No.	Title	Page No.	Date	Staff Member's Signature
1.	Demonstrate the use of different file accessing modes. different attributes read method	21	29/11/19	
2.	Programs with iterables and iterator	24	6/12/19	
3.	Program to demostrate exception handling	27	20/12/19	
4.	Demonstrat the use of regular expression	30	3/1/2020	
5.	GUI components	37	17/1/2020	
5	Components of GUI	44	1/2/2020	
5.	GUI component	51	6/2/2020	
6.	Data connectivity	58	19/2/2020	

Pratical - 1

Objective :- Demonstrate the use of different file accessing modes, different attributes, read method.

Algorithm :-

step 1 :- Create a file object using open method and use the write accessing mode followed by writing some contents onto the file and than closing the file.

step 2 :- 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 3 :- Now use the file object for finding the name of the file, the file mode in which it opened whether the file is still open or close and finally the output of the softspace attribute

step 4 :- Now open the file obj in write mode write some another content, close subsequently. Then again open the file obj in 'w+' mode that is the update mode and write contents.

```
file obj = open ("abc.txt", "w")
file obj.write ("Science Subjects" + "\n")
file obj.write ("Physics\n, Chemistry\n, Maths\n")
file obj.close ()

file obj = open ("abc.txt","r")
# read ()
str1 = file obj.read ()
print ("The output of read method :", str1)
file obj.close ()
```

>>> The output of read method : 'science subject\n Physics\n chemistry\n Maths\n'

```
# readline ()
file obj = open ("abc.Text","r")
str 2 = file obj.readline ()
print ("The output of readline method :", str 2)
file obj.close ()
```

>>> ('The output of readline method :', 'Science subjects\n')

```
# readlines ()
file obj = open ("abc.txt","r")
str 3 = file obj.readlines ().
print ("The output of readlines method :", str 3)
file obj.close ()
```

>>> ('The output of readlines method :' ['Science subjects\n', 'Physics\n', 'Chemistry\n', 'Maths\n']

```
# file attributes
a = file obj.name
print ("name of file (name attribute) :", a)
```

>>> (name of file (name attribute) :', 'abc.txt')

```
b = file obj.closed
print ("(close) attribute :", b)
```

>>> ('(close) attribute :', True')

Step 5 :- Open file obj in read mode display the update written content and close open again in 'r+' mode with parameter passed and display the output subsequently.

Step 6 :- Now open file obj in append mode open write method write content close the file obj again open the file obj in read mode and display the appending output.

Step 7 :- Open the file obj in read mode. declare a variable and perform fileobject dot tell method and store the output consquently in variable.

```
# tell()
file obj = open("abc.txt","r")
pos = file obj.tell()
print("tell():", pos)
file obj.close()
>>> ('tell():', 0L)

# seek()
file obj = open("abc.txt","r")
st = file obj.seek(0,0)
print("seek(0,0) is :",st)
file obj.close()
>>> (seek(0,0) is :', None)
file obj = open("abc.txt","r")
st 1 = file obj.seek(0,1)
print("seek(0,1) is :", st 1)
>>> ('seek(0,1) is :', None)
file obj = open("abc.txt","r")
st 2 = file obj.seek(0,2)
print("seek(0,2) is :", st 2)
file obj.close()
>>> ('seek(0,2) is :', None)

# finding length of different lines exist within lines
file obj = open("abc.txt","r")
st at = file obj.readlines()
print("output :", stat)
for line in stat:
        print(len(line))
file obj.close()
>>> ('output :', ["Physics"])
```

```
c = file obj. mode
print ("file mode", c)
>>> ("file mode", 'r')
d = file obj . softspace
print ("softspace', d')
>>> ('softspace :', 0)
```

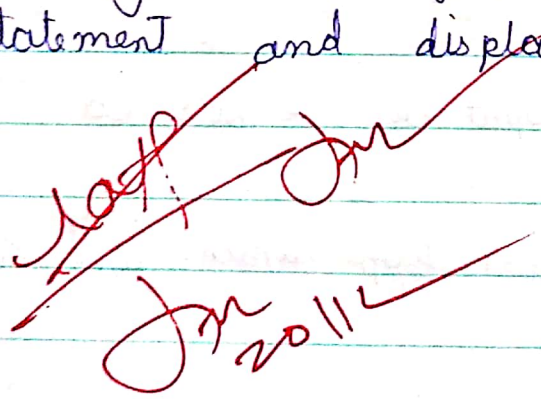
```
# w+ mode
file obj = open ("abc.txt","w+")
file obj . write ("loukik sir")
file obj . close ()
```

```
# write mode
file obj = open ("abc.txt", "w")
file obj . write ("DBMS")
file obj . close ()
```

```
# read mode
file obj = open ("abc. txt", "r")
  s = file obj . read ()
  print ("output of read mode!", s)
>>> ('output of read mode !', 'loukik sir')
```

step 8 : Use the seek method with the arguments with opening the file obj in read mode and closing susquently.

step 9 : Open file obj with read mode also use the readlines method and store the output subsquently in and print the same for counting the length use the for conditional statement and display the length



Pratical no - 2

Aim :- Programs with iterables and iterators.

Step 1 :- Create a tuple with the elements that will need to iterate using the iter and next method. The number of time we use the iter and next method we will get the next iterating element in the tuple display the same.

