```
fileobj = open ("abc.txt", "w")
fileobj.write ("Dance tunes" + "\n")
fileobj.write ("bangawang in revolution in gerhoo")
fileobj.close()

fileobj = open ("abc.txt", "r")
# read
str1 = fileobj.read()
print ("The output of read method:", str1)
fileobj.close()
>>> ('The output of read method:', 'Dance tunes\n
                bangawang in revolution in gerhoo\n')

#readline()
fileobj = open ("abc.txt", "r")
str2 = fileobj.readline()
print ("output:", str2)
fileobj.close()
>>> ('output:', 'Dance tunes\n')

#readlines()
fileobj = open ("abc.txt", "r")
str3 = fileobj.readlines()
print ("output:", str3)
fileobj.close()
>>> ('output:', ['Dance tunes\n', 'bangawang in revolution in gerhoo'])

# 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 = fileobj.mode
print ("file mode", c)
>>> ('file mode', 'w')
d = fileobj.softspace
print ("softspace :", d)
>> ('softspace :', 0)
```

```
# w+ mode
fileobj = open ("abc.txt", "w+")
fileobj.write ("skhillex")
fileobj.close()
```

```
# r+ mode
fileobj = open ("abc.txt", "r+")
s1 = fileobj.read(5)
print ("output of r+", s1)
fileobj.close()
>> ('output of r+', 'skhil')
```

```
#append mode
fileobj = open ("abc.txt", "a")
fileobj.write (" Ragga bomb")
fileobj.close()
fileobj = open ("abc.txt", "r").
s3 = fileobj.read()
print (" output of append mode :", s3)
fileobj.close()
>>> ('output of append mode :', 'skhillex Ragga bomb')
```

```
# write mode
fileobj = open ("abc.txt", "w
fileobj.write (" Diplo ")
fileobj.close()
```

```
# read mode
fileobj = open ("abc.txt", "r")
s = fileobj.read()
print ("output of read mode",
>>('output of read mode :', 'diplo
```

Step 7 - open the fileobject in read mode, declare a variable and perform fileobject.read() [illegible] and store the output consecutively in variable.

Step 8 - Use the seek method with the arguments after opening the fileobject in read mode and print subsequently.

Step 9 - open fileobject with read mode also use the readlines method and store the output [illegible] in and print the same. For counting the length use the for conditional statement & display the length.

6/12/M 75:

Practical No. 2.

Objective : To demonstrate the use of iterators & iterable.

odd numbers.

Algorithm :

Step 1 : Define a iter method with an argument and initialize the value and return the value.

Step 2 : Define the next method with an argument and compare the upper limit by using a conditional statement. Increment the value by

Step 3 : Now create an object of the given class and pass this object in the iter method.


```
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 Stop Iteration
y = iter(odd())
while True:
    print(next(y))
```

Output:

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

```
class myiter :
    def __iter__(pow):
        pow.n = 1
        pow.n1 = int(input("Enter the number"))
        pow.n2 = int(input("Maximum limit of powers"))
        return pow
    def __next__(pow):
        if pow.n <= pow.n2 :
            num = pow.n1 ** pow.n
            pow.n += 1
            return num
        else :
            raise StopIteration
y = iter(myiter())
while True:
    print(next(y))
```

Output:

```
>>> Enter the number = 2
>>> Maximum limit of powers = 4
2
4
8
16
```


Powers

Algorithm.

Step 1: Define iter method with 3 arguements. Initialize the first arguement as 1. Initialize the other two arguements as "Enter the number" and "Maximum limit of powers" respectively.

Step 2: Define the next method with an arguement and compare it by using a conditional statement. Increment the value by 1.

