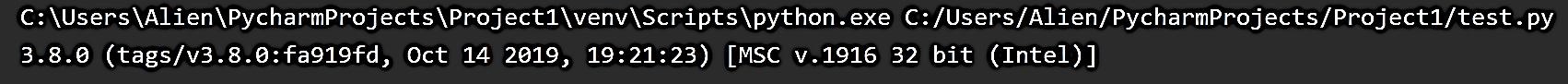
***Aim: Programs to understand the basic Data type and control structures of python.***

***A. Write a python programme to print version of system.***

**Program:**

**import sys**

**print(sys.version)**

 **Output:**

***B. Write a program to print prime numbers from 1 to 1000.***

**Program:**

**for num in range(1, 1000):**

**if num > 1:**

**for n in range(2, num):**

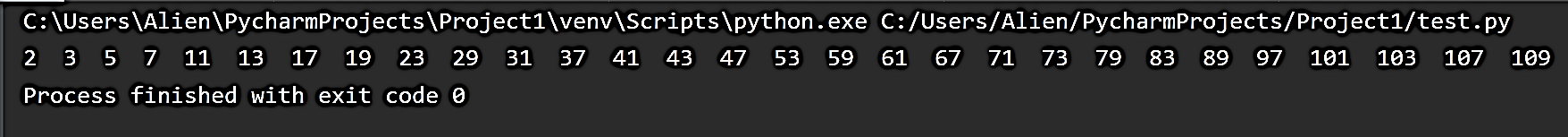
**if num % n == 0:**

**break**

**else:**

**print(num,' ',end='')**

**Output:**



***C. Write a program which will take input from user for the value of Celsius and convert it to the Fahrenheit value. Print both the value in output.***

**Program:**

**while 1:**

**print('\n1.Celsius To Fahrenheit \n2.Fahrenheit To Celsius')**

**ch = int(input('\nSelect\n'))**

**if ch == 1:**

**cel = float(input('Enter Temperature In Celsius\n'))**

**fah = (cel \* (9 / 5)) + 32**

**print('\nCelsius= ', cel, '\nFahrenheit= ', fah)**

**elif ch == 2:**

**fah = float(input('Enter Temperature In Fahrenheit'))**

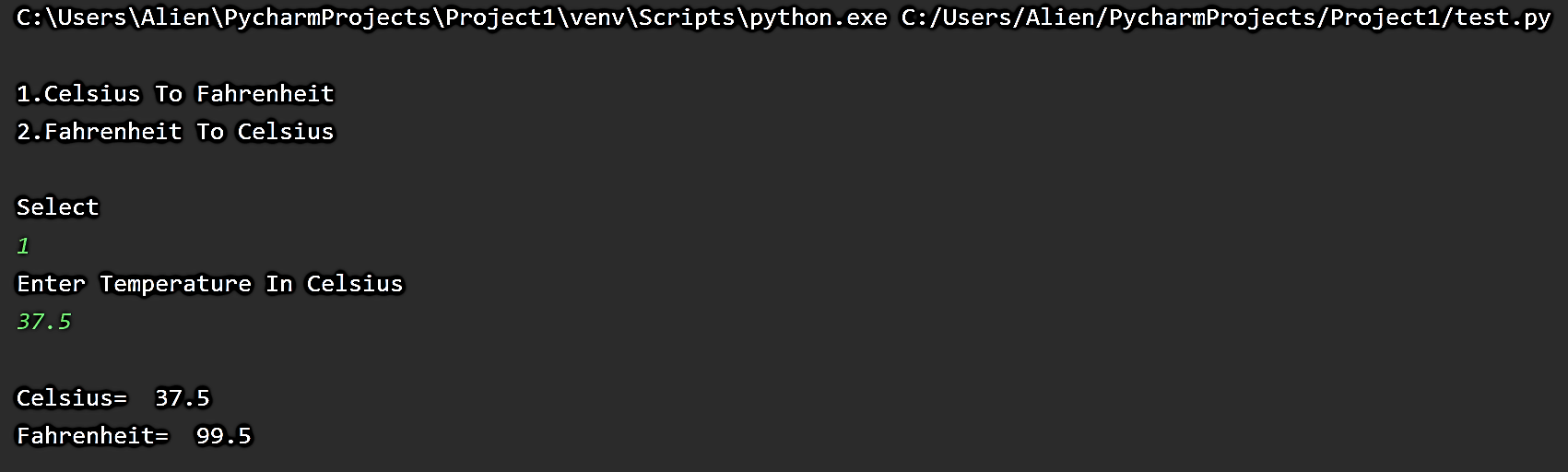
**cel = (fah - 32) \* (5 / 9)**

**print('\nFahrenheit= ', fah, '\nCelsius= ', cel)**

**else:**

**print('\nEnter Correct Choice')**

**Output:**



***D. Write a program which will take three number as an input form user and accordingly print number of # in new line. For example: if user enter 3, 5, 2 then output is as follows:***