Step 2 :- The similar output can be obtained by using for conditional statement for iterable variable is to be developed in for loop which will iterate

Step 3 :- Define a function name square with a parameter which will obtain output of square value of the given numbers in similar way declare cube to get the value raised by 3 and return the same

Step 4 :- Call the declared function using function call.

iter() and next()

```
vowels = ['a', 'e', 'i', 'o', 'u']
vowels iter = iter(vowels)
print (next(vowels iter))
print (next(vowels iter))
print (next(vowels iter))
print (next(vowels iter))
print (next(vowels iter))
print (next(vowels iter))
```

```
>>> a
    e
    i
    o
    u
```

Odd numbers.

```
def __iter__(self):
    self.num = 1
    return self
def __next__(self):
    if self.num <= 10:
        num = self.num
        self.num += 2
        return num
    else:
        raise StopIteration

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

```
>>> 1
    3
    5
    7
```

Power 2

```
class power :
    def __iter__(self):
        self.p = 0
        return self
    def __next__(self):
        if self.p <= 10:
            num = self.p
            self.p += 1
            p0 = 2 ** num
            print("2 ** ", self.p-1, " = ", p0)
            return p0
        else:
            raise StopIteration
```

```
>>> p = power()
>>> x = iter(p)
>>> x.next()
    2 ** 0 = 1
>>> x.next()
    2 ** 1 = 2
```

Step 5 : Using for conditional statement specifying the range use the list tuple casting with map method declare a 'lambda' i.e anonymous function and print the sum.

Step 6 : Define a list num variable and declare some elements then use the map method with help of lambda function give two arguments display the output.

Step 7 : Define a function even with a parameter then using conditional statement do check whether the number is even and odd and return respectively.

Step 8 : Define a class and within that define the iter() method which will initialize the first element within the conditional object.

Step 9 : Now use the next() and define the logic for displaying odd value.

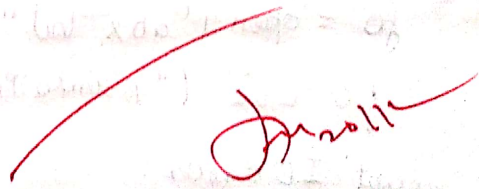
step 10 : Define an object of a class

step 11 : Accept an number from the user till the limit which we want to display the given condition.

factorial

```
class factorial :
    def __iter__(self):
        self.a = 1
        self.b = 1
        return self
    def __next__(self):
        if self.a <= 10
            self.b = self.a * self.b
            x = self.b
            self.a += 1
            return x
        else :
            raise StopIteration

my class = factorial()
my data = iter(my class)
for x in my data:
    print(x)
```



Program

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

Output -

Enter class : 467

Program

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

Output :

Operation carried out successfully

Pratical - 3

Aim :- Program to demonstrate exception handling.

1] Write a program using the exception method of the natural Arithmetic error

Step 1 :- Use the try block and accept the input using the raw input method and convert it into the integer datatype and subsquently terminate the block.

Step 2 :- Use the except block with the exception nature 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 :- With in 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.

Write a program for using the assert() to check if the list elements are empty.

step 1 :- Define a function which accepts our arguments and then check using the assert statement whether the given list is an empty list and accordingly return the message.

step 2 :- Close the function and in the body of the program define certain elements in the list & take some appropriate actions

Write a program to check the range of the age of the students in a given class and if the age do not fall in the given range, use the value Error exception otherwise return the valid number.

step 1 :- Define a function which will accept the age of the students from the standard input.

step 2 :- Use the if condition to check whether the input range is false in the given range & so return the age else use the Value Error exception

Assert error

```
def assert_(n):
    assert (len(n) == 0)
    print ("list is empty")
var1 = []
print (assert_(var1))
print ("division is : ", division)
```

Output :

List is empty

Program

```
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()
        valid = True
        print ("Valid age")
    except ValueError:
        print ("Not a valid age")
```

Output :

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

step 3 :- Define the while loop to check whether the boolean expression holds True, use the try block to accept the age of the students and terminate the looping condition.

step 4 :- Use for except with the ValueError and print the message not a valid age.

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 re module and common usage of regular expression involves following functionalities.

* Searching a given string
* Finding a given string
* Breaking a string into smaller substring.
* Replacing part of string.

Write a regular expression regarding numeric and alphabetic values from a given string.

Algorithm

Step 1 :- Now apply string & pattern in findall() and display the output

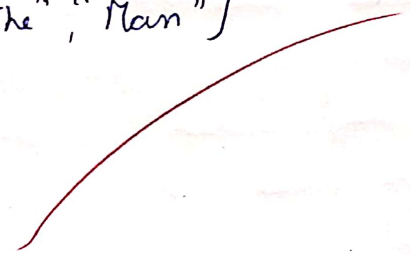
Step 2 :- \d is used for matching all decimal digit whereas is used to match non-decimal digits

code

```
import re
string = "The Man1"
result = re.findall ["\d", string]
result = re.findall [":\", string]
print (result)
print (result+1)
```

Output

```
>>> ["1"]
>>> ["The", "Man"]
```



code 2

```
import re
string = "Python is an important language"
result = re.search ["\A python", string]
print (result)
if result :
    print ("Match found")
else :
    print ("Match not found")
```

Output

```
>>> re match object span = (0,6)
                  match = "Python"
>>> match found.
```

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

Step 1 :- Import re module and apply a string.

Step 2 :- use search() with "\Apython" and string as the parameter.

Step 3 :- Now display the output.

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

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

Algorithm.

Step 1:- Import re module and apply a string of module no. 8.

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

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

code 3:-

```
import re
li = ["9876543215", "8855915721", "765432108",
      "6543210896"]
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
    9876543215
    Correct mobile no :
    8855915721
    Incorrect mobile no:
    Incorrect mobile no.
```

code 4

```
import re
string = "Python is important"
string1:
result 1 = re.findall ("\w", string)
result 2 = re.findall ("\w+", string)
print (result 1)
print (result 2)
```

Output

```
>>> ["Python", " is ", " important"]
>>> ['Python', 'is', 'important']
```

4] Write a regular expression for extracting a word from given string along with space characters in between the word and subsquently entered the word without space character

Step 1:- Import re module and apply a string.

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

Step 3:- Use "\w" to extract a word along with space and use "\w," to extrad word without space

Step 4:- Now display the output.

Write a regular expression for extracting first and last word from a string.

Step 1 :- Import re module and apply a string

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

Step 3 :- Display the output

code 5

```
import re
string = "Python is important"
result = re.findall ['^\w+', string]
result = re.findall ['\w+', string]
print (result)
print (result+1)
```

Output :

```
>>> ['Python']
>>> ['important']
```

code 6

```
import re
string = "Ram 45 24-12-2019"
result = re.findall ["\d{2}-\d{2}-\d{4}", string]
print (result)
```

Output

```
>>> ['24-12-2019']
```

6) Write a regular expression for extracting the data in format dd mm yyyy by using the findall() where the string has following format 24-12-2019

step 1:- Import re module and apply string

step 2:- Use findall method and use '\d 23-\d 23-d 23' as parameter.

step 3:- Now display the output.

7. Write a re module for extracting the
① User name from email-id
② Host name from email-id
③ both username & host name from email-id.

Algorithm :-

Step 1 :- Import re module and apply a string

Step 2 :- Use findall() to find username, hostname and both as email-id

Step 3 :- Use "\w+" for username use "@\w.\w+" for hostname and use "[\w\.]+" for as parameter in findall()

Step 4 :- Display the output

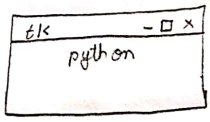
code 7

```
import re
seq = 'abc.tcsc@edu.com, xyz@gmail.com'
pattern = r'[\w\.-]+[\w\.-]'
output = re.findall(pattern, seq)
print(output)
```

creation of parent window

```
from Tkinter import *
root = Tk()
l = Label(root, text = "python")
l.pack()
root.mainloop()
```

OUTPUT :-



#2

```
from Tkinter import *
root = Tk()
l = Label(root, text = "python")
l.pack()
l1 = Label(root, text = "CS", bg = "grey", fg = "black",
           font = "10")
l1.pack(side = LEFT, padx = 20)
l2 = Label(root, text = "CS", bg = "light blue", fg = "black",
           font = "20")
l2.pack(side = LEFT, pady = 40)
root.mainloop()
```


Pratical - 5 (A)

Gui Components.

Algorithm

1. Use the tkinter library 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 and use the text method.

step 4 :- Use the mainloop() for triggering of the corresponding above mention units.

~~step 5~~ :-

2 :-

step 1 :- Use the Tkinter library for importing the features of the text editor.

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 created from the text() and use the parameter.

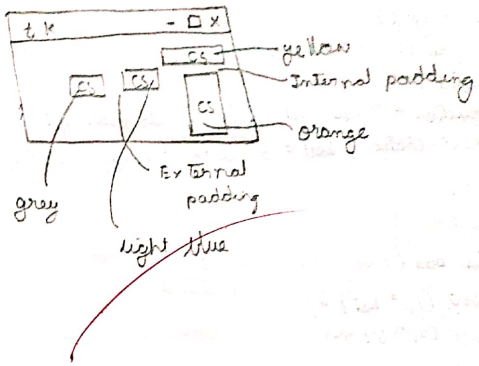
1) side = LEFT , padx = 20
2) side = LEFT , pady = 30
3) side = TOP , ipadx = 40
4) side = TOP , ipady = 50

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

Step 5 :- Now repeat above steps with the label() which takes the following argument.

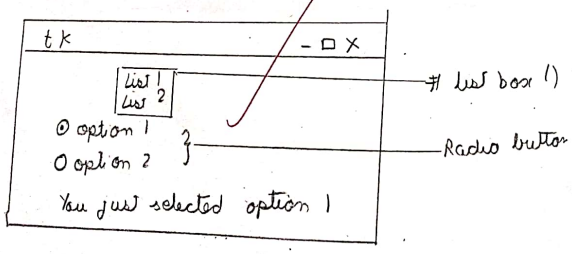
1) Name of the parent window
2) Text attribute which defines the string.
3) The background colour (bg) use the pack(). with a revelant padding attributes.

OUTPUT :-



```
# Radio Button
from tkinter import *
root = tk()
root.geometry("500x500")
def select():
    selection = "you just selected" + str(var.get())
    label.config(text = selection, justify = LEFT)
root = Tk()
var = IntVar()
l1 = Listbox()
l1.insert(1, "list 1")
l2.insert(2, "list 2")
l1.pack(anchor = W)
r1 = RadioButton(root, text = "option 1", variable = var,
        value = 1 "Option 1", command = select)
r1.pack(anchor = W)
r2 = RadioButton(root, text = "option 2", variable = var,
        value = "option 2", command = select)
r2.pack(anchor = W)
root.mainloop()
```

Output :



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 specific pixel size of the parent window

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

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

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

Step 6 : Create an object from radio button which will take following arguments & parent window object text variable which will take the values option no1, 2, 3, ..., variable argument, corresponding value &

Step 7 :- trigger the function declared

Step 8 :- Now call the pack() for radio object is created and specify the grey argument using anchor attributes.

Step 9 :- 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 window

Step 4 :- Create an object and use the scrollbar()

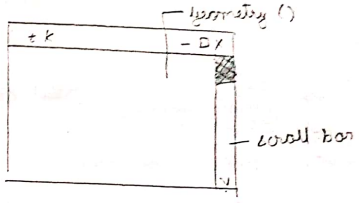
Step 5 :- Use the pack() along with the scrollbar object with side and fill attribute.

Step 6 :- Use the mainloop with the parent object.

2 scroll bar

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

Output :

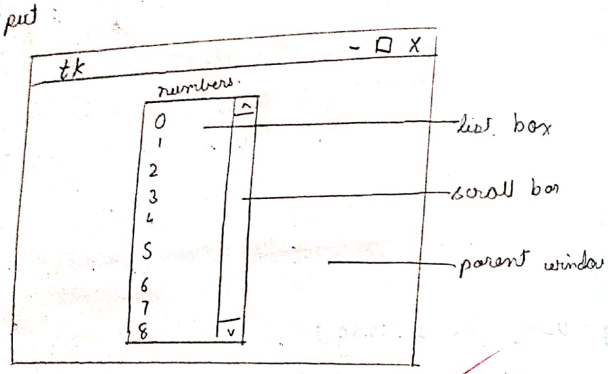


3 Using frame widget

```
from tkinter import *
window = Tk()
window.geometry("680x500")
Label(window, text = "numbers :").pack()
frame = Frame(window)
frame.pack()
list nodes = List box(frame, width = 20, height = 20,
                      font = "Times new Roman", 10)
list Nodes.pack(side = "LEFT", fill = Y)
scroll bar = Scroll bar(frame, orient = "vertical")
scrollbar.config(command = listnodes.yview)
```

```
scrollbar.pack(side="RIGHT", fill=Y)
for x in range(100):
    list nodes.insert(END, str(x))
window.mainloop()
```

Output :



3 :

Step 1 : Import the relevant ~~f~~ 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 × 500) or any other suitable pixel value.

Step 4 : Use the Label widget along with the parent object created and subsquently 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 attributes like width, height, font. To 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 : Trigger the events using mainloop().

#4.

Step 1 :- Import relevant methods libraries from tkinter.

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

Step 3 :- Now define the frame object from the method and place it on the parent window.

Step 4 :- Create another frame object termed as the left frame and init it on the parent window on it LEFT side.

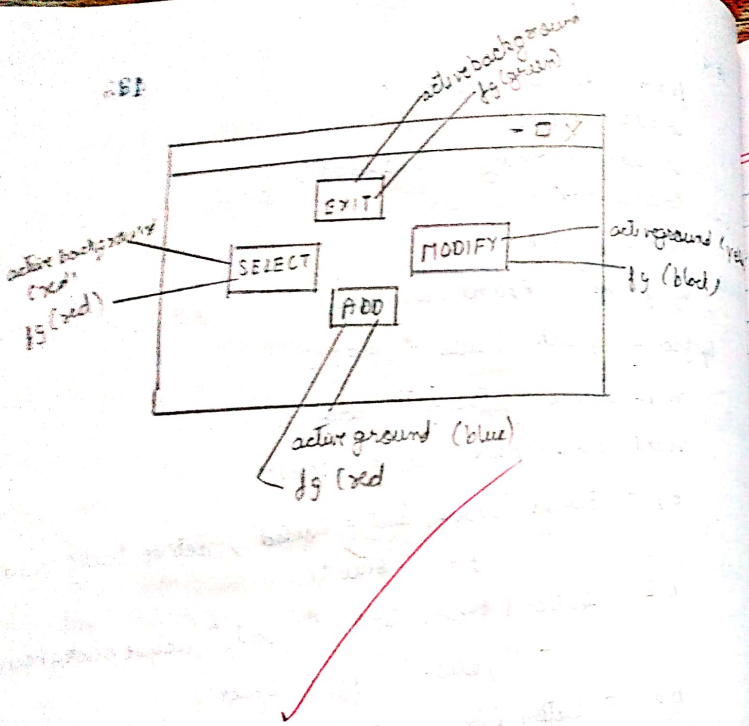
Step 5 : Similarly define the RIGHT frame and subsequently define the button object placed onto the given frame with the attribute as text, active background and fore ground.

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

Step 7 :- Similarly create the button object corresponding to the MODIFY operation it into frame object on side = "RIGHT".

#4.

```
from Tkinter import *
window = Tk()
window.geometry("680x500")
frame = Frame(window)
frame.pack()
leftframe = Frame(window)
leftframe.pack(side="LEFT")
rightframe = Frame(window)
rightframe.pack(side="RIGHT")
b1 = Button(frame, text="select", activebackground="red",
            fg="blue")
b2 = Button(frame, text="Modify", activebackground
            ="yellow", fg="black")
b3 = Button(frame, text="ADD", activebackground
            ="red", fg="green")
b4 = Button(frame, text="EXIT", activebackground
            ="red", fg="yellow")
b1.pack(side="LEFT", padx=20)
b2.pack(side="RIGHT", padx=30)
b3.pack(side="BOTTOM", pady=20)
b4.pack(side="TOP"))
```



step 8 :- Create another button object & place it on to the RIGHT frame & label the button as ADD

step 9 :- Add another button & place it on to the top of frame and label it as EXIT

step 10 :- Use the pack() simultaneously for all the objects & finally use the mainloop().

Pratical - 5

Components of GUI (Button, attribute, message box)

→ WAP on various attributes which a button widget assume related to the relief attribute.

① define a button object and place it on the top of object 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.

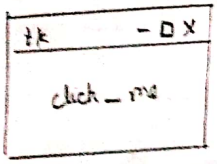
④ For positioning the widget object on to the parent window and trigger the corresponding event by calling the ~~pe~~ mainloop method.

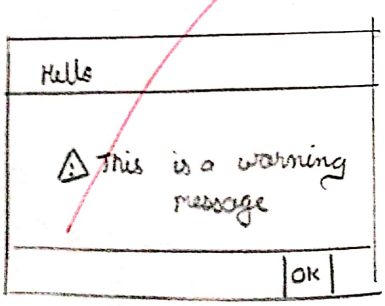
Code


```
from tkinter import *
top = Tk()
b1 = Button (top, text = "Sunken", relief = SUNKEN) . pack ()
b2 = Button (top, text = "raised", relief = RAISED) . pack ()
b3 = Button (top, text = "groove", relief = GROOVE) . pack ()
b4 = Button (top, text = "ridge", relief = RIDGE) . pack ()
b5 = Button (top, text = "flat", relief = FLAT) . pack ()
top . mainloop ()
```

```
from tkinter import *
msg1 = Tk()
def msg():
    print("ok")
    messagebox.showwarning("hello", "This is a warning
                message").pack()

b = button(msg1, text = "click_me", command = msg).pa
msg1.mainloop()
```





⇒ showwarning()

step 1 : Define a function which will use the showwarning() derived from the messagebox library.

step 2 : The attributes which a given method takes will specify 2 strings one related to the message box displayed (11) corresponding to the message.

step 3 : Now create an object from the method and place it on the parent window with the title of the button object separated and finally use the command attribute to execute the revelant function.

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

2. WAP to implement the message box widget.

→ showinfo()

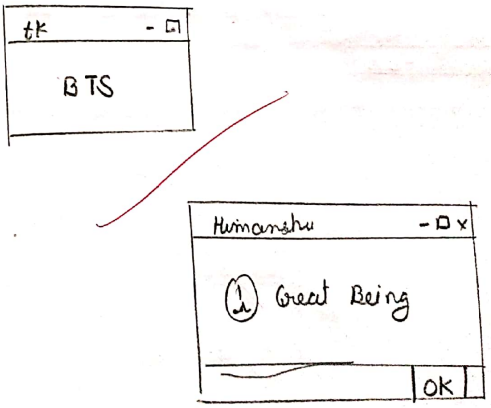
step 1 : Define a function which will use the showinfo() derived from the message box class.

step 2 : The attributes which a given method takes will specify two string. One related to the title and correspond to the message display.

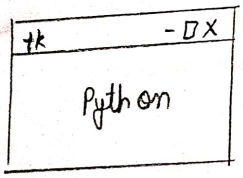
step 3 : Now create an object from the button method and place it on to the parent window with title of the button object and finally use command attribute.

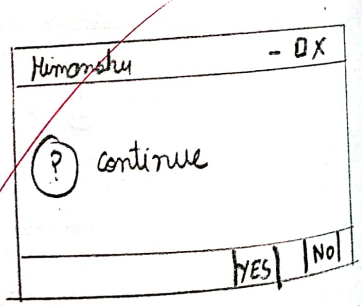
step 4 : Terminate by calling mainloop method.


```
from tkinter import *
msg1 = Tk()
def msg():
    print("ok")
    messagebox.showinfo("hello", "Great Being")
b1 = Button(msg1, text = "Bts", command = msg).pack()
msg1.mainloop()
```



```
from tkinter import *
top = Tk()
def msgb():
    messagebox.askyesno("Himanshu", "continue")
b1 = Button(top, text = "Python", command = msgb)
b1.pack()
top.mainloop()
```





→ askyesno()

step 1 : Define a function which will use the askyesno() derived from the messagebox library

step 2 : The attribute which a given method takes of the window (1) related to the title of the window (11) corresponding to the message displayed

step 3 : Now create an object from the button method & place to the parent window with the title of the button object specified and finally use the command attribute to execute function

step 4 : Terminate the program by calling the mainloop()

→ showError()

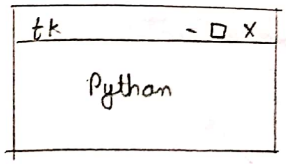
step 1 :- Define a function which will use the showerror() of the derived from the message box library

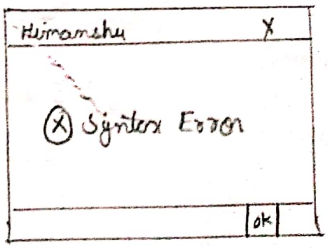
step 2 :- The attributes which a given method takes will specify the 2 string :
(i) related to the title
(ii) corresponding to the message box.

step 3 :- Now create another object from the button method and place it on to the parent window with the help of the button object

step 4 :- Terminate the program by using mainloop()

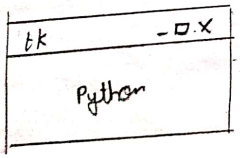
```
from tkinter import *
top = Tk()
def msgb():
    messagebox.showerror("Himanshu", "Syntax Error")
b1 = Button(top, text = "python", command = msgb)
    .pack()
top.mainloop()
```

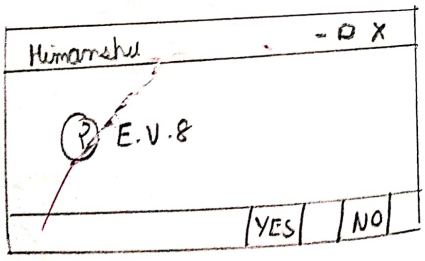




8)

```
from tkinter import *
top = Tk()
def msgb():
    messagebox.askquestion("Himanshu", "E.V.S")
b1 = Button(top, text = "python", command = msgb).
pack()
top.mainloop()
```





→ askquestion()

step 1 : define a function which will use the askquestion() from the message box library.

step 2 : The attributes which a given method takes will specify the 2 string.
(i) related to the title of the window
(ii) corresponding to the message displayed

step 3 : Now create another object from the button method and place on to parent window.

step 4 : Terminate the program by using mainloop().

→ ask ok cancel ()

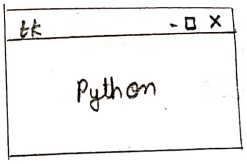
Step 1 : Define a function which will use the ask ok cancel from the message box library.

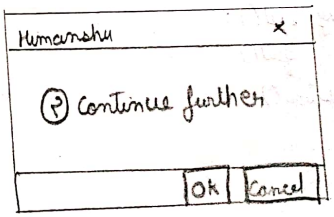
Step 2 : The attribute which gives method takes will specify 2 strings
(i) related to the title of parent window.
(ii) corresponding to the message displayed.

Step 3 : Now create another object from the button method and place it on the parent window

Step 4 : Terminate the program by using mainloop()


```
from tkinter import *
top = Tk()
def msgb():
    messagebox.askokcancel("Himanshu", "Continue further")
b1 = Button(top, text="python", command=msgb).pack()
top.mainloop()
```





```
# 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)
    b1.pack(padx=10, pady=10)
    root.mainloop()

def new1():
    tree = Tk()
    tree.config(bg="black")
    tree.title("second window")
    tree.minsize(250, 200)
    b2 = Button(tree, text="secondwindow", command=exit).
        pack(padx=20, pady=40)

def exit():
    quit()
selection()
```


5(D)

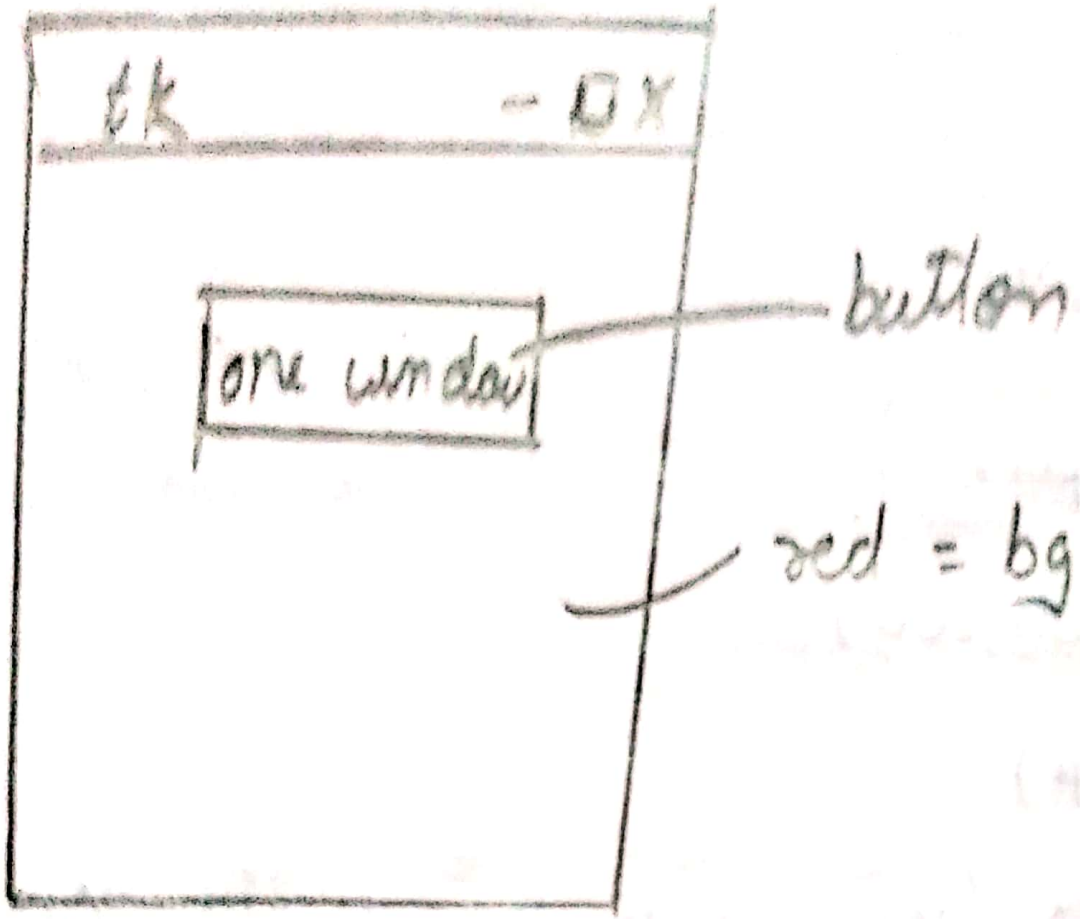
Write a program to move from one window to another window with the help of button widget.

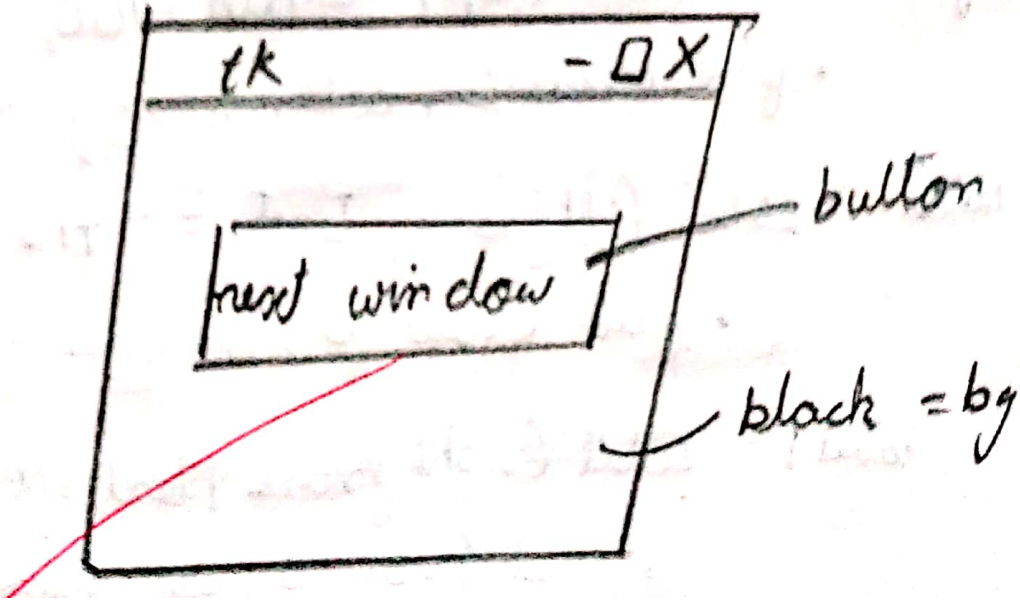
Step 1 :- Define a function & create a parent window object and use the config, title, minsize

Step 2 :- Now define a button object place it on to the parent window with a suitable title and use command attribute to call the next function using the ~~guild~~ grid method specifying external paddin

Step 3 :- Now define a function corresponding to second window and create another parent window object wit method config, title and minsize and again place the button object calling the next function.

Step 4 :- similarly create the function and use the button widget and finally create a function which will terminate the aggregate function by using Quit method.





```
from tkinter import *
top = Tk()
top.title("The MAN")
top.config(bg = "pink")
top.minsize = (200,200)
frame = Frame(top)
leftframe = Frame(top, width=100, height=100, bg="green"
    .grid(row=0, column=0)
rightframe = Frame(top, width=100, height=100, bg="yellow"
    .grid(row=0, column=0)
label = Label(leftframe, text="The", bg="yellow")
    .grid(row=0, column=0)
label1 = Label(rightframe, text="MAN", bg="pink")
    .grid(row=0, column=0)
image = PhotoImage(file="Desert.gif")
ori_image = image.subsample(1,2)
```


WAP to insert an image in the frame widget using the other widget

Step 1 :- Create the parent window object and use the method title, config and minsize with this object

Step 2 : Create an object from the frame method and place it onto the parent window object with width, height and bg color and use the grid() method along with row and column attributes as (0,0) with same external padding.

Step 3 : similarly create the rightframe object from the frame method with row and column attribute making the value (0,1)

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

Step 5 :- similarly create the label for the right frame and use the title and row, column value as (0,

Step 6 :- Use the photo() with the file attribute specified and subsequently subsample() for

specifying the image object.

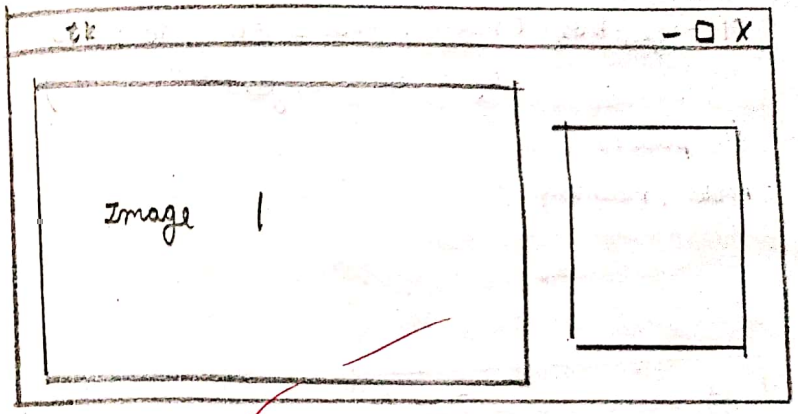
step 7 :- Now use the label() using the left frame and the image attribute and the row, column value specified in grid()

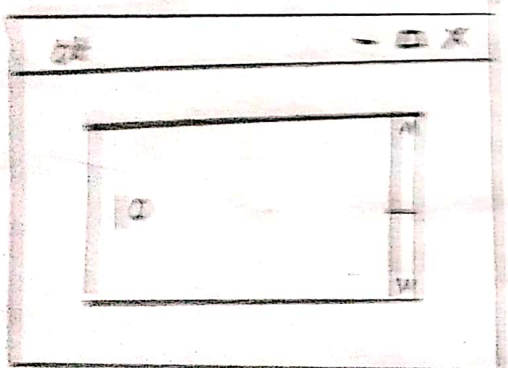
step 8 :- similarly create the label() using right frame object with the image attribute & same bg colour with specified as (0,0)

step 9 :- Now define a function using print statement which shall be called on clicking the window.

step 10 :- Create the label method position it onto the Toolbox with some title.

step 11 :- Now use the mainloop method to Terminate the given program.





Step 1 :- Use the tkinter library to import the required methods

Step 2 :- Create the parent window object

Step 3 :- Create an object by using geometry method and place it on the parent window with specified size

Step 4 :- Now use the pack method() to make object visible on the parent window and then use the mainloop method to execute the program

Practical - 5 (E)

* Paned window

Step 1 :- Create an object from the paned window method and use the pack method to make this object visible

Step 2 :- Now create an object from the entry widget and placed it onto the paned window and use the add method, similarly, create an object of a paned window

Step 3 :- Create an object from the scale widget and place it onto the preceding paned window and use the add method accordingly.

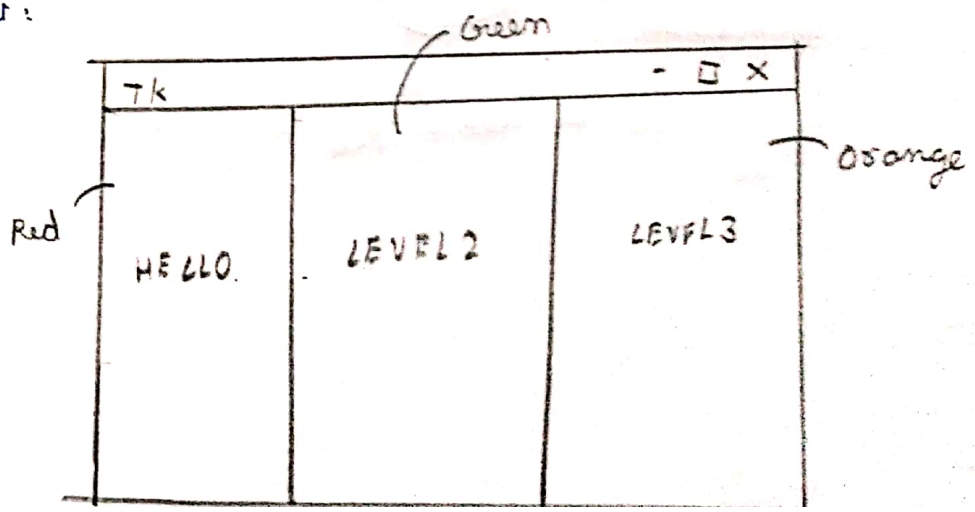
Step 4 :- Create a button widget and place it onto the paned window define a functionality along with the button widget.

Step 5 :- Use the pack method & mainloop method for the corresponding event to trigger.

code

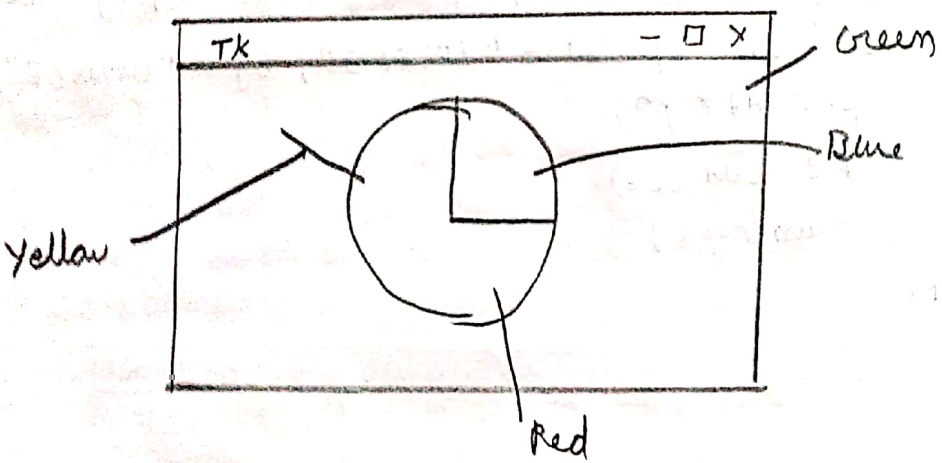
```
from tkinter import *
root = Tk()
p1 = PanedWindow(bg = "Pink").pack(fill = BOTH, expand = 10)
l1 = Label(p1.text = "HELLO", bg = "RED")
p1.add [l1]
p2 = PanedWindow(p1.orient = VERTICAL , bg = "YELLOW")
p1.add (p2)
l2 = Label (p2.text = "LEVEL 2", bg = "GREEN")
p2.add (l2)
p3 = PanedWindow (p1.orient = HORIZONTAL , bg = "BLUE")
l3 = Label (p1.text = "LEVEL 3", bg = "ORANGE")
p1.add (p3)
p3.add (l3)
mainloop()
```

Output :



Code:

```
from tkinter import *
root = Tk()
cl = Canvas(root, height = 5000, width = 5000, bg = "green")
oval = cl.create_oval(300, 12, 12, 400, fill = "RED")
line = cl.create_line(30, 20, 70, 60, fill = "Yellow")
arc = cl.create_arc(300, 18, 18, 400, fill = "Blue")
cl.pack(side = TOP)
root.mainloop()
```



canvas widget.

step 1 :- Create an object from the canvas widget by using the attribute height width bg color & parent window object.

step 2 :- use the corresponding method for drawing the simple geometrical shape like arc, oval & line and specifie the co-ordenat values.

step 3 :- similarly use the create line & create oval method along with the co-ordinate value & the fill attribute for specifying the color

step 4 :- finally use the pack & main loop method.

Scanned with CamScanner

```
import os, sqlite3
con = sqlite3.connect ("student.db")
cur = con.cursor ()
cur.execute ("create table dss (name char, Roll no
                                int)")
cur.execute ("insert into dss value ('Himanshu', 1845),
              ('Ashay', 1852)")
con.commit ()
cur.execute ("select * from dss")
print (cur.fetchall ())
con.close ()
```

output :

[('Himanshu', 1845), ('Ashay', 1852)]

2

step 1 :- Import corresponding library to make data base connection, os and sql lib -3

step 2 : Now create the connection object using sql lib 3 library and the connect () for creating new data base

step 3 : Now create cursor object using the cursor and from the connection object create.

step 4 : Now use the the execute () for creating the table with the column name and respective datatype

step 5 : Now with cursor object use the insert statement for entering the values corresponding to different fields, corresponding the data type.

step 6 : Use the commit () to complete the transaction

step 7 : Use the execute statement along with cursor object for accessing the value from the data base. using the select from where clause

step 8 : Finally use the fetch() or fetchall () for displaying the values from the table using the cursor object. Execute () and drop table syntax for terminating the data base and finally use the close.