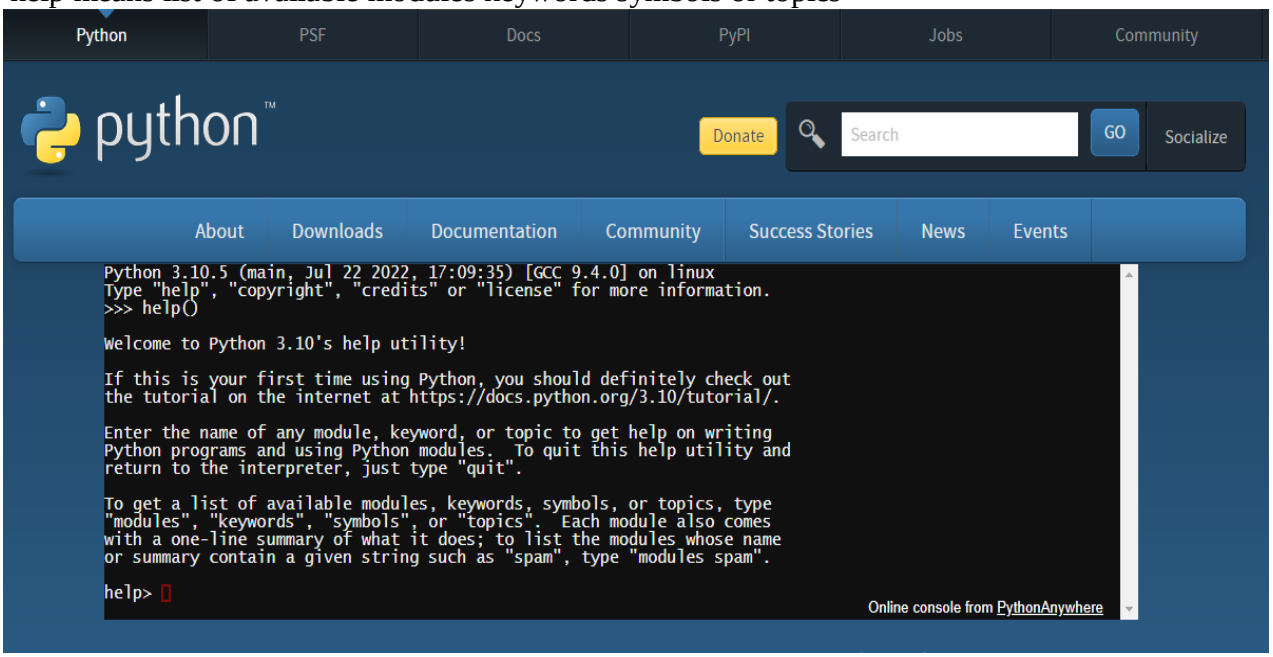
Week 1:

PROGRAM 1.1:

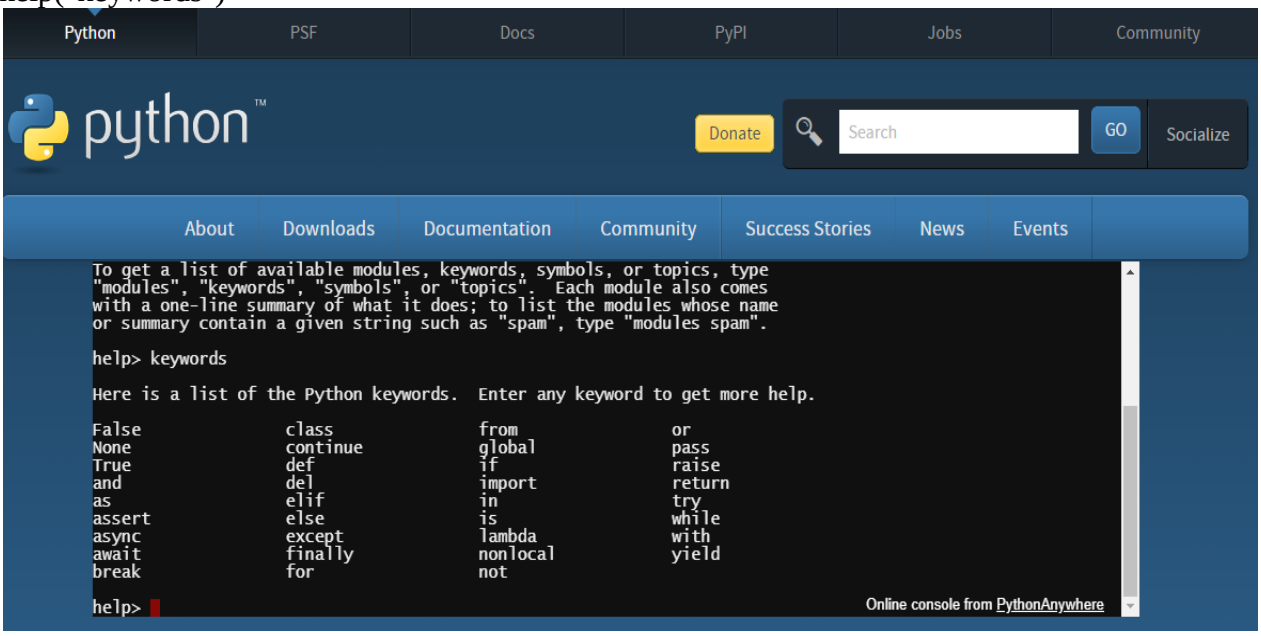
i) Use a web browser to go to the Python website http://python.org. This page contains information about Pythonand links to Python-related pages, and it gives you the ability to search the Python documentation.
ii) Start the Python interpreter and type help() to start the online help utility.

-open web browser http://python.org
-then type help()
-help means list of available modules keywords symbols or topics

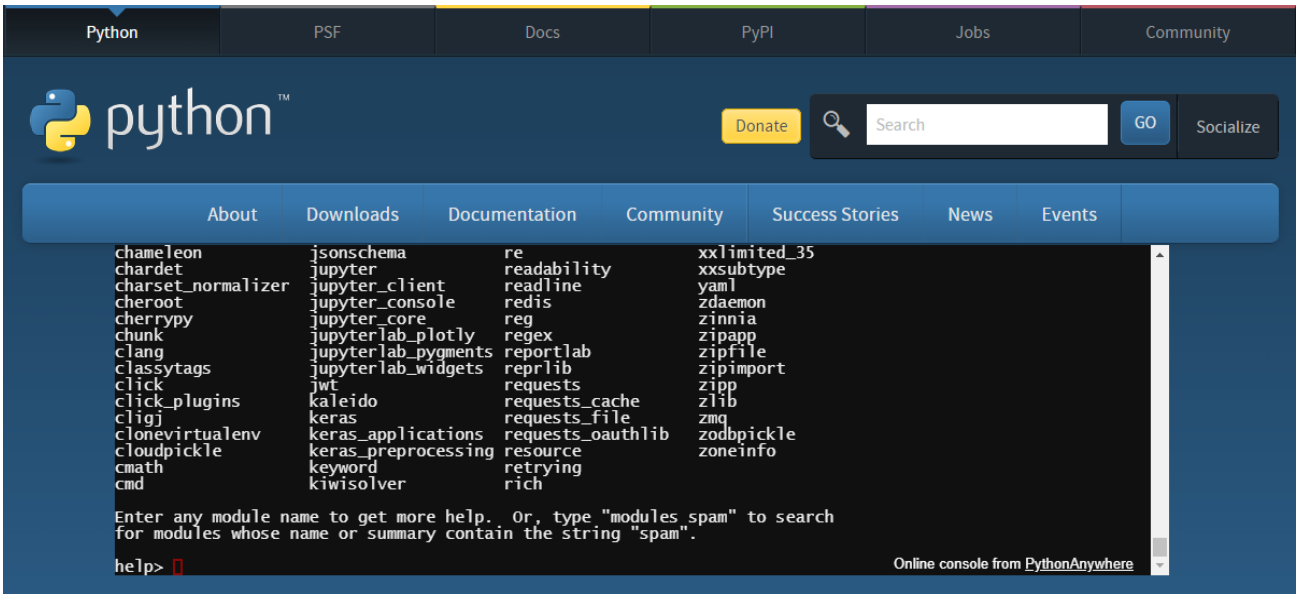


-go back = type quit or cntrl+D

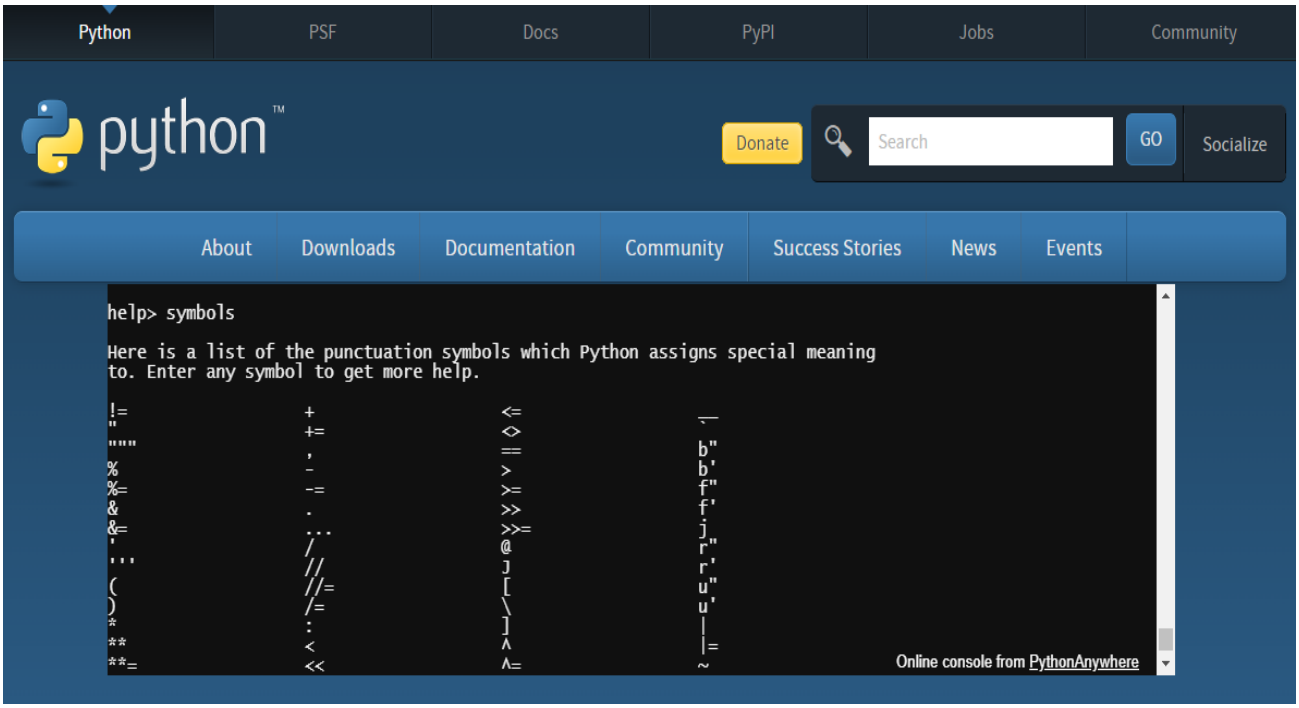
like
help("keywords")



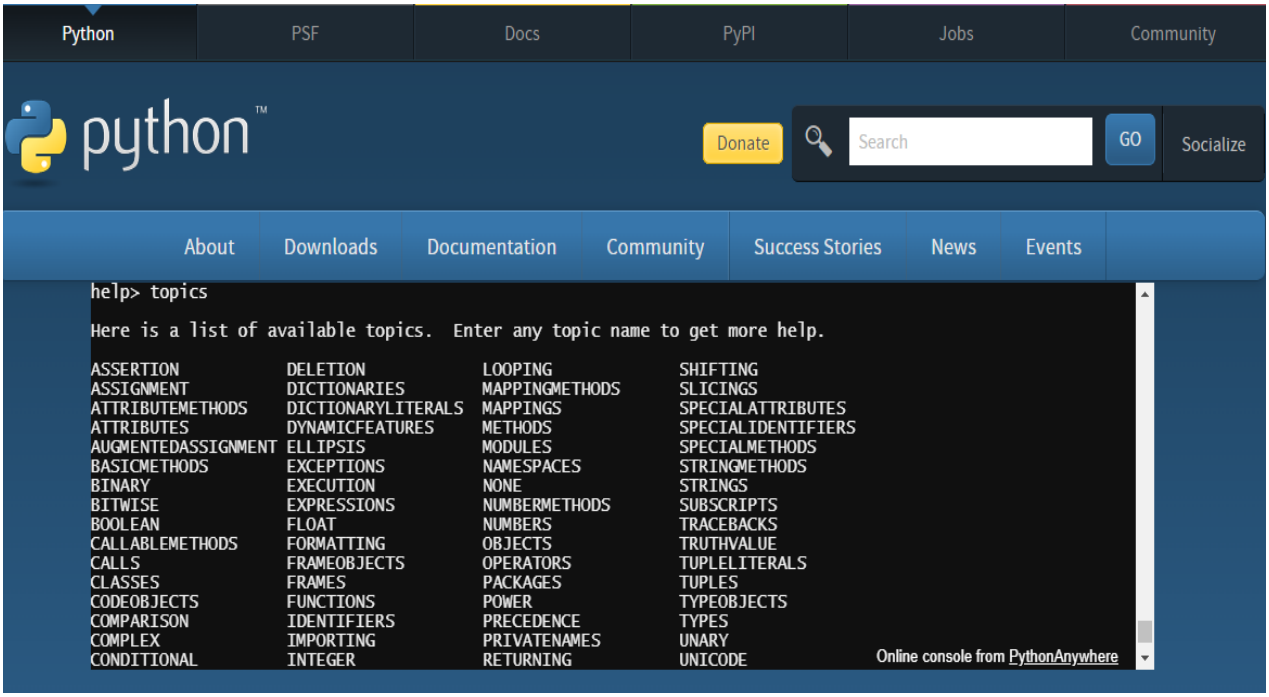
help("modules")



help("symbols")



help("topics")


Python
PSF
Docs
PyPI
Jobs
Community
python
Donate
Search
GO
Socialize
About
Downloads
Documentation
Community
Success Stories
News
Events
help> topics
Here is a list of available topics. Enter any topic name to get more help.
ASSERTION DELETION LOOPING SHIFTING
ASSIGNMENT DICTIONARIES MAPPINGMETHODS SLICINGS
ATTRIBUTEMETHODS DICTIONARYLITERALS MAPPINGS SPECIALATTRIBUTES
ATTRIBUTES DYNAMICFEATURES METHODS SPECIALIDENTIFIERS
AUGMENTEDASSIGNMENT ELLIPSIS MODULES SPECIALMETHODS
BASICMETHODS EXCEPTIONS NAMESPACES STRINGMETHODS
BINARY EXECUTION NONE STRINGS
BITWISE EXPRESSIONS NUMBERMETHODS SUBSCRIPTS
BOOLEAN FLOAT NUMBERS TRACEBACKS
CALLABLEMETHODS FORMATTING OBJECTS TRUTHVALUE
CALLS FRAMEOBJECTS OPERATORS TUPLELITERALS
CLASSES FRAMES PACKAGES TUPLES
CODEOBJECTS FUNCTIONS POWER TYPEOBJECTS
COMPARISON IDENTIFIERS PRECEDENCE TYPES
COMPLEX IMPORTING PRIVATENAMES UNARY
CONDITIONAL INTEGER RETURNING UNICODE
Online console from PythonAnywhere

PROGRAM 1.2:

Start a Python interpreter and use it as a Calculator.

Source Code:

```
# This function adds two numbers
def add(x, y):
    return x + y

# This function subtracts two numbers
def subtract(x, y):
    return x - y

# This function multiplies two numbers
def multiply(x, y):
    return x * y

# This function divides two numbers
def divide(x, y):
    return x / y


print("Select operation.")
print("1.Add")
print("2.Subtract")
print("3.Multiply")
print("4.Divide")

while True:
    # take input from the user
    choice = input("Enter choice(1/2/3/4): ")

    # check if choice is one of the four options
    if choice in ('1', '2', '3', '4'):
        try:
            num1 = float(input("Enter first number: "))
            num2 = float(input("Enter second number: "))
        except ValueError:
print("Invalid input. Please enter a number.")
            continue

        if choice == '1':
print(num1, "+", num2, "=", add(num1, num2))

elif choice == '2':
print(num1, "-", num2, "=", subtract(num1, num2))

elif choice == '3':
print(num1, "*", num2, "=", multiply(num1, num2))
```

```
elif choice == '4':
print(num1, "/", num2, "=", divide(num1, num2))

# check if user wants another calculation
        # break the while loop if answer is no
next_calculation = input("Let's do next calculation? (yes/no): ")
        if next_calculation == "no":
          break
    else:
print("Invalid Input")
```

Output:

```
Select operation.
1.Add
2.Subtract
3.Multiply
4.Divide
Enter choice(1/2/3/4): 1
Enter first number: 5
Enter second number: 3
5.0 + 3.0 = 8.0
Let's do next calculation? (yes/no): yes
Enter choice(1/2/3/4): 2
Enter first number: 5
Enter second number: 3
5.0 - 3.0 = 2.0
Let's do next calculation? (yes/no): yes
Enter choice(1/2/3/4): 3
Enter first number: 5
Enter second number: 3
5.0 * 3.0 = 15.0
Let's do next calculation? (yes/no): yes
Enter choice(1/2/3/4): 4
Enter first number: 5
Enter second number: 3
5.0 / 3.0 = 1.6666666666666667
Let's do next calculation? (yes/no): no
```

PROGRAM 1. 3:

i) Write a program to calculate compound interest when principal, rate and number of periods are given.

Source Code:

```
import math
p=float(input("Enter Principal Amount: "))
r=float(input("Enter Rate of Interest: "))
t=float(input("Enter Time Period in years: "))
SI=(p*t*r)/100
print("Simple Interest: ",SI)
CI=p *((1 + (r/100))**t)
print("Compound interest:",CI)
```

Output:

```
Enter Principal Amount: 10000
Enter Rate of Interest: 2
Enter Time Period in years: 1
Simple Interest:  200.0
Compound interest: 10200.0
```

ii) Given coordinates (x1, y1), (x2, y2) find the distance between two points

Source Code:

```
import math

x1 = 5
x2 = 3
y1 = 8
y2 = 4

distance = math.sqrt(((x1 - x2) ** 2) + ((y1 - y2) ** 2))

print(distance)
```

Output:

4.47213595499958

PROGRAM 1.4:

Read name, address, email and phone number of a person through keyboard and print the details.

Source Code:

```
name = input("Enter your name: ")
address = input("Enter your Address: ")
email = input("Enter your emailid: ")
mobno = input("Enter your Mobile Number: ")
print(name)
print(address)
print(email)
print(mobno)
```

Output:

```
Enter your name: Vaagdevi
Enter your Address: Bollikunta
Enter your emailid: vaag@gmail.com
Enter your Mobile Number: 123456789
Vaagdevi
Bollikunta
vaag@gmail.com
123456789
```

Week - 2:

PROGRAM 2.1:

Print the below triangle using for loop.
5
4 4
3 3 3
2 2 2 2
1 1 1 1 1

Source Code:
```
n = 6
for i in range(n,0,-1):
    for j in range(n-i,0,-1):
print(i, end=' ')
print(")
```

Output:

```
5
4 4
3 3 3
2 2 2 2
1 1 1 1 1
```

PROGRAM 2.2:

Write a program to check whether the given input is digit or lowercase character or uppercase character or a special character (use 'if-else-if' ladder)

Source Code:
```
ch = input("Enter a character: ")
if ch>= '0' and ch<= '9':
  print("Digit")
elifch.isupper ():
print("Uppercase character")
elifch.islower ():
print("Lowercase character")
else:
print("Special character")
```

Output:

```
Enter a character: 1234
Digit
```

PROGRAM 2.3:

Python Program to Print the Fibonacci sequence using while loop ?

Source Code:

```
def fib(number):
  count = 0
  first = 0
  second = 1
  temp = 0
  while count <= number:
    print(first)
    temp = first + second
    first = second
    second = temp
    count = count + 1
fib(10)
```

Output:

```
0
1
1
2
3
5
8
13
21
34
55
```

PROGRAM 2.4:

Python program to print all prime numbers in a given interval (use break)?

Source Code:

```
start = 2
n = int(input("enter n numer: "))

print("Prime numbers between", start, "and", n, "are:")

for num in range(start, n + 1):
  # all prime numbers are greater than 1
  if num > 1:
      for i in range(2, num):
          if (num % i) == 0:
              break
      else:
          print(num)
```

Output:

```
enter n numer: 10
Prime numbers between 2 and 10 are:
2
3
5
7
```

WEEK-3

PROGRAM 3.1:

i) Write a program to convert a list and tuple into arrays?

Source Code:

```
from array import array
list1=[1,7,0,6,5,6]
tuple1=(4,2,6,7,1)
listarray=array("i",list1)
tuplearray=array("i",tuple1)
print("list:",list1)
print("tuple:",tuple1)
print("listarray:",listarray)
print("tuplearray:",tuplearray)
```

Output:

```
list: [1, 7, 0, 6, 5, 6]
tuple: (4, 2, 6, 7, 1)
listarray: array('i', [1, 7, 0, 6, 5, 6])
tuplearray: array('i', [4, 2, 6, 7, 1])
```

ii) Write a program to find common values between two arrays?

Source Code:

```
from array import array
a1=array("i",[5,6,7,8])
a2=array("i",(9,6,5,3))
result=[i for i in a1 if i in a2]
print(result)
```

Output:

```
[5, 6]
```

PROGRAM 3.2:

Write a function called gcd that takes parameters a and b and returns their greatest common divisor?

Source Code:

```
def gcd(a, b):
if(b == 0):
      return a
   else:
      return gcd(b, a % b)

a = int(input("Enter a number: "))
b = int(input("Enter a number: "))
print("The greatest common divisor of ",a," and ",b," is : ", end="")
print(gcd(a, b))
```

Output:

```
Enter a number: 5
Enter a number: 10
The greatest common divisor of  5  and  10  is : 5
```

PROGRAM 3.3:

Write a function called palindrome that takes a string argument and returnsTrue if it is a palindrome and False otherwise. Remember that you can use the built-in function len to check the length of a string.?

Source Code:

```
def isPalindrome(s):
   return s == s[::-1]

# Driver code
s = "malayalam"
ans = isPalindrome(s)

if ans:
   print("Yes")
else:
   print("No")
```

Output:

```
Yes
```

WEEK 4:

PROGRAM 4.1:

Write a function called is_sorted that takes a list as a parameter and returns True if the list is sorted in ascendingorder and False otherwise?

Source Code:

```
def is_sorted(lst):
    n = len(lst)
    for i in range(n):
        for j in range(0, n-i-1):
            if lst[j] >lst[j+1]:
lst[j], lst[j+1] = lst[j+1], lst[j]

    if y == lst:
        p = True
    else:
        p = False

    return p

y = [4,6,2,8,1,3]
print(is_sorted(y))
```

Output:

```
True
```

PROGRAM 4.2:

Write a function called has_duplicates that takes a list and returns True if there is any element that appearsmore than once. It should not modify the original list.

Source Code:

```
def has_duplicates(ms):
    n=len(ms)
    k=[]
    for i in range(n):
        for j in range(i+1,n):
            if (ms[i]==ms[j] and ms[i] not in k):
k.append(ms[i])
print("Duplicate values in list: ",k)
    if k==[]:
        return True
    else:
        return False
```

```
p=[4,3,2,5,6,7,3,5]

print("original list: ",p)
print(has_duplicates(p))
```

Output:

```
original list:  [4, 3, 2, 5, 6, 7, 3, 5]
Duplicate values in list:  [3, 5]
False
```

i). Write a function called remove_duplicates that takes a list and returns a new list with only the unique elements from the original. Hint: they don't have to be in the same order?

Source Code:

```
def remove_duplicates(rc):
    n=len(rc)
    k=[]
    for i in range(n):
        for j in range(i+1,n):
            if (rc[i]!=rc[j] and rc[i] not in k):
k.append(rc[i])
print("Unique elements from list")
    return k

d=[4,3,2,5,6,7,3,5]
print("Elements from original list: ",d)
print(remove_duplicates(d))
```

Output:

```
Elements from original list:  [4, 3, 2, 5, 6, 7, 3, 5]
Unique elements from list
[4, 3, 2, 5, 6, 7]
```

ii). The wordlist I provided, words.txt, doesn't contain single letter words. So you might want to add "I", "a", andthe empty string.

Source Code:

```
fp = open("word.txt","r")
y = fp.read()
print(y)
s = y.split()
print(s)
p=list(s)
x="a"
for i in range(len(s)):
    if len(s[i])==1:
        p[i].append(x)
print(p)
```

Output:

StudentsIa

iii). Write a python code to read dictionary values from the user. Construct a function to invert its content. i.e.,keys should be values and values should be keys.

Source Code:

```
d={}
n=int(input("Enter items number"))
for i in range(n):
    k=input("Enter key: ")
    v=input("Enter value:")
d.update({k:v})

print("Original Dictionary: ",d)
rd={}
for k,v in d.items():
    p=v
    t=k
rd.update({p:t})

print("\nNew dictionary: ",rd)
```

Output:

```
Enter items number2
Enter key: 1
Enter value:apple
Enter key: 2
Enter value:banana
Original Dictionary:  {'1': 'apple', '2': 'banana'}

New dictionary:  {'apple': '1', 'banana': '2'}
```

PROGRAM 4.3:

i) Add a comma between the characters. If the given word is 'Apple', it should become 'A,p,p,l,e'

Source Code:

```
r=[]
r.extend("Apple")
#r = sm.split(",")
print(r)
```

Output:

```
['A', 'p', 'p', 'l', 'e']
```

ii) Remove the given word in all the places in a string?

Source Code:

```
sm="Pleasant Morning Students"
r=sm.split()
x=r.remove("Pleasant")
print(r)
```

Output:

```
['Morning', 'Students']
```

iii) Write a function that takes a sentence as an input parameter and replaces the first letter of every word withthe corresponding upper case letter and the rest of the letters in the word by corresponding letters in lowercase without using a built-in function?

Source Code:

```
def cap(s):
sp=s.split()
    x=""

 for i in sp:
        x=x+i.capitalize()+" "
    return x
p="vaagdevi engineering college"
print(cap(p))
```

Output:

```
Vaagdevi Engineering College
```

PROGRAM 4.4:

Writes a recursive function that generates all binary strings of n-bit length

Source Code:

```
def genbin(n, bs=''):
  if len(bs) == n:
    print(bs)
  else:
genbin(n, bs + '0')
genbin(n, bs + '1')

genbin(2)
```

Output:

```
00
01
10
11
```

Week 5:

PROGRAM 5.1:

i) Write a python program that defines a matrix and prints

Source Code:

```
Row = int(input("Enter the number of rows:"))
Column = int(input("Enter the number of columns:"))

# Initialize matrix
matrix = []
print("Enter the entries row wise:")

# For user input
# A for loop for row entries
for row in range(Row):
    a = []
    # A for loop for column entries
    for column in range(Column):
        a.append(int(input()))
    matrix.append(a)

# For printing the matrix
for row in range(Row):
    for column in range(Column):
        print(matrix[row][column], end=" ")
    print()
```

Output:

```
Enter the number of rows:2

Enter the number of columns:2

Enter the entries row wise:

5

6

7

8

5 6

7 8
```

ii) Write a python program to perform addition of two square matrices

Source Code:

```
X = [[12,7,3],
    [4 ,5,6],
    [7 ,8,9]]
Y = [[5,8,1],
    [6,7,3],
    [4,5,9]]

result = [[X[i][j] + Y[i][j]  for j in range(len(X[0]))] for i in range(len(X))]

for r in result:
   print(r)
```

Output:

```
[17, 15, 4]
[10, 12, 9]
[11, 13, 18]
```

iii) Write a python program to perform multiplication of two square matrices

Source Code:

```
X = [[12,7,3],
    [4 ,5,6],
    [7 ,8,9]]
Y = [[5,8,1],
    [6,7,3],
    [4,5,9]]
result = [[X[i][j] * Y[i][j]  for j in range(len(X[0]))] for i in range(len(X))]
for r in result:
   print(r)
```

Output:

```
[60, 56, 3]
[24, 35, 18]
[28, 40, 81]
```

PROGRAM 5.2:

How do you make a module? Give an example of construction of a module using different geometrical shapesand operations on them as its functions.

Source Code:

```
#using module finding area
import gshapes

gshapes.square_area(5)
gshapes.rect_area(5,6)
gshapes.circle_area(10)
gshapes.triange_area(20,3)


#FInding areas of Square,Rectangle, CIrcle and Triangle

def square_area(side):
    area = side*side
print("Area of Square:",area)
    return

def rect_area(x,y):
    area = x*y
print("Area of Rectangle:",area)
    return

def circle_area(radius):
    area = 3.14*radius**2
print("Area of Circle:",area)
    return

def triange_area(b,h):
    area = 0.5*b*h
print("Area of Triangle:",area)
    return
```

Output:

```
Area of Square:25

Area of Rectangle:50

Area of Circle:9.42

Area of Triangle:25.0
```

PROGRAM 5.3:

Use the structure of exception handling all general purpose exceptions.

Source Code:

```
try:
    x=int(input("Enter a Value"))
    y=int(input("Enter a Value"))
    z=x/y
print("z value",z)
except ZeroDivisionError:
print("Do not give denominator zero")
except InvalidError:
print("Give number type values:")
except Exception:
print("Kindly give valid input")
finally:
print("End of program")
```

Output:

```
Enter a Value1
Enter a Value0
Do not give denominator zero
End of program
```

Week-6:

PROGRAM 6.1:

a. Write a function called draw_rectangle that takes a Canvas and a Rectangle as arguments

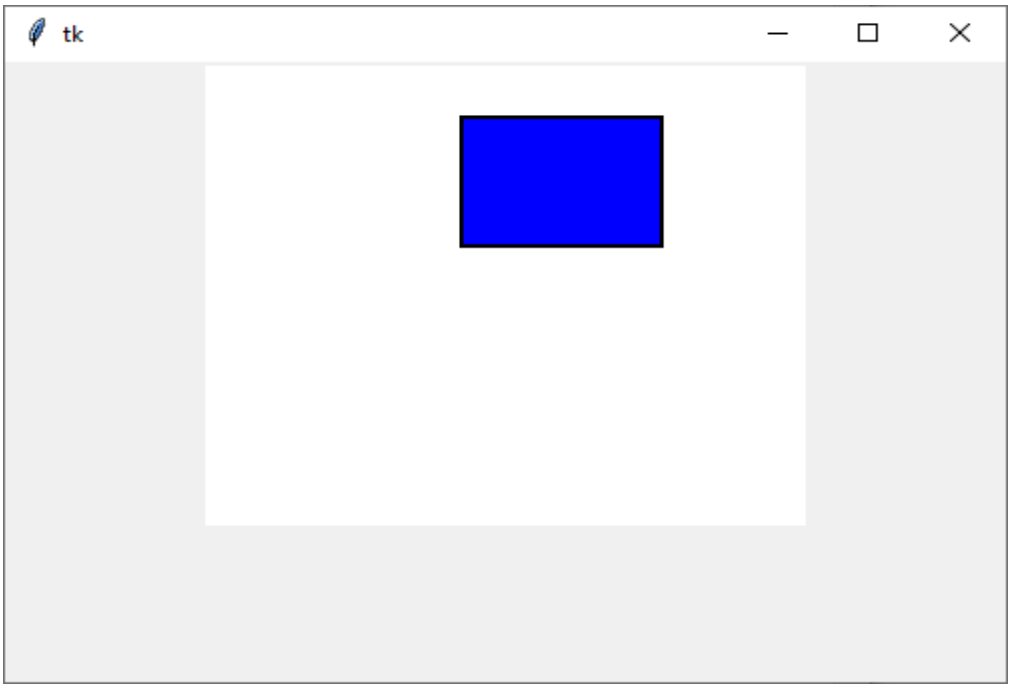
b. and draws arepresentation of the Rectangle on the Canvas.

b. Add an attribute named color to your Rectangle objects and modify draw_rectangle so that it uses the colorattribute as the fill color.

Source Code:

```
from tkinter import *
top = Tk()
top.geometry("500x900")
C = Canvas(top, bg="white", height=250, width=300)

#coord = 10, 50, 240, 210
#arc = C.create_arc(coord, start=0, extent=150, fill="red")

C.pack()

a = C.create_rectangle(110, 10, 210, 80,outline = "black", fill = "blue",width = 2)
C.move(a, 20, 20)

top.mainloop()
```

Output:

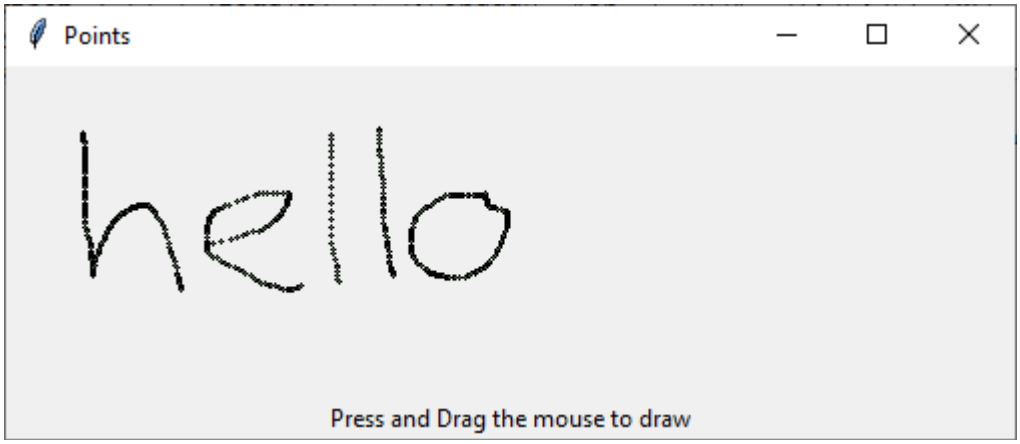


c. Write a function called draw_point that takes a Canvas and a Point as arguments and draws a representationof the Point on the Canvas.

Source Code:

```
from tkinter import *
canvas_width = 500
canvas_height = 150

def paint(event):
python_green = "#476042"
    x1, y1 = (event.x - 1), (event.y - 1)
    x2, y2 = (event.x + 1), (event.y + 1)
w.create_oval(x1, y1, x2, y2, fill=python_green)

master = Tk()
master.title("Points")
w = Canvas(master,
           width=canvas_width,
           height=canvas_height)
w.pack(expand=YES, fill=BOTH)
w.bind("<B1-Motion>", paint)

message = Label(master, text="Press and Drag the mouse to draw")
message.pack(side=BOTTOM)

mainloop()
```

Output:

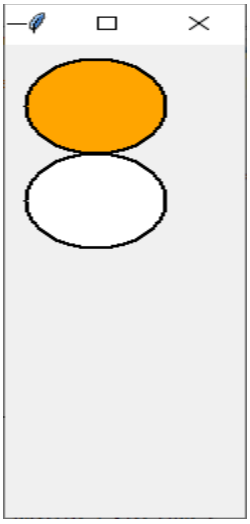


d. Define a new class called Circle with appropriate attributes and instantiate a few Circle objects. Write afunction called draw_circle that draws circles on the canvas.

Source Code:

```
from tkinter import *

class Circle:
    def __init__(self, master = None):
self.master = master
self.draw_circle()

    def draw_circle(self):
self.c = Canvas(self.master)
        #diameter as 80
        self.c.create_oval(10,10,80,80,outline="black",fill="orange",width=2)

        #diameter as 100
        self.c.create_oval(10,150,80,80,outline="black",fill="white",width=2)
self.c.pack(fill = BOTH, expand = 1)


if __name__ == "__main__":

    master = Tk()
    s = Circle(master)
master.title("Circle")
master.geometry("50x350")
mainloop()
```

Output:



PROGRAM 6.2:

Write a Python program to demonstrate the usage of Method Resolution Order (MRO) in multiple levels ofInheritances.

Source Code:

```
class GrandParent:
   def display(self):
print("I am a class GrandParent")

class Parent(GrandParent):
   def display(self):
      #super().display()
print("I am a class Parent")

class Child(Parent):
   def display(self):
      #super().display()
print("I am a class Child")

c = Child()
c.display()
```

Output:

```
I am a class Child
```

PROGRAM 6.3:

Write a python code to read a phone number and email-id from the user and validate it for correctness.

Source Code:

```
import re
def validate_phone(phone):
   if re.match(r"[0-9]{10}",phone):
print("Valid Phone Number")
   else:
print("Invalid Phone Number")

def validate_email(email):
   if re.match(r"[^@]+@[^@]+\.[^@]+", email):
print("Valid email")
   else:
print("Invalid email")
```

```
ph = "7569696504"
email = "mady@gmail.com"

#For Dynamic values
#ph = input("Enter a phone number")
#email = input("Enter email")

validate_phone(ph)
validate_email(email)
```

Output:

```
Valid Phone Number
Valid email
```

Week- 7

PROGRAM 7.1:

Write a Python code to merge two given file contents into a third file.

Source Code:

```
fp1=open("f1.txt","r")
fp2=open("f2.txt","r")
x=fp1.read()
y=fp2.read()
fp3=open("f3.txt","w")
fp3.write(x)
fp3.write(y)
fp1.close()
fp2.close()
fp3.close()
```

Output:

```
f1= Python
f2=Programming
f3=Python Programming
```

PROGRAM 7.2:

Write a Python code to open a given file and construct a function to check for given words present in it and display on found.

Source Code:

```
fp=open("f1.txt","r")
data=str(fp.read())
print(data)
def found(y):
   if y in data:
      print("Found")
   else:
print("Not found")
y=input("enter a string")
found(y)
```

Output:

```
Data in words[“hi”,”students”]
enter a string: hi
Found
```

PROGRAM 6.3:

Write a Python code to Read text from a text file, find the word with most number of occurrences

Source Code:

```
fp=open("words.txt","r")
x=fp.read()
y=x.split()
print("Data in words:\n",y)
z={}
for i in y:
   z[i]=y.count(i)
print("words in a given file with occurances:\n",z)
   res=max(z.key=z.get)
print("\n most number of occurance word in given list is:",res)
```

Output:

```
Data in words
[‘hi’,’hi’,’students’]
words in a given file with occurances:
[‘hi’:2,’students’:1]
most number of occurance word in given list is:hi
```

PROGRAM 6.4:

Write a function that reads a file file1 and displays the number of words, number of vowels, blank spaces, lowercase letters and uppercase letters.

Source Code:

```
import re
def file_operations():
fp = open("f1.txt","r")
   x = fp.read()
   y = str(x.split())
print("Data in a file")

pws = "\s+"
   words = re.findall(pws,y)
print("No. of words:",len(words)+1)
print("No. of whitespaces:",len(words))

pv = "[aeiouAEIOU]"
   vowels = re.findall(pv,y)
print("No. of vowels:",len(vowels))
```

```
pwr = "[a-z]"
lowr = re.findall(pwr,y)
print("No. of lowercase:",len(lowr))
pupr = "[A-Z]"
uppr = re.findall(pupr,y)
print("No. of uppercase:",len(uppr))

    return

file_operations()
```

Output:

```
Data in a file
No. of words:1
No. of whitespaces:0
No. of vowels:4
No. of lowercase:8
No. of uppercase:6
```

Week - 8:

PROGRAM 8.4:

Import numpy, Plotpy and Scipy and explore their functionalities.

Source Code:

```
#NumPy Array All Functions
#1.NumPy Array Creation Functions
import numpy as np

# create an array using np.array()
array1 = np.array([1, 3, 5])
print("np.array():\n", array1)

# create an array filled with zeros using np.zeros()
array2 = np.zeros((3, 3))
print("\nnp.zeros():\n", array2)

# create an array filled with ones using np.ones()
array3 = np.ones((2, 4))
print("\nnp.ones():\n", array3)

#2. numpy array manipulation functions
# create a 1D array
array1 = np.array([1, 3, 5, 7, 9, 11])

# reshape the 1D array into a 2D array
array2 = np.reshape(array1, (2, 3))

# transpose the 2D array
array3 = np.transpose(array2)

print("Original array:\n", array1)
print("\nReshaped array:\n", array2)
print("\nTransposed array:\n", array3)

#3. numpy array mathematical functions
# create two arrays
array1 = np.array([1, 2, 3, 4, 5])
array2 = np.array([4, 9, 16, 25, 36])

# add the two arrays element-wise
arr_sum = np.add(array1, array2)

# subtract the array2 from array1 element-wise
arr_diff = np.subtract(array1, array2)
```

```
# compute square root of array2 element-wise
arr_sqrt = np.sqrt(array2)

print("\nSum of arrays:\n", arr_sum)
print("\nDifference of arrays:\n", arr_diff)
print("\nSquare root of first array:\n", arr_sqrt)

#4. numpy array statistical functions
# create a numpy array
marks = np.array([76, 78, 81, 66, 85])

# compute the mean of marks
mean_marks = np.mean(marks)
print("Mean:",mean_marks)

# compute the median of marks
median_marks = np.median(marks)
print("Median:",median_marks)

# find the minimum and maximum marks
min_marks = np.min(marks)
print("Minimum marks:", min_marks)

max_marks = np.max(marks)
print("Maximum marks:", max_marks)

#5. numpy Array Input/Output Functions
# create an array
array1 = np.array([[1, 3, 5], [2, 4, 6]])

# save the array to a text file
np.savetxt('data.txt', array1)

# load the data from the text file
loaded_data = np.loadtxt('array1.txt')

# print the loaded data
print(loaded_data)

#Scipy and plot functions
import numpy as np
from scipy.stats import multivariate_normal, norm
import matplotlib.pyplot as plt

mean = [0, 0]            # zero mean
cov = [[1, 0.8],[0.8, 1]] # covariance matrix
X1 = np.random.default_rng().multivariate_normal(mean, cov, 5000)
X2 = multivariate_normal.rvs(mean, cov, 5000)
```

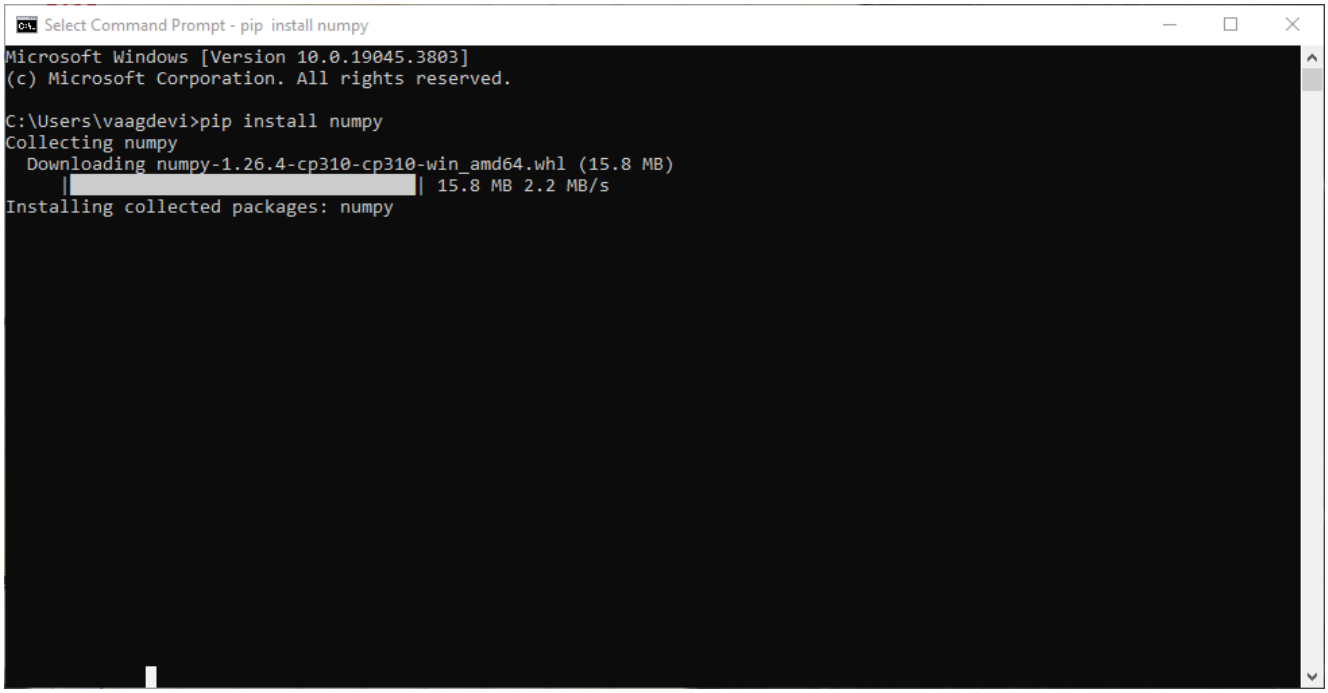
```
fig = plt.figure(figsize=(12,6))
ax = plt.subplot(121)
ax.scatter(X1[:,0], X1[:,1], s=1)
ax.set_xlim([-4,4])
ax.set_ylim([-4,4])
ax.set_title("NumPy")

ax = plt.subplot(122)
ax.scatter(X2[:,0], X2[:,1], s=1)
ax.set_xlim([-4,4])
ax.set_ylim([-4,4])
ax.set_title("SciPy")

plt.show()
n = norm.cdf([1,2,3,-1,-2,-3])
print(n)
print(n[:3] - n[-3:])

print(norm.ppf(0.99))
```

Output:

```
np.array():
 [1 3 5]

np.zeros():
 [[0. 0. 0.]
 [0. 0. 0.]
 [0. 0. 0.]]

np.ones():
 [[1. 1. 1. 1.]
 [1. 1. 1. 1.]]

Original array:
 [ 1  3  5  7  9 11]

Reshaped array:
 [[ 1  3  5]
 [ 7  9 11]]

Transposed array:
 [[ 1  7]
 [ 3  9]
 [ 5 11]]

Sum of arrays:
 [ 5 11 19 29 41]

Difference of arrays:
 [ -3  -7 -13 -21 -31]

Square root of first array:
 [2. 3. 4. 5. 6.]

Mean: 77.2
Median: 78.0
Minimum marks: 66
Maximum marks: 85
Data in data.txt file :
  [[1. 3. 5.]
 [2. 4. 6.]]
```

PROGRAM 8.2:

a) Install NumPy package with pip and explore it.



Source Code:

```
#NumPy Array All Functions
#1.NumPy Array Creation Functions
import numpy as np

# create an array using np.array()
array1 = np.array([1, 3, 5])
print("np.array():\n", array1)

# create an array filled with zeros using np.zeros()
array2 = np.zeros((3, 3))
print("\nnp.zeros():\n", array2)

# create an array filled with ones using np.ones()
array3 = np.ones((2, 4))
print("\nnp.ones():\n", array3)
```

Output:

```
np.array():
 [1 3 5]

np.zeros():
 [[0. 0. 0.]
 [0. 0. 0.]
 [0. 0. 0.]]

np.ones():
 [[1. 1. 1. 1.]
 [1. 1. 1. 1.]]
```

PROGRAM 8.3:

Write a program to implement Digital Logic Gates – AND, OR, NOT, EX-OR

Source Code:

```
def AND(a,b):
    return a and b

def OR(a,b):
    return a or b

def NOT(a):
    return not a

def EX_OR(a,b):
    if a != b:
        return True
    else:
        return False

print("and: ",AND(0,1))
print("or: ",OR(1,0))
print("not: ",NOT(0))
print("Ex_Or: ",EX_OR(0,0))
```

Output:

```
and:  0
or:  1
not:  True
Ex_Or:  False
```

PROGRAM 8.4:

Write a program to implement Half Adder, Full Adder, and Parallel Adder

Source Code:

```
# Function to print sum and carry for Half adder
# sum = A XOR B
# carry = A AND B
def halfAdder(A, B):
    sum = A ^ B
    carry = A & B

print("Half Adder:")
print("Sum ", sum)
print("Carry", carry)
    return

def parallelAdder(A, B, C, D):
    sum = (A ^ B) & (C ^ D)
    carry = (A & B) & (C & D)
print("Parallel Adder:")
print("Sum ", sum)
print("Carry", carry)
    return

# Function to print sum and C-Out
def fullAdder(A, B, C):
    sum = C ^ (A ^ B)
c_Out = (A & B) & (A ^ B)
print("Full Adder:")
print("sum = ", sum)
print("c_out = ", c_Out)
    return

A = 0
B = 0
C = 1
D = 1
halfAdder(A,B)
parallelAdder(A,B,C,D)
fullAdder(A, B, C)
```

Output:

```
Half Adder:
Sum  0
Carry 0
Parallel Adder:
Sum  0
Carry 0
Full Adder:
sum =  1
c_out =  0
```

PROGRAM 8.5:

Write a GUI program to create a window wizard having two text labels,two text fields and two buttons as Submit and Reset.

Source Code:

```
from tkinter import *
from tkinter import messagebox

master = Tk()

usr = Label(master, text = "Username").grid(column = 0, row = 0)
tfu = Entry(master, text = "").grid(column = 1, row = 0)
pwd = Label(master, text = "Password").grid(column = 0, row = 1)
tfp = Entry(master, text = "").grid(column = 1, row = 1)
def submitted():
messagebox.showinfo("Message","Submitted Successfully")

def reset():
messagebox.showinfo("Message","Reset Successful")

submit = Button(master, text = "Submit", command = submitted).grid(column = 0,row = 2)
reset = Button(master, text = "Reset", command = reset).grid(column = 1,row = 2)

master.mainloop()
```

<u>Output:</u>