☆☆ INDEX ☆☆

No.	Title	Page No.	Date	Staff Member's Signature
1.	Demonstrate the use of different file accessing mode, attributes and 'read' methods.	19	29.11.19	
2.	Demonstrate the use of different file accessing iterators and iterables	22	6.12.19	
3.	Demonstrate the exceptional handling in python.	25	20.12.19	
4.	Demonstrate the use of regular expressions in python.	28	3.12.19	
5.	GUI Components :	33	14.2.20	
	• Demonstrate tkinter			
	• Radio Button			
	• ScrollBar Widget			
	• Frame Widget			
	• Message box			
	• Button Attributes			
	• Window Traversal			
	• Image Insertion			

TITLE : Demonstrate the use of different file accessing modes, attributes and 'read' methods.

Algorithm: (i) Create a file object using open() and use the write access mode followed by writing some contents onto the file and then close the file.

(ii) Now open the file in read mode and then use the read(), readline() and readlines(). Store the output in variable and finally display the contents of the variable.

(iii) Now use the file object for finding the name of the file and the file mode in which it is opened, whether the file is still open or close and finally the output of the softspace attribute.

(iv) Now open the file object in write mode and rewrite content onto the file subsequently.
Reopen the file in 'w+' mode which is also known as update mode and write content onto the file.

(v) Open file object in read mode, display the update written content and close. Open ope again in 'r+' mode within parameter passed and display the output subsequently.

(vi) Now open file object in append mode. Open write method write contents onto the file. Close the file and reopen it in read mode to display the 'appended' content as output.

(vii) Declare a ~~Open~~ variable and store the file objects onto it so as to demonstrate the '.tell()' on the output screen.

(viii) With the file open in read mode, use the 'seek()' with appropriate arguments.

(ix) While the file is open in read mode, use the readlines() to store the output of the traversed objects onto a variable. Print the contents of the variable onto the screen.

(x) use the length() to count the number of characters in the variable and print it onto the output screen.

Program :

```
// Demonstrating read() & write() :
fileobj = open ("SAMPLE.txt ", "w")
fileobj. write ("File contents are : " + "\n")
fileobj. close()
fileobj= open ("SAMPLE. txt ", "r")
Line 1 = fileobj. read ()
fileobj. close ()
print (line1)

// Demonstrating append "a" mode / readline() :
fileobj = open ("SAMPLE.txt", "a")
fileobj. write ("\n This statement is printing using 'a' mode")
fileobj. close()
fileobj = open ("SAMPLE.txt", "r")
line2 = fileobj. readline (15)
fileobj. close ()
print (line2)

// Demonstrating seek() and tell () :
fileobj = open ("SAMPLE.txt ", "r")
pos = fileobj. tell ()
print ("\n Position of tell() is: ", pos)
seeker1 = fileobj. seek (50, 0)
print ("\n File contents after using seek() are :", seeker1)
fileobj. close()
```

Output: // Implementing read() & write()

File contents are:

// Implementing append mode

This statement is printing using the 'a' mode.

// Implementing readline(13)

File contents

// Implementing tell() & seek()

The position of tell() is 0

File contents after using seek() are: using the 'a' mode.

Output: (on screen)

File contents are:

File contents

The position of tell() is 0

File contents after using seek() are: using the 'a' mode

incomplete

6/12/17

PRACTICAL NO. 2

Aim: To demonstrate the use of iterator and iterables.

Algorithm: (i) Define a iter method with an argument and initialize the value and return that value

(ii) Define the next method with an arguement and compare the upper limit by using a condn. statement.

(iii) Now create an object of the given class and pass the object in the iter method.

// Demonstrating seek():

```
OS
seeker 2 = fileobj.seek (0, 1)
```

Output:

File contents are:
File contents
File contents after using seek are: File contents are:
This statement is printing using the 'a' mode.

// Demonstrating seek():

```
seeker 3 = fileobj.seek (0,2)
```

Output:

File contents are:
File contents
The position of tell() is : 0
File contents after using seek are:

6/12/19

Algorithm : (Odd Numbers)

(i) Define a iter method with an argument and individual initialize the value and return the value.

(ii) Define the next method with a argument and compare the upper limit by using a conditional statement.
Increment the value by 2.

(iii) Create an object of the given class and pass this object in the iter method.

Source code: (Range)

```
def __iter__(self):
    self.a = 1
    return self
def __next__(self):
    if self.a <= 15:
        x = self.a
        self.a += 1
        return x
    else:
        stopIteration
Myclass = myrange()
mydata = iter(myclass)
for x in mydata:
    print x
```

Output:

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

Algorithm: (Power)

(i) Define iter method with 3 arguments. Initialize the first argument as and set variable value as 1.

(ii) Accept an input value from the user for the base number as well as the limit value of its power respectively.

(iii) Define the next method with argument and compare it by using an conditional statement. increment the value by 1.

(iv) Create an object of the given class and pass this object in the iter method.

(v) Print the respective output using the 'while' conditional statement.

Source Code : (Odd Numbers)

```
class odd
    def __iter__(self):
        self.num = 1
        return self
    def __next__(self):
        if self.num <= 0 :
            num = self.num
            self.num += 2
            return num
        else :
            raise StopIteration

x = iter(odd())
y = next()
```

Output :

```
1
3
5
7
9
11
13
15
17
19
```

Source Code:

```
class myiter:
    def __iter__(pow):
        pow.n = 1
        pow.n1 = int(input("Enter a no. : "))
        pow.n2 = int(input("Limit of power: "))
        return pow
    def __next__(pow):
        if pow.n1 = pow.n2:
            num = pow.n1 ** pow.n
            pow.n += 1
            return num
        else:
            raise StopIteration
y = iter(my_iter())
while True:
    print(next(y))
```

Output:

```
Enter a no. : 2
Limit of power: 4

2
4
8
16
```

Aim: Program - to demonstrate exception handling.

Algorithm: (Exception method for Arithematic Error)

(i) Use the try block and accept the input using the raw input method and convert it into the integer datatype and subsiquently terminate the block.

(ii) Use the 'except' block with the exception name as value error and display the appropriate message if the suspecious code is part of the try block.

Algorithm: (Exception method for Environment Error)

(i) Within the try block, open the file using the write mode and write mode and write some content on the file.

(ii) Use the except block with IO errors and display the message regarding missing of the file or incompatibility of the mode. Use the else block to display a message that the operation is carried out successfully.

Q. Write a program demonstrating assert() to check if list elements are empty.

Algorithm: (i) Define a function which accepts an arguements.
Check using the assert() whether the given list is empty list and accordingly return output message.

(ii) Close the function and define elements in the body of the program. Take appropriate action of the list elements.

Source Code : (Arithematic Error)

```
while True
    try :
        x = int(input("Enter class : "))
        break
    except ValueError
        print ("Enter numeric value !")
```

Output :

Enter class : 450

Output 2 :

Enter class : ABC

Enter numeric value!

Source Code : (Environment Error)

```
try :
    fopen
    fileobj = open ("ABC.txt", "w")
    fileobj.write (" Hello would ")
except IOError
    print (" Error writing on file ! ")
else :
    print ("Operation successful ! ")
    fileobj.close()
```

Output :

Operation successful !

Source code :

```
def assert_(u):
    assert(len(u)==0)
    print("List is empty")
var1 = []
print(assert_(var1))
```

Output :

List is empty.

Q. Write a program using the ~~assert~~ range ?? to check age of ~~student~~ in a given class and if the age does not fall in given range raise exception error.

Algorithm: (i) Define a function which will accept the age of the student from the input.

(ii) use the 'if' condition to check whether the ~~iptt~~ input age falls in the range and thus return the age. Else, use the error exception.

(iii) Define the while loop to check whether the boolean expression holds true. Use the try block to accept the age of student and terminate the looping condition.

(iv) Use except with value Error and print the message not a valid range.

Source Code:

```
def acceptage()
    age = int(input("Enter age : "))
    if age > 30 or age < 16 :
        raise ValueError
    return age

valid = False
while not valid:
    try :
        age = acceptage()
        value = True
    except ValueError :
        print ("Not a valid age!")
```

Output :

```
Enter age : 4
Not a valid age
```

Output 2 :

```
Enter age : 18
```

TS

PRACTICAL NO. 4

Aim: Demonstrate the use of regular expression.