**###**

**#####**

**##**

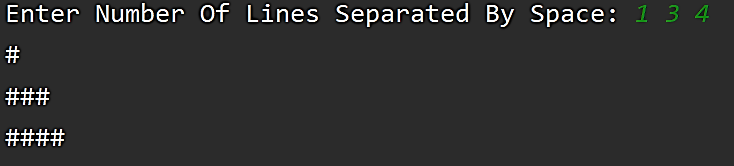
**Program:**

**userList = input("Enter Number Of Lines Separated By Space: ").split()**

**for i in userList:**

**print('#'\*int(i))**

**Output:**



Practical-2

***Aim: Programs to learn concept of functions scoping, recursion and list mutability.***

***A. Write the function to print largest odd number from 10 entered number.***

**Program:**

**n = int(input("Enter Number of Element "))**

**b=[]**

**print("Enter ",n," Element")**

**for i in range(0,n):**

**a=int(input())**

**b.append(a)**

**l=0**

**for num in b:**

**if num % 2 != 0 and num>l:**

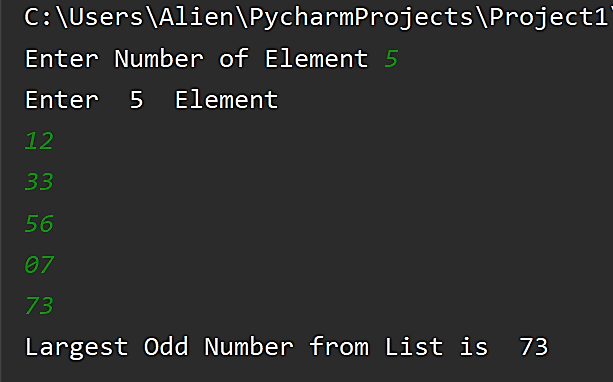
**l=num**

**if(l==0):**

**print("No Odd Number Found")**

**else:**

**print("Largest Odd Number from List is ",l)**

**Output:**

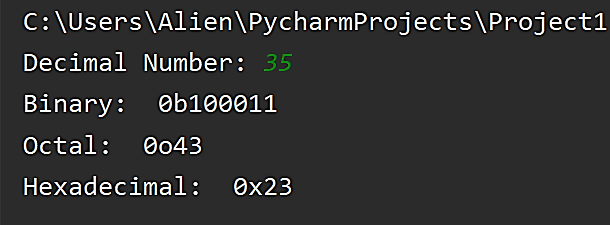
***B. Write Python Program to Convert Decimal to Binary, Octal and Hexadecimal using inbuilt function.***

**Program:**

**num = int(input('Decimal Number: '))**

**print('Binary: ', bin(num), '\nOctal: ', oct(num), '\nHexadecimal: ', hex(num))**

**Output:**



***C. Print sum of all numbers within given range using function. The rage of numbers is defined by argument to function(First and second argument become start and end rage respectively).***

**Program:**

**print("Enter Lower and Upper Bound")**

**l=int(input("Lower Bound = "))**

**u=int(input("Upper Bound = "))**

**def sum(l,u):**

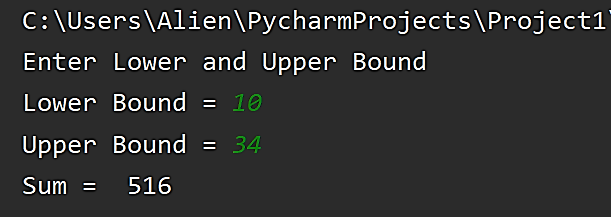
**add=0**

**for i in range(l,u):**

**add=i+add**

**print("Sum = ",add)**

**sum(l,u)**

**Output:**

***D. Write a Python program to check whether the given no is Armstrong or not using user defined function***

**Program:**

**num = int(input("Enter A Number: "))**

**sum = 0**

**temp = num**

**while temp > 0:**

**digit = temp % 10**

**sum += digit \*\* 3**

**temp //= 10**

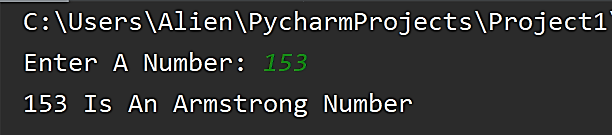
**if num == sum:**

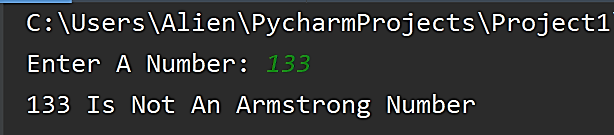
**print(num, "Is An Armstrong Number")**

**else:**

**print(num, "Is Not An Armstrong Number")**

**Output:**





***E. Reverse a given string using recursion***

**Program:**

**def reverse(s):**

**if len(s) == 0:**

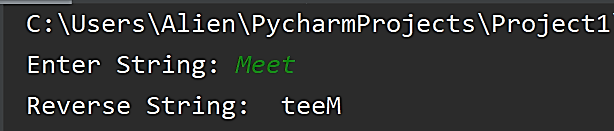
**return s**

**else:**

**return reverse(s[1:]) + s[0]**

**str1 = input('Enter String: ')**

**print('Reverse String: ', reverse(str1))**

**Output:**

***F. Write a Python program to find reverse of given number using user defined function***

**Program:**

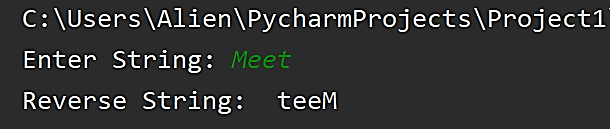
**def reverse(temp):**

**return temp[::-1]**

**str1 = input('Enter String: ')**

**print('Reverse String: ',reverse(str1))**

**Output:**



Practical-3

***A. Write a python program to find the x^y of using recursion..***

**Program:**

**def power(x, y):**

**if y == 0:**

**return 1**

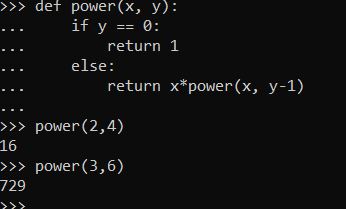
**else:**

**return x\*power(x, y-1)**

**power(2,3)**

**power(3,4)**

**Output:**

****

***B. Write a python program to create a function to find the second minimum number from a list of 10 numbers.***

**Program:**

**def second\_smallest(numbers):**

**m1, m2 = float('inf'), float('inf')**

**for x in numbers:**

**if x <= m1:**

**m1, m2 = x, m1**

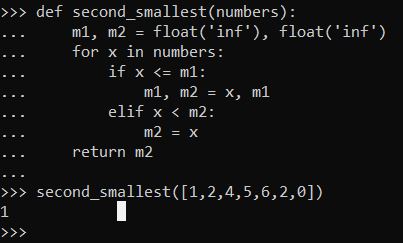
**elif x < m2:**

**m2 = x**

**return m2**

**second\_smallest([1, 2,3,4])**

**Output:**

****

***C. Write a python program to create a function that returns only odd number from an original list given to it as an argument.***

**Program:**

**def remove\_even(numbers) :**

**new\_list = []**

**for i in numbers:**

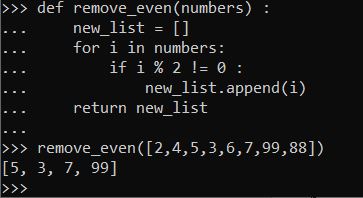
**if i % 2 != 0 :**

**new\_list.append(i)**

**return new\_list**

**remove\_even([2,4,5,6,77,88,99])**

**Output:**

****

***D.*** ***Write a python program to reverse a string using slicing.***

**Program:**

**Name=input(‘enter your name’)**

**Print(“reverse of your name is :”,name[::-1] )**

**Output:**

**Enter your name:nirmal**

**Reverse of your name: lamrin**

***E. Write a python program to show the use of a global variable x inside the local scope of a***

***function (use Global Keyword ).***

**Program:**

**globvar = 0**

**def set\_globvar\_to\_one():**

**global globvar # Needed to modify global copy of globvar**

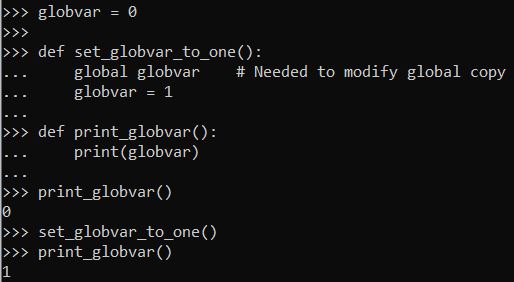
**globvar = 1**

**def print\_globvar():**

**print(globvar)**

**set\_globvar\_to\_one()**

**print\_globvar()**

**Output:**

***F. Write a Python program to find reverse of given number using user defined function***

**Program:**

**List = [1, 2, 3, 4, 5]**

**print(List.append(20544))**

**print(List.insert(2,10087) )**

**print(List.extend([2,3,4,5]))**

**print(sum(List))**

**print(List.count(2))**

**print(len(List))**

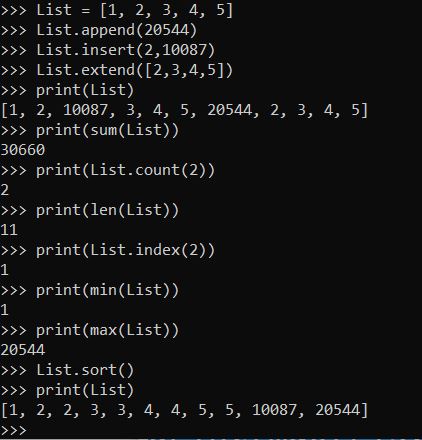
**print(List.index(2))**

**print(min(List))**

**print(max(List))**

**List.sort()**

**print(List)**

**Output:**

Practical-4

***A. Write a Python program to append text to a file and display the text.***

**Program:**

**with open("test.txt", "a") as myfile:**

**myfile.write("appended text")**

****Output:**

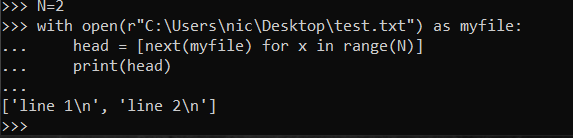
***B. Write a Python program to read first n lines of a file.***

**Program:**

**with open(r"C:\Users\nic\Desktop\test.txt") as myfile:**

**head = [next(myfile) for x in range(N)]**

**print(head)**

**Output:**

***C. Write a python program to find the longest words from file.***

**Program:**

**def find\_longest\_word(word\_list):**

**longest\_word = ''**

**for word in word\_list:**

**if len(word) > len(longest\_word):**

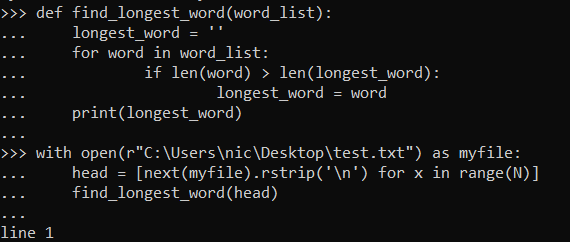
**longest\_word = word**

**print longest\_word**

**with open(r"C:\Users\nic\Desktop\test.txt") as myfile:**

**head = [next(myfile).rstrip('\n') for x in range(N)]**

**find\_longest\_word(head)**

**Output:**

***D. Write a Python program to count the number of lines in a text file.***

**Program:**

**with open(r"C:\Users\nic\Desktop\test.txt") as myfile:**

**head = [next(myfile).rstrip('\n') for x in range(N)]**

**print(“len:”,len(head))**

**Output:**

**Len:3**

***E. Write a Python program to write a list to a file.***

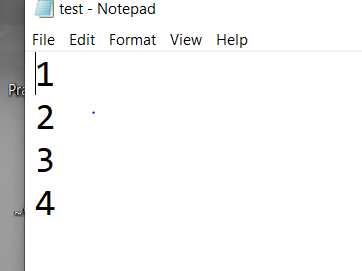
**Program:**

**my\_list=[1,2,3,4]**

**with open(r"C:\Users\nic\Desktop\test.txt", 'w') as f:**

**for item in my\_list:**

**f.write("%s\n" % item)**

**Output:**

***F.*** ***Write a Python program open file and show the difference between read(), readline() and readlines().***

**Reading from a file**

There are three ways to read data from a text file.