Step 3: Now create an object of the given class and pass that object to the iter method & use the while conditional statement to print.

```
class myrange:
    def __iter__(self):
        self.a = 1
        return self
    def __next__(self):
        if self.a <= 30
            x = self.a
            self.a += 1
            return x
        else:
            raise StopIteration.

myclass = myrange()
mydata = iter(myclass).
for x in mydata
    print(x)
```

output:

1	16
2	17
3	18
4	19
5	20
6	21
7	22
8	23
9	24
10	25
11	26
12	27
13	28
14	29
15	30

15

Range

Algorithm

Step 1: Define a iter method with an argument & initialize the value & return that value.

Step 2: Define the next method with an argument & compare the upper limit by using a conditional statement.

Step 3: Now create an object of the given class & pass that object in the iter method & use for loop statement to print.

#1.

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

output :

Enter class : 467.

#2

```
try :
    fo = open ("abx. txt", "w")
    fo. write (" skwillex ")
except IOError :
    print (" Error while writing on the file")
else :
    print (" operation carried out successfully")
    fo. close ()
```

output :

operation carried out successfully.

Practical No. 3.

Aim : programs to demonstrate exception handling.

1) write a program using the exception method of the name Arithmetic error.

→ Step 1 : Use the try block and except the input using the raw input method and convert it into the integer datatype and subsiquently terminate the block.

Step 2: 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.

2) write a program for accepting the file in a given mode and use the environment error as an exception for the given input.

→ Step 1: write the try block, open the file using the write mode and write some content on the file.

Step 2 : Use the except block with IO error and display the message regarding missing of the file or incompability of the mode use the else block to display a message that the operation is carried out successfully.

3) write a program using the assert() to check if t list elements are empty.

→ step 1: Define a function which excepts an argument and check using the assert statement whether the given list is empty list and accordingly return the message.

step 2: close the function and in the body of the pro and define certain elements in list and take some appropriate action.

4) write a program to check the range of the age of th students in given class and if the age do not fall in given range use the value error exception otherwi return the valid no.

→ step 1: Define a function which will accept the age o the student from the standard input.

step 2: use the if condition to check whether the input falls in the range and so return the age else the value error exception

step 3: Define the while loop to check whether the boolean expression holds true use the try blo accept the age of student and terminate the looping condition.

step 4: use except with ValueError and print the mes not a valid num.

code

```
import re
string = "hello 1234 abc456"
result = re.findall("\d", string)
result = re.findall("\D", string)
print(result)
print(result[1])
```

output

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

5) write a regular expression pgm finding the match string at the begining of given sequence.

Algorithm:

Step 1: Import re module & apply a string.

Step 2: use search with "^a python" and string as two parameters.

Step 3: Now display the output.

Step 4: Now use if conditional statement to know whether match is found or not.


```
#code
import re
string = "python is an intended language"
result = re.search ["^a python", string]
print (result).
if result:
    print ("Match found").
else:
    print ("Match not found").

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

```
# code
import re
li = ["9653177650", "8828707485", "9892400000"]
for element in li:
    result = re.match("[8-9][0-9][9]",element)
if result:
    print("connect mobile no")
    print(result.group())
else:
    print("incorrect mobile no.")

#output
>>> connect mobile no.
    9653177650.

>>> connect mobile no.
    8828707485
```


3) write a regular expression to check whether the given mobile number starts with 8 or 9 & the total length 10.

Algorithm:

step 1 : Import the module & apply a string of mobile no. &.

step 2 : Now use for conditional statement to find if the number starts with 8 or 9 & the total number should be of length 10. use match() inside for statement to find the match in given string.

step 3 : use if conditional statement to know whether we have to match or not if we have use group() to display the output & if we don't display incorrect mobile no.

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

Algorithm:
Step 1: Import re module & apply a string.

Step 2: Use findall() to extract a word from given string.

Step 3: Use "\w*" to extract word along with space & use "\w+" to extract word without space.

Step 4: Now display the output.

5) Write a regular expression for extracting first & last word from a string.

Algorithm:
Step 1: Import re module & apply a string.

Step 2: Use findall() in which use "^\w+" as one parameter to find first word of string and use "\w+$" as parameter to find last word of string.

Step 3: Display the output