Theory: Regular expression represents the sequence of characters which is mainly used for finding and replacing the given pattern in a string and for this we import removable and common usage of regular expression involved following functionalities.

- Searching a given string.
- Finding a string
- Finding a string into smaller string
- Replacing part of a staing

Q. Write a regular expression segregating numeric and alphabetic value from a given string Algorithm.

Algorithm: (i) Now apply string and pattern in findall() and display the respective output.

(ii) Use the \d method for matching all decimal digits whereas is used to match non-decimal digits.

Code:

```
import re
string = "hello 1234abc 456"
result = re.findall("\d", string)
result1= re.findall("\w", string)
print(result)
print(result1)
```

Output:

```
>> ['1234', '4567']
>> ['hello', 'abc']
```

Q. Write a regular expression for finding the matched string at the begining of given sequence.

Algorithm: i) Import the remodule files and apply a string.

(ii) Search two parameters in the given string. Let the two parameters be 'Python' and a string value.

(iii) Once the string parameters are found, print them as output for the user.

(iv) make the use of 'if' conditional statement for user to know whether match is found or not.

8S Code :

```
impout re
steing = "python is an intended language"
result = re.search ["\A python", steing]
print (result)
if result :
    print ("Match found")
else :
    print ("Match NOT found")
```

Output:

```
>> rematch object span = (0, 6)
                match = "python"
>> match found.
```

Q. Write a regular expression to check whether given mobile number starts with 8 or 9 and the total length is 10.

Algorithm:

(i) Import re module and apply a string of mobile number 8.

(ii) Now use 'for' conditional statement to find in the number starts with 8 or 9 and the total number should be of length 10.
Use match() inside for statement to find the match in given string.

(iii) Use 'if' conditional statement to know whether we have to match or not. If we have a match, use the group() to display the output and if we dont, display appropriate output to the user.

Code :

```
import re
li = ["9653177650", "8820547891", "9819224989"]
for element in li :
    result = re.match ("[8-9]- {1} [0-9] {9},"
                       element )
 if result :
      print ("correct mobile no.")
      print (result group (1))
  else :
      print ("Incorrect mobile no.")
```

Output :

» correct mobile no.

Q. Write a regular expression for extracting a word from given string along with space characters in between the word and subsequently extract the word without space between.

Algorithm: (i) Import re module and apply a string.

(ii) Use findall() to extract a word from given string.

(iii) Use "\w*" to extract word along with space and use the "\w+" to extract word without space.

(iv) Display the output to the user.

Code:

```
import re
string = "Python is a indented language"
x1 = re.findall ("\w* ", string)
x2 = re.findall ("\w+", string)
print (x1)
print (x2)
```

Output:

>> ['Python.' ' ', 'is', ' ', 'an', ' ', 'indented', ''
'language',]

>> ['Python', 'is', 'an', 'indented', 'language']

Q. Write a regular expression for extracting the data in format dd-mm-yy by using the findall() where the string has following format

KartikRaj 28-05-1998

Algorithm:

(i) Import the re module and apply a string in given program.

(ii) Use the findall() and use \d{2} - \d{4} as respective parameters in the code.

(iii) Now display the output to the user.

Code :

```
import re
string = "KaushikRaj 28-05-1998"
result = re.findall ("\d{2}-\d{2}-\d{4}",
                     string)
print (result)
```

Output :

```
>> ['28-05-1998']
```

PRACTICAL NO. 5

TOPIC : GUI Components

Q. Create a parent window in python using tkinter.

Algorithm : (i) Use the tkinter library for importing the features of the the text widget.
(ii) Create an object using the Tk().
(iii) Create a variable using the widget label and use the text().
(iv) Use the mainloop() for triggering of the corresponding above mention events.
(v) Ensure appropriate output is observed.

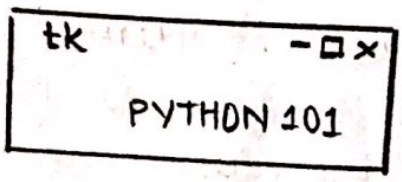
Q. Create a label() and demonstrate padding.

Algorithm : (i) Use the tkinter library for importing the features of the text widgets.
(ii) Create a variable from the text method and position it on the parent window.
(iii) Use the pack() along with padding parameters on the object created from the text().
(iv) Use the mainloop() for triggering the execution of created window application.

Demonstrate tkinter

Code:

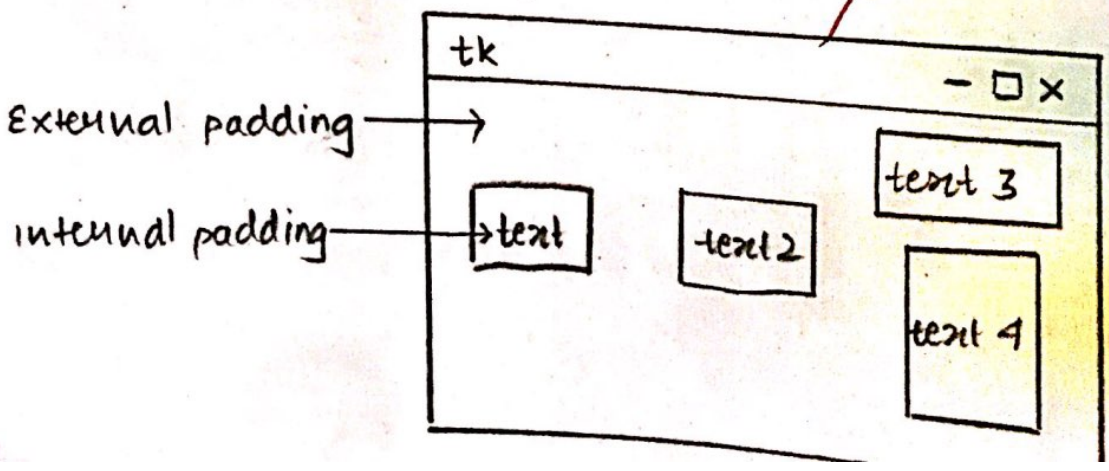
```
from Tkinter import *
root = TK()
label1 = Label (root, text = "Python 101")
label1.pack()
root. mainloop()
```



Demonstrate Label () and padding.

Code:

```
from Tkinter import *
root = TK()
Label1 = Label (root, text = "PYTHON PROGRAMMING")
label1. pack()
label2 = Label (root, text = "FYBSc", font = 10)
label2. pack (side = LEFT, padx = 15, pady = 25)
root. mainloop ()
```



Q. write a program to demonstrate the listbox(), and pad it onto the parent window.

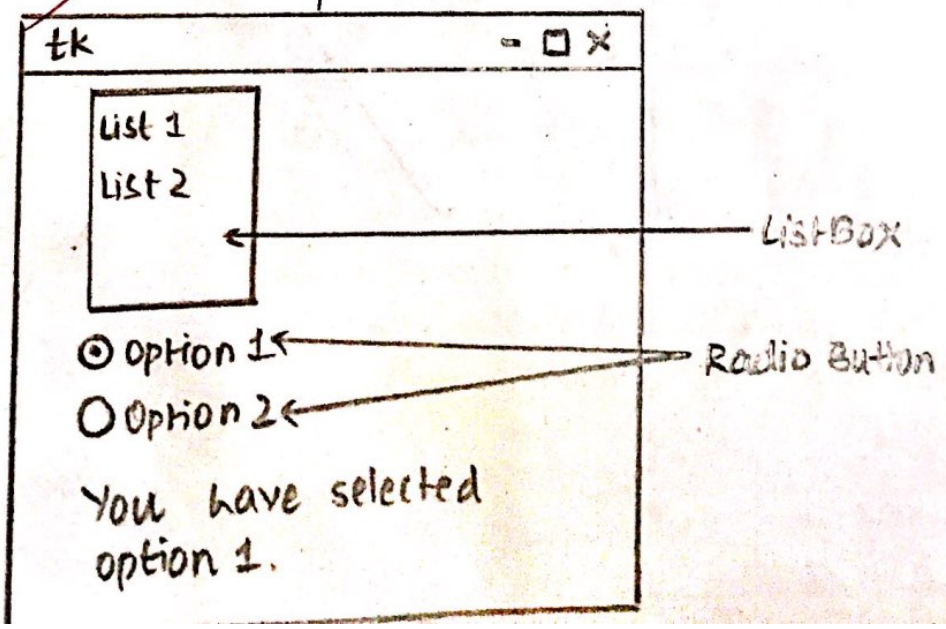
Algorithm: (i) import the relevant methods from the tkinter library and create an object within the parent window.
(ii) Define a function which tells the user about the given selection made from multiple options available.
(iii) Define the options within the parent window with control variable.
(iv) Use the listbox() and insert options onto the parent window along with the pack() with specifying anchor attri
(v) Create an object from radiobutton that will accept parent window object, text variable argument and corresponding value as arguments.
(vi) Call the pack() for the radio object and specify the anchor attribute.
(vii) Run the parent window object with the mainloop().