1. **read() :** Returns the read bytes in form of a string. Reads n bytes, if no n specified, reads the entire file.

File\_object.read([n])

1. **readline() :** Reads a line of the file and returns in form of a string.For specified n, reads at most n bytes. However, does not reads more than one line, even if n exceeds the length of the line.

File\_object.readline([n])

1. **readlines() :** Reads all the lines and return them as each line a string element in a list.

File\_object.readlines()

**Program:**

**file1 = open(r"C:\Users\nic\Desktop\test.txt","r+")**

**print("Output of Read function is ")**

**print(file1.read())**

**file1.seek(0)**

**print("Output of Readline function is ")**

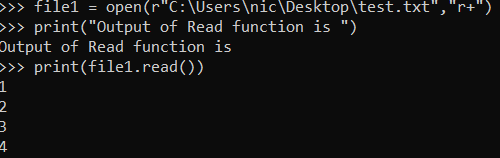
**print(file1.readline())**

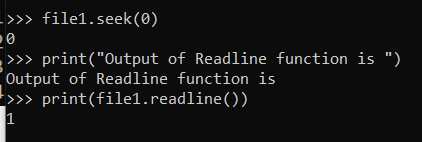
**file1.seek(0)**

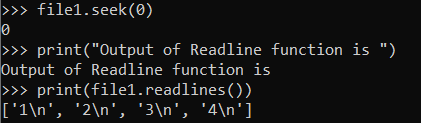
**print("Output of Readline function is ")**

**print(file1.readlines())**

**file1.close()**

**Output:**





***G.*** ***Write a Python program to show how to copy a file into another file in python***

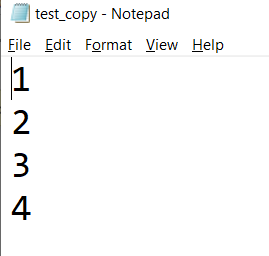
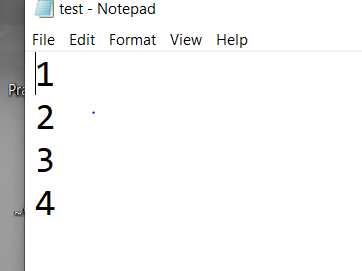
**Program:**

**source=open(r"C:\Users\nic\Desktop\test.txt","r+")**

**dest=open(r"C:\Users\nic\Desktop\test\_copy.txt","w")**

**dest.write(source.read())**

**Output:**



***H.*** ***Write a Python program to delete the last two lines from file***

**Program:**

**with open(r"C:\Users\nic\Desktop\test.txt",’r+’) as myfile:**

**head = [next(myfile).rstrip('\n') for x in range(N)]**

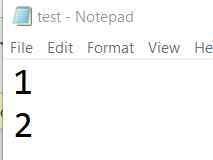
**print(“len:”,len(head))**

**head=head[:(len(head)-2):]**

**myfile= open(r"C:\Users\nic\Desktop\test.txt",’w’)**

**for item in head:**

**f.write("%s\n" % item)**

**Output:**

***I.*** ***Write a Python program to delete the last two lines from file***

**Program:**

**combine =[]**

**with open(r"C:\Users\nic\Desktop\test.txt") as xh:**

**with open(r"C:\Users\nic\Desktop\test\_copy.txt") as yh:**

**with open(r"C:\Users\nic\Desktop\test\_result.txt","w") as zh:**

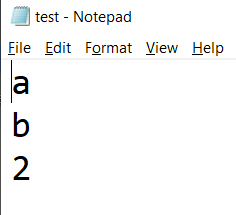
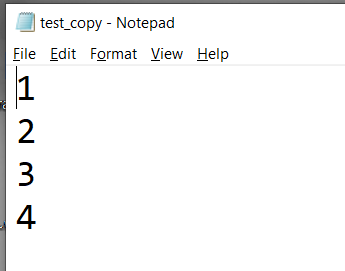
**xlines = xh.readlines()**

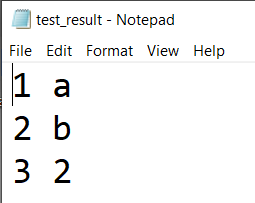
**ylines = yh.readlines()**

**for i in range(len(xlines)):**

**line = ylines[i].strip() + ' ' + xlines[i]**

**zh.write(line)**

**Output:**



***j. Write a Python program to assess if a file is closed or not.***

**Program:**

**try:**

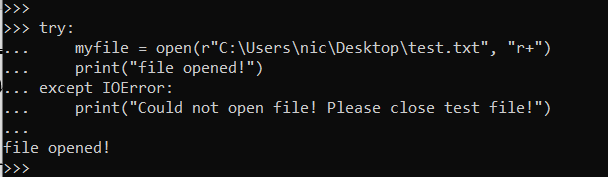
**myfile = open(r"C:\Users\nic\Desktop\test.txt", "r+")**

**print("file opened!")**

**except IOError:**

**print("Could not open file! Please close test file!")**

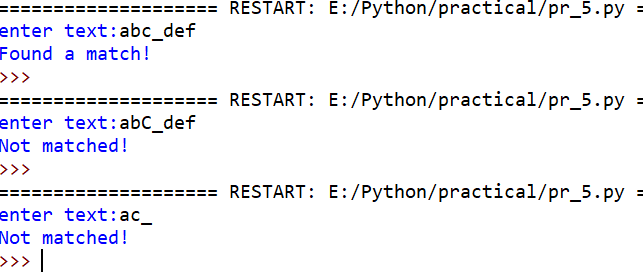
**Output:**



Practical-5

***A. Write a Python program to find sequences of lowercase letters joined with an underscore.***

**Program:**

**Output:**

**import re**

**def match(text):**

**patterns = '^[a-z]+\_[a-z]+$'**

**if re.search(patterns, text):**

**return 'Found a match!'**

**else:**

**return('Not matched!')**

**text= input("enter text:")**

**print(match(text))**

***B. Write a Python program to remove leading zeros from an IP address.***

**Program:**

**import re**

**def remove(ip):**

**#ip= re.split('^[0]\*',ip)[1]**

**string = re.sub('^[0]\*|', '',ip)**

**string = re.sub('\.[0]\*', '.',string)**

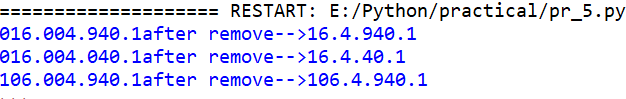
**return string**

**print('016.004.940.1'+'after remove-->'+remove('016.004.940.1'))**

**print('016.004.040.1'+'after remove-->'+remove('016.004.040.1'))**

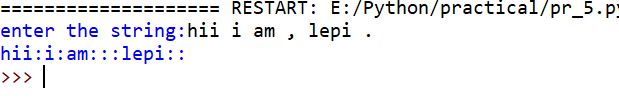
**print('106.004.940.1'+'after remove-->'+remove('106.004.940.1'))**

**Output:**



***C. Write a Python program to replace all occurrences of space, comma, or dot with a colon.***

**Program: Output:**



**import re**

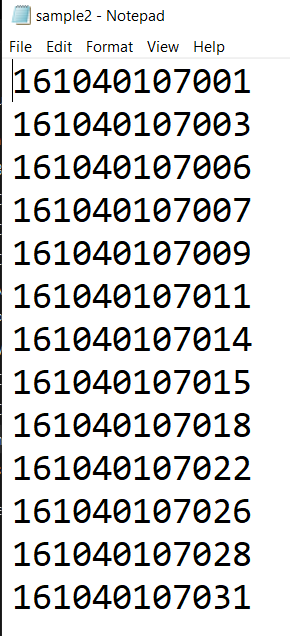
**text = input("enter the string:")**

**print(re.sub("[ ,.]", ":", text))**

***D. Write a Python program to count the number of lines in a text file.***

***a.***   **Extract the enrolment number of the students having enrollment number starting with "1610401070"  and store it in a new file.** 

**Program: Output:**



**import re**

**samples = []**

**with open('sample.txt') as myfile:**

**for line in myfile.readlines():**

**if re.search(r'[0-9A-Za-z]\*', line):**

**samples.append(line)**

**#[print(i) for i in samples]**

**with open("sample2.txt",'w') as my:**

**for i in samples:**

**num=re.search(r'\d{12}',i)**

**if num ==None:**

**print('not matched')**

**elif '1610401070' in num.group(0):**

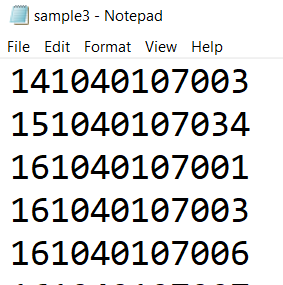
**my.write(num.group(0)+'\n')**

**else:**

**print('not matched '+num.group(0))**

***b.* extract list of enrolment number, sort it in descending order and store it in a new file.**

**Program: Output:**



**import re**

**samples = []**

**with open('sample.txt') as myfile:**

**for line in myfile.readlines():**

**if re.search(r'[0-9A-Za-z]\*', line):**

**samples.append(line)**

**enroll=[]**

**for i in samples:**

**num=re.search(r'\d{12}',i)**

**if num ==None:**

**print('not matched')**

**else:**

**enroll.append(int(num.group(0)))**

**enroll.sort()**

**with open("sample3.txt",'w') as my:**

**[my.write(str(i)+'\n') for i in enroll]**

***c. Extract email addresses from the sample.txt file, and store it in a new file.***

**Program: Output:**

**import re**

**samples = []**

**with open('sample.txt') as myfile:**

**for line in myfile.readlines():**

**if re.search(r'[0-9A-Za-z]\*',line):**

**samples.append(line)**

**for i in samples:**

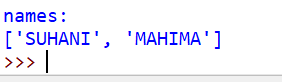
**email=re.findall('\S+@\S+',i)**

**print(email)**



**d. Extract the list of students having a length of a name is exactly 6.**

**Program: Output:**



**import re**

**samples = []**

**with open('sample.txt') as myfile:**

**for line in myfile.readlines():**

**if re.search(r'[0-9A-Za-z]\*', line):**

**samples.append(line)**

**names=[]**

**for i in samples:**

**name=re.findall(r'\s\*([a-zA-Z]+)\s\*',i)**

**try:**

**if len(name[1]) == 6:**

**names.append(name[1])**

**except:**

**print('names:')**

**print(names)**

**e. Extract the email addresses from a webpage using urllib2 module**

**Program:**

**from bs4 import BeauifulSoup**

**from urllib.request import urlopen,Request**

**import re**

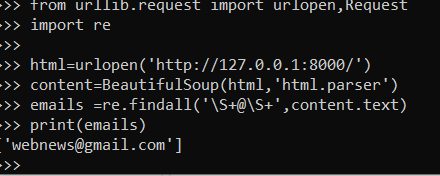
**html=urlopen('http://127.0.0.1:8000/')**

**content=BeautifulSoup(html,'html.parser')**

**emails = re.findall('\S+@\S+',content.text)**

**print(emails)**

**Output:**



Practical-6

***A. Demonstrate the concept of class and object using python.***

**Program:**

**def classVol():**

**def \_\_init\_\_(self):**

**print("")**

**def volume(self,data1 = None, data2 = None):**

**if data1 and not data2:**

**print("Volume of cube is ", data1\*\*3)**

**elif data1 and data2:**

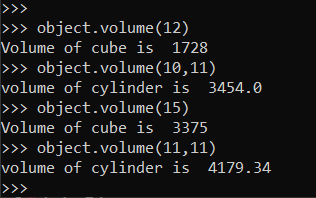
**print("volume of cylinder is ", 3.14\*data1\*data1\*data2)**

**object = Vol()**

**object.volume(12)**

**object.volume(10,11)**

**Output:**



***b. Implement operator overloading to overload addition (+) operator using python.***

**Program:**

**class A:**

**def \_\_init\_\_(self, a):**

**self.a = a**

**# adding two objects**

**def \_\_add\_\_(self, o):**

**return self.a + o.a**

**ob1 = A(1)**

**ob2 = A(2)**

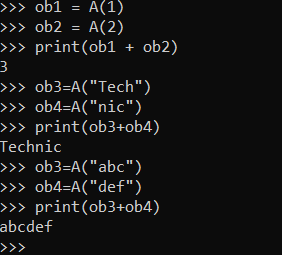
**ob3 = A("tech")**

**ob4 = A("nic")**

**print(ob1 + ob2)**

**print(ob3 + ob4)**

**Output:**



Practical-7

***A. Write socket program to demonstrate the echo server functionality***

**Program:**

***SERVER.PY***

**import socket**

**HOST='localhost'**

**PORT=50001**

**s=socket.socket(socket.AF\_INET,socket.SOCK\_STREAM)**

**s.bind((HOST,PORT))**

**s.listen(1)**

**conn,addr=s.accept()**

**print("connected with:",conn)**

**while True:**

**data=conn.recv(1024).decode()**

**print(data)**

**if not data: break**

**conn.send(data.encode())**

**s.close()**

***CLIENT.PY***

**import socket**

**HOST='localhost'**

**PORT=50001**

**s=socket.socket(socket.AF\_INET,socket.SOCK\_STREAM)**

**s.connect((HOST,PORT))**

**msg=input("enter msg(quit to stop):")**

**while True:**

**s.send(msg.encode())**

**data=s.recv(1024).decode()**

**print("server:",data)**

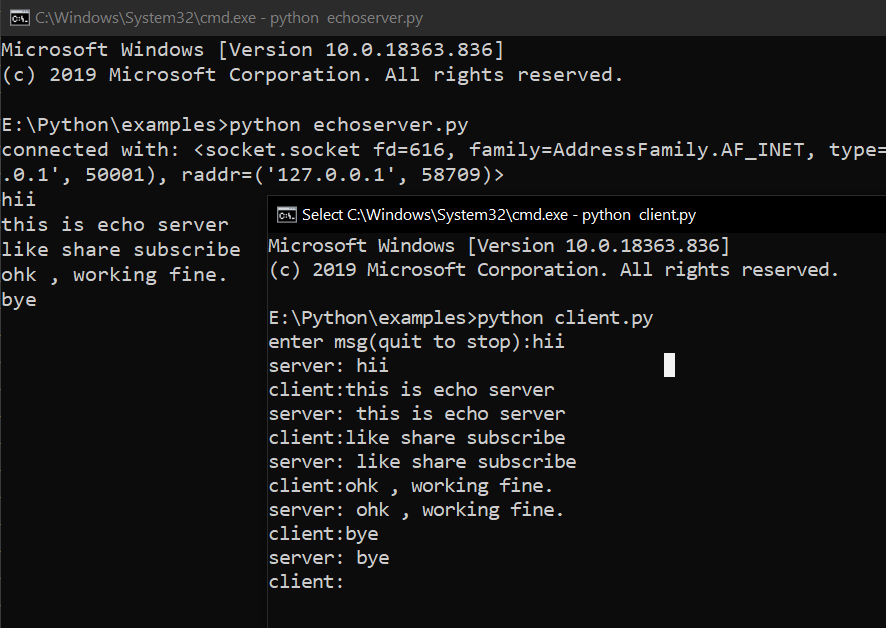
**if msg == 'quit':**

**break**

**msg=input("client:")**

**s.close()**

**Output:**



Practical-8

***A. Chat application using socket programming.***

**Program:**

***SERVER.PY***

**import socket**

**HOST='localhost'**

**PORT=50001**

**s=socket.socket(socket.AF\_INET,socket.SOCK\_STREAM)**

**s.bind((HOST,PORT))**

**s.listen(1)**

**conn,addr=s.accept()**

**print("connected with:",conn)**

**while True:**

**data=conn.recv(1024).decode()**

**print("client:",data)**

**if not data: break**

**data=input("server:")**

**conn.send(data.encode())**

**s.close()**

***CLIENT.PY***

**import socket**

**HOST='localhost'**

**PORT=50001**

**s=socket.socket(socket.AF\_INET,socket.SOCK\_STREAM)**

**s.connect((HOST,PORT))**

**msg=input("enter msg(quit to stop):")**

**while True:**

**s.send(msg.encode())**

**data=s.recv(1024).decode()**

**print("server:",data)**

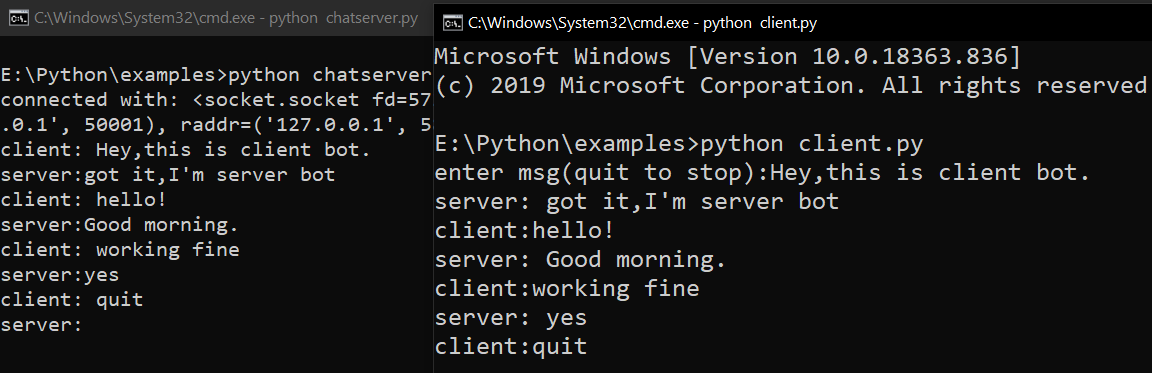
**if msg == 'quit':**

**break**

**msg=input("client:")**

**s.close()**

**Output:**



Practical-9

1. ***To study about PyPlot.***

**Intro to pyplot**

[matplotlib.pyplot](https://matplotlib.org/api/_as_gen/matplotlib.pyplot.html#module-matplotlib.pyplot) is a collection of command style functions that make matplotlib work like MATLAB. Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc.

In [matplotlib.pyplot](https://matplotlib.org/api/_as_gen/matplotlib.pyplot.html" \l "module-matplotlib.pyplot" \o "matplotlib.pyplot) various states are preserved across function calls, so that it keeps track of things like the current figure and plotting area, and the plotting functions are directed to the current axes

EXAMPLE

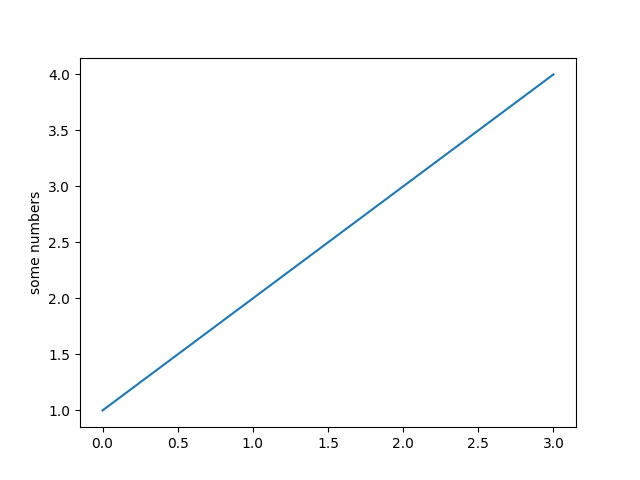
import matplotlib.pyplot as plt

plt.plot([1, 2, 3, 4])

plt.ylabel('some numbers')

plt.show()

OUTPUT



## Plotting with keyword strings

There are some instances where you have data in a format that lets you access particular variables with strings. For example, with [numpy.recarray](https://docs.scipy.org/doc/numpy/reference/generated/numpy.recarray.html" \l "numpy.recarray" \o "(in NumPy v1.17)) or [pandas.DataFrame](https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.html" \l "pandas.DataFrame" \o "(in pandas v1.0.3)).

Matplotlib allows you provide such an object with the data keyword argument. If provided, then you may generate plots with the strings corresponding to these variables.

EXAMPLE

data = {'a': np.arange(50),

'c': np.random.randint(0, 50, 50),

'd': np.random.randn(50)}

data['b'] = data['a'] + 10 \* np.random.randn(50)

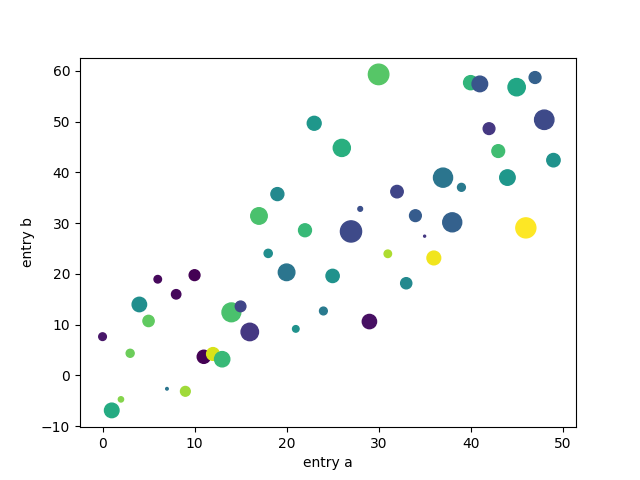
data['d'] = np.abs(data['d']) \* 100

plt.scatter('a', 'b', c='c', s='d', data=data)

plt.xlabel('entry a')

plt.ylabel('entry b')

plt.show()

OUTPUT

Practical-10

1. ***To Study about Turtle, TKinter.***

**Introduction**

Turtle graphics is a popular way for introducing programming to kids. It was part of the original Logo programming language developed by Wally Feurzeig, Seymour Papert and Cynthia Solomon in 1967.

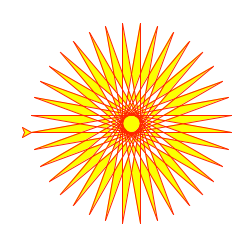
Imagine a robotic turtle starting at (0, 0) in the x-y plane. After an import turtle, give it the command turtle.forward(15), and it moves (on-screen!) 15 pixels in the direction it is facing, drawing a line as it moves. Give it the command turtle.right(25), and it rotates in-place 25 degrees clockwise.

By combining together these and similar commands, intricate shapes and pictures can easily be drawn.

The [turtle](https://docs.python.org/3/library/turtle.html#module-turtle) module is an extended reimplementation of the same-named module from the Python standard distribution up to version Python 2.5.

It tries to keep the merits of the old turtle module and to be (nearly) 100% compatible with it. This means in the first place to enable the learning programmer to use all the commands, classes and methods interactively when using the module from within IDLE run with the -n switch.

The turtle module provides turtle graphics primitives, in both object-oriented and procedure-oriented ways. Because it uses [tkinter](https://docs.python.org/3/library/tkinter.html" \l "module-tkinter" \o "tkinter: Interface to Tcl/Tk for graphical user interfaces) for the underlying graphics, it needs a version of Python installed with Tk support.

 **Turtle star**

**from** **turtle** **import** \*

color('red', 'yellow')

begin\_fill()

**while** **True**:

forward(200)

left(170)

**if** abs(pos()) < 1:

**break**

end\_fill()

done()

**METHODS IN TURTLE.**

To move forward with given distance.

* turtle.**forward**(*distance*)
* turtle.**fd**(*distance*)

To move backwoard with given distance.

* + turtle.**back**(*distance*)
  + turtle.**bk**(*distance*)
  + turtle.**backward**(*distance*)

To move right side.

* turtle.**right**(*angle*)
* turtle.**rt**(*angle*)

To move left side.

* turtle.**left**(*angle*)
* turtle.**lt**(*angle*)

**distance** – a number (integer or float)

# **TKINTER:-** Python interface to Tcl/Tk.

The [tkinter](https://docs.python.org/3/library/tkinter.html" \l "module-tkinter" \o "tkinter: Interface to Tcl/Tk for graphical user interfaces) package (“Tk interface”) is the standard Python interface to the Tk GUI toolkit. Both Tk and [tkinter](https://docs.python.org/3/library/tkinter.html" \l "module-tkinter" \o "tkinter: Interface to Tcl/Tk for graphical user interfaces) are available on most Unix platforms, as well as on Windows systems. (Tk itself is not part of Python; it is maintained at ActiveState.)

Running python -m tkinter from the command line should open a window demonstrating a simple Tk interface, letting you know that [tkinter](https://docs.python.org/3/library/tkinter.html" \l "module-tkinter" \o "tkinter: Interface to Tcl/Tk for graphical user interfaces) is properly installed on your system, and also showing what version of Tcl/Tk is installed, so you can read the Tcl/Tk documentation specific to that version.

## Tkinter Modules

Most of the time, [tkinter](https://docs.python.org/3/library/tkinter.html" \l "module-tkinter" \o "tkinter: Interface to Tcl/Tk for graphical user interfaces) is all you really need, but a number of additional modules are available as well. The Tk interface is located in a binary module named \_tkinter. This module contains the low-level interface to Tk, and should never be used directly by application programmers. It is usually a shared library (or DLL), but might in some cases be statically linked with the Python interpreter.

In addition to the Tk interface module, [tkinter](https://docs.python.org/3/library/tkinter.html" \l "module-tkinter" \o "tkinter: Interface to Tcl/Tk for graphical user interfaces) includes a number of Python modules, tkinter.constants being one of the most important. Importing [tkinter](https://docs.python.org/3/library/tkinter.html" \l "module-tkinter" \o "tkinter: Interface to Tcl/Tk for graphical user interfaces) will automatically import tkinter.constants, so, usually, to use Tkinter all you need is a simple import statement:

Example:

**import tkinter as tk**

**class Application(tk.Frame):**

**def \_\_init\_\_(self, master=None):**

**super().\_\_init\_\_(master)**

**self.master = master**

**self.pack()**

**self.create\_widgets()**

**def create\_widgets(self):**

**self.hi\_there = tk.Button(self)**

**self.hi\_there["text"] = "Hello World\n(click me)"**

**self.hi\_there["command"] = self.say\_hi**

**self.hi\_there.pack(side="top")**

**self.quit = tk.Button(self, text="QUIT", fg="red",**

**command=self.master.destroy)**

**self.quit.pack(side="bottom")**

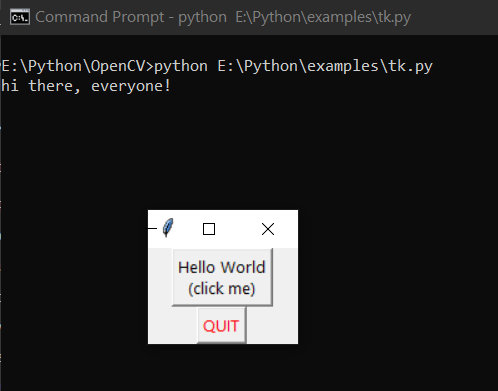
**def say\_hi(self):**

**print("hi there, everyone!")**

**root = tk.Tk()**

**app = Application(master=root)**

**app.mainloop()**

 **OUTPUT:**