```
# code
import re
string = " VISHNU24 12-12-2019 "
result = re.findall ("\d{2}-\d{2}-\d{4}", string)
print (result)

# output
>>> ['12-12-2019']

# code
import re
string = "abc@tcsc.edu"
result1 = re.findall ("^\w+", string)
result2 = re.findall ("+\w+.\w+$", string)
result3 = re.findall ("[\w%.-]+", string)
print (result1)
print (result2)
print (result3)

# output
>>> ['abc']
>>> ['tcsc.edu']
>>> ['abc', 'tcsc.edu']
```


6) write a regular expression for extracting the data in format dd-mm-yyyy by using the findall() where the string has following format.
VISHNU24 12-12-2019.

Algorithm:
Step 1: Import re module & apply a string.

Step 2: use findall method & use \d{2}-\d{2}-\d{4} as a parameter.

Step 3: Now display the output.

7) write a re for extracting the
① username from email id.
② Hostname from email id.
③ Both username & hostname from email id.

Algorithm:
Step 1: Import re module & apply string.

Step 2: use Findall() to find username, hostname & both from email-id.

Step 3: use "^\w+" for username use "+\w+.\w+$" for hostname & use "[\w%.-]+" for both as parameter in findall().

Step 4: Display the output.

17/1/2020.

Practical No. 5

Topic : GUI components.

Step 1: Use the tkinter libraries for importing the features of the text widget.

Step 2: Create an object using the Tk().

Step 3: Create a variable using the widget label & use the text method.

Step 4: Use the mainloop() for triggering of the corresponding above mentioned events.

&

#2

Step 1: Use the tkinter library for importing the features of the that widget.

Step 2: Create a variable from the text method and position it on the parent window.

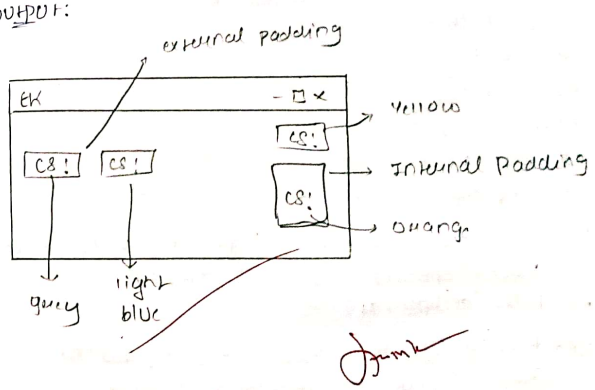
Step 3: Use the pack() along with the object name from the text() and use the parameters.
1) side = LEFT, padx = 20
2) side = LEFT, pady = 80
3) side = TOP, ipadx = 40
4) side = TOP, ipady = 50

Step 4: Use the mainloop() for the triggering of the corresponding events.

Step 5: Now repeat above steps with the label() which takes the following arguments
1) Name of the parent window.
2) Text attribute which defines the string.
3) The background color (bg).
4) The foreground fg and then use the pack() with a relevant padding attributed.


```
l4 = Label (root, text = "CS 1", bg = "orange",
            fg = "black", font = "110")
l4. pack (side = TOP, ipady = 50)
root. mainloop ( )
```

OUTPUT:



#Radio Button