Radio button

Code:

```
from Tkinter import *
root = Tk()
root
def sel():
    sel = "You have selected : " + str(var.get())
    t1 = Label (text = sel, bg = "Yellow")
    t1.pack (side = TOP)

var = stringvar()
L1 = listbox()
L1.insert (1, "List 1")
L2. insert (2, "List 2")
L1.pack (anchor = N)
R1 = RadioButton (root, text = "option 1", variable = var,
    value = "option 1", command = sel)
R1.pack (anchor = N)
R2. RadioButton (root, text = "Option 2", variable = var,
    value = "Option 2", command = sel)
R2. pack (anchor = N)
root. mainloop()
```



Write a program to demonstrate use of scrollbar.

Algorithm:

(i) Import relevant methods from the tkinter library.

(ii) Create a parent object corresponding to the parent window.

(iii) Create an object and implement the scrollbar() onto it.

(iv) Use the pack() along with the scrollbar object with side and fill attributes.

(v) Use the mainloop() with the parent window.

Scrollbar :

Code:

```
from tkinter import *
root=Tk()
scroll = scrollbar()
scroll.pack( side = RIGHT , fill = "Y")
root. mainloop ()
```

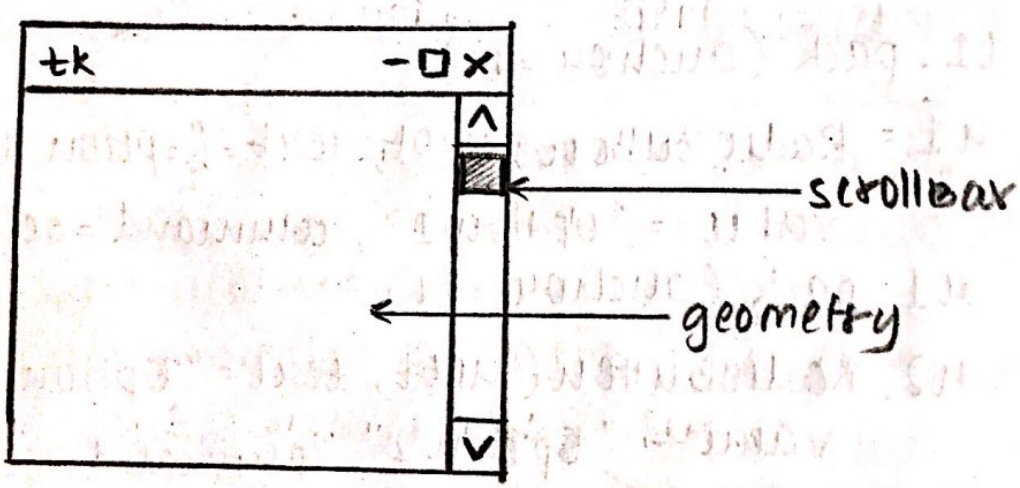


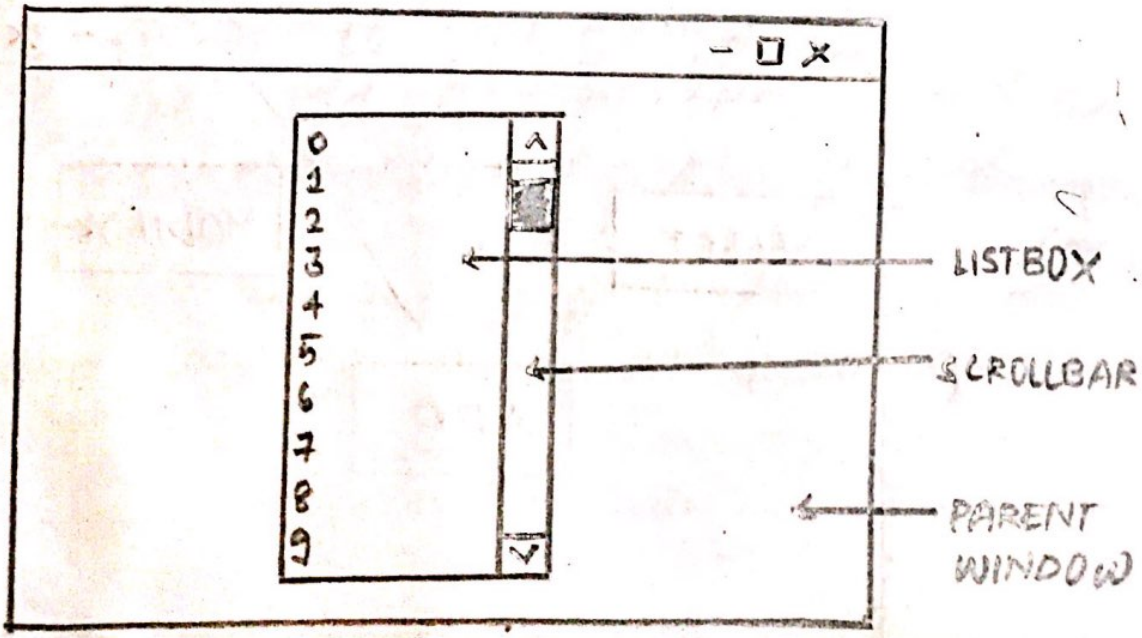
fig. scrollBox.

Q3. Demonstrate implementation of scrollbar using frame widget.

Algorithm: (i) Import the relevant libraries from the tkinter method.
(ii) Create a corresponding object of the parent window.
(iii) Use the label widget along with the parent object created and subsequently use the pack().
(iv) Use the frame widget along with the parent object created and use the pack().
(v) Use the listbox method along with the attributes like width, height, font.
(vi) Create a listbox method object and use pack() ~~from~~ for the same.
(vii) Use the scrollbar() with a scroll object and use the vertical attribute.
(viii) Configure the same with object created from the scrollbar() and implement pack().
(ix) Run the parent object through the mainloop().

Scrollbar

```
from tkinter import *
window = Tk()
window.geometry("680x500")
Label.pack(window, text = "Number:")
fname.Frame(window).
fname.pack()
ListNodes = Listbox(fname, width=20, weight=20,
                    font=("Time New Roman")
ListNodes.pack(side=LEFT, Fill='y')
Scrollbar1 = scrollbar(fname, oriant="vertical")
Scrollbar1.config(command = listNodes.Yview)
Scrollbar.pack(side="RIGHT", fill='Y')
for x in range(100):
        ListNodes.insert(END, str[x])
window.mainloop()
```



Q. Demonstrate the Frame() in python GUI

Code Algorithm: (i) Import relevant methods from tkinter library.

(ii) Define the object in parent window and define the size of parent window in terms of no. of pixels.

(iii) Now, define the frame object from the method & place it onto parent window.

(iv) Define the right frame and define the bottom object placed onto the given frame with the attribute as text, background colors etc.

(v) Now use the pack() along with the side attribute.

(vi) Similarly, create the button object corresponding to MODIFY operation and place it onto the frame object on the right ~~rig~~ side.

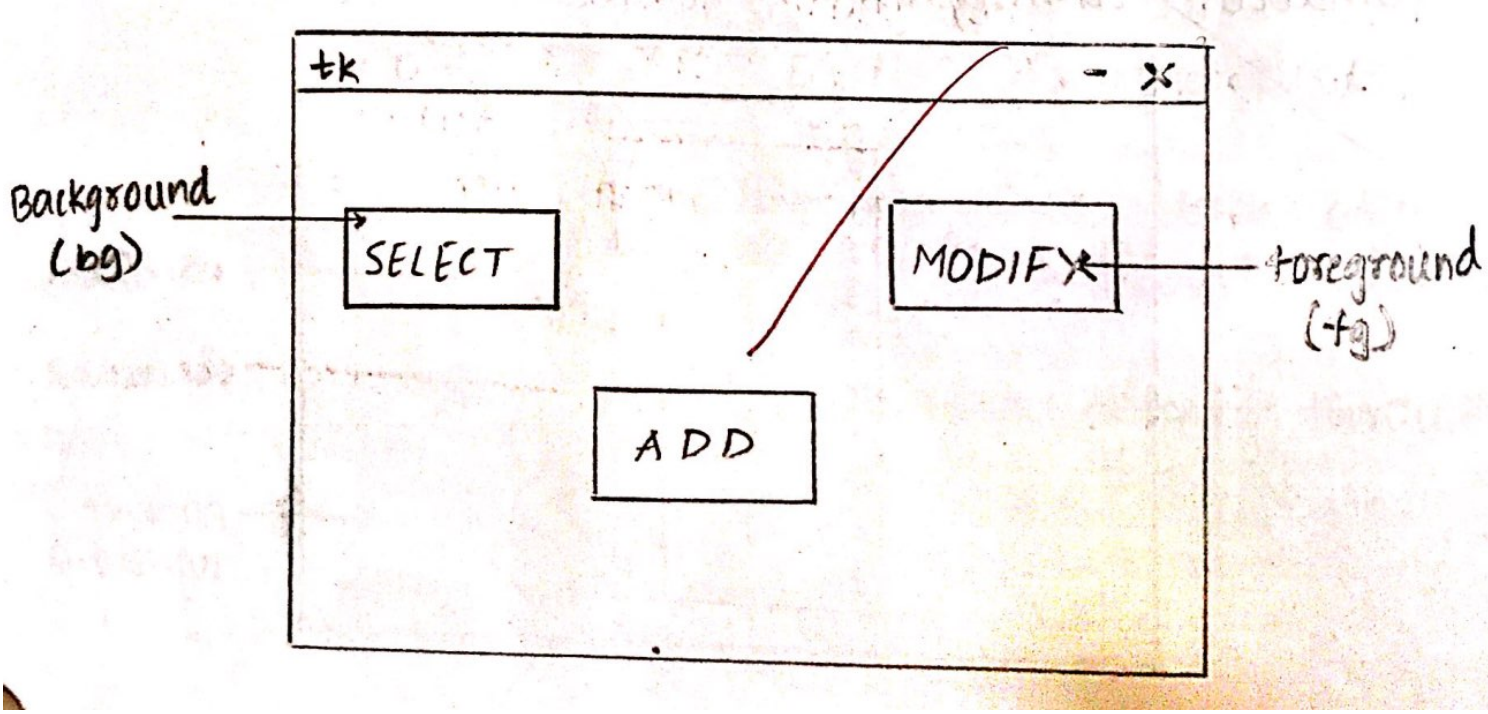
(vii) Create another button object, and place it on to the right.

(viii) Add another button and put it onto the frame and label it as EXIT.

(ix) Use the pack() for all the object and finally use the mainloop().

#6 Fname Window.

```
from tkinter import *
window = Tk()
window = geometry (680x500)
fname = Fname (window)
fname.pack()
leftfname = Fname (window)
leftfname • pack.(side = 'LEFT')
rightfname= Fname (window)
rightfname.pack (side = 'RIGHT')
b1 = Button (fname, text='SELECT', fg = 'blue')
b2 = Button (fname, text = 'SELECT', fg = 'yellow')
b3 = Button (fname, text= 'ADD', fg="red")
b1. pack (side = 'LEFT', padx = 20)
b2. pack (side = 'RIGHT', padx = 30)
b3. pack (side = 'BOTTOM', padx = 25)
```



Q. Demonstrate Message box in Python.

(i) Showinfo:

Algorithm: (i) Import the relevant method from tkinter library.

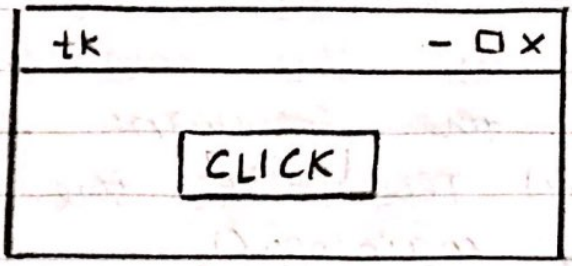
(ii) Define function and use the attributes which the given method will take and specify the strings corresponding to the title of the message displayed.

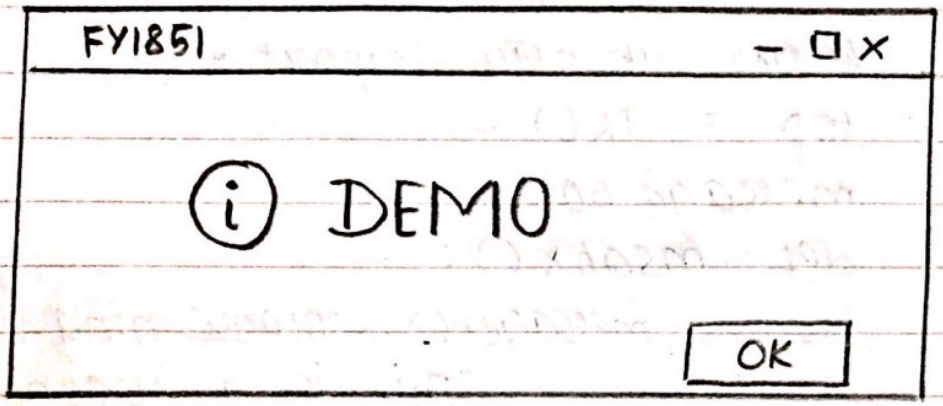
(iii) Create an object from the button method and place it onto the parent window with the title of the button object and use the command attribute.

(iv) Terminate the program by calling the mainloop().

Code:

```
from tkinter import *
root = Tk()
def messgbx():
    messagebox.showinfo("FY1851", "DEMO")
b1 = Button(top, text = "CLICK", command = msgbx)
b1.pack()
top.mainloop()
```





(ii) show warning :

Algorithm :

(i) Define a function which will use the showwarning() derived from the messagebox library.

(ii) The attributes which a given method take will specify the 2 strings on related to the message displayed corresponding to the message.

(iii) Now create an object from the method and place it on the parent window

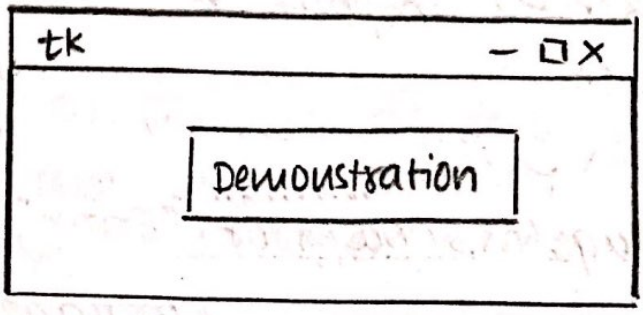
with the title of the button object specified and finally use the command attribute to execute the relevant function.

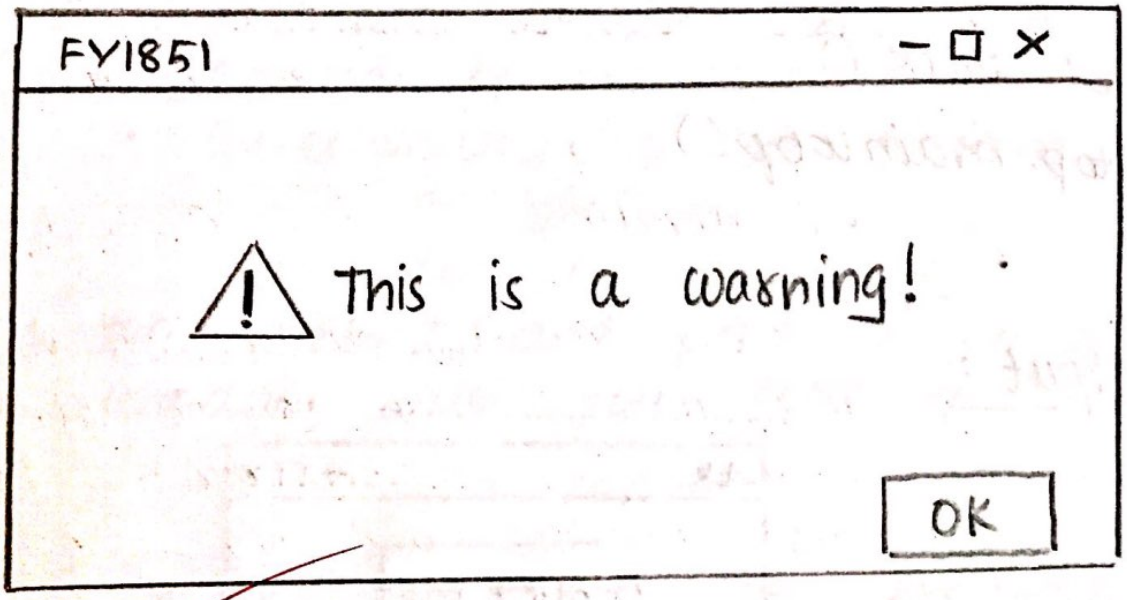
(iv) Terminate the program by calling the mainloop().

Code :

```
from tkinter import *
top = Tk()
message box
def msgbx():
        messagebox.showwarning ("FY1851",
                "This is a warning!")
bl. Button (top, text = "Demonstration",
                command = msgbx )
bl. pack ()
top. mainloop ()
```

Output:





(iii) showerror():

Algorithm():

(i) Define a function which will use the showerror() derived from the messagebox library.

(ii) The attribute which a given method takes will specify the 2 strings:

- Related to the title
- Corresponding to the message.

(iii) Now create an object from the button method and play it on the parent window with the title of the button.

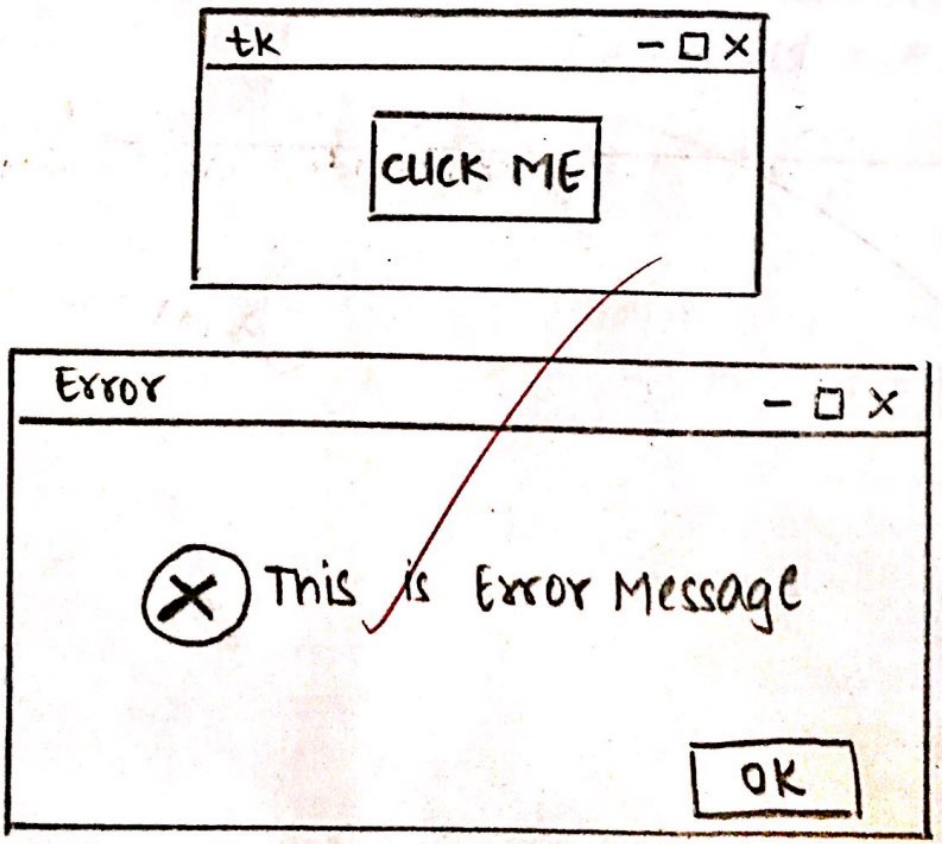
(iv) Terminate the program by using mainloop().

PE

Code:

```
from tkinter import *
top = Tk()
def msgbx():
    messagebox.showerror("Error", "This is error
                                    message")
bl. Button (top, text="CLICK ME", command = msgbx)
bl. pack()
top. mainloop()
```

Output:



(iv) askyesorno():

Algorithm:

(i) Define a function which will use the askyesno() ~~def~~ derived from the messagebox library.

(ii) The attributes which a given method takes of the window

- Related to the title of the window.
- Corresponding to the message displayed

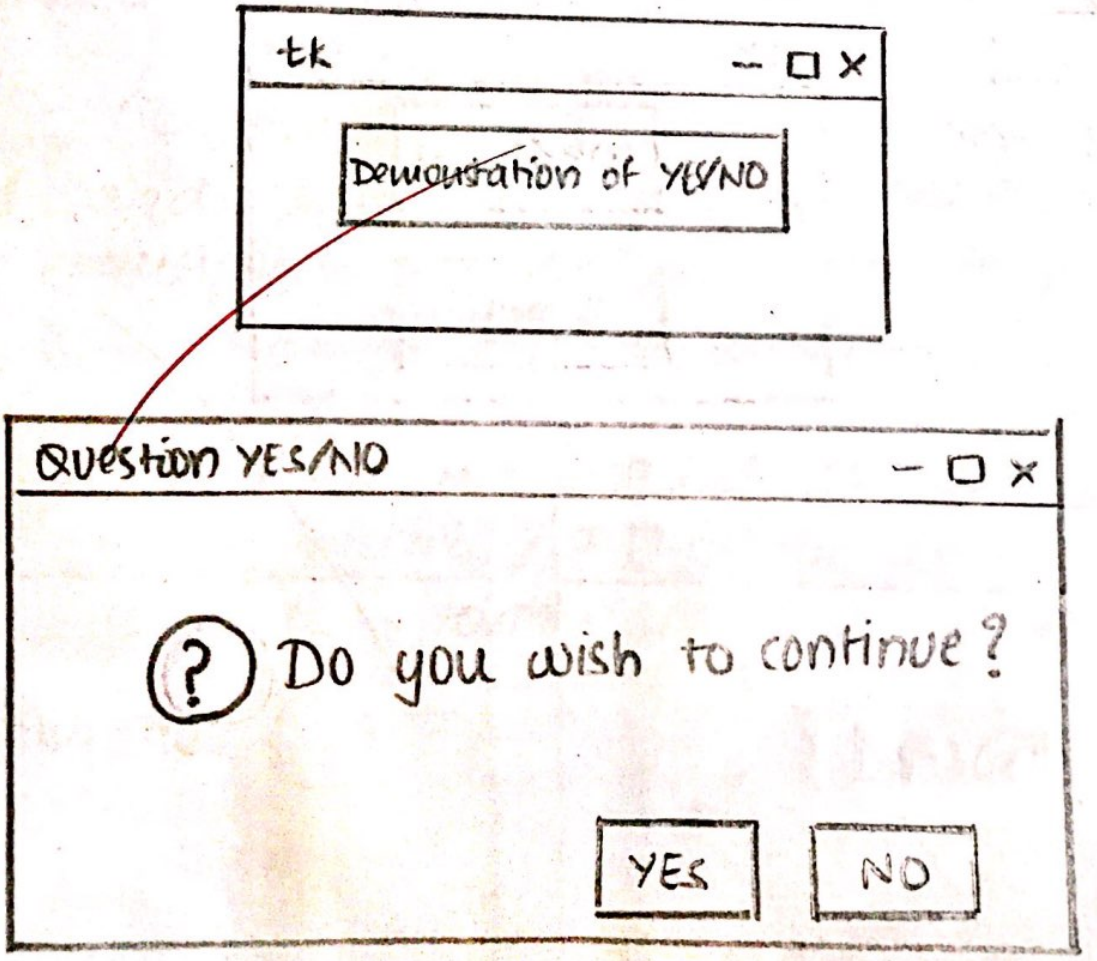
(iii) Now create an object from the button method and place ~~it~~ to the parent coindow with title of the button object specified and finally use the command attribute to execute the function.

(iv) Terminate the program by calling the mainloop().

Code:

```
from tkinter import *
top = Tk()
messagebox
def msgbx():
    messagebox.askyesno("QuestionYesNO", "Do you
                wish to continue?")
bl=Button(top, text="Demonstration of YES/NO",
                command = msgbx)
bl.pack()
top.mainloop()
```

Output:



(v) askokcancel & askquestion

Algorithm:

(i) Define a function which uses the askokcancel() from the messagebox library. Similarly define a function that uses the askquestion() from the message box library.

(ii) The attribute which a given method takes will specify the 2 strings:

- Related to the title
- Corresponding to the message displayed.

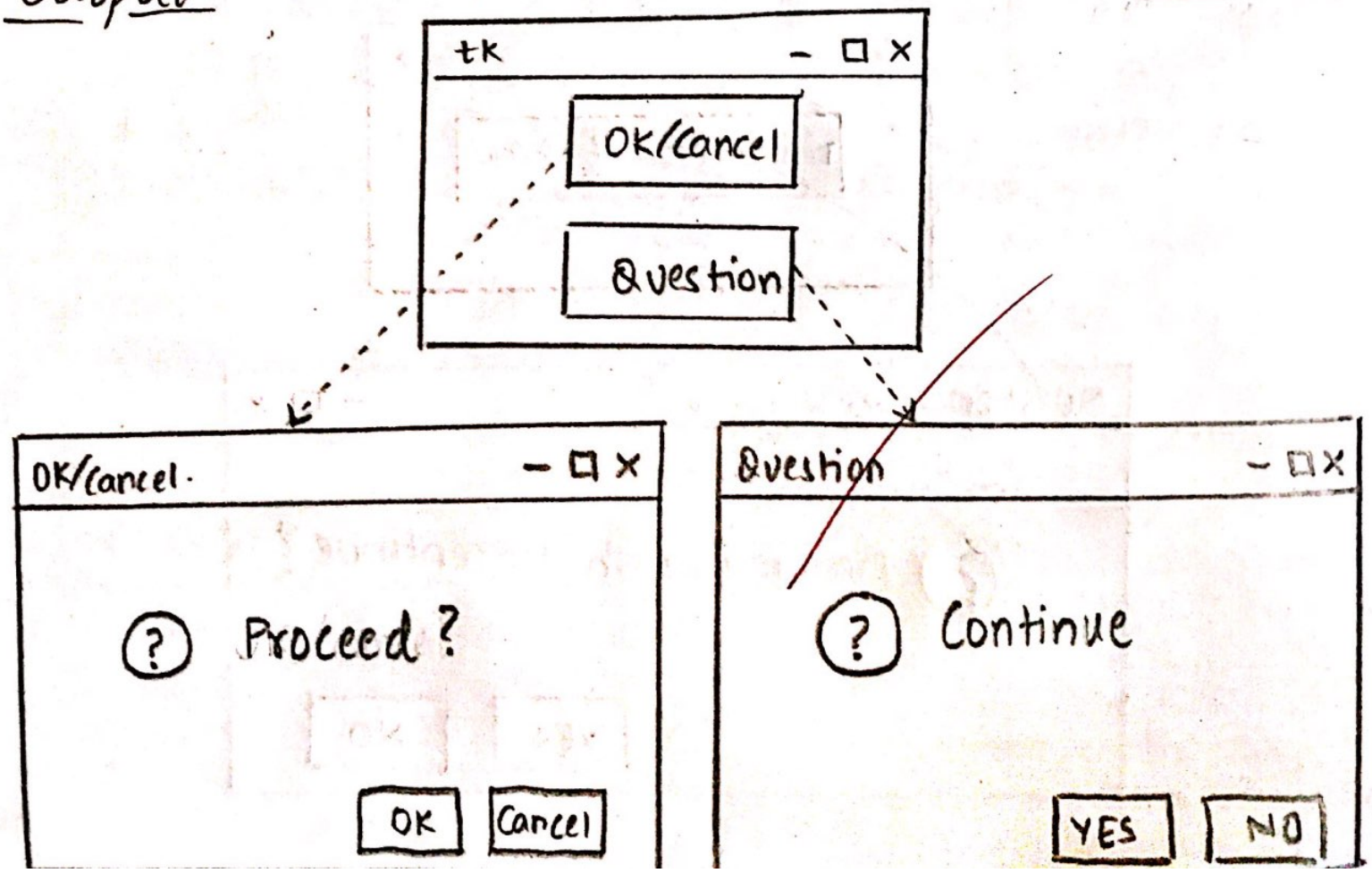
(iii) Create an object from the button method and place it onto the parent window with corresponding attributes to trigger the two message box functions.

(iv) Use the pack() to place the objects visibly on the window and run the mainloop() to terminate the program.

Code :

```
from tkinter import *
root = Tk()
def msgbx1():
    messagebox.askokcancel("OK/Cancel", "Proceed ?")
def msgbx2():
    messagebox.askquestion("Question", "Continue?")
b1 = Button(root, text="OK/Cancel", command = msgbx1)
b2 = Button(root, text="Question", command=msgbx2)
b1.pack()
b2.pack()
root.mainloop()
```

Output :



(vi) Button Attribute:

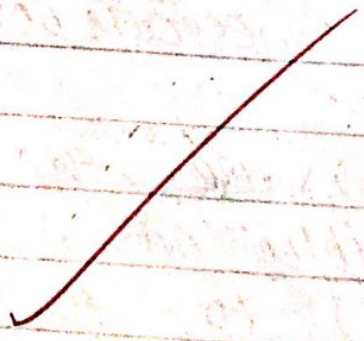
Algorithm:

(i) Define a button object and place it onto the corresponding parent window.

(ii) Use the text attribute for specifying the title of the button object.

(iii) Use the relief attribute with one style at a time with different objects.

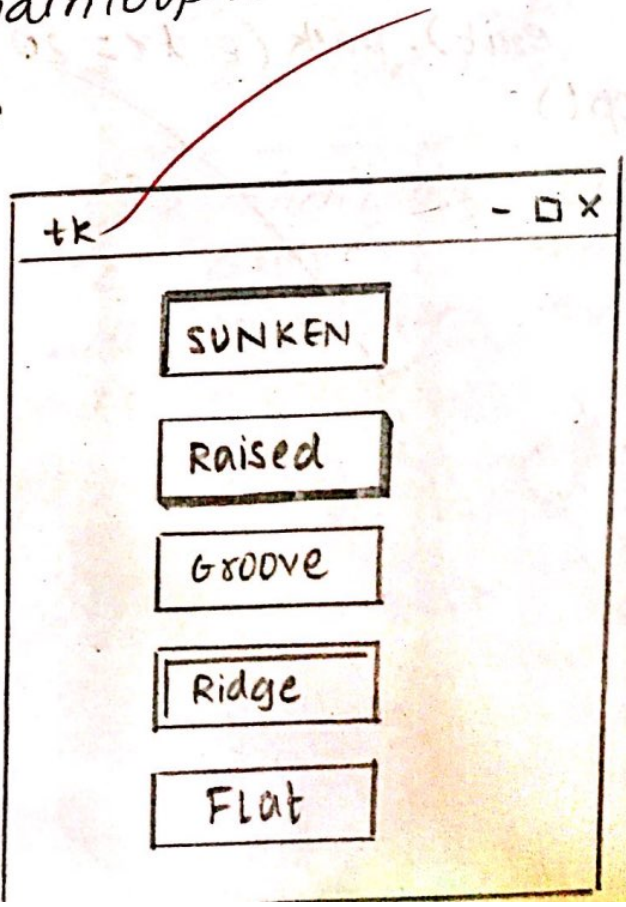
(iv) For positioning the widget object on the parent window and trigger the corresponding event by calling on the mainloop() function.



Code:

```
from tkinter import *
top = Tk()
b1 = Button (top, text = "sunken", relief = SUNKEN)
b1. pack()
b2 = Button (top, text = "Raised", relief = RAISED)
b2. pack()
b3 = Button (top, text = "Ridge", relief = RIDGE)
b3. pack()
b4 = Button (top, text = "Flat", relief = FLAT)
b4. pack()
b5 = Button (top, text = "Groove", relief = GROOVE)
b5. pack()
top. mainloop()
```

Output:



Traversing Multiple windows:

Algorithm:

(i) Define a function and create a parent window object and use the config, title and minsize attribute.

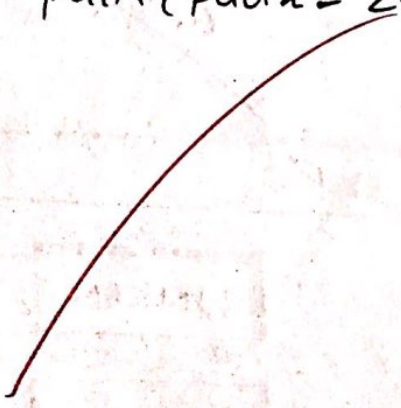
(ii) Define a button object and place it on to the parent window with a suitable title and use command attribute to call the next function using the grid() specifying padding.

(iii) Now define a function corresponding to the second window and create another parent window object with config(), title and minsize. Place the button object and call the next function.

(iv) similarly create the function and use the 'button()' and create a function which will terminate the aggregate function using quit method.

Code:

```
from tkinter import *
def selection ():
        root = Tk()
        root.config (bg = "red")
        root.title ("One Window")
        root.minsize (200, 200)
        b1 = Button (root, text = "Next window", command =
                    new1).pack (padx = 10 , pady = 20)
        root.mainloop()
def New1 ():
        top = Tk()
        top.config (bg = "Pink")
        top.title ("second window")
        top.minsize (150, 150)
        b2 = Button (top , text = "second window", command =
                            exit).pack (padx = 20, pady = 20)
        top.mainloop()
def exit ():
        quit ()
selection ()
new1()
new2()
```



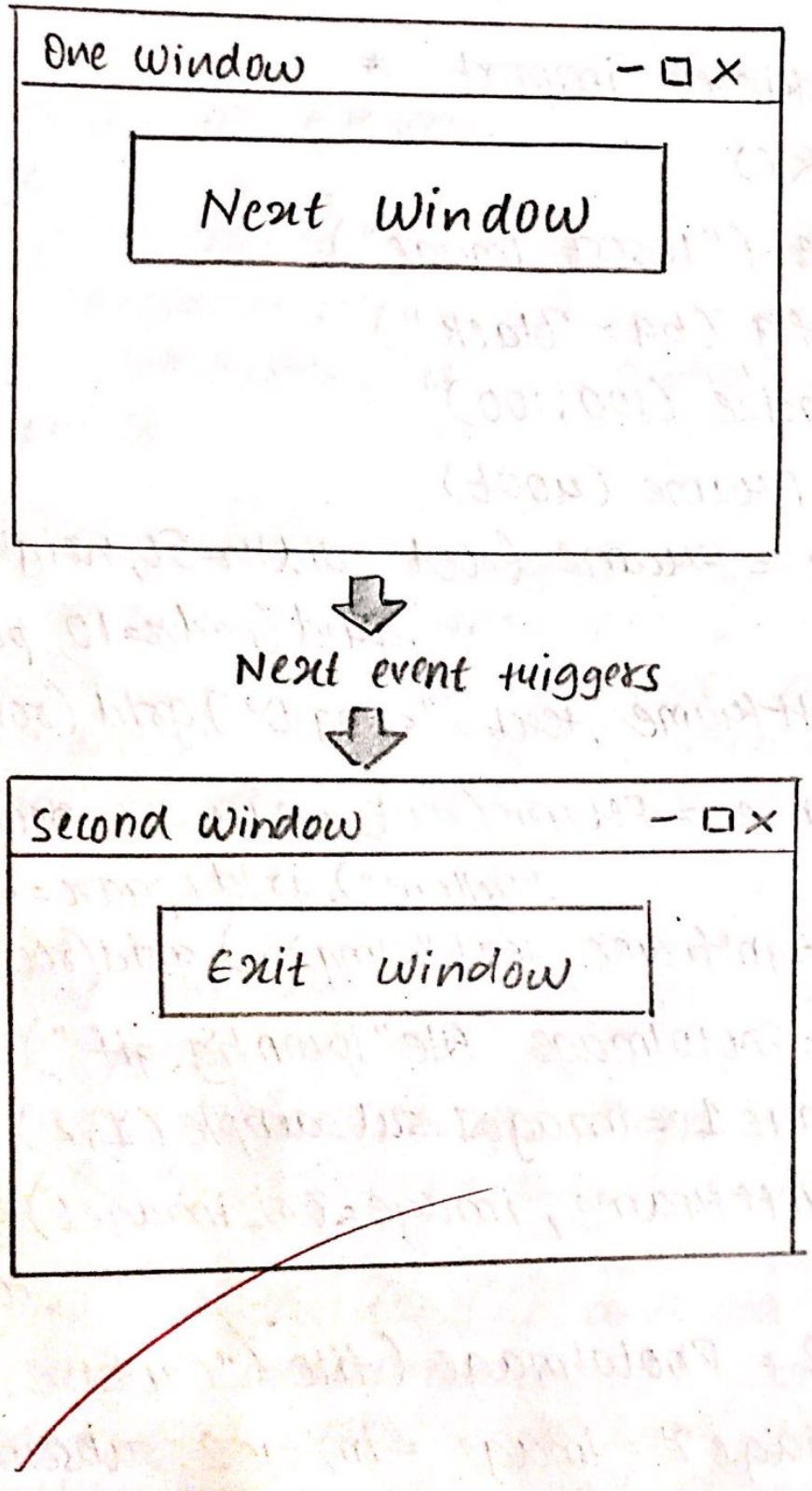


fig. window Traversal

Insert image in Frame.

Algorithm:

(i) Create a parent window by importing necessary files from tkinter library. Use the attributes title, config and minsize with object.

(ii) Create an object from the frame method and place it onto the parent window object with width, height and colour and use the grid() along with column and row attribute set to (0,0) with appropriate padding.

(iii) Similarly create the right frame ~~and~~ object from the frame() with row and column attribute making their respective values (0,1).

(iv) Use the label() and the parent window object corresponding to the left frame with text and relief attribute and use the grid method with row and column value as (0,0)

(v) Similarly create the label for the right frame and use the title. Set row and column values as (0,1).

(vi) Use the photo() with the file attribute specified and subsequently subsample() for specifying the image object.

(vii) Now use the label() using leftframe and the image attribute, row, column value specified in grid().

(viii) Create the label() using rightframe object with the image attribute and background colour with specified values (0,0

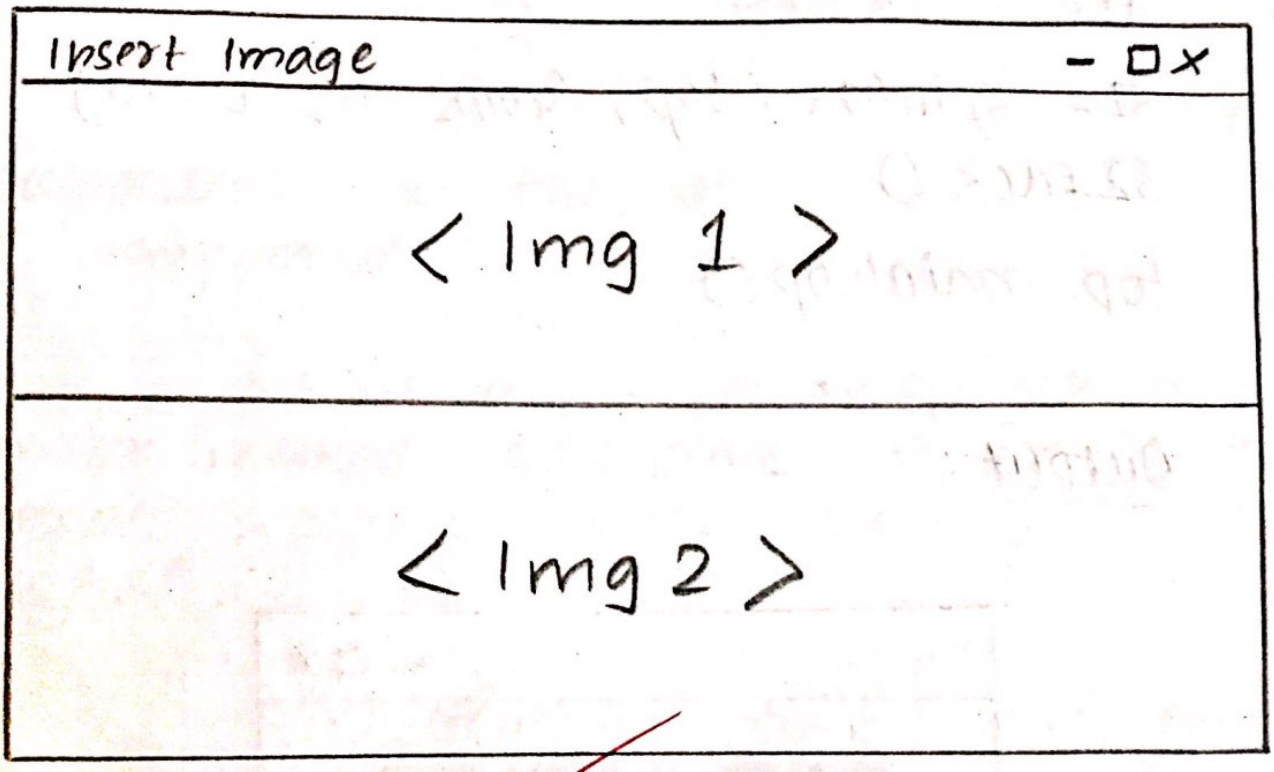
(ix) Define a function using print statement which shall be called on by clicking the window.

(x) Now use the mainloop() to terminate the program.

Code:

```
from tkinter import *
root = Tk()
root.title ("Insert Image")
root.config (bg = "black")
root.minsize (100,100)
fram1 = Frame (root)
leftframe = Frame (root, width=50, height=50, bg="blue")
                .grid (padx=10, pady=10)
Label (leftframe, text = "<Img 1>").grid (row=0, column=0)
rightframe = Frame(root, width=50, height=50, bg =
                "yellow").grid (padx = 10, pady = 10)
Label (rightframe, text"<Img2>").grid(row=0, column=0)
image1 = PhotoImage (file "ipconfig.gif")
ori_image1 = image1.subsample (1,2)
Label (leftframe, image=ori_image1).grid (row=0,
                column = 0)
image2 = PhotoImage (file = "capture.gif")
ori_image2 = image = image2.subsample (1,2)
Label (rightframe, image = ori_image2).grid(row=1,
                column = 0)
root.mainloop()
```

Output :



Spinbox :

Algorithm:

(i) Use the tkinter library to import the relevant method.

(ii) Create the parent window corresponding to the necessary events to be triggered.

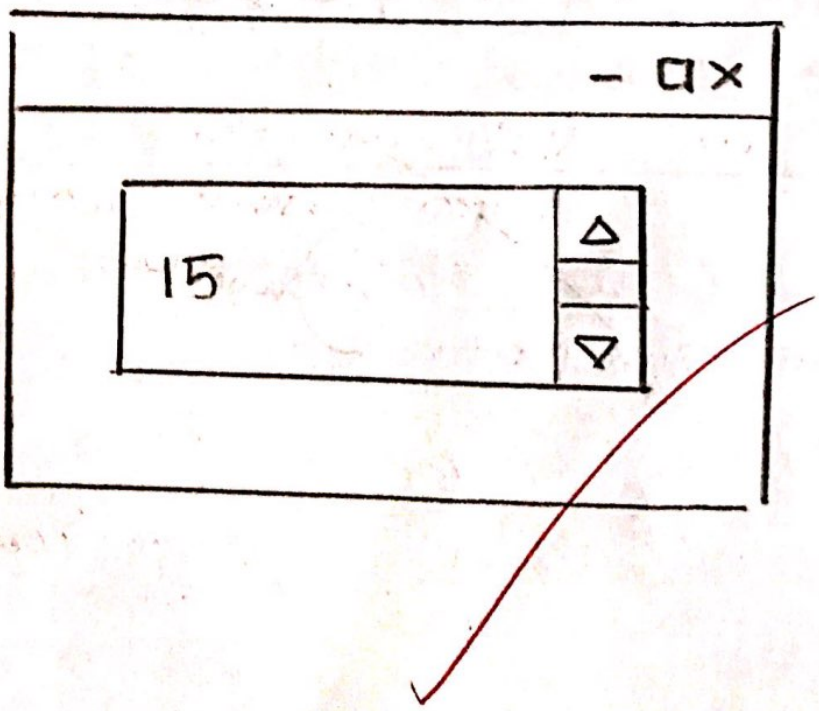
(iii) Create an object from a spinbox method and place it on to the parent window with the option specified.

(iv) Now use the pack method to make the object visible onto the parent window and ~~caused~~ call the mainloop method.

Code :

```
from tkinter import *
top = Tk()
s1 = spinbox(top, from_=0, to=10)
s2.pack()
top.mainloop()
```

Output :



Paned window:

Algorithm: (i) create an object from the paned window method and use the pack method to make this object visible.

(ii) Now create an object from the entry widget and place it onto the paned window and use the add method. Similarly create an object of a paned window.

(iii) create an object from the scale ~~pack~~ widget and place it onto the preceding paned window and use the add method accordingly.

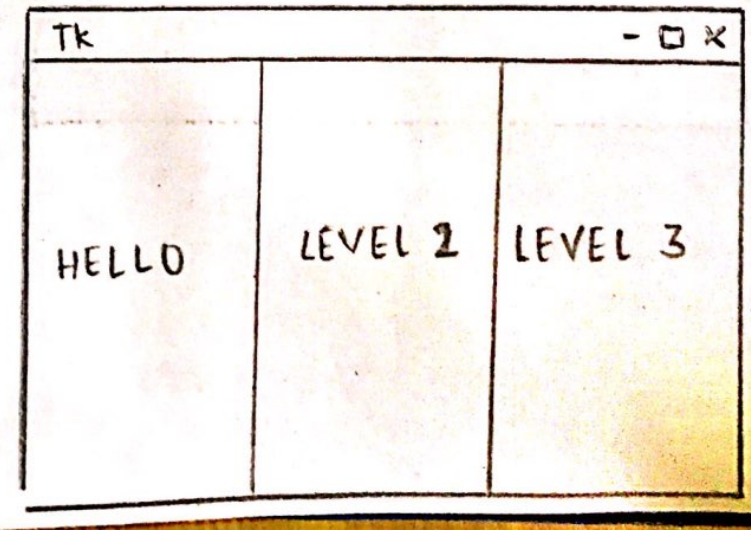
(iv) create a button widget and place it onto the paned window. define a functionality along with the button widget.

(v) use the pack method and mainloop method for the corresponding event to trigger.

Code :

```
from tkinter import *
root = Tk()
p = PanedWindow (bg = "yellow")
p.pack (fill = BOTH , expand = 10)
Label1 = Label (p.text = "Hello", bg = "red")
p.add (Label1)
p1 = PanedWindow (p.orient = VERTICAL , bg = "blue")
p.add (p1)
Label2 = Label (p1.text = "LEVEL 2", bg = "green")
p1.add (label2)
p2 = PanedWindow (p.orient = HORIZONTAL ,
                  bg = "pink")
label3 = Label (p.text = "LEVEL 3", bg = "orange")
p.add (p2)
p2.add (label3)
mainloop ()
```

Output :



Canvas Widget:

Algorithm: (i) Create an object from the canvas widget by using the attribute height, width, bg color and parent window object.

(ii) Use the corresponding method for drawing the simple geometrical shape like circle, oval and line and specify the co-ordinate values.

(iii) Similarly use the create line and create oval method along with the co-ordinate values and the fill attributes for specifying the colour.

(iv) Finally use the pack() and mainloop() to trigger the corresponding event.

Code:

```
from tkinter import *
root = Tk()
can1 = canvas (root, height = 50, width = 50)
oval = c1. create_oval (500, 12, 12, 40, fill = "RED")
line = c1. create_line (50, 20, 70, 60, fill = "BLUE")
arc = c1. create_arc (30, 18, 18, 40, fill = "YELLOW
c1. pack ( side = TOP)
root. mainloop ()
```

Output: