Exam Seat No._____

THAKUR COLLEGE OF SCIENCE & COMMERCE

NAAC Accredited with Grade "A" (3rd Cycle)

THAKUR TRUSTS

ISO 9001 : 2015 Certified

Degree College

Computer Journal

CERTIFICATE

SEMESTER II UID No. _____

Class FyBSc. CS Roll No. 1859 Year 2019-20

This is to certify that the work entered in this journal is the work of Mst. / Ms. Rohra. Rishi Anil

who has worked for the year 2019-20 in the Computer Laboratory.

Teacher In-Charge

Head of Department

Date :_____

Examiner

Scanned with CamScanner

Scanned with CamScanner

	Remarks	Teacher's Signature
I	Incomplete	12/10/1
II	Completed	

Scanned with CamScanner

Scanned with CamScanner

Sem - II

019

Practical no - 1 29-11-19

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

Step I : Create a file using open method then write some access mode continuing by writing some contents on the file and the close the file.

Step II : Open the file in read mode and then use read(), readline() and readlines()

Step III : Store the output in variables and finally display the contents.

Step IV : Use the file object to find the name of the file mode in which it is opened file can be either open or close and the output of softspace attribute

Scanned with CamScanner

Scanned with CamScanner

```
Sikobj = open("abc.txt", "w") # file open (write m
file obj.write ("CS subjects" + "\n")
file obj write (" Data structure in python \n DS \n")
file obj close()

file obj = open ("abc.txt", "r")
String1 = file obj. read()
print (" The output of read method:" String1
file obj. close()

file obj = open ("abc.txt", "r")
String2 = file obj. readline()
print (" print output of readline method:", String2
file obj. close()

file obj = open ("abc.txt", "r")
String3 = file obj. readlines()
print (" The output of readlines method:" String3
file obj. close()

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

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

Scanned with CamScanner

Step V : open file in write mode write more contents the close. then again open the file object in 'wr' mode then update and write more contents.

Step VI : open the file obj which was created in the read mode and display the updated written contents and close again and open again in 'rt' mode with parameters pursel and display the output.

Step VII : Again open the file obj is opened mode write method write the contents close the file obj, again open the file obj, again open it is read mode and display the appending output

```
c= fileobj mode
print("file mode", c)

d= fileobj - softspace
print("softspace", d

fileobj = open("abc.txt","wt")
fileobj.write("Ashish sir")
fileobj.close()

# readmode
fileobj = open("abc.txt","rt")
si = fileobj.read(6)
print("output of rt", si)
fileobj.close()

fileobj = open("abc.txt","a")
fileobj.write("data structure")
fileobj.close()
fileobj = open("abc.txt","r")
s3 = fileobj.read()
print("output of append:", s3)
fileobj.close()

fileobj = open("abc.txt","w")
fileobj.write("python")
fileobj.close()

fileobj = open("abc.txt","r")
s = fileobj.read()
print("output of read mode", s)
```


Scanned with CamScanner

050

```
fileobj = open("abc.txt", "r")
pos = fileobj.tell()
print("tell(): ", pos)
fileobj.close
```

```
fileobj = open("abc.txt", "r")
st = fileobj.seek(0,0)
print("seek(0,0) is: ", st)
fileobj.close
```

```
fileobj = open("abc.txt", "r")
st = fileobj.seek(0,1)
print("seek(0,1) is: ", st)
fileobj.close()
```

```
fileobj = open("abc.txt", "r")
st2 = fileobj.seek(0,2)
print("seek(0,2) is: ", st2)
```

```
fileobj = open("abc.txt", "r")
start = fileobj.readlines()
print("output: ", start)
for line in start:
    print(line, end=)
fileobj.close()
```

Scanned with CamScanner

Scanned with CamScanner

Step VIII : Open the file obj in read mode name a variable, and perform file obj dot read method and store the output consequently in variable

Step IX : Execute the read method with arguments with opening the file obj in read mode and closing.

Step X : Open the file obj with read mode also use readlines and store the output consequently is and pass the scan for counting the length of the statement and display the length.

2.112

Practical no 2.

Objective: Iterators.

Step1: Create a tuple with the elements that we need to iterate using the iter and next methord the number of time we use the iter and next methord we will get the next iterating elements in the tuple Display the same.

Step2: The similar output can be obtained by using conditional statement. An iterable variable is checked for loop which will be iterate.

Step 3: Def a function name Square with a parameter which will optain output of square value of the given number. In similar fashion checked with to get the value raised 3 and return the same.

Step 4: call the defined function using function call.

Scanned with CamScanner

Scanned with CamScanner

```
mytuple1 = ("yellow", "orange", "purple")
myiter1 = iter(mytuple)
print(next(myiter1))
myiter2 = iter(mytuple)
print(next(myiter1))
myiter3 = iter(mytuple)
print(next(myiter))
>>> yellow
>>> orange
>>> purple

mytuple1 = ("xyz", "yzx", "zyx")
for x in mytuple
    print(x)
>>> xyz
>>> yzx
>>> zxy

def square(x)
    y = x * x
    return y
def cube(x)
    z = x * x * x
    return z
```

Scanned with CamScanner

Step 5: use for conditional statement specifing the range use the list tuple casting with map method declare a lambda and a anonymous function.

Step 6: Define a list num variable and define some elements with the help of lambda a function give two argument display the output

Step 7: def a function with a give parameter then use conditional statement do check weather the num is even or odd and return respectively.

Step 8: Def a class and within that def the init () method with self initialize the field element within the constructor object.

Step 9: now use the reset () and def the logic for displaying odd value.

Step 10: Def a object for class.

Scanned with CamScanner

Scanned with CamScanner

```
for x in range(3):
value = list(map(lambda x: x(x), funcs))
    print(value)
>>> [22, 23]
    [4, 9]
    [1, 1]

lstnum = [0, 4, 5, 7, 9, 11, 13, 15, 20, 19, 25, 30, 31]
lstnum = list(map(lambda x: x % 5, lstnum))
print(lstnum)
def even(x):
    if (x % 2 == 0):
        return "Even"
    else:
        return "odd"

list(map(even, lstnum))

class odd:
    def __iter__(self):
        self.num = 1
        return self
    def __next__(self):
        num = self.num
        num += 2
    def __next__(self)
    num = self.num
```

Scanned with CamScanner

Scanned with CamScanner

```
PSO:
    self.num += 2
    return num
myobj = odd()
myiter = iter(myobj)
x = int(input("Enter elements:"))
for i in myiter
    if (i<x):
        print(i)
>>> Enter elements: 15
1
3
5
7
9
11
13
15
19
25
31
```

Scanned with CamScanner

practical no 3.

Aim: Find the avg using exception.

~~Step I~~:

1. Write a program using exception method of the native arithmetic error

Step I: Use the try block and except the input using the raw input method and convert it into the integer data type and subtraction terminate the block.

Step 2: use a except block with the exception name as a value error and display the appropriate message if the suspicious code is part of the 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.

Scanned with CamScanner

Scanned with CamScanner

Step 1: within the code block try to open the file using the write mode and write some content on the file.

Step 2: in the except block with no error and display the message including missing of the file or inappropriately of the mode use the else block to display a msg that the operation is correct or

arithmetic error.

```
while True:
    try:
        x = int(input("Enter a class"))
        break.
    except value error:
        print("Enter numeric class")
```

output

Enter a class: TYBSC-CS
Enter a num char:
Enter class: 13

environment error.

```
try:
    fo: open("abc.txt", "w")
    fo: write("RiRo7000")
except IOError:
    print("Error writing on file")
else:
    print("operation carried out successfully")
    fo.close()
```

output: operation carried out successfully.

Scanned with CamScanner

assert error

```
def assert(n):
    assert(len(n) == 0)
    print("list is empty")
val1 = []
print(assert(val1))
```

Output: list is empty
None

4)

```
def ampsage():
    age = int(input("Enter your age:"))
    if age is >30 or age is <18;
        raise value error
    return age
value = False
while not value:
    try:
        age = ampsage()
        value = True
    except value error:
        print("not valid age")
```

Output: Enter your age 19.

Scanned with CamScanner

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

Step 1: Def a function which accepts an arguments and check using the assert statement the given list is empty list and accordingly return the message

Step 2: close the function and in the body of program and def certain elements the list and give some appropriate action.

4. wap to check the range of the age of the students in the given class if the age do not fall in the age do the raise error exception otherwise return the valid no.

Step 1: Def a function which will accept the age of the student from the standard input.

Step 2: use if condition to check the whether the input age falls in the range and do the raise the raise exception.

Scanned with CamScanner

Scanned with CamScanner

Step 3: If the while loop to check the boolean expression holds true in the first block to compare to the age of the student and terminate the loop condition

Step 4: an the except with name error and print the msg not valid age.

Scanned with CamScanner

Scanned with CamScanner

