```
# Write method
file obj = Open ("abc.txt", "w")
fileobj.write ("Subjects in" + "Cs")
fileobj.write ("\n C \n Calculus \n Stats \n Python")
fileobj.close()
# read method
fileobj = Open ("abc.txt", "r")
str1 = file obj.read()
print("The output of read method is: ", str1)
file obj.close()
>>> The Output of read method is : Subjects in Cs
    C
    Calculus
    Stats
    Python
# readline method
file obj = Open ("abc.txt", "r")
str2 = fileobj.readline()
print(" The output of readline method is : ", str2)
file obj.close()
>>> The Output of readline method is: Subject in Cs
# read lines method
file obj = Open ("abc.txt", "r")
str3 = fileobj.readlines()
print("The output of readlines method is : ", str3)
>>> The output of readlines method is : ['Subjects in
Cs \n', 'C\n', 'Calculus\n', 'Stats\n', 'Python']
```

# Practical - 1



Aim :-

Demonstrate the use of different file accessing modes different attributes & read methods.

Algorithm :-

Steps

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

2. Now, Open the file in read mode & then use read( readline(), & readlines() & store the Output in Variable & finally display the Contents of Variable.

3. Now use the fileobject for finding the name of file, the file name, the file mode in which it Opened whether the file is still Open or closed & finally the Output of the software attribut

4. Now Open the fileobj in write mode, write Some another close Subsequently then again Open the fileobj in ~~wt~~ 'wt' mode that is the update mode & write Contents.

5. Open fileobj in read mode display the update written Contents & close, Open again in 'rt' mode that is with parameter passed & display the Output Subsequently.

6. Now Open fileobj in append mode open write method, write Contents close the fileobj again. Open the fileobj in read mode & display the append outputs.



```
# file attributes
a = fileobj.name
b = fileobj.closed
c = fileobj.mode.
print("Name of file : ", a)
print("closed ", b)
print("File mode : ", c)
>>> Name of file : abc.txt
>>> closed True
>>> File mode : r
# write mode
fileobj = open("abc.txt", "w")
fileobj.write("c")
file obj.close()
# read mode
fileobj = open("abc.txt", "r")
x = fileobj.read()
print("The output of read mode is : ", x)
>>> The output of read mode is : c
# wt
fileobj = Open("abc.txt", "wt")
fileobj.write("Sagar")
file obj.close()
```

```
# rt
file obj = Open ("abc.txt", "rt")
x1 = file obj.read(10)
print("The output of rt is ", x1)
>>> The output of rt is : Sagar
# Append method
fileobj = Open("abc.txt", "a")
fileobj.write("\n Avengers")
file obj.close()
file obj = Open("abc.txt", "r")
x2 = fileobj.read()
print("The output of append mode is : ", x2)
file obj.close()
>>> The output of append mode is : Sagar
    Avengers.
# Tell method
fileobj = Open("abc.txt", "rt")
pos = file obj.tell()
print("Tell", pos)
fileobj.close()
>>> Tell 0
```



7. Open the fileobj in readmode declare a variable & perform fileobject do tell method & store the output consequently in variable.

8. Use seek method with the arguements with opening the fileobj in read mode & closing subsequently.

9. Open fileobj with read mode also use the readlines() method & print the same for counting the length use for conditional statements & display the length.

5/2/19

```
# seek method
file obj = Open ("abc. txt", "r")
s = file obj. seek (0,0)
print ("seek (0,0)", s)
file obj. close( )
>>> seek (0,0) 0
file obj = Open ("abc. txt", "r")
s1= file obj. seek(0,1)
print ("seek (0,1)", s1)
file obj. close( )
>>> seek (0,1) 0
file obj= Open ("abc. txt", "r")
s2 = file obj. seek (0,2)
print ("seek (0,2)", s2)
file obj. close( )
>>> seek (0,2) 14
# length of different lines
fileobj = Open ("abc. txt", "r")
str0= fileobj. readlines( )
print (" length of different lines ", str0)
for line in str0:
    print (len(line))
file obj. close( )
>>> length of different lines ['Sagar \n', 'Avengers']
9
4
```

Scanned with CamScanner

```
# iter() & next()
mytuple1 = ("apple", "orange", "banana").
myiter1 = iter(mytuple1)
print(next(myiter1))
myiter2 = iter(mytuple1)
print(next(myiter2))
myiter3 = iter(mytuple1)
print(next(myiter1))
>>> apple
    orange
    banana
# for loop :-
mytuple1 = ("Tom", "Stuart", "Jerry")
for x in mytuple1:
        print(x)
>>> Tom
    Stuart
    Jerry
# Square & Cube:
def Square(x):
        y = x * x
        return y
def cube (x):
        z = x * x * x
        return z
Count 1 = [Square, Cube]
for r in range (5):
        Value = list(map( lambda x: x(r)), fun(1))
        print (Value)
>>> [0, 0]
    [1, 1]
    [4, 8]
    [9, 27]
    [16, 64]
```



# Practical - 2

## Iterators

Step 2.

1. Create a tuple with elements that we need to iterate using the iter & next method. The numbers of time we use the iter & next method we will get the next iterating element in the tuple. Display the same.

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.

3. Define a function name square with a parameter which will obtain output of square value of the given number. In similar way declare cube to get the value raised 3 & return the same.

4. Call the declared function using function call.

5. Using for conditional statement specifying the range use the list typecasting with map method, declare a 'lambda' i.e. annoymous function & print the same.

6. Declare a list num variable & declare one element then use the map method with the help of lambda function give two arguement display the output.

Scanned with CamScanner

7. Define a function even with a parameter then using Conditional statements. So, check whether the number is even & odd & return the Value respectively.

8. Define a class & within that define the iter() method which will initialize the first element within the Container object

9. Now use the next() & define the logic for displaying odd Value.

10. Define an object of a class

11. Accept an number from the user till which we want to display the odd number

12. Define a function with if condition, if exactly equal to one return else on parameters.

```
# map()
listnum = [0,4,5,7,9,11,13,15,20,19,25]
listnum = 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, listnum))
>>> [0,4,0,2,4,1,3,0,0,4,0]
# odd numbers :
class odd:
        def __iter__(self):
                self.num = 1
                return self
        def __next__(self):
                num.self = num
                num.self += 2
        def __next__(self):
                num = self.num
                self.num += 2
                return num

my obj = odd()
my iter = iter(my obj)
x = int(input("Enter a number:"))
for i in my iter:
        if (i < x):
                print(i)
>>> Enter a number : 15
1
3
5
7
9
11
13
```

# Factorial

```
def fact(n):
    if (n == 1):
        return 1
    else:
        return (n*fact(n-1))
n = int(input("Enter a number:"))
list = [ ]
list.append(n)
a = map(fact, )
print("The factorial is:", a)
```

```
>>> Enter a number: 5
    The factorial is: [120]
```

#

```
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

my class = myrange()
mydata = iter(my class)
for x in mydata:
    print(x)
```

output

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



13. Enter a input from the declare an empty list use append method, for appending the input value.

14. Print the output variable.

15. Define a iter method with an argument & initialize it to the 1st value.

16. For extracting the next element from the container, use the next method with an argument & compare the no. of elements required in a container by using the conditional statement

17. Now create an object from the given class & pass this object as an argument to the iter method.

18. Now using the conditional statement display all the values from the given container.

19/12/19

```
# Factorial

class fact:
    def __iter__(m):
        m.value = 1
        return m
    def __next__(m):
            factr = 1
            for i in range (1, m.value+1):
                factr = factr * (i)
            print ("The factorial ", m.value, "is", factr)
            m.value += 1
            if (m.value == 7):
                raise StopIteration

y = iter (fact())
while True:
    next (y)
```

Output:

```
The factorial 1 is 1
The factorial 2 is 2
The factorial 3 is 6
The factorial 4 is 24
The factorial 5 is 120
The factorial 6 is 720
```

14/11

```
# IOError:
try:
    fo = open("abc.txt","w")   #R
    fo.write("python is an independent language")
except IOError:
    print("Enter appropriate mode for file operation!")
else:
    print("Operation is Successful!")
>>> Operation is Successful!
# R Output:
>>> Enter appropriate mode for file Operation!

# Value Error:
try:
    x = int(input("Enter a statement:"))
except ValueError:
    print("Arithmetic Error!")
else:
    print("Operation is Successful!")
>>> Enter a statement: abc
    Arithmetic Error!
>>> Enter a Statement: 123
    Operation is Successful!
```

# PRACTICAL- 3



Topic :- Exceptions

Steps:

1. In the try block Open an file in the Open mode with write mode write Some Content in file.
2. In except(IO Error) block use the error in IOError use the appropriate message to display.
3. Else display the Operation is Successful!
4. In try block accept an input from the user
5. In except block use Value error and print the Same message
6. Else display the Operation is Successful!

Scanned with CamScanner

7. Accept an integer value from the user. In the try use the division method.

8. For the exception to be raised use the except keyword (and) i.e Type error print In Compatible Values

9. Use except with error of zero division Error & print the message according that is if entered number is zero not able to perform Operation !

10. Declare Static Variables & Values

11. For Multiple exception use the error types by separating them with a Comma.



```
# Type error, Zero division error.
a=1
b= input("Enter:")
try:
    print(a/b)
except TypeError:
        print("Incompatible Value")
except ZeroDivisionError:
        print("Denominator is Zero so operation cannot be
              performed")
```

```
>>> Enter: 2
    0.5
>>> Enter: 0
    Denominator is Zero so operation cannot be performed.
```

```
# Multiline exception
a,b = 1, 0
try:
    print(1/10)
    print('10'+10)
except(TypeError, ZeroDivisionError):
        print('Invalid Input!')
```

```
>>> 1.0
>>> 1010
>>> Invalid Input!
```

10/07

Scanned with CamScanner

```
# Using except keyword
try:
    a = open("abc.txt","w")
    a.write("python")
except IOError:
        print("Error!")
    else:
        print("Successful")
def x():
        l=[ ]
        print(len(l))
def y():
        li=[2,4,4,1]
        print(len(li))
print(x())
print(y())
```

Output: Successful
0
None
4
None

```
# raise keyword:
try:
    a = int(input("Enter a number:"))
    raise ValueError
except ValueError:
        print("Enter integer Value!")
```

>>> Enter: XYZ
Enter integer Value!



12. Use try block Open a file in write mode & subsequently enter value in the file

13. Use the IOError & display appropriate message

14. Define a function with empty list & Calculate the length of the lists

15. Define another function y initialise or declare some elements in list & Calculate the length of same & display the same

16. In try block element accept input from the user & if the user enters character values raise an error that is saying up enter integer values.

Scanned with CamScanner

17. Define a function with an arguement & initialize an empty list.

18. Print the last element in a try block

19. Use accept for index error & print appropriate message, lite list is empty

20. Define a function with an arguement & initialize with some values

21. Print the last element in the try block

22. Use except for Index error & print appropriate message

23. Else print same Operation on the list

24. Call both the function defined



```
def list(a):
    a=[ ]
    try:
        print (a[-1])
    except Index Error:
        ~~print ("list is empty")~~
        print("list is empty")
    else:
        print (a*2)
def list2(b):
    b=[1, 2, 3, 4]
    try:
        print (b[-1])
    except IndenError:
        print ("list is empty")
    else:
        print(b* 2)
list1("abc")
list2("xyz")
```

```
>>> list is empty
>>> [2, 4, 6, 8]
```

10/07

```
# match
import re
pattern = r"FYCS"
Sequence = "FYCS represent Computer Science"
if re.match (pattern, Sequence):
        print ("matched pattern found!")
else:
        print (" NOT FOUND!")
>>> matched pattern found!
# numerical values (Segreation)
import re
pattern = r"\d+"
String = "hello123, howdy 789, 45 howru"
output = re.findall (pattern, String)
print (output)
>>> ['123', '789', '45']
# Split
import re
pattern = r"\d+"
string = "hello123, howdy 789, 45 howru"
output = re.split (pattern, string)
print (output)
>>> ['hello', ', howdy ', ', howru']
```



Aim: Regular Expression

Steps

1. Import re module declare pattern and declare Sequence use match method with declare arguements. if arguements matched than print the Same otherwise print pattern NOT FOUND!.

2. Import re module declare pattern with literal & meta character. Declare string value. Use the findall() with arguements & print the Same.

3. Import re module declare pattern with meta character use the split() & print the Output.

Scanned with CamScanner

4. Import re module declare string & accordingly declare pattern replace the blank space with no_space. Use sub() with 3 arguements & print the string without spaces.

5: Import re module declare a sequence use search method for finding subsequently use the group() with dot operator as search() gives memory location using group() it with show up the matched string.

6. Import re module declare list with numbers. Use the conditional statement here we have used up the for condition statement. Use if condition for checking first number is either 8 or 9 & next number are in range of 0 to 9 & check whether the entered number are equal to 10. If criteria matches print cell number matches otherwise print failed.



```
#no_space:
import re
string="abc def ghi"
pattern = r"\s+"
replace = ""
v1= re.sub(pattern, replace, string)
print(v1)
>>> abcdefghi
```

```
# group()
import re
sequence="Python is an ~~student~~ interesting language"
v= re.search("\A Python", sequence)
print (v)
v1 = v.group()
print (v1)
>>> <_sre.SRE_Match object at 0x0281DF0>
    Python
```

```
# Vrag' Verifying the given set of phone number
import re
list1=['8530802458','9594186500','9766391938',
       '8879324437']
for value in list1:
        if re.match(r'[8-9]{1}[0-9]{9}',
           value or len(value)==10):
            print ("Criteria matched for cell numbers!")
        else:
            print (" Criteria failed!")
>>> Criteria matched for cell number!
>>> Criteria matched for cell number!
>>> Criteria matched for cell number!
>>> Criteria matched for cell number!
```

Scanned with CamScanner

```
# Vowels
import re
Str1 = "plant is like" overall"
output = re.findall(r'\b[aeiouAEIOU]\w+', Str1)
print(output)
>>> ['is', 'overall']

# host & domain
import re
Seq="abc.tcse@edu.com, XYZ@gmail.com"
pattern=r'[\w\.-]+[\w\.-]'
Output = re.findall(pattern, Seq)
print(output)
>>> ['abc.tcse', 'edu.com', 'XYZ', 'gmail.com']

# Counting of first 2 letters
S="mr.a, ms.b, ms.c, mr.t'
P=r'[/ms/mr/]+
O= re.findall(p,s)
print(o)
m=0
f=0
for v in O:
    if(v=='ms'):
        f=f+1
    else
        m=m+1
print("No. of males is: ", m)
print("No. of females is: ", f)
>>> ['mr', 'ms', 'ms', 'mr']
No. of males is: ',2
No of females is: ',2
```



7. Import re module declare a string use the module with findall() for finding the Vowels in the string & declare the same.

8. Import re module declare the host & domain name declare pattern for separating the host & domain name. Use the findall() & print the output respectively.

9. Import re module enter a string use pattern to display only two elements of the particular string. Use findall() declare two variables with initial value as zero ~~as~~ use for condition & subsequently use the if condition check whether condition satisfy add up the or else increment value. & display the values subsequently.

10/10

Scanned with CamScanner

## PRACTICAL- 5 (A)

### GUI Components

Steps:

1. Use the tkinter library for importing the feature of text widget.

2. ~~Crat~~ Create an object using the Tk().

3. Create a Variable using the Widget label & the use the text method

4. Use the mainloop() for trigerring of the Corresponding above mention events.

#2:

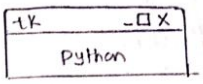
1. Use the tkinter library for importing the features of that widgets

2. Create a Variable from the text method & position it on the parent window



```
# creation of parent window
from tkinter import *
root = Tk()
l = Label (root, text = " python")
l.pack ()
root.mainloop()
```

Output:

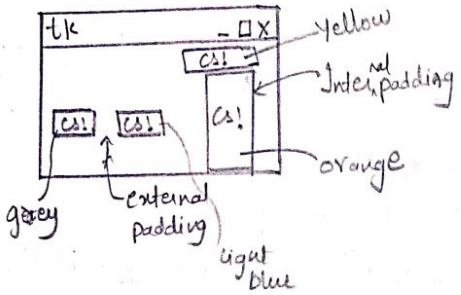



```
#2
from Tkinter import *
root = Tk()
l = label (root, text = "python")
l.pack()
l1 = Label (root, text = "cs", bg = "grey", fg = "black",
        font = "10")
l1.pack (side = LEFT, padx = 20)
l2 = label (root, text = "cs", bg = "light blue", fg = "black",
        font = "20")
l2.pack (side = LEFT, pady = "30")
l3 = Label (root, text = "cs", bg = "yellow", fg = "black",
        font = "10")
l3.pack (side = TOP, ipadx = 40)
```

Scanned with CamScanner

```
l1 = label(root, text = "cs1", bg = "orange", fg = "black",
           font = "10")
l4.pack(side = TOP, ipady = 50)
root.mainloop()
```

Output :






3. Use the pack() along with the object created from the text() & use the parameter

i Side = LEFT, padx = 20
ii Side = LEFT, pady = 30
iii Side = TOP, ipadx = 40
iv Side = TOP, ipady = 50

4. Use the mainloop() for the triggering of the Corresponding events.

5. Now repeat above steps with the label() which takes the following arguements.

i Name of the parent window
ii Text attribute which defines the string
iii The background color (bg)
iv The foreground fg & than use the pack() with a relevant padding attributes

Scanned with CamScanner

# PRACTICAL - 5 (B)

## GUI Components

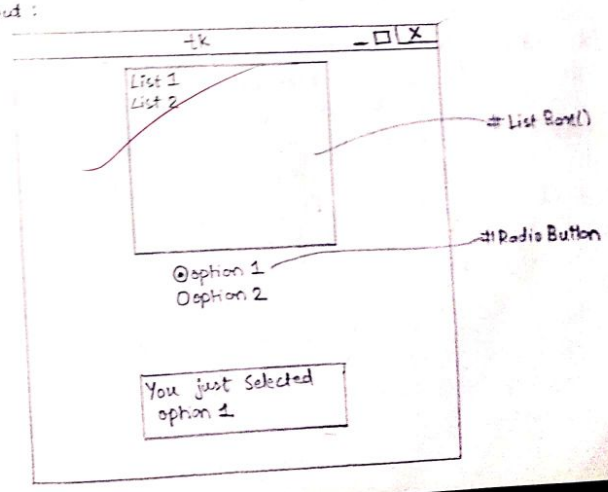
Steps:

1. Import the relevant methods from the tkinter library. Create an object with the parent window
2. Use the parent window object along with the geometry declaring specific pixel size of the parent window
3. Now define a function which tells the user about the given selection made from multiple option available
4. Now define the parent window & define the option with Control Variable
5. Use the listbox() & insert options on the parent window along with the pack() with specifying anchor attribute
6. Create an object from radio button which will take following arguments, parent window object, text variable which will take the values option no 1, 2, 3,... variable argument, corresponding value & the triggered the function declared.
7. Now call the pack() for radio object to created & specify the arguments using anchor attribute.



```
#Radio Button
from tkinter import *
root = Tk()
root.geometry("300x300")
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," List 1")
l2.insert(2, "List 2")
l1.pack(anchor = N)
r1 = Radio Button(root, text = " option 1", variable = var,
                  value = " option 1", Command = select)
r1.pack(anchor = N)
r2 = Radio Button(root, text = " option 2", variable = var,
                  value = " Option 2", Command = select)
r2.pack(anchor = N)
root.mainloop()
```

Output :

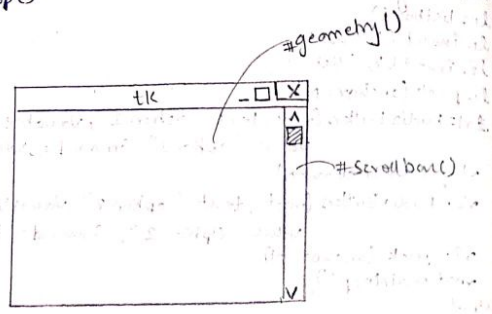



Scanned with CamScanner

#2:

```
~~root~~ Scrollbar()
from tkinter import *
root = Tk()
root.geometry("500 x 500")
S = Scrollbar()
S.pack(side="right", fill="y")
root.mainloop()
```

Output:






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

#2:

Steps

1. Import relevant methods from the tkinter library
2. Create a parent object corresponding to the parent window
3. Use the geometry() for laying of the window
4. Create an object & Use the Scroll bar().
5. Use the pack() along with the Scroll bar object with side & fill attributes.
6. Use the mainloop() with the parent object.

Scanned with CamScanner

# 3:

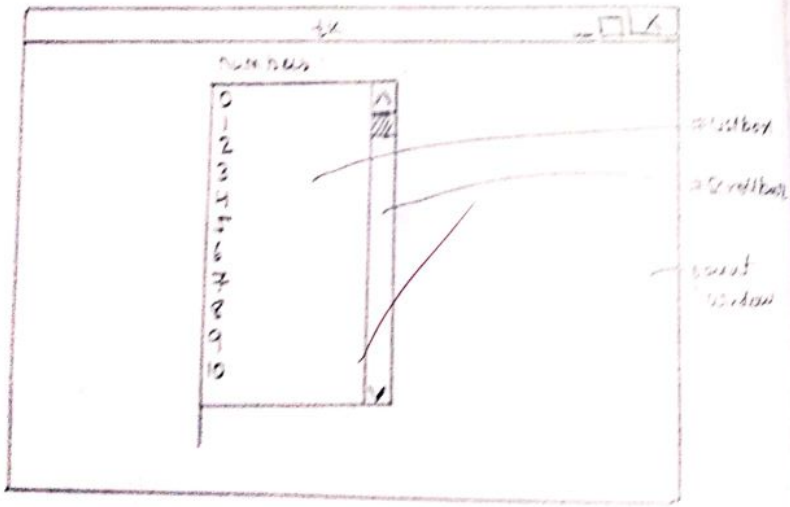
Steps

1. Import the relevant libraries from the tkinter method
2. Create an corresponding object of the parent window
3. Use the geometry manager with pixel size (650x500) or any other suitable pixel value
4. Use the label widget along with the parent object created & subsequently use the pack method
5. Use the frame widget along with the parent object created & use the pack method.
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.
7. Use the ScrollBar() with an object use the attribute of vertical. then configure the same with object created from the scrollbar() & use pack().
8. Trigger the events using mainloop

Scanned with CamScanner

```
# 3:
# Frame widget
from tkinter import *
root = Tk()
root.geometry("640x480")
label(root, text="numbers:").pack()
frame1 = Frame(window)
frame.pack()
list_nodes = Listbox(frame, width=20, height=20, font=("Times New
                                                        Roman", 10))
list_nodes.pack(side="LEFT", fill='y')
Scrollbar = Scrollbar(frame, orient="Vertical")
Scrollbar.Config(Command=list_nodes.yviews)
Scroll bar.pack(side="RIGHT", fill="y")
for x in range(100):
        list_nodes.Insert(END, str(x))
root.mainloop()
```

Output:






# 4

Step

1. Import relevant methods from tkinter library

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

3. Now define the frame object from the method & place it on to the parent window

4. Create another frame object termed as the left frame & put it on the parent window on its LEFT side

5. Similarly define the RIGHT frame & subsequently define the button objects placed on to the given frame with the attributes as text, active background & foreground.

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

7. Similarly Create the button object corresponding to the Modify operation put it onto frame object on side = "right".

8. Create another button object & place it on to the right frame. & label the button as ADD

No.

7.

9. Add another button & put it on the top of frame & label it as EXIT.

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

8

9.

10

11.

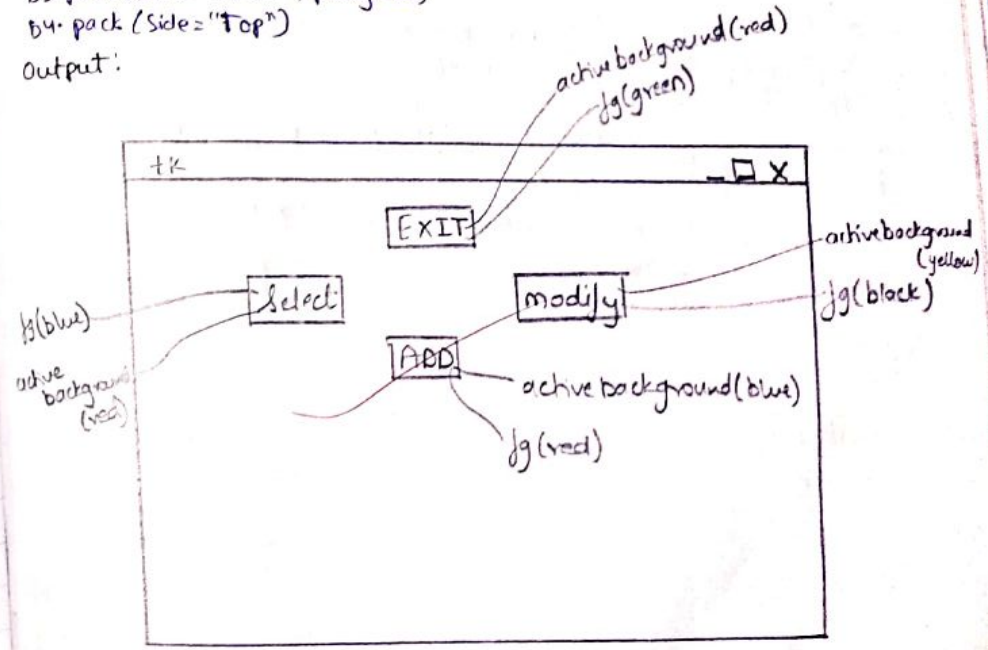
12.



#4:

```
from tkinter import *
root = Tk()
root.geometry("680x500")
frame = Frame(window)
frame.pack()
left frame = Frame(window)
left frame.pack(side="left")
right frame = Frame(window)
right frame.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="blue", fg="red")
b4 = Button(frame, text="EXIT", activebackground="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")
```

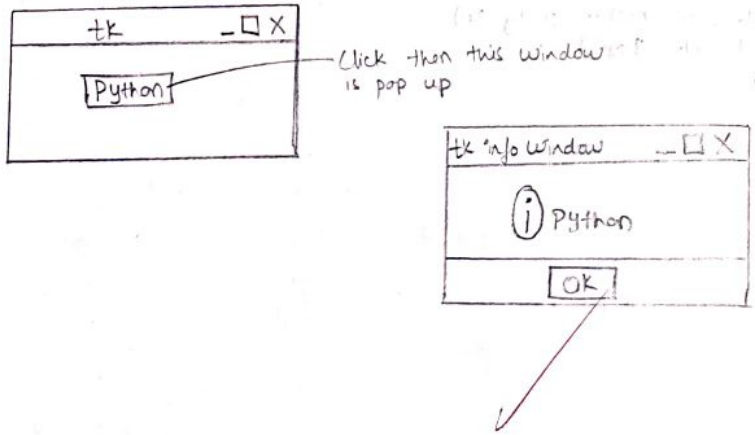
Output:




Scanned with CamScanner

```
#message box
from tkinter import *
import tkMessageBox
root=Tk()
def function():
    tkMessageBox.showinfo("info window","python")
    b1=Button(root, text="python", command=function)
    b1.pack()
root.mainloop()
```

Output :






#5:

Steps

1. Import the relevant methods from tkinter library
2. Import tk message box
3. Define a parent window object along with the parent window
4. Define a function which will use tk message box with showinfo method along with info window attribute.
5. Declare a button with parent window object along with the Command attribute.
6. Place the button widget onto the parent window & finally call mainloop() for trigerring of the events called above.

6/2

1. Import the relevant methods from the tkinter library along with parent window object declared

2. Use parentwindow object along with minsize function for window size.

3. Define a function main, declare parent window object & use Config(), title(), minsize(), Label() as well as button() & use pack() & mainloop Simultaneously.

4. Similarly define the function Second & use the attributes accordingly.

5. Declare another function button along with parent object & declare button with attributes like FLAT, RIDGE, GROOVE, RAISED, SUNKEN along with the relief widget

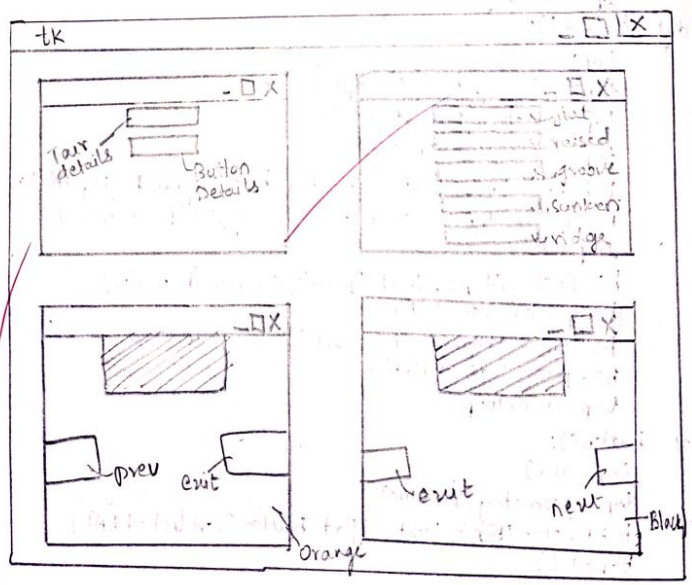
6. Finally called the mainloop() for even driven programming



```
# Multiple window
# Different button (Relief())

from Tkinter import *
root = Tk()
root.minsize(300,300)
def main():
    top=Tk()
    top.Config(bg="black")
    top.title("HOME")
    top.minsize(100,100)
    l=Label(top,text="India" \n Places of Interest: \n Delhi
                                \n Mumbai \n Chennai \n Kerala")
    l.pack()
    b1=Button(top,text="next",Command=Second),
    b1.pack(Side=RIGHT)
    b2=Button(top,text="exit",Command=terminate)
    b2.pack(Side=LEFT)
    top.mainloop()
def Second():
    top2=Tk()
    top2.Config(bg="Orange")
    top2.title("About US")
    top2.minsize(100,100)
    l1=Label(top2,text="Created by: Sagar Rajapak \n
    For more details contact to our official account")
    l1.pack()
    b3=Button(top2,text="prev",Command=main)
    b3.pack(Side=LEFT)
    b2=Button(top2,text="exit",Command=terminate)
    b2.pack(Side=RIGHT)
    top2.mainloop()
def button():
    top3=Tk()
    top3.geometry(300x100)
    b1=Button(top3,text="flat button",relief=FLAT)
    b1.pack()
```

```
b2 = Button(top3, text = "groove button", relief = GROOVE)
b2.pack()
b3 = Button(top3, text = "raised button", relief = RAISED)
b3.pack()
b4 = Button (top3, text = "Sunken button", relief = SUNKEN)
b4.pack()
b5 = Button (top3, text = "Ridge button", relief = RIDGE)
top3.mainloop()
def terminate():
    quit()
b5 = Button(root, text = "TOUR DETAILS", command = main)
b5.pack()
b6 = Button (root, text = "BUTTON DETAILS", command = button
b6.pack()
root.mainloop()
```



8.A

#2

1. Import relevant method from the tkinter library

2. Create parent window object & use the Config method along with background color attribute specified.

3. Define a function finish with the message box widget which will display a message i.e a warning message & subsequently terminate the program.

4. Define a function info use a listbox widget along with the object of the same. Use the listbox object along with insert method & insert the same & finally use the grid() with ipadx attribute.

5. Define a function about us with label widget & text attribute & subsequently use the grid().

6. Use photoimage widget with file and filename with gif attribute.

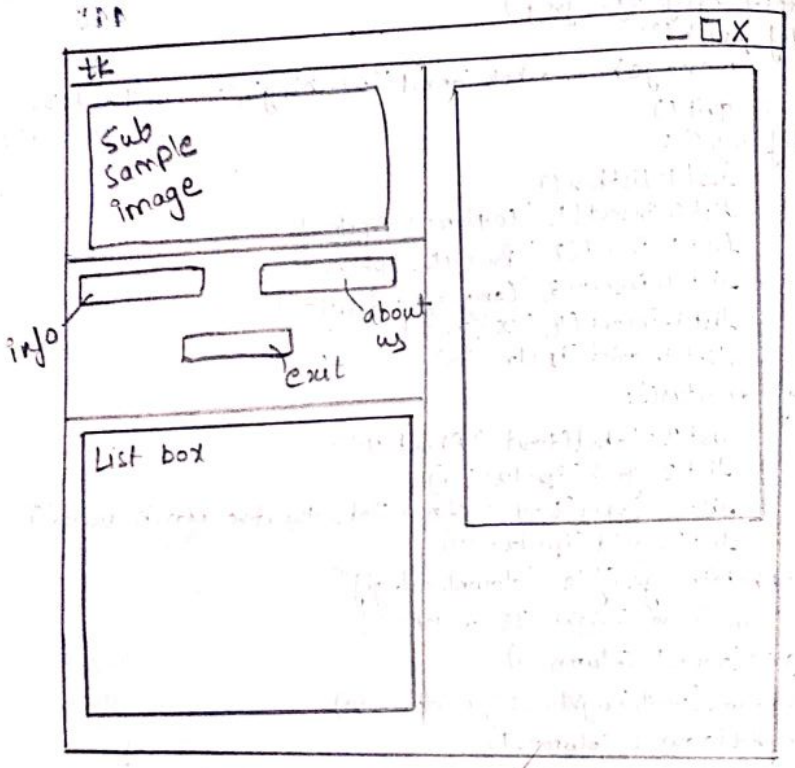
7. Create a frame object along with the Frame() along with parent Window object height & width specified & subsequently use the grid() with row & Column attribute specified.

8. Similarly Create another frame object as declared by Step 7.



```
# image
from Tkinter import *
root = Tk()
root.config(bg="grey")
def finish():
    messagebox.askokcancel("Warning", This will end the program")
    quit()
def info():
    list1 = Listbox()
    list1.insert(1, "Co-Name: Apple")
    list1.insert(2, "Products: iphone")
    list1.insert(3, "Language: Swift")
    list1.insert(4, "OS: IOS")
    list1.grid(ipadx = 30)
def aboutus():
    list2 = Label(text = " About Us")
    list2.grid(ipadx = 30)
    list3 = Label(text = " Steve Jobs theatre March 2020")
    list3.grid(ipadx = 24)
p1 = PhotoImage(file = "download.gif")
f1 = Frame(root, height = 35, width = 5)
f1.grid(row=1, Column=0)
f2 = Frame(root, height = 250, width=500)
f2.grid(row=1, column=1)
p2 = p1.subSample(5,4)
l1 = Label(f1, image = p2, relief = FLAT)
l1.grid(row=1, Column=0, padx = 20, pady=15)
l2 = Label(f2, image = p1, relief = SUNKEN)
l2.grid(padx = 25, pady=10)
b1 = Button(f1, text = "Information", relief = SUNKEN, Command = info)
b1.grid(row=1, column=0)
b2 = Button(f1, text = "About us", relief = SUNKEN, Command = about us)
b2.grid(row=1, Column = 2, padx = 5)
b3 = Button(f1, text = "Exit", relief = RAISED, Command = finish)
b3.grid(row = 2, Column = 1, ipadx = 15)
root.mainloop()
```

Scanned with CamScanner






9. Create another object & use the Subsample (5,4)

10. Use label widget along with the frame object relief attribute & subsequently use the grid().

11. Now create button object dealing with different section of frame.

# PRACTICAL - 5 (B)

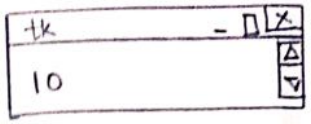
## Spin box.

1. ~~Source~~ Import relevant (): method from the tkinter library.
2. Create parent window object along with TK()
3. Create an object from Spin box method & use attributes parent window object, from_ and to attributes
4. Subsequently Call the pack method along with Spin box object & called the main loop ().

## Paned Window

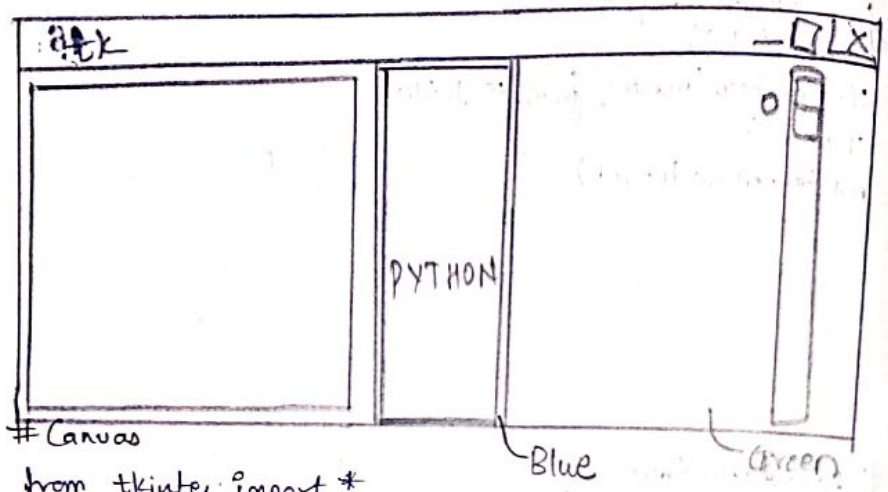
1. Import relevant methods from the tkinter library also Create a parent window object
2. Create an object along with paned window & Subsequently use the pack() along with paned window object along with attributes like fill & expand.
3. Create an Label object along with labe, use paned window object, Orient & background color
4. Similiarly Create another label method & use the add() subsequently.

```
# Spin box
from tkinter import *
master=Tk()
s1=Spinbox(master, from_=0,to=10)
s1.pack()
master.mainloop()
```



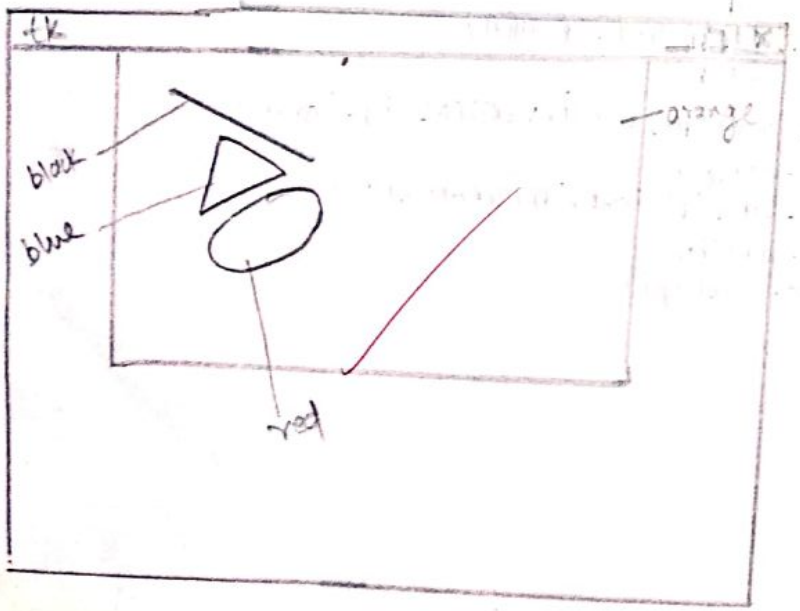

```
# Paned window
from tkinter import *
root=Tk()
p=PanedWindow()
p.pack(fill=BOTH, expand=100)
e=Entry(p, bd=10)
p.add(e)
p2=PanedWindow(p, orient=HORIZONTAL, bg="blue")
p.add(p2)
l=Label(p,text="PYTHON")
p2.add(l)
p1=Scale(p2, orient=VERTICAL, bg="green")
p2.add(p1)
l1=Label(p1, text="PROGRAMMING")
p1.add(l1)
root.mainloop()
```

Scanned with CamScanner




# Canvas

```
from tkinter import *
root=Tk()
C= Canvas (root, height =100, width =200 , bg="orange")
Carc = C.Create_arc( 10, 20, 30, 40 ,start=10, extent =50, fill="blue")
Coval= C.Create_oval (20, 31, 45, 50, fill="red")
Cline= C.Create_line( 10, 15, 30, 20, fill="orange")
C.pack()
root.mainloop ()
```




5. Finally called the mainloop () for event trigered.

# Canvas

1. Import relevant ()'s from tkinter library declare a parent window object.

2. Create a Canvas object along with the Canvas() with attributes parent window object, height, width, background color.

3. Use Create_arc() along with Canvas object declared along with Start & extent & Co-ordinates.

4. Similarly for oval & line use the pack() & called mainloop() for event driven programming.

# PRACTICAL- 6

## DATABASE CONNECTIVITY

#1

Steps

1. Import the (DBM) dbm library & use the open() for creating the database by specfying the name of the database along with the corresponding flag.

2. Use the object so created for accessing the given website & corresponding regular name for the website.

3. Check whether the given ~~ass~~ url address matches with the regular name of the page is not equal to none then display the message that particular found/match or else not found/unmatched.

4. Use the close() the terminate database library



```
>>> import dbm
>>> db = dbm.open('database', flag='c', mode=438)
>>> db ["name"] = "name",
>>> if db ["name"] != None:
        print("database not empty //match!")
    else:
        print("database empty!" // Not Match")
>>> match
>>> db.close()
```

Output

```
import os, sqlite3
Conn = Sqlite3.Connect ("record.db")
cur = Conn.cursor()
cur.execute('Create table dos (Name char(10),
                                Roll_no int(10)')

Cur.execute ('insert into dos Values ("Sagar", "1784")')
Cur.execute ('insert into dos values ("Abhang", "1727")')
Cur.execute ('insert into dos values ("Yash", "1704")')
Conn.commit()
cur.execute ('Select * from dos')
Cur.fetchall()
[('Sagar', 1784), ('Abhang', 1727), ('Yash', 1704)]
```



#2. 1. Import Corresponding library to make database Connection, os & sqlite3

2. Now Create the Connection object using Sqlite-3 library & the Connect() for creating new database.

3. Now Create Cursor object using the cursor() & from the Connection object created.

4. Now use the execute() for creating the table with the Column name & respective datatype.

5. Now with cursor-object use the insert statement for entering the Values Corresponding the different fields, Corresponding the datatype

6. Use the Commit() to Complete the transaction using the Connection object

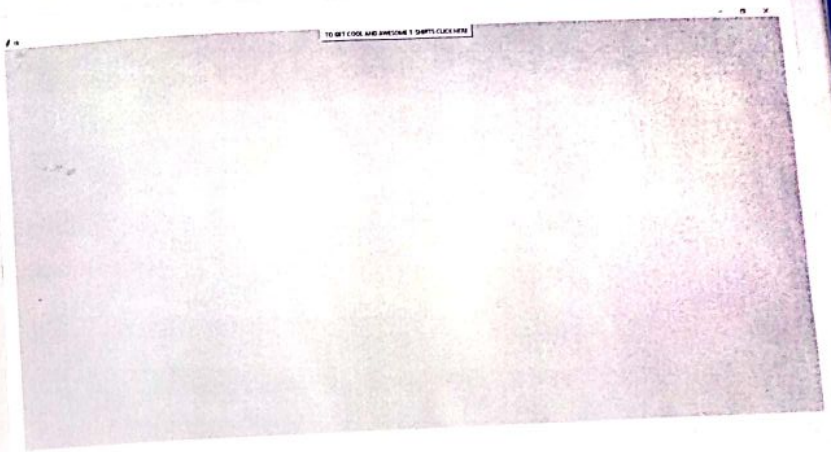
7. Use the execute statement do along with Cursor object for accessing the Values from the database using the Select from where Clause

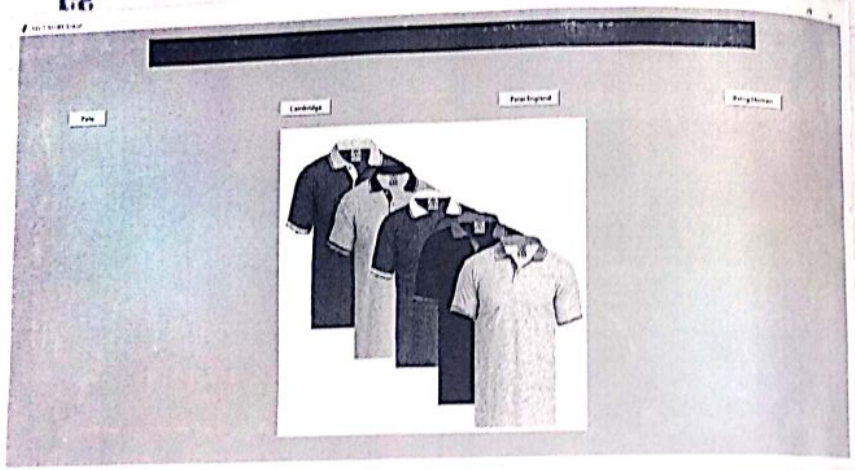
8. Finally use the fetch() or fetchall() for displaying the Values from the table using the Cursor-object

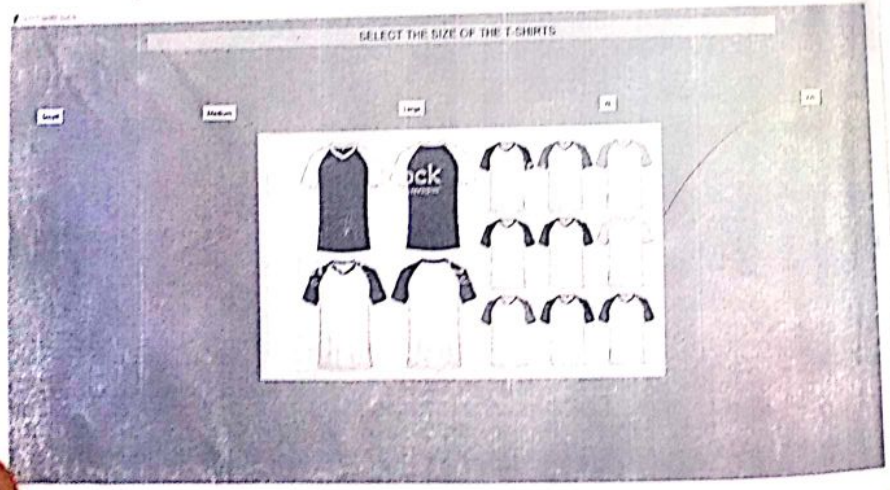
9. Execute() & drop table Syntax for terminating the database & finally use the close().

C.P GUI PROJECT

```
from tkinter import *
root=Tk()
root.minsize(200,200)
def main():
  top=Toplevel()
  top.config(bg="light green")
  top.title(" MY T-SHIRT SHOP ")
  top.minsize(100,100)
  l=Label(top,text=" MY T-SHIRT SHOP \n Our shop contains T-Shirt of every size. \n Our shop contains
T-Shirts of every brand. \n Our shop contains T-Shirts of different prices as per the brand and size. \n"
,width=100,font=("bold",15),bg="green")
  l.place(x=250,y=20)
  b=Button(top,text= "T-Shirts ranging from 500-1000 ",command=first)
  b.place(x=100,y=150)
  b1=Button(top,text= "T-Shirts ranging from 1000-1500 ",command=second)
  b1.place(x=100,y=350)
  b2=Button(top,text= "Don't want to buy ",command=terminate)
  b2.place(x=1250,y=350)
  b3=Button(top,text= "T-Shirts ranging from 1500-2000 ",command=third)
  b3.place(x=1250,y=150)
  image1=PhotoImage(file="t.gif")
  sub=image1.subsample(1,1)
  Label(top,image=sub).pack(side=BOTTOM)
  top.mainloop()
def first():
  top1=Toplevel()
  top1.config(bg="light blue")
  top1.title(" MY T-SHIRT SHOP ")
  top1.minsize(100,100)
  l=Label(top1,text=" T-SHIRTS RANGING FROM 500-1000 \n PLEASE SELECT THE
BRAND",width=100,font=("bold",15),bg="blue")
  l.place(x=250,y=20)
  b4=Button(top1,text="    Polo    ",command=size)
  b4.place(x=100,y=150)
  b5=Button(top1,text="   Cambridge   ",command=size)
  b5.place(x=500,y=150)
  b6=Button(top1,text="    Peter England   ",command=size)
  b6.place(x=900,y=150)
  bu=Button(top1,text="  Being Human   ",command=size)
  bu.place(x=1300,y=150)
  image=PhotoImage(file="t1.gif")
  sub1=image.subsample(2,2)
```





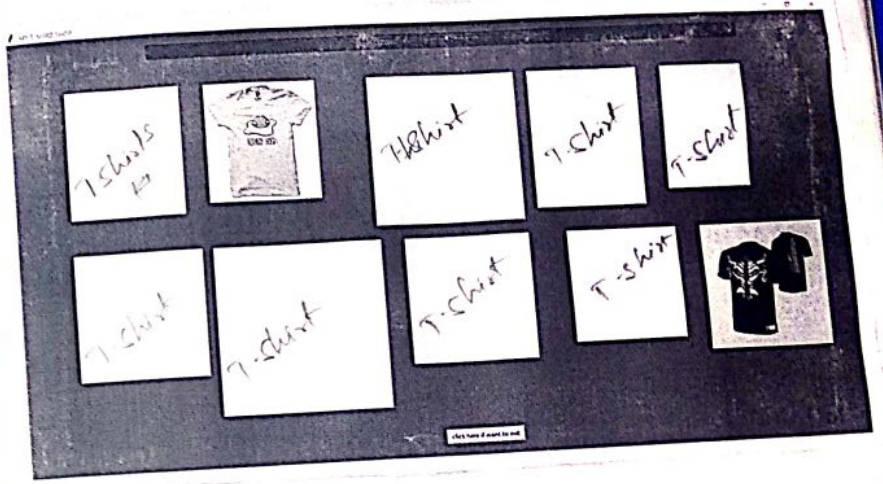



```
  Label(top1,image=sub1).place(x=500,y=200)
  top1.mainloop()
def second():
  top2=Toplevel()
  top2.config(bg="yellow")
  top2.title(" MY T-SHIRT SHOP ")
  top2.minsize(100,100)
  l=Label(top2,text=" T-SHIRTS RANGING FROM 1000-1500 \n PLEASE SELECT THE
BRAND",width=100,font=("bold",15),bg="red")
  l.place(x=250,y=20)
  b7=Button(top2,text=" Polo ",command=size)
  b7.place(x=100,y=150)
  b8=Button(top2,text=" Cambridge ",command=size)
  b8.place(x=450,y=150)
  b9=Button(top2,text=" Peter England ",command=size)
  b9.place(x=850,y=150)
  ba=Button(top2,text=" Being Human ",command=size)
  ba.place(x=1200,y=150)
  image3=PhotoImage(file="t4.gif")
  sub1=image3.subsample(1,1)
  Label(top2,image=sub1).pack(side=BOTTOM)
  top2.mainloop()
def third():
  top3=Toplevel()
  top3.config(bg="purple")
  top3.title(" MY T-SHIRT SHOP ")
  top3.minsize(100,100)
  l=Label(top3,text=" T-SHIRTS RANGING FROM 1500-2000 \n PLEASE SELECT THE
BRAND",width=100,font=("bold",15),bg="brown")
  l.place(x=250,y=20)
  b10=Button(top3,text=" Polo ",command=size)
  b10.place(x=100,y=150)
  b11=Button(top3,text=" Cambridge ",command=size)
  b11.place(x=450,y=150)
  b12=Button(top3,text=" Peter England ",command=size)
  b12.place(x=850,y=150)
  bs=Button(top3,text=" Being Human ",command=size)
  bs.place(x=1200,y=150)
  image4=PhotoImage(file="t6.gif")
  sub1=image4.subsample(1,1)
  Label(top3,image=sub1).place(x=450,y=200)
  top3.mainloop()
```

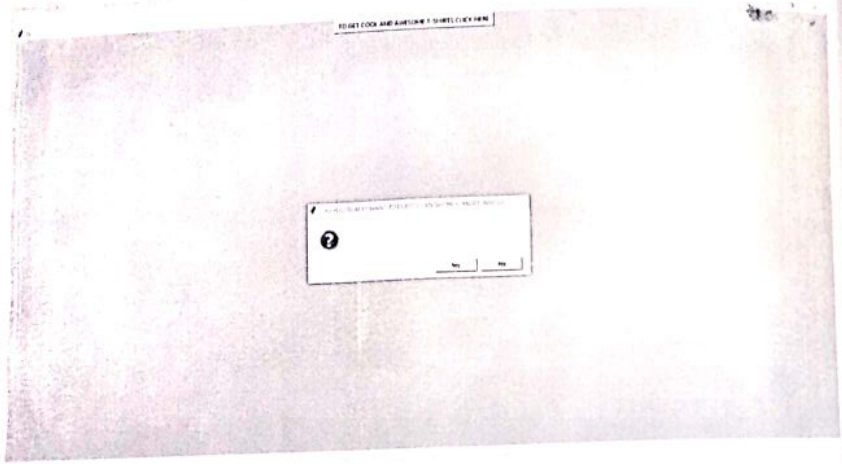
Scanned with CamScanner

```
def size():
    top4=Toplevel()
    top4.config(bg="orange")
    top4.title(" MY T-SHIRT SHOP ")
    top4.minsize(100,100)
    l=Label(top4,text=" SELECT THE SIZE OF THE T-SHIRTS ",width=100,font=("bold",15),bg="pink")
    l.place(x=250,y=20)
    b13=Button(top4,text=" Small ",command=p1)
    b13.place(x=50,y=150)
    b14=Button(top4,text=" Medium ",command=p1)
    b14.place(x=350,y=150)
    b15=Button(top4,text=" Large ",command=p1)
    b15.place(x=700,y=150)
    b16=Button(top4,text=" XL ",command=p1)
    b16.place(x=1050,y=150)
    b17=Button(top4,text=" XXL ",command=p1)
    b17.place(x=1400,y=150)
    image5=PhotoImage(file="t7.gif")
    sub1=image5.subsample(1,1)
    Label(top4,image=sub1).place(x=450,y=200)
    top4.mainloop()
def p1():
    top5=Toplevel()
    top5.config(bg="brown")
    top5.title(" MY T-SHIRT SHOP ")
    top5.minsize(100,100)
    l=Label(top5,text=" SELECT THE T-SHIRT U LIKE ",width=100,font=("bold",15),bg="blue")
    l.place(x=250,y=20)
    image6=PhotoImage(file="t9.gif")
    sub1=image6.subsample(3,3)
    Label(top5,image=sub1).place(x=350,y=100)
    image09=PhotoImage(file="t10.gif")
    sub2=image09.subsample(2,2)
    Label(top5,image=sub2).place(x=100,y=100)
    img=PhotoImage(file="ts.gif")
    sub2=img.subsample(4,4)
    Label(top5,image=sub2).place(x=650,y=100)
    image01=PhotoImage(file="t11.gif")
    sub2=image01.subsample(1,1)
    Label(top5,image=sub2).place(x=950,y=100)
    image02=PhotoImage(file="t12.gif")
    sub2=image02.subsample(2,2)
```






Scanned with CamScanner






```
Label(top5,image=sub2).place(x=1200,y=100)
image03=PhotoImage(file="t13.gif")
sub2=image03.subsample(1,1)
Label(top5,image=sub2).place(x=100,y=400)
image04=PhotoImage(file="t14.gif")
sub2=image04.subsample(3,3)
Label(top5,image=sub2).place(x=350,y=400)
image05=PhotoImage(file="t15.gif")
sub2=image05.subsample(1,1)
Label(top5,image=sub2).place(x=700,y=400)
image06=PhotoImage(file="t16.gif")
sub2=image06.subsample(3,3)
Label(top5,image=sub2).place(x=1000,y=400)
image07=PhotoImage(file="t18.gif")
sub2=image07.subsample(1,1)
Label(top5,image=sub2).place(x=1250,y=400)
b19=Button(top5,text=" click here if want to exit ",command=terminate)
b19.place(x=750,y=750)
top5.mainloop()
def terminate():
  messagebox.askyesno(" DO YOU REALLY WANT TO LEAVE I CAN SHOW U MORE AWESOME T-SHIRTS")
  quit()
root.config(bg="pink")
b18=Button(root,text=" TO GET COOL AND AWESOME T-SHIRTS CLICK HERE ",command=main)
b18.pack()
root.mainloop()
```

Scanned with CamScanner

# 5.3 DATABASE CONNECTIVITY

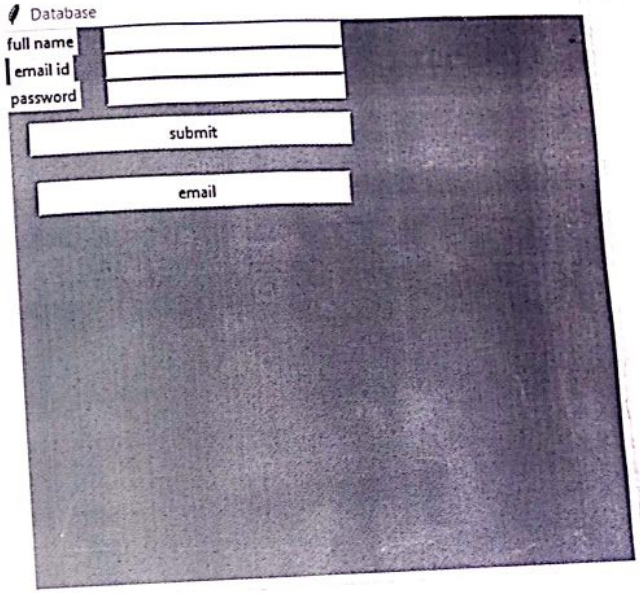
```
from tkinter import *
from tkinter import messagebox
root=Tk()
root.geometry("400x400")
root.config(bg="grey")
root.title("Database")
import os,sqlite3
db=sqlite3.connect("email.db")
c=db.cursor()
try:
  c.execute('create table email(full_name,email_id,password)')
except:
  print()
def new_window():
  c.execute('insert into email
values(:full_name,:email_id,:password)',{'full_name':full_name.get(),'email_id':email_id.get(),'password'
:password.get()})
  db.commit()
  messagebox.showinfo("tk","your email has been registered")
  full_name.delete(0,END)
  email_id.delete(0,END)
  password.delete(0,END)
def query():
  import os,sqlite3
  db=sqlite3.connect("email.db")
  c=db.cursor()
  c.execute('select * from email')
  fetch=c.fetchall()
  for i in fetch:
    print(i)
full_name=Entry(root,width=30)
full_name.grid(row=1,column=1,padx=20)

email_id=Entry(root,width=30)
email_id.grid(row=2,column=1)

password=Entry(root,width=30)
password.grid(row=3,column=1)

full_name_label=Label(root,text="full name")
full_name_label.grid(row=1,column=0)
```

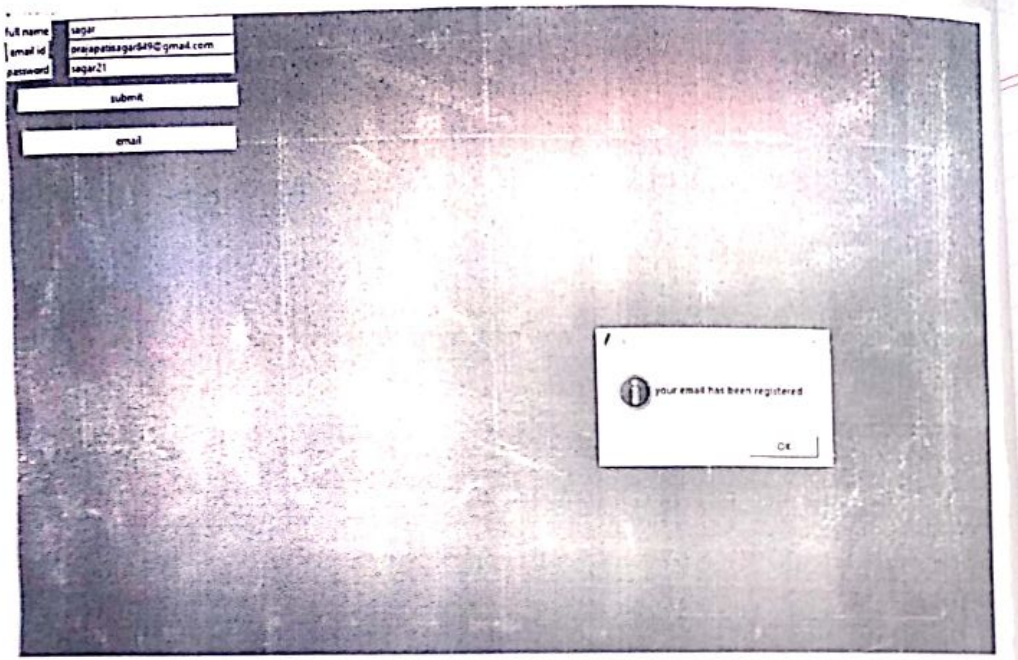





Enter the details asked and then press Submit. your details will be saved & the spaces will get empty again.

After you press e-mail your details will be shown on python Shell.

Scanned with CamScanner






```
email_id_label=Label(root,text="email id")
email_id_label.grid(row=2,column=0)

password_label=Label(root,text="password")
password_label.grid(row=3,column=0)


submit=Button(root,text="submit",command=new_window)
submit.grid(row=4,column=0,columnspan=5,padx=10,pady=10,ipadx=100)
see_your_email=Button(root,text="email",command=query)
see_your_email.grid(row=5,column=0,columnspan=2,padx=10,pady=10,ipadx=100)

mainloop()
```

Scanned with CamScanner