```
from tkinter import *
root = Tk().
root.geometry("500x500")
def select():
    selection = "You just selected " + str(var.get())
    l1 = Label(text = selection, bg = "white", fg = "green")
    l1.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 = select)
l1.pack(anchor = N).
R2 = Radiobutton(root, text = "option 2", variable = var,
    value = "option 2", command = select)
l2.pack(anchor = N)
root.mainloop().
```


Practical -5(B)

AIM : GUI components.

#1 Step 1 : Import the relevant methods from the tkinter library. Create an object with the parent window.

Step 2 : Use the parent window object along with the geometry() declaring specify pixel size of the parent window.

Step 3 : Now define a function which tells the user about the given selection made from multiple option available.

Step 4 : Now define the parent window and define the option with control variable

Step 5 : Use the listbox() and insert options on the parent window along with the pack() with specifying anchor attribute.

Step 6 : Create an object from Radiobutton which will take following arguments (parent window object, text variable which will take the values option no. 1, 2, 3... variable arguments, corresponding values & trigger the function declared.

Step 7 : Now call the pack() for radio object to created and specify the arguement using anchor attribute.

Step 8 : Finally make use of the mainloop() along with parent object.

#2

Step 1 : Import relevant methods from the Tkinter library.

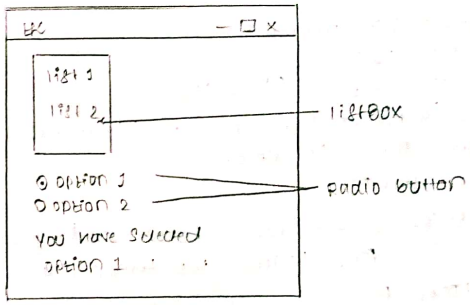
Step 2 : Create a parent object corresponding to the parent window.

Step 3 : Use the geometry() for laying of the comp

Step 4 : Create an object and use the Scrollbar().

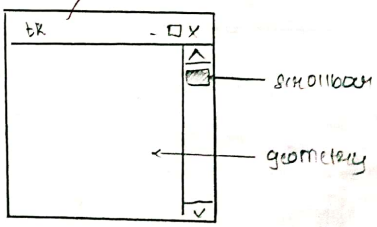
Step 5 : Use the pack() along with the Scrollbar object with side & fill attributes.

Step 6 : Use the mainloop with the parent objec



#2

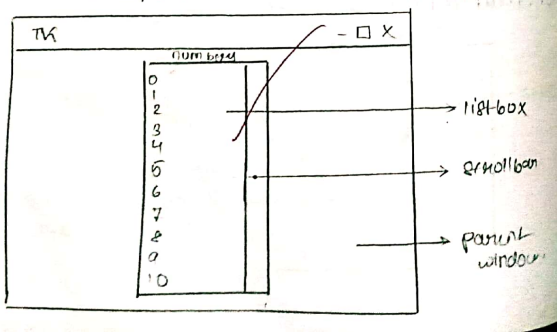
```
scrollbar()
from Tkinter import *
root = Tk()
root.geometry ("500x500")
s= scrollbar()
s. pack (side = "RIGHT", fill = "Y")
root. mainloop()
```



Scrollbar.

#3 Ex

```
# using frame widget.
from Tkinter import *
window = Tk()
window.geometry ("680 x 500")
label (windows Text = "numbers"). pack()
frame = Frame (window)
frame.pack()
listnodes = listbox (frame, width = 20, height = 20,
                     font = ("Times New Roman", 10))
listnodes.pack (side = "LEFT", fill = 'Y')
scrollbar = scrollbar( frame, width = 20, hei
scrollbar = Scrollbar (frame, orient = "VERTICAL")
scrollbar.config (command = listnodes.yview)
scrollbar.pack (side = "RIGHT", fill = "Y")
for x in range (100):
    listnodes.insert (END, str(x))
window.mainloop().
```



#3 Step 1: Import the relevant libraries from the Tkinter method.

Step 2: Create an corresponding object of the parent window

Step 3: Use the geometry manager with pixel size (680 x 500) or any other suitable fixed value.

Step 4: Use the label widget along with the parent object created & simultaneously use the pack() method.

Step 5: Use the frame widget along with the parent object created and use the pack method.

Step 6: Use the listbox method along with the attribute like width, height font. Do create a listbox method's object. Use pack() for the same

Step 7: Use the Scrollbar() with an object use the attribute of vertical then configure the same with object created from the Scrollbar() and use pack()

Step 8: Triggers the events using mainloop.

#4:

Step 1: Import relevant methods from Tkinter library.

Step 2: Define the object corresponding the parent window & define the size of parent window in terms of no. of pixels.

Step 3: Now define the Frame object from the method & place it onto the parent window.

Step 4: Create another Frame object termed as the left Frame & put it on the parent window on its left side.

Step 5: Similarly define the RIGHT Frame and subsequently define Frame with the attributes as text, active background & foreground.

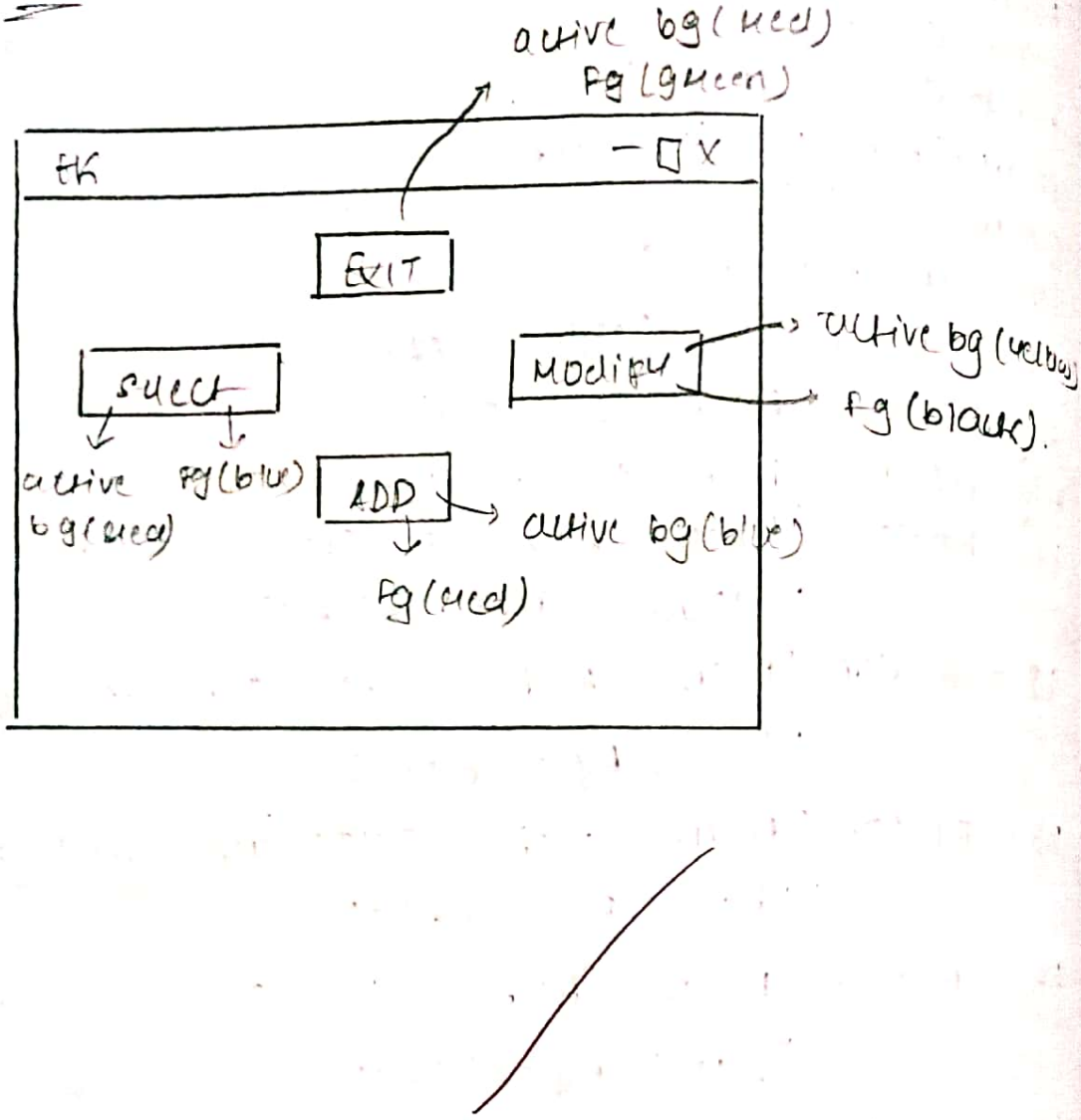
Step 6: Now use the pack() along with the side attribute.

Step 7: Similarly create the button Object connected to the MODIFY operation put it into Frame object on side = "right".

#4:

```
From tkinter import *
window = Tk()
window.geometry("680 x 500")
Frame = Frame(window)
Frame.pack()
leftFrame = Frame(window)
leftFrame.pack(side = "LEFT")
rightFrame = Frame(window)
rightFrame.pack(side = "RIGHT")
b1 = Button(Frame, text = "modify", active background =
     "yellow", fg = "black")
b2 = Button(Frame, text = "select", active background =
     "red", fg = "blue")
b3 = Button(Frame, text = "EXIT", active background =
     "blue", fg = "red")
b4 = Button(Frame, text = "EXIT", active background =
     "red", fg = "green")
b1.pack(side = "LEFT", padx = 20)
b2.Pack(side = "RIGHT", padx = 30)
b3.Pack(side = "BOTTOM", pady = 20)
b4.Pack(side = "TOP")
```

OUTPUT :-



components OF GUI (BUTTON, attribute, messagebox)

Q. write a program on various attributes which a button widget may assume related to the relief attribute.

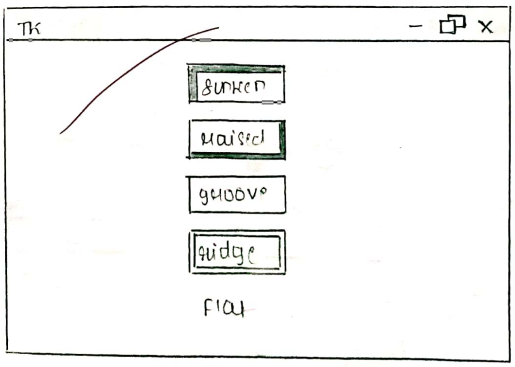
1) define a button object & place it on to the obj corresponding to the parent window.

2) use the text attribute for specifying the title to the button object.

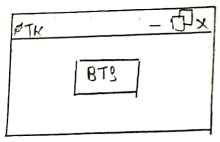
3) use the relief attribute with one style at a time with different button objects.

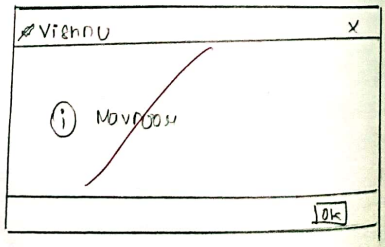
4) for positioning the widget object on to the parent window & triggering the corresponding event by calling the mainloop method.


```
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 = "groove", relief = GROOVE)
b3.pack()
b4 = Button (top, text = "ridge", relief = RIDGE)
b4.pack()
b5 = Button (top, text = "flat", relief = FLAT)
b5.pack()
top.mainloop().
```



```
from tkinter import *
top = Tk()
messagebox
def msgb():
    messagebox.showinfo("Vishnu", "Navroosu")
b1 = Button(top, text = "BTS", command = msgb)
b1.pack()
top.mainloop()
```





Q.) write a program to implement the messagebox widget & the different method which these particular widget may assume.

→ showinfo()

1) define function which will use the showinfo() derived from the messagebox library.
2) The attributes which a given method takes will specify the 2 strings one selected to the title as composed to the message displayed.
3) Now create an object from the button method & place it on to the parent window with title of the button object and finally use the command attribute.
4) Terminate the program by calling the mainloop().

→ showwarning().

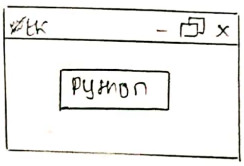
1) define a function which will use the showwarning() derived from messagebox library

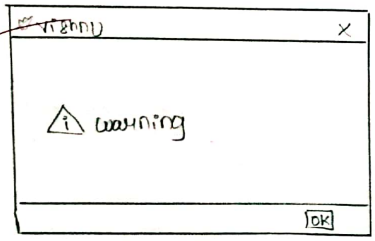
2) The attribute which a given method takes will specify the 2 strings one related to the message displayed (1) corresponding to the message

3) 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.

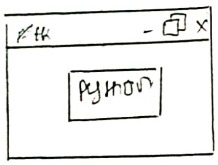
4) Terminate the program by calling the mainloop().

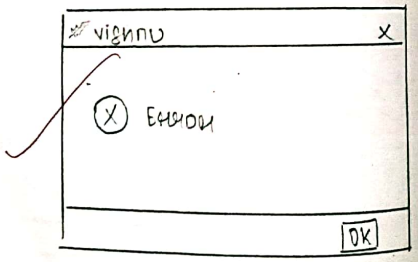

```
from tkinter import *
top = TK()
messagebox
def msgb():
    messagebox.showwarning("vishnu", "warning")
b1 = BUTTON(top, text = "Python", command = msgb)
b1.pack()
top.mainloop()
```





```
from tkinter import *
top = Tk()
messagebox
def msgb()
    messagebox.showerror("vishnu", "Error")
b1 = Button(top, text = "python", command = msgb)
b1.pack()
top.mainloop().
```





→ showerror().

1) define a function which will use the showerror() derived from the messagebox library.

2) The attribute which a given method take will specify the 2 string.
   (1) related to the title.
   (2) corresponding to the message.

3) Now create an object from the button method & place it onto the parent window with the title of the button object.

4) Terminate the program by using mainloop()

→ askyesno()

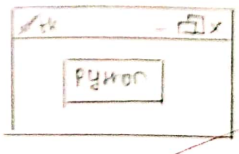
1) Define a function which will use the askyesno from the messagebox library.

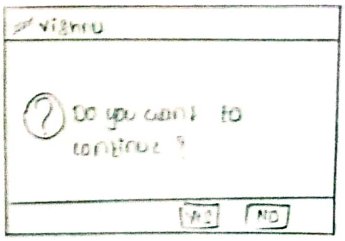
2) The command which a given method takes is the callback to be called if the user clicks on the button corresponding to the message displayed.

3) Now create an object from the button method, place it in the parent window with the help of the button object specified and finally use the command attribute to execute function.

4) Terminate the program by calling the mainloop

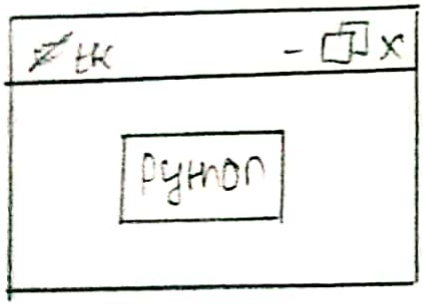
```
from tkinter import *
top = Tk()
import messagebox
def msg():
    messagebox.askyesno("virtual", "do you want to continue")
b = Button(top, text = "Python", command = msg)
b.pack()
top.mainloop()
```

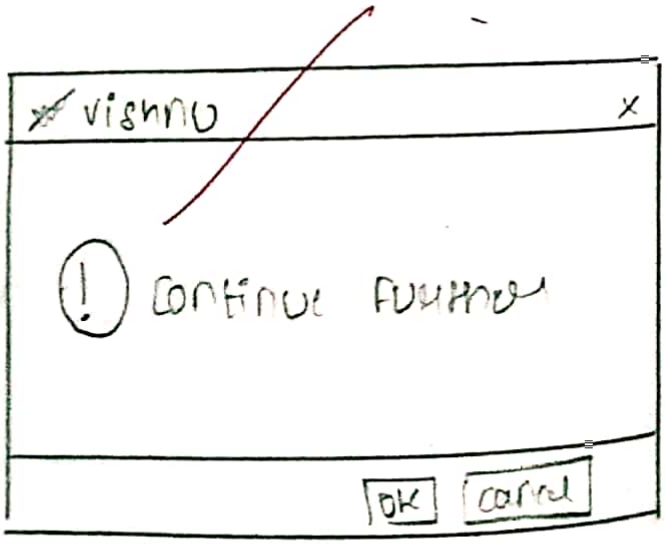




SA

```
from tkinter import *
top = Tk()
messagebox X
def msgb():
    messagebox. showinfo
    messagebox. askokcancel("vishnu" "continue running")
b1 = Button(top, text = "python", command = msgb
b1. pack().
top. mainloop().
```





→ askquestion().

1) define a function which will use the askquestion() from the messagebox library.

2) The attributes which a given method takes will specify the 2 strings.
   (1) related to the title of the window.
   (2) corresponding to the message displayed.

3) Now create an object from the button() and place it onto the parent window.

4) Terminate the program by calling mainloop().

```
from tkinter import *
def function():
    root = Tk()
    root.config(bg = "red")
    root.title("one window")
    root.minsize(200, 200)
b1 = Button(root, text = "Next window", command
        = new).pack(padx = 10, pady = 80)
root.mainloop().
def new()
    two = Tk()
    two.config(bg = "black")
    two.title("second window")
    two.minsize(100, 100)
b2 = button(two, text = "second window", command = exit
        .pack(padx = 20, pady = 40)
two.mainloop().
def exit():
    quit()
function().
```

```
from tkinter import *
top = Tk()
top.title("Vishnu")
top.config(bg="Red")
top.minsize(200,200)
Frame = Frame(top)
leftFrame = Frame(top, width=100, height=100, bg="green"
            .grid(row=0, column=0, padx=10, pady=15)
rightFrame = Frame(top, width=75, height=75, bg="yellow"
            .grid(row=0, column=3, padx=15, pady=20)
label(leftFrame, text="left", bg="Yellow")
    .grid(row=0, column=0)
label(rightFrame, text="Right", bg="Black")
    .grid(row=0, column=1)
image = PhotoImage(file="Desert.gif")
on_image = image.subsample(3,3)
label(leftFrame, img=on_image)
    .grid(row=0, column=1, padx=10, pady=20)
label(rightFrame, img=on_image)
    .grid(row=0, column=1, padx=10, pady=20)
label(toolbox, text="Title", relief=SUNKEN)
    .grid(row=0, column=1)
def obj()
    print("New Title")
toolbox=Frame(leftFrame, width=100, height=100)
    .grid(row=2, column=0, padx=10, pady=10)
```


write a program to insert an image in the Frame widgets using the other widgets.

1) Create the parent window object and use the method title, config and minsize with this object.

2) Create an object from the Frame method & place it onto the parent window object with width, height & bg colour and use the grid method() along with row & column attribute as (0,0) with some external padding.

3) Similarly create the rightFrame object from the Frame method with row & column attribute making the values (0,1).

4) Use the label() & the parent window object corresponds to left Frame with text & relief attribute and use the grid method with row & column value as (0,0)

5) Similarly create the label for the right Frame and use the title & row, column values as (0,1).

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

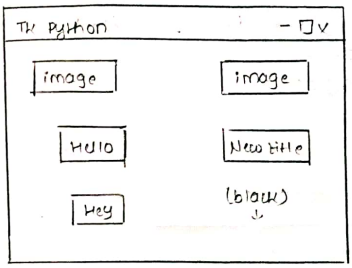
7) Now use the label() using the leftframe & the image attribute and the row, column value specified in the grid().

8) Similarly create the label() using rightframe object with the image attribute & same bg colour with specified at (0,0).

9) Now define a function using print statement which shall be called on clicking the combobox

10) Now use the mainloop method to terminate the given program.

OUTPUT:-



12.

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

output: