```
fileobj.open("abc.text", "a0")
fileobj.write("Dance tunes "+"\n")
fileobj.write("bangarang in revolution \n getlow")
fileobj.close()
file=open("abc.text", "r")
#read
str1=fileobj.read()
print("The output of read method:", str1)
fileobj.close()
>>>('The output of read method:', 'Dance tunes \n bangarang \n
                    revolution \n getlow \n')
#readline()
fileobj.open("abc.text", "r")
str2 = fileobj.readline()
Print("Output:", str2)
fileobj.close()
>>> ('Output:', 'Dance tunes \n')

#readlines()
fileobj=open("abc.text", "r")
str3 = fileobj.readlines()
Print("Output:", str3)
fileobj.close()
>>> ("Output:', ['Dance tunes \n bangarang \n revolution \n getlo
# file attributes
a: fileobj.name
Print("name of file(name attribute):", a)
>>> (name of file (name attribute): 'abc.text')
b: fileobj.closed
Print("(close) attribute:", b)
>>> ("(close) attribute:", 'true')
```



## PRACTICAL NO: 1

OBJECTIVE:- Demonstrate the use of different file accessing mode different attributes read methods.

Step 1: Create a file object using open method and use the write access mode followed by writing some contents onto the file and then closing the file.

Step 2: Now open the file in read mode and then use read(), readline() and readlines(). and readlines() and store the output in variable and finally display the contents of variable.

Step 3: Now use the fileobject for finding the name of the file, the file mode in which it is opened whether the file is still open or close and finally the output of the softspace attribute.

Step 4: Now open the fileobj in cwrite mode write some another content close subsequently then again open the fileobj in 'w+' mode that is the update mode ad write contents.

Step 5:- Open fileobj in read mode display the update written contents ad close. Open again in 'r+' mode with parameter passed and display the output subsequently.

Step 6: Now open fileobj in append mode open write method write contents close the fileobj again open the fileobj in read mode and display the appending output.

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



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

# r+ mode
fileobj = open ("abc.text", "r+")
s1 = file.obj.read(5)
print ("output of r+", s1)
fileobj.close()
>>> (output of r+, 'skill')
# append mode.
fileobj = open ("abc.txt", "a")
fileobj.write (" Ragga bomb")
fileobj.close()
fileobj.open ("abc.txt", "r")
s3 = file.read()
print ("Output of append mode :", s3)
fileobj.close()
>>> ['output of append mode :', ' skillu Ragga bomb')
```

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

# read mode
fileobj.open ("abc.txt","r")
s = file.obj.read()
Print ("output of read mode :"s)
>>output of read mode :', ' Diple')
```

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

# seek()
filobj = open ("abc.txt", "r")
St = filobj.seek (0,0)
print ('seek(0,0) is :" St)
filobj, close()
>>> C'seek (0,0) is:', none)
filobj = open ("abc.txt", "r")
pos2 = filobj.seek (0,1)
St2 = filobj.readline (8)
Print ('seek (0,1) is:", St2)
filobj.close().
>>> ('seek (0,1) is', 'Skitter')
# Finding length of different line exist
filobj = open ("abc.txt", "r")
Stat = filobj.readline()
Print ("output:", Stat)
for line in Stat:
  print (len(line))
filobj, close()
>>> ('Output:' [skillet Rega Lomb])
```



Step 7 : Open the filobj in read mode, declare a variable and perform filobj dot tell method and store the output consequently in variable.

Step 8 :- ~~Open filobj with read mode also use~~ Use the the seek method with the arguments with opening the file obj in the read mode and closing subsequently.

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

# PRACTICAL NO: 2

Objective : Iterators

Step 1: Create a tuple with elements that we need to iterate using the iter and next method. no. of times 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. An iterable variable is to be declared 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 number. In similar fashion declare cube to get the value raised 3 and return the same

Step 4: Call the declared function using function call.



```
# iter() and next()
mytuple1 = ("banana", "orange", "apple")
myiter1 = iter(mytuple)
print(next(myiter1)
myiter2 = iter(mytuple)
print(next(myiter2)
myiter3 = iter(mytuple)
print(next(myiter3)
>>> banana
    orange
    apple
# for loop
mytuple1 = ("kevin", "shaun", "bob")
for x in mytuple1
print(x)
>>> kevin
    Shaun
    bob
# square and cube
def square(x)
y = x**x
return y
def cube(x):
z = x****x
return z
wit1 = [square, cube]
```

```
for x in range(5):
    value = list(map(lambda X: X(x), funct))
    print(value)

>>> [0,0]
    [1,1]
    [4,8]
    [9,07]
    [16,64]
```

```
# map[]
list num= [0,4,5,7,9,11,13,15,20,19,25]
list num= list map(lambda x: x*.5, list num))
print(list num)
def even(x):
  if (x%2==0)
     return "EVEN"
  else:
     return "ODD"
list(map(even list num))
```

```
# odd numbers
class odd:
    def __iter__(self):
    self.num1
      return self
    def __next__(self):
      num.self = num
      num.self+=2
    def __next__(self):
      num = self.num
      self.num+=2
        return num
```



Step 5: Using for conditional statement specifying and range use the list type casting with map method declare a 'lambda' ie annonyms functions and print the same.

Step 6: Declare a list num variable and declare some element then use the map method with help of lambda function given two argument and display the output.

Step 7: define a function given with a parameter then using conditional statement do check whether the number is even, odd & return respectively.

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

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

Step I: define an object of a class.

Step II: Accept an number from the user till which he want to display the odd numbers.



```
my obj = odd()
my iter = iter(myobj)
x = int(input("Enter a number"))
for i in myiter
    if(i<x):
        print(i)
```

```
>>> Enter a number : 15
1
3
5
7
9
11
13
```

```
# arithmatic error
while True:
    try:
        x = int(input("Enter class"))
        break
    except ValueError:
        Print("Enter numeric char")
```

Output:

Enter class : fycs

Enter numeric char:

Enter class : 13.

```
# environment error
try:
    fo = open("abc.txt","w")
    fo.write("Nisha")
except IOError:
    print("error writing on the file")
else:
    Print("Operation carried out successfully")
    fo.close()
```

Output:

Operation carried out successfully.

## PRACTICAL - 3

AIM :- To demostrate exception handling.

1] Write a program using the exception method of the nature Arithmatic error.

Step 1: Use the try block and except the input using the raw input method and convert it into the integers datatype and subsequent 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 title in a given mode and use the environment error as an exception for the given input.

Step 1: Within 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 incompatibility of the mode. Use the else block to display to display a message that the operation is carried out respectfully

3) WAP using the assert() to check if the list elements are empty.

Step 1: Define a function which accepts 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 program add define certain elements in the list and take some appropriate action

4) WAP to check the range of the age of the students in given class and if the age do not fall in given range else the value error exception otherwise return the valid no.

```
# assert error
def assert_(n):
    assert (len(n) == 0)
    print ("list is empty")
val l = [ ]
print (assert_(val l))
```

Output

list is empty

none

```
4] def acceptage():
       age = int(input("Enter your age:"))
       # age >30 or age <16.
         raise ValueError
       return age
   Valid: False
   while not valid:
     try:
       age = acceptage()
       valid = true
     except ValueError:
       Print ("not valid age")
```

Output:

Enter your age: 18.



Step 3: 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.

Step 4: Use except with ValueError and print the message not a valid range.

## Practical - 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 & for this we import re module and common useage of regular expression. It involves following functionalities.

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

Algorithm:

Step1: Now apply string & pattern in find all () and display the output.

Step2: \d is used for matching all decimal digits whereas \D is used to match non-decimal digits.



# code.

```
import re
string = "Hello 1234 abc 4567"
result = re.findall("\d", string)
result = re.findall("\d", string)
print(result)
print(result)
```

# output

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

Q6] Write a regular expression for finding the match string at the beginning of given sequence

Algorithm:

Step 1: Import remodule and apply a string.

Step 2: Use search () with "\A Python" and string as two 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.

```
# code 2
import re
string = "Python is an important language"
result = re.search("\A Python", string)
print(result)
# result:
  print("match found")
else:
  print("match not found")
```

# Output

» rematch object .span = (0,6)

match = "Python"

» match not found.

Q3 Write a regular expression to check whether the given mobile number starts with 8 or 9 and the total legth is 10.

Algorithm:

Step 1: Import the module and apply a String of mobile no. 8.

Step 2: Now use for conditional statement to find if the number starts with 8 or 9 and the total number should be length of 10 Use match() inside for the 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

```
import re
li=['9867854321098',"875432109","7654321098","654321075"
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.
9867854 3210
correct mobile no.
8676543 2109
Incorrect mobile no.

```
# code
import re
string = "Python is important"
result1 = re.findall ("\wt", string)
result2 = re.findall ("\wt", string)
print (result1)
print (result2)

# Output
>>> [python']
```



1) 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 character

Algorithm :-

Step 1: Import the module and apply a string

Step 2: Use find all () to extract a word from given string

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

Step 4: Now display the output.

Q.5] Write a regular expression for extracting first and last word from a string.

Algorithm :-

Step 1 : Import re module and apply a string.

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

Step 3 : Display the output.



# code.

```
import re
String = "Python is important"
result = re.findall("\n\w+", string)
result1 = re.findall("\w+$", string)
print(result)
print(result1)
```

# output

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

```
#code
import re
string = "Nisha. 201. 24-12-2019"
result = re.findall ('\d{2}-\d{2}-\d{4}', string)
print (result)

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



Q.6] Write a regular expression for extracting the data in form dd-mm-yyyy by using the findall() even the string has following format NISHA. 201. 24-12-2019

Algorithm:

Step 1: Import re module and apply string.

Step 2: Use findall method and use \d 23-\d{2}-\d{4}") as an parameter.

Step 3: Now display the output.

07) Write a re for extracting the
(i) Username from email-id.
(ii) Host name from email-id
(iii) Both username & hostname from email-id

Algorithm:-

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

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

Step 3: Use "\w+" for username use "t\w.\. wt$" for hostname and use "[\w\.-]+" for both as parameter in findall[].

Step 4: Display the output



# code

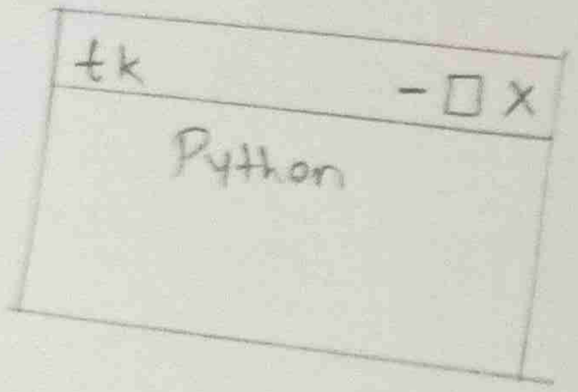
```
import re
string = "abc@tcsc.do"
result1 = re.findall("\w+.\w+$",string)
result2 = re.findall("\w+.\w+.\w+$",string)
result3 = re.findall("[\w\.-]+",string)
print(result1)
print(result2)
print(result3)
```

# output

```
>>> ['abc']
>>> ['tcsc.pdw']
>>> ['abc', 'tcsc', 'edu']
```

```
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 = "30")
l3.label (root, text = "CS", bg = "yellow", fg = "black"
    font = "10")
l3.pack (side = Top, ipadx = 40)
```



## PRACTICAL - 5

TOPIC - gui components

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

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

Step 3: Create an variable using the widget Label and use the text method.

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

#2 :

Step 1: Use the tkinter library for importing the features of the text 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 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

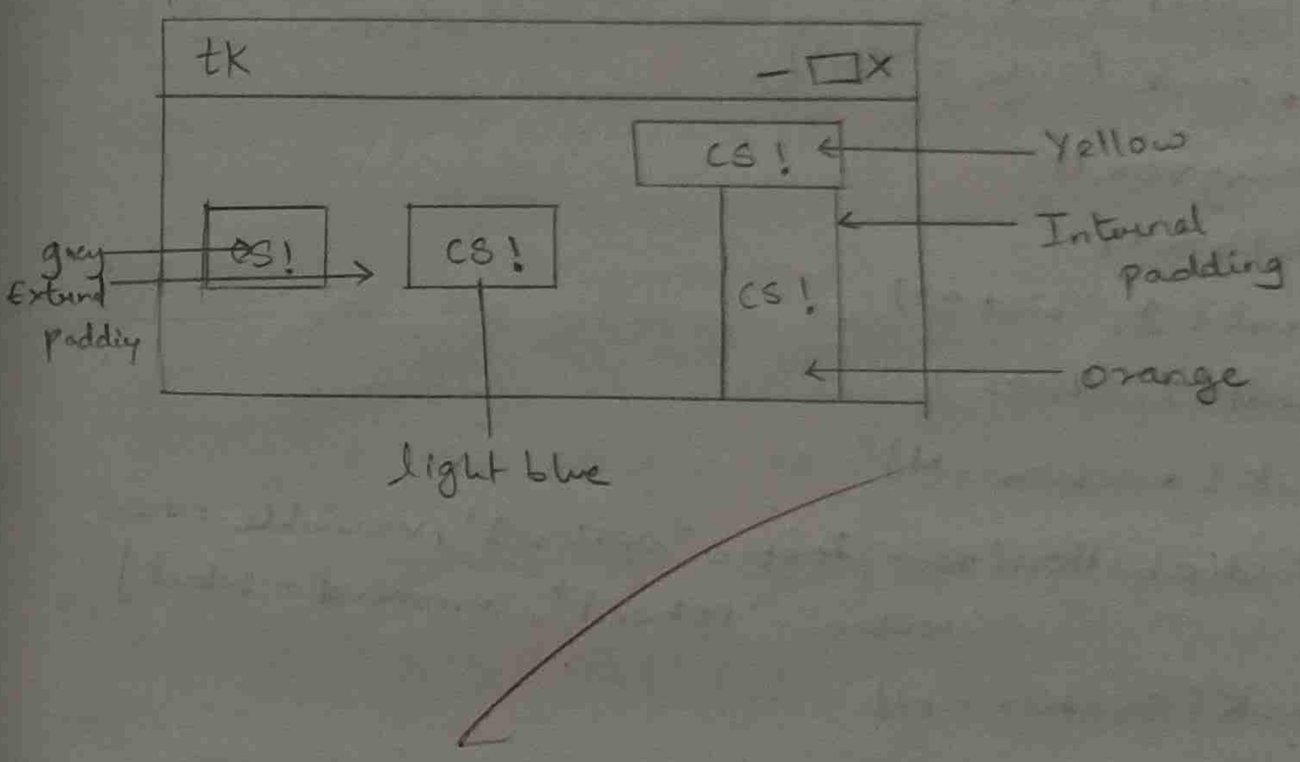
Step 4: Use the mainloop () for the triggering of the corresponding events.

Step 5: No 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 than use the pack () with a relevent padding attributes.

```
l4= Label (root, text = "CS"!, bg = "orange"
    fg= "black", font = "10")
l4.pack (side = TOP, ipady = 50)
root. mainloop ()
```



Output:



```
from tkinter import *
root = Tk()
root.geometry("500x500")
def select():
    Selection = "You just selected" + str(var.get())
    t1 = Label(text = selection, bg = "white", fg = "green")
    t1.pack(side = TOP)
var = StringVar()
l1 = Listbox()
l1.insert(1, "List1")
l2.insert(2, "List2")
l1.pack(anchor = N)
r1 = Radiobutton(root, text = "option1", variable = var
                 value = "option1", command = select)
r1.pack(anchor = N)
r2 = Radiobutton(root, text = "option2", variable = var
                 value = "option2", command = select)
r2.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 specific 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 argument, corresponding value & trigger the function declared-

Step 7: Now call the pack() for radio object so created and specify the argument 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.

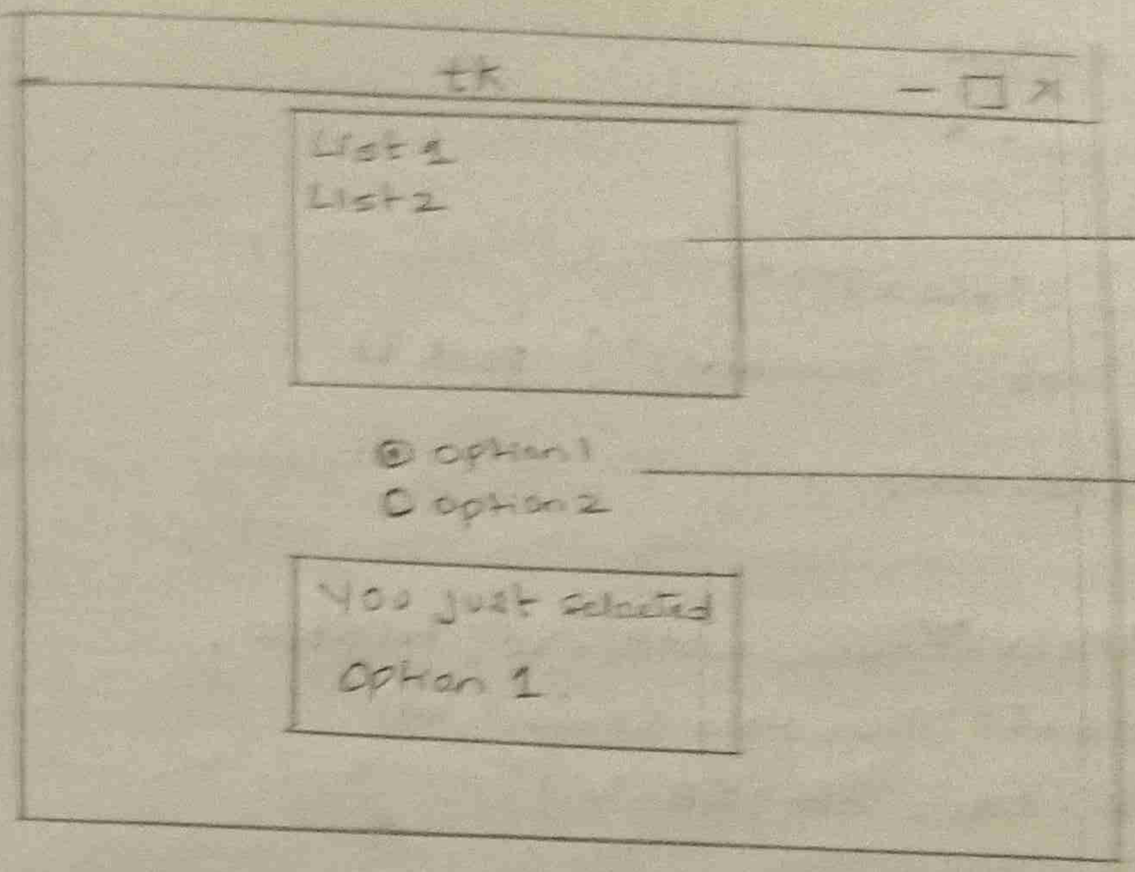
Step 3: Use the geometry() for laying of the window.

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

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

Step 6: Use the mainloop with the parent object.

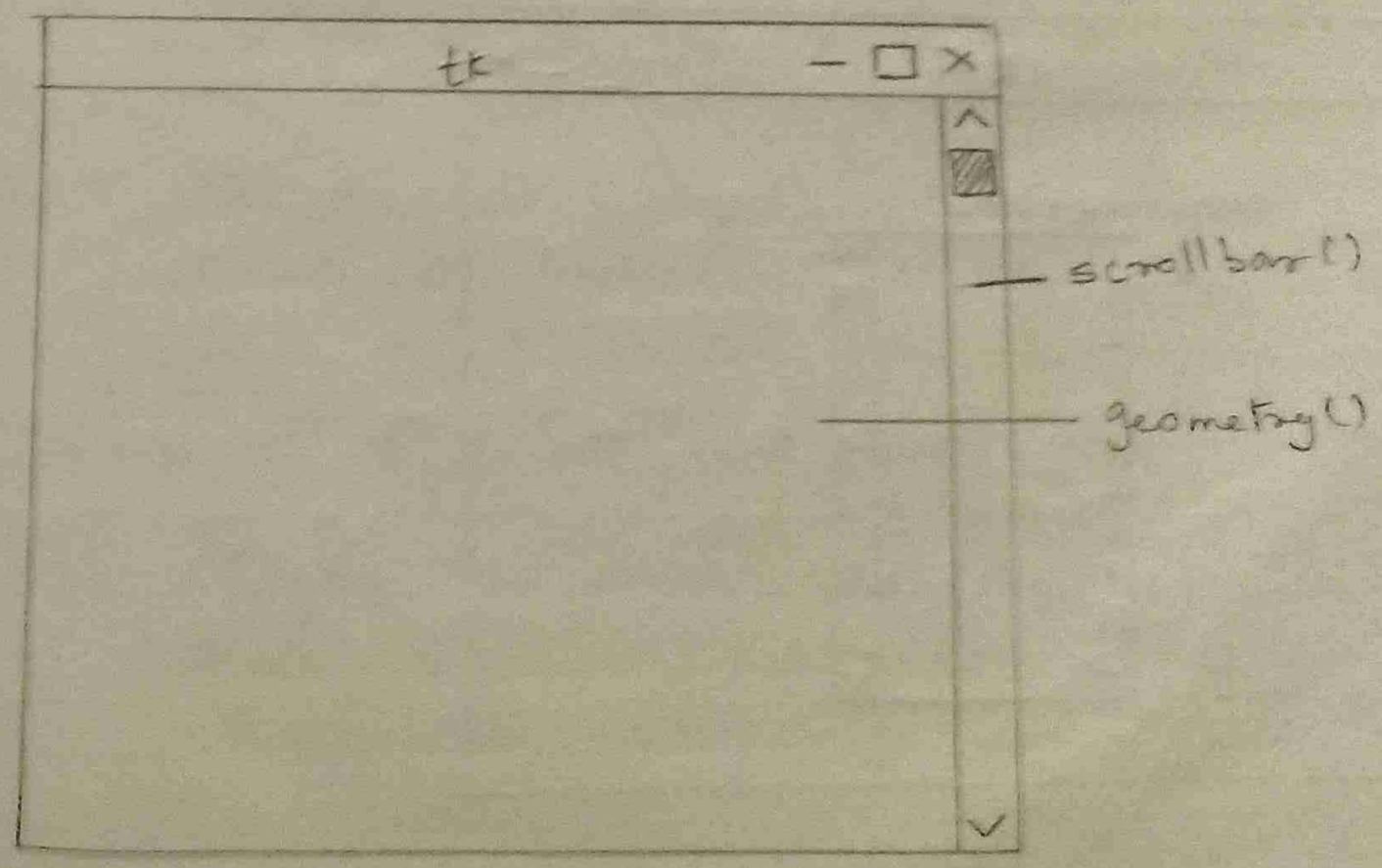
Output :-






#2.

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



#5

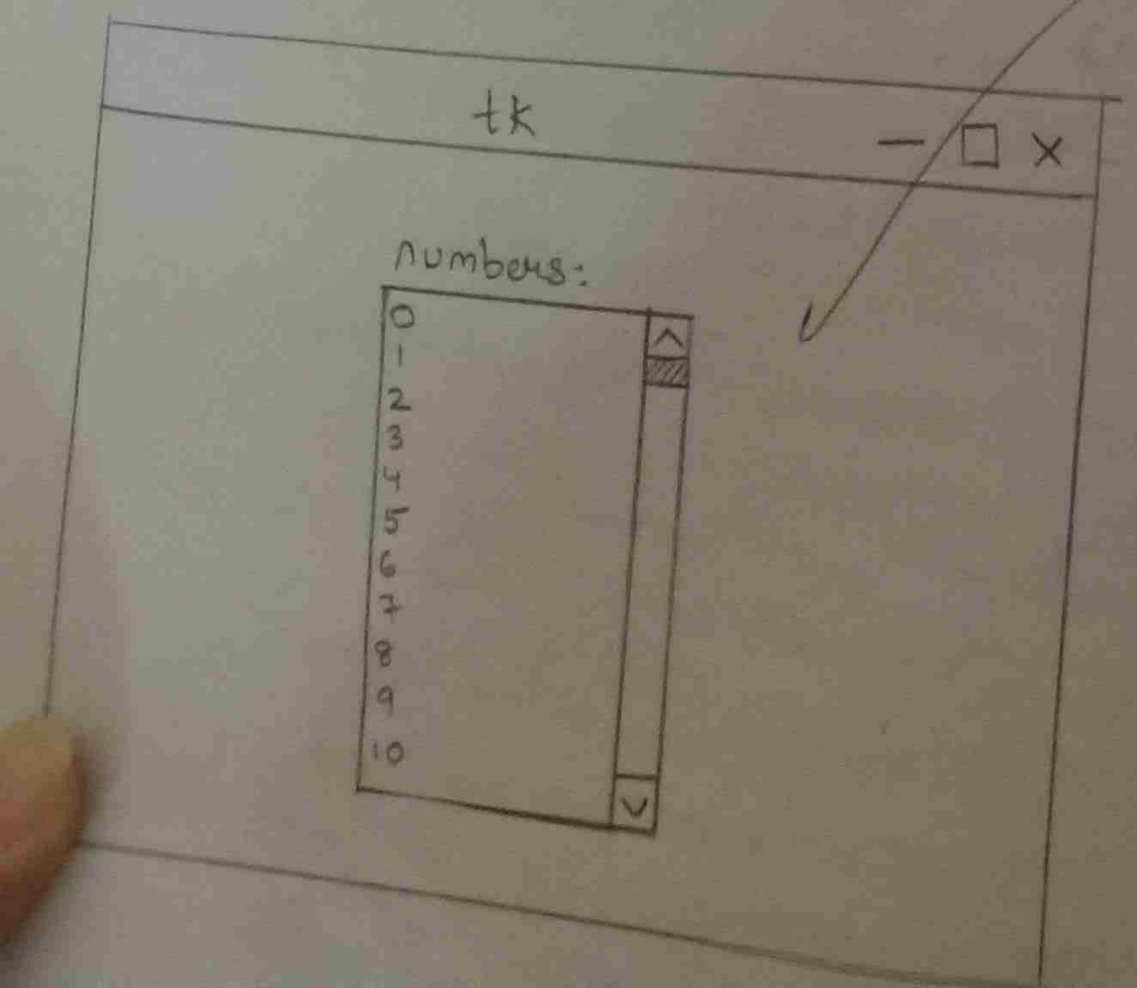
# using frame widget

```
from tkinter import *
window = Tk()
window.geometry("680x500")
Label(window, 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, 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()
```

Output:-






#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 (650x500) or any other suitable pixel value.

Step 4:- Use the label widget along with the parent object created and subsequently 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. 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: Trigger the events using mainloop.

#4:

Step 1: Import relevant methods from tkinter library.

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 to the parent window.

Step 4: Create another frame object termed as the left frame and put it on the parent window on i/z LEFT side.

Step 5: Similarly, define the RIGHT frame and ~~subsequ~~ Subsequently define the button object placed on to the given frame with the attribute as text, active background and foreground.

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

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

#4:

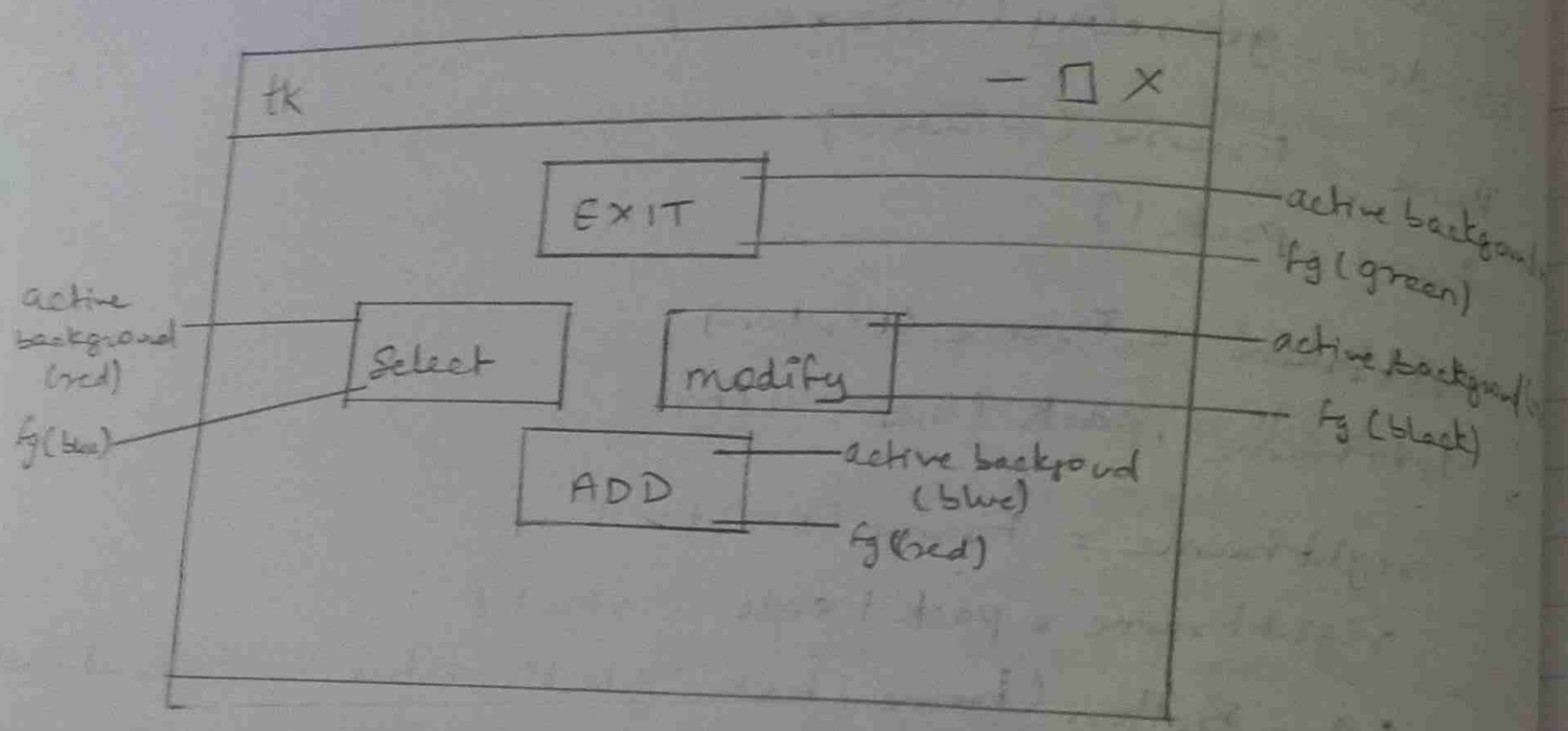
```
from tkinter import *
window = Tk()
window.geometry("680x500")

frame = Frame(window)
frame.pack()

leftframe = Frame(window)
leftframe.pack(side="LEFT")

rightframe = Frame(window)
rightframe.pack(side="right")
b1 = Button(frame, text="select", active background="
                fg="blue")
b2 = Button(frame, text="modify", active background="
                fg="black").
b3 = Button(frame, text="ADD", active background="
                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 :-






Steps :- Create another button object & place it onto the RIGHT frame & label the button as ADD.

Step 9 :- Add another button & pob it on the top of frame and label it as EXIT.

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

# PRACTICAL - 6

Aim: Demonstrate the use of GUI by creating a human face and converting celsius into

Q.1) Write a program to draw human face using GUI.

Algorithm :-

Step 1: Import relevant methods from tkinter library

Step 2: Create an object corresponding to the parent window from TK()

Step 3: Create an object from canvas() & place onto parent window along with height & width.

Step 4: Now use pack() for positioning of widget onto parent ~~widget~~ window.

Step 5: Now create an object face & use object create_oval() with coordinates 50, 50, 350, 350 & outline = "black", fill = "yellow" as attribute to create face.

Step 6: Now create eye 1 object & again use objet create_oval() with appropriate coordinats along with fill as attribute to create left eye.

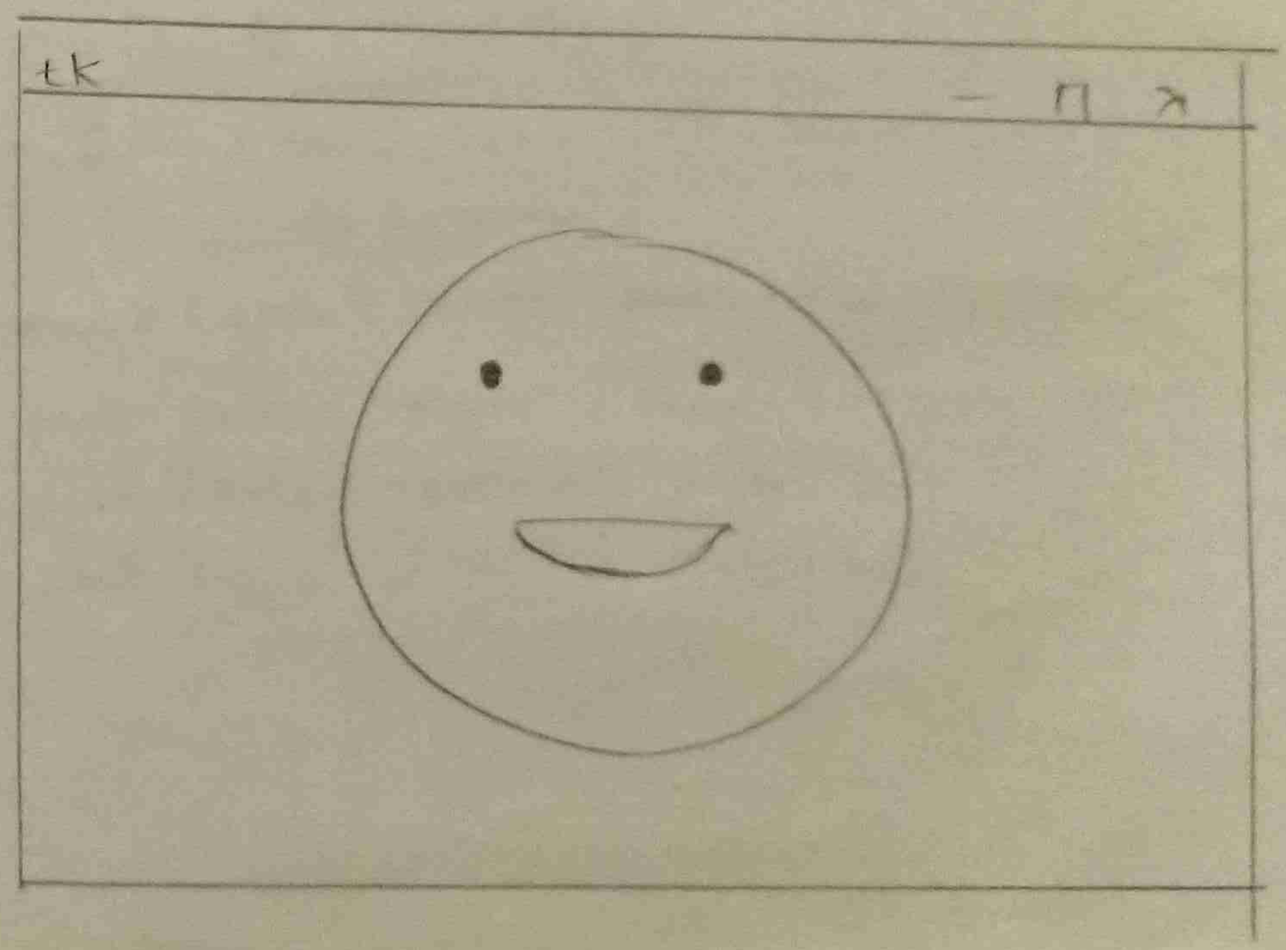
Step 7: Now repeat same step 6 to create right eye.



## code.

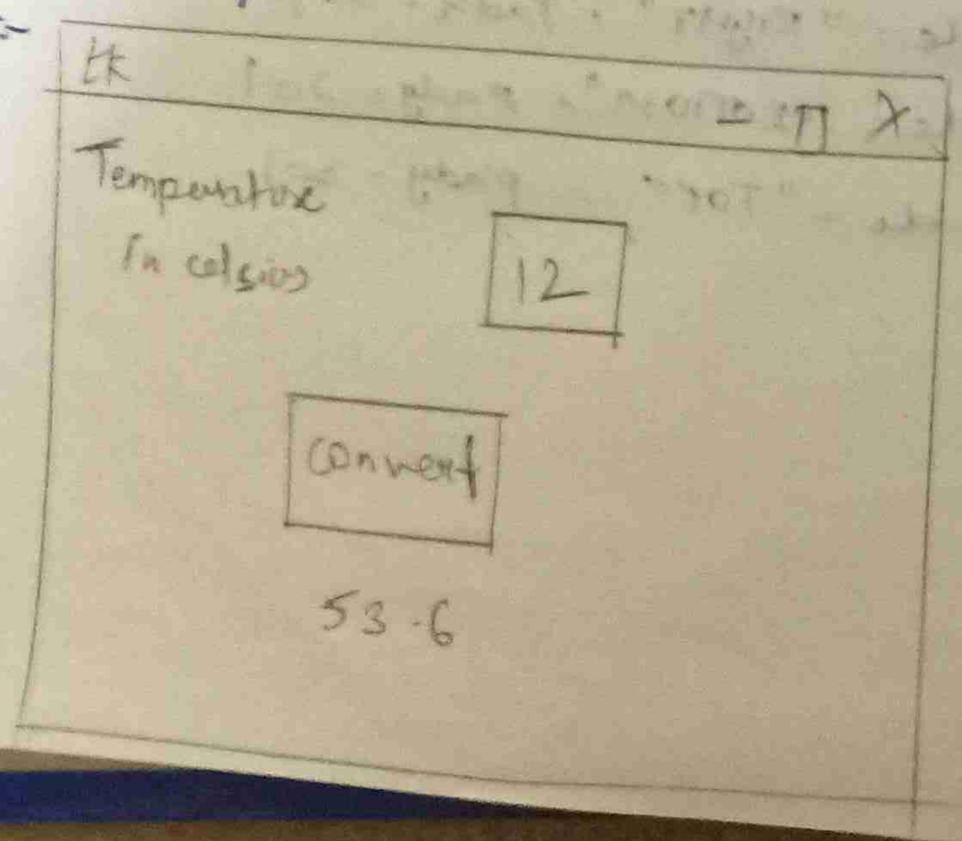
```
from tkinter import *
root = TK()
c = Canvas (root, width = 500, height = 500)
c = Packed()
face = C.create.oval (50,50,350,350, outline = "black", fill = "yellow")
eye1 = C.create_oval (125, 125, 175, 175, fill = "black").
eye2 = c.create.oval (225, 125, 275, 175, fill = "black"))
mouth = c.create_oval (125, 125, 275, 175, start = 0,
                       extent = 180, width = 5, fill "red")
root.mainloop()
```

## output:



```
# code
from tkinter import *
window = Tk()
fahrenheit = DoubleVar()
fahrenheit.set(92.0)
def convert(celsius):
    fahrenheit.set(9.015) * celcius + 32)
l1 = label(window, text = "Temperature in celsius:")
l1.grid(row=0, column=0)
e = entry(window, text variable = celsius)
e.grid(row=0, column=1)
celsius = IntVar()
l2 = label(window, textvariable = fahrenheit)
l2.grid(row=2, column=0, columnspan=2)
B = Button(window, text = "calculate", command = lambda:
                                    convert(celsius.set(17)).
B.grid(row=1, column=0, columnspan=2)
window.mainloop().
```

# output :-






Step 8: Create an object mouth and use object create_oval() with appropriate co-ordinates, start=0, extent=180 & fill = "red", width=5 as attribute to create mouth

Step 9: Finally use the mainloop().

02) Write a program to convert celsius into fahrenheit GUI.

Algorithm:

Step 1: Import all the relevant methods in the tkinter library.
Step 2: Create object corresponding to the parent window.
Step 3: Now initialize fahrenheit as DoubleVar() & set it to 320
Step 4: Now define a function 'convert' with argument celsius to convert celsius into fahrenheit using set().
Step 5: Now create an object l2 using label() & place it onto parent window & use text attribute as onto as 20.
Step 6: Now use grid() for position the object onto the parent window.
Step 7: Initialize celsius as integer using IntVar().
Step 8: Create another object & use entry widget to enter the input & place it onto the parent window.
Step 9: Now use grid() for positioning the object onto parent window with text variable attribut.
Step 10: Now again label() along with text variable used attribute to display output & use grid() for positioning
Step 11: Finally use mainloop().