☆☆ INDEX ☆☆

No.	Title	Page No.	Date	Staff Member's Signature
1)	Demonstrate the use of different file accessing mode, different attributes read method.		25/1/19	
2)	To display element of tuple using the iterator method.		2/12/19	
3)	Do demonstrate exep exception handling.		9/12/19	
4)	Regular Expression.		16/12/19	incomplete

Parent's Sign :

Contact No :

Reason for Absence :

Mithilesh Jha
9987424184

☆☆ INDEX ☆☆

No.	Title	Page No.	Date	Staff Member's Signature
5)	program to make use of GUI application along with basic pack method		27/1/20	
6)	TO make use of message box and traverse windows			

PRACTICAL 1

Objective:- Demonstrate the use of different file accessing mode, different attribute read method.

Step 01 : Create a file object using open method and use the write access mode followed by writing some contents onto the file and then closing the file.

Step 02: Now open the file in read mode and then use read( ), readline( ) and readlines( ) and store the output in variable and finally display the contents of variable.

Step 03: Now use the file object for finding the name of the file, the file mode in which it is opened whether the file is still open or close and finally the output of the software attribute.

Step 04: Now open the file object in write mode ~~write some~~ and some another content close subsequently then again open the file object in 'w+' mode that is the update mode and write contents

Step 05:- Open the file object in read mode, display the update written contents and close open again in 'r+' mode with parameter passed and display the output subsequently.

```
Fileobj = open ("abc.text", "w") # File open (write mode)
Fileobj.write("computer science subjects" + "\n")
Fileobj.write (" DBMS in python in DS \n") # File write
Fileobj. close () # File close.

Fileobj = open ("abc.text", "r") # read mode
# read ()
  str1 = Fileobj.read ()
Print (" The output of read method: ", str1)
Fileobj.clase()
>>> (" The outputs of read method: ', 'computer science subje
      in DBMS in python in DS \n' )
# readline ()
Fileobj = open ("abc.txt", "r")
str2 = Fileobj.readline ()
print ("The output of readline method :" str2)
Fileobj. close ()
>>> ('The output of readline method : 'computer science
      subject \n' )
# readlines ()
Fileobj = open ("abc.txt", "r")
Str3 = File obj. readlines ()
print ("The output of readlines method :" str3)
   Fileobj. close ()
>>>('The Output of readlines method :' ["computer science
   subjects \n, ' DBMS \n' 'python \n' 'DS \n']
```

```
# File attributes
a= Fileobj.name
print ("name of file (name attribute): ", a)
>>> (name of file (name attribute): 'abc.txt')
b = Fileobj.closed
 print ("(close) attribute: ", b)
 >>> ('(close) attribute: ' (True)
 c= Filobj.mode.
 print ("File mode",c)
 >>> ($file mode; 'r')
 d= Fileobj. softspace
 print ("softspace",d)
 >>> ('softspace:', 0)

 # w+ mode
   Fileobj = Open ("abc.txt", "w+")
   fite str1 = Fileobj.read (5)
   print ("output of r+", str1)
    Fileobj. close()
   >>> (output of r+, '$Abhishek')

# write mode
   Fileobjob =open ("abc.txt", "w")
   Fileobj. write ("bBMJ")
   Fileobj. close ()
# read mode.
   Fileobj = open ("abc.txt","r")
    str2 = Fileobj.read ()
    Print ("output of read mode !", str2)
  >>> ('output of read mode !' 'Abhishek')
```

Step-6: Now open fileobject in appened mode open write method write content close the fileobject again. open the fileObject in read mode and display the appened output.

Step 7: Open the fileobject in read mode. declare a variable and perform file object in read mode. declare a variable and perform fileObject dot tell mell method and store the output consequently in variable.

Step 8: use the seek method with the arguments with opening the fileobject in read mode and closing subsequently.

Step 9: Open fileObject with read mode also, use the readlines method and store the output consequently in and print the same for counting the length use the for condition statement and display the length.

```
# append mode
fileobj = open ("abc.txt", "a")
fileobj.write ("data structure")
fileobj.close ()
fileobj = open ("abc.txt","r")
str3 = fileobj.read ()
print ("output of append mode :", str3)
fileobj.close ()
>>> (output of appent mode:', ('Abhishek', 'data structure')
```

```
# tell ()
fileobj = open ("abc.txt", "r")
pos = fileobj.tell ()
print ("tell (): ", pos)
fileobj.close
>>> ('tell (): ', pos)
```

```
# seek ()
fileobj = open("abc.txt", "r")
str4 = fileobj.seek(0,0)
str8 = fileobj.read(10)
print ("The beginning of the file:", str8)
```

last 31

```
# finding length of different lines exist within lines
fileobj = open ("abc.txt," "r")
str9 = fileobj.readlines ()
print ("output:", str9)
for line in str9:
        print (len(line))
    fileobj.close ()
>>>('output:', ['college database'])
```

PRACTICAL 2

1) Aim: To display elements of tuple using the iterator method.

Algorithm:

Step 1: Form a tuple by and containing some elements in it.

Step 2: use the iter method followed by the next method to iterate the varicble.

Step 3: print the elements.

2) Aim: To use the iterator method with for loop.

Algorithm:

Step 1: form a tuple containing elements in it

Step 2: use the for conditional statements to iter the elements

Step 3: print the elements.

3) Aim: To print odd numbers using iterator method.

Algorithm:

Step 1: Define a class and within that define a iter method which will iterating the first

Variable within the containg object.

Step 2: Now use the next method and defi... by cla... the large for displaying the odd value

Step 3: Creat an object to print the values

4) Aim:- using iterate display set of the values.

4) Aim:- using iterate display set of first 2 numbers

Step 1: Define a iter method with a argum and iterating ~~in~~ to a first value.

Step 2: for ~~it~~ extracting the next element from the container use the next method with an argument and compare of elements recieved in a object from the given class and pass the object as an argument to the iter method.

Step 4: Now using the conditional statem display all the value.

3) class odd:

```
    def __iter()__(self):
        self.num = 1
        return self
    def __next__(self):
        num = self.num
        self.num = 2
        return num
myobj = odd()
myiter = iter(myobj)
a = int(input("enter no"))
for x in myiter:
    if x<a
        print x.
```

output : enter no = 20

1
3
5
7
9
11
13
15
17
19

4) class myclass :

```
def -- iter -- (self) :
    self.a = 1
    return self
def -- next --(self) :
    if self a< =20
      x = self .a
       self .a+=1
      return x
      else.
      raise Stop iteration

myobject = myclass ()
  myiter = iter (myobject)
  for x in myiter
       print ()
```

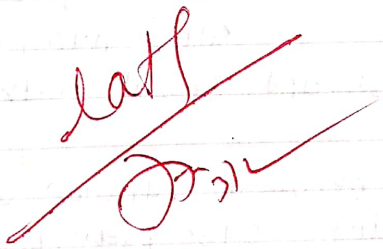
output :

```
1  11
2  12
3  13
4  14
5  15
6  16
7  17
8  18
9  19
10 20
```

5) Aim: To find the first 10 numbers factorial using iter method.

Algorithm:

Step 1



PRACTICAL 3

TO Demonstrate exception handling.

1) Aim:- TO find the number is even or ~~odd~~ odd using map method.

Algorithm:-

Step 1) Take a ~~first~~ list datatype of same numbers by user by using list and input method subsequently.

Step 2: Define a function with a single argume and subsequently use if statement. return even when the parameters is modular by 2 or else return odd.

Step 3. By using map method map the input list datatype with given function ~~and~~ and convert it into list datatype. and display it using print statements

2) Aim: TO find square and cube of a number in a list using map method

Algorithm:-

Step 1 : Take input of numbers by user and convert it into list datatype.

1) List 1 = list (input ("enter:"))

```
define even (x):
    if x % 2 == 0
    return ("even")
else.
    return ("odd")
print list (map(even, list))
```

output.

enter : 0, 4, 5, 7, 9, 11, 15, 20, 25

['even', 'even', 'odd', 'odd', 'odd', 'odd', "even
'odd']

9)
```
list 1 = list (input ("enter "))
def square (x):
    return x ** 2
def cube (x):
    return x ** 3
funct = [square, cube]
for i. in range listl:
    valueout = map (lambda, x: x(i) funct
print (list (valueout))
```

output: enter : 1,2,3,4, 5,6

[1,1]

[4,8]

[9,27]

[25, 125]

[36, 216]

Step 2: define a function for square using single parameters which will return (2**2) square of given data by the user. simultaneously, again define a function for cube using single parameter which return (***3)

Step 3: In list take square and cube and assign it to a variable.

Step 3: use for statement for i in list given by users. by using map method with lambda function using appropriate logic for mapping the given input with the define function.

Step 4: After mapping convert it into list datatype and display it using print method.

3) Aim: To display the list of the square of numbers using appened method.

Algorithm:-

Step 1: take input of number by user and convert it into list datatype.

Step 2: assign a variable with empty list

Step 3: use for statement for a loop variable i in list of given numbers subsequently use the append method using appropriate logic of to square the given user list by appending it in the empty list and display it using print statement.

3)

3) list = list (input ("enter:"))
list 2 = [ ]
for i in list 1:
    list 2 - appened (i **2)
print list 2.

output : enter : 1, 4, 2, 5, 8, 8, 12, 9

[1, 16, 4, 25, 64, 144, 81]

PRACTICAL NO:3

Aim:- programs to demonstrate exception handling

1> programs to demonstrate the use of I/O error.

Step 1:- use the try block to define the normal course of action eg:- define the file object and open the file in the write or read mode and write convert onto the file.

STEP 2:-

use the except block with the I/O error as an environment errors and convey the appropriate message to the user else display the me

PRACTICAL -4

Topic : regular expression.

STEP 01 : Import the module declare pattern and declare sequence use match method with declare arguments if arguments matched than print the same ortherwise print pattern NOT FOUND

STEP 02 : Import re module declare pattern with literal and meta character. declare String value use the window () with argume and print the same.

STEP 3 : Import re - module declare pattern with meta character use the split () and print the output.

STEP 4 : Import re - module declare String and accordingly decare pattern replace the blank space with no - space use sub() with 3 arguments and print the string without spaces.

STEP 5 : Import re module declare a sequen use search method for finding seq subsequent() use the group() with dot operator as search () guess memory co-relation

```
# match ()
import re
pattern = r "fycs"
Sequence = "fycs represents computer science stream"
if re. match ( pattern, sequence);
        print ( " matched pattern found!")
else :
   print ("Not FOUND!")
  >>> matched pattern found !
# numerical values (segration)
 import re
 pattern = r'\d+'
 string = 'hellow 123, howdy 789, 45 howra.'
  output = re. findall (pattern , string)
  print (output)
  >>> ['123', '789', '45']
# split()
 import re
  pattern = r'\d+'
 string = 'hellow 123, howdy 789 ,45 howra.
  output = re. split (pattern , string)
  print (output)
>>> ['hellow', ' howdy', ' ', ' howra']
```

```
# no. space:
import re
string = 'abc def gwi'
pattern = r'\s+'
replace = ''
v1 = re.sub(pattern, replace, string)
print(v1)
>>> abc defgwi
# group()
import re
sequence = 'python is an interesting language.'
v = re.search('\Apython', sequence)
print(v)
v1 = v.group()
print(v1)
>>> <_sre.SRE_Match object at 0x0281bf00>
    python
# verifying the given set of phone numbers
import re
list1 = ['8004567891', '+41 45 673210', '+1 786 543248',
         '9876543201']
for value in list1:
if re.match(r'[6-9]{1}[0-9]{9}',
value or len(value) == 10):
print("criteria matched for all numbers")
else:
print("criteria failed !!")
```

using group() it will show up the matched string.

STEP 6: Import re-module delare list with numbers. use the conditional statement here all have used up the for conditional statement. use if condition statement for checking first number is either 4 or 9 and check whether the entered numbers are equal to 10. if criteria matches print ceu numbers matches otherwise print failed.

STEP 7: Import re-module declare a string use the module with findall() for finding the values in the string and declare the same.

STEP:8 Import re module declare the host and domain name declare pattern for seprating the host & domain name. use the window() and print the output respectively.

STEP:9 Import re module enter a string use pattern to display only the elements of the particular String use window() delare two variables with initial value as zero we for conditions and subsequen use the if condition check whether condition satisfy add up the or else increment value. And display the values subsequently.

```
>>> criteria matched for all number
    criteria matched for all number
     criteria failed.
      criteria matched for all number

# values
    import re
    str1 = 'plant is life overcall'
    output = re.findall(r'\b[qei ouA E7oU]\w+', str1)
    print (output)
   ??? [' is', 'overcall']
  # host & domain
    import re
    seq = 'abc.tesc@ed.com, xyz@gmail.com'
    pattern = r'[\w\.-]+[\w\.-]'
     outpw = re.findaw findall (pattern, seq)
    print (output)
    >>> ['abc.tesc', 'edu.com', 'xyz', 'gmail.com']

# counting of first 2 letters.
 import re
    s = 'mr.a, ms.b, ms.c, mr.t'
    p = r'\ms |mr\]+
    o = re.findall (p,s)
    print(o)
      m=0
      f=0
      for v in o:
```

```
if (u == 'ms'):
    f = f+1
else
    m = m+1
print ("No of males is:", m)
print ("No of females is:", f)
>>> ('mr', 'ms', 'ms', 'mr')
('No. of males is:', 2)
('No. of females is:', 2)
```

Scanned with CamScanner

PRACTICAL NO: 5

Aim: To make use of scroll bar widget of the GUI application.

Algorithm:-

STEP 1 : Imoport tkinter Library to use scroll Bar widget

STEP 2 : Create on object corresponding to scroll parent window of create on object From scroll bar & place it on the parent window so created.

STEP 3: Create on object of Label method to provide a heading and place it on parent window.

STEP 4:- use pack method along with object of scroll bar method and use argument side & fill.

STEP 5:- Create on object of ListBox method and place it onto parent window with attribute / command.

STEP 6: Use for loop to insert values in the object of list box by using insert method.

STEP 7:- use config method along with scroll Bar object and use command attribute.

STEP 8:- finally call the mainloop method.

Program :-

```
from tkinter import *
root = TK()
root.geometry ('450x400')
L = Label (root, text = (B Batch roll numbers:", bg = "black",
                                                  Fg = "blue")
L.pack ()
scroll = scrollbar (root)
scroll.pack (side = RIGHT, Fill = Y)
mylist = listBox (root, yscroll command = scroll.set, bg = "Light
                                                          Blue")
for num in range (41, 81):
mylist.insert (END, "* Roll Number : 17", str (num))
mylist.pack (side = LEFT, Fill = BOTH)
scroll. config (command = mylist.gview)
root.mainloop ()
```

Output :

bg = black Fg = White

tk

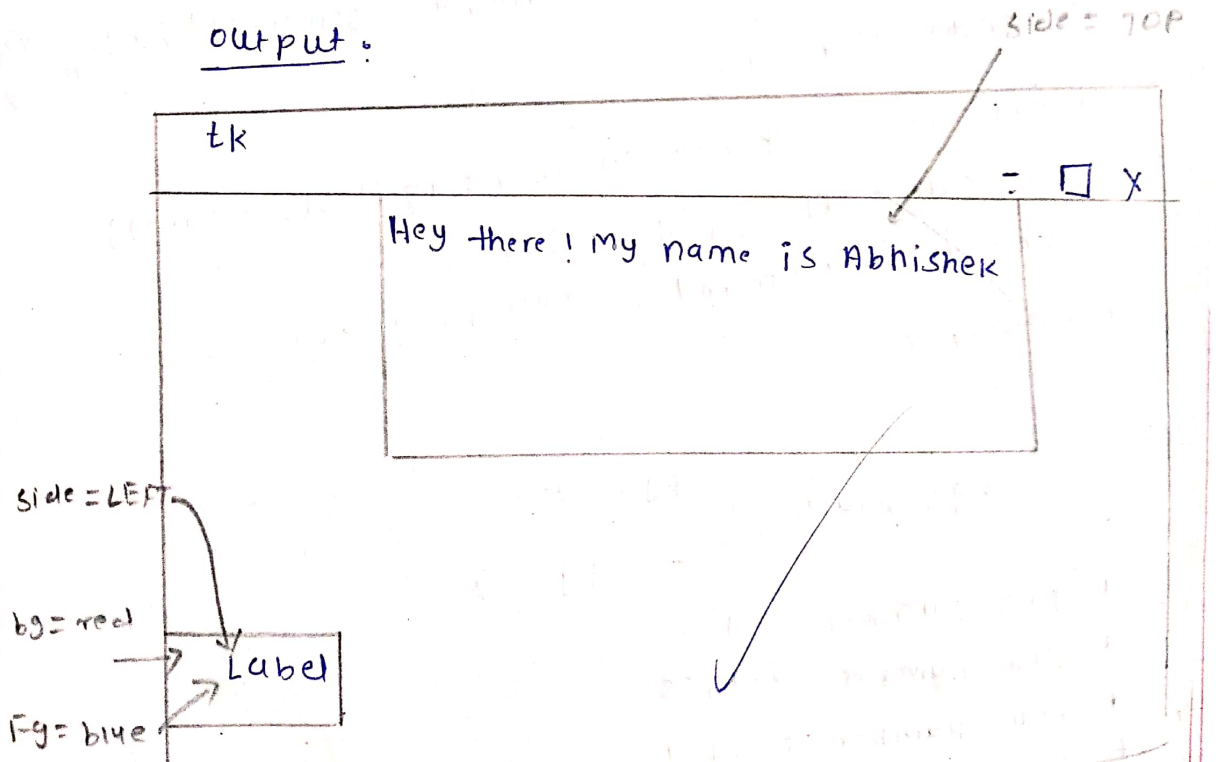
* Roll Number : 1752
* Roll number : 1753
* Roll Number : 1754
* Roll Number : 1755
* Roll Number : 1756

Bg = light Blue.

PROGRAM :-

```
From tkinter import *
root = TK ( )
T1 = Text(root)
T1.insert(END, "Hey There ! my name is Abhishek.")
T1.pack(side = TOP, padx = 20, pady = 30, ipadx = 40, ipady
L1 = Label(root, text = "Label", bg = "red", fg = "blue")
L1.pack (side = LEFT, pady = 10, ipadx = 20, ipady = 30)
root.mainloop ( )
```

output :-



Aim : To make use of GUI application along with the basic pack method.

Algorithm:-

STEP 1:- Use the tkinter library for importing the features of text widget.

STEP 2:- Create a variable from a text variable of position it onto the parent window.

STEP 3:- Use the pack() along with the object created from text method and use the parameter.

i) side = TOP , padx =20 , ipadx = 40, ipady =50

STEP 4:- Use the mainloop method for triggering corresponding event.

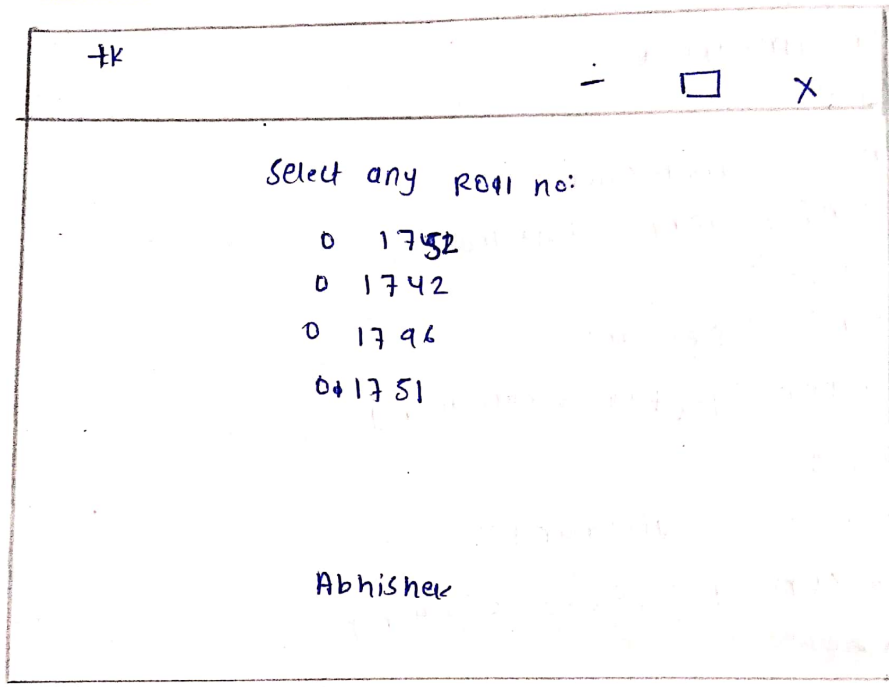
STEP 5:- Now repeat above step with a label method which takes the following argument.

i) Name of parent window
ii) Text attribute which defines the string.
iii) The background colour (bg)
iv) The Foreground colour (Fg)
v) Now use pack() with relevant attributes.

PROGRAM:-

```
From tkinter import *
def sel 1 ( ):
    Selection = "Abhishek"
    Label . config (text = selection)
def sel 2 ( ):
    Selection = "Deepu"
    label . config (text = selection)
def sel 3 ( ):
    Selection = " Vikrant"
    Label. config ( text = selection)
def sel 4 ( ):
    Selection = "sachin"
    label = . config (text = selection)
root = Tk ( )
root = Intvar ( )
L1 = Label (root, text = "select any root number")
L1 = . pack (side = TOP)
R1 = Radiobutton (root, text = "1752"; variable = var, value = 0,
                                                command = sel)
R1. pack (anchor = N)
R2 = Radiobutton (root, text = "1742"; variable = var, value = 1,
                                                command = sel 2)
R3 = Radiobutton (root, text = "1796"; variable = var, value = 2,
                                                command = sel 3)
R3. pack ( anchor = N)
R4 = Radiobutton (root, text = "1751"; variable = var, value = 3, comm = sel4
R4. pack (anchor = N)
label = Label (root)
label. pack ( side = BOTTOM)
root mainloop ( )
```

Output:-



Aim:- To make use PRACTICAL – 6

~~To~~ Aim:- To make use of messagebox method of the GUI application.

Algorithm:-

STEP 1:- Import relevant method from tkinter library

STEP 2:- Define a function and use messagebox along with different methods available which one or more arguments.

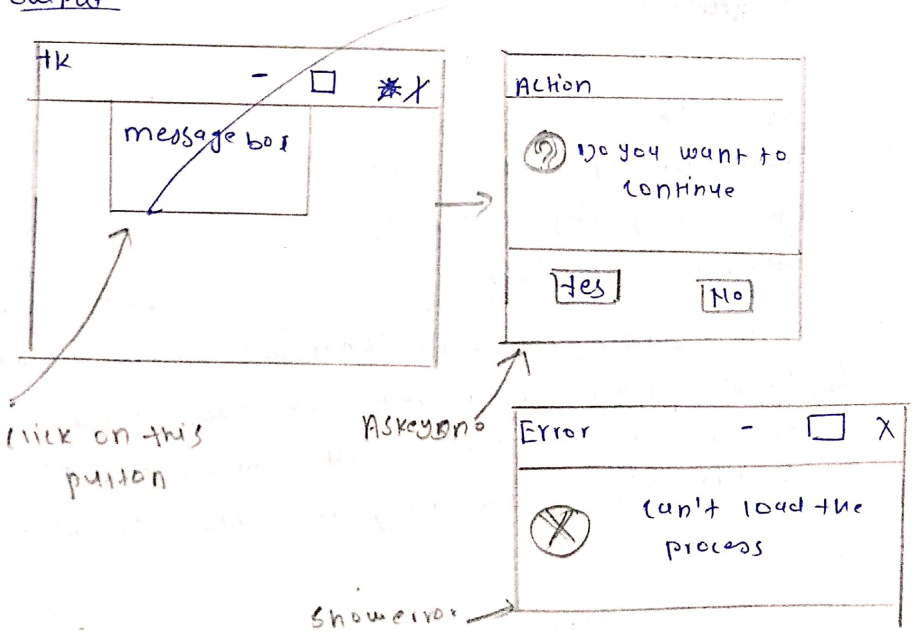
STEP 3:- create an object from button method and place it onto the parent window with text and command attributes specified.

STEP 4:- use pack and finally use the mainloop method.

Program :

```
From tkinter import*
From tkinter import messagebox
def msgbox( ):
    messagebox.showaskerror("Action","Do you want to contin")
    messagebox.showerror("Error","can't load the process!")
root = Tk( )
root.config( bg="grey")
B1 = Button(root, text ="messagebox", bg="light blue", comm-
                                        = msgbox)
B1.pack( )
root.mainloop( )
```

Output :



PROGRAM:-

```
from tkinter import *
def main():
    root = Tk()
    root.geometry("450x500")
    root.config(bg="Light green")
    root.title("window 1")
B1 = Button(root, text="Next", command=main1)
B1.grid(ipadx=50, ipady=40, padx=20, pady=30)
B2 = Button(root, text="exit", command=term)
B2.grid(ipadx=50, ipady=40, padx=20, pady=30)
def term():
    quit()
tos = Tk()
tos.geometry("450x500")
tos.config(bg="purple")
tos.title("main window")
B3 = Button(tos, text="On Hing", command=main)
B3.grid(ipadx=50, ipady=40, padx=20, pady=30)
B2 = Button(tos, text="Exit", command=term)
B2.grid(ipadx=50, ipady=40, padx=20, pady=30)
```

Aim:- program to traverse various windows using the button widget.

Algorithm:-

STEP 1 : Import the relevant method from tkinter library.

STEP 2:- Define a function and create a object of given window by using the three methods namely config, title, size.

STEP3:- Define a button object which will be placed on the corrent window to traverse and define another button which will be used to exit from the window and place it onto corrent window.

STEP 4:- Define another function which will use the quit method to terminate the program.

STEP 5:- Now create an object of main window and use various methods like config, title, geometry etc

STEP 6:- Define two buttons which will be placed on the main window, one to traverse another window and the other to terminate the program.

STEP 7 : Define another function which will carry various button placed on third window. Define two buttons respectively and use the grid method along with the two buttons.

STEP 8:- finally call the mainloop method.

```
def main():
    top = Tk()
    top.geometry("450x500")
    top.config(bg="purple")
    top.title("window 2")
    By = Button(top, text="main page", command=main)
    By.grid(ipadx=50, ipady=10, padx=20, pady=30)
    B2 = Button(top, text="Exit", command=term)
    B2.grid(ipadx=50, ipady=10, padx=20, pady=30)
    mainloop()
```

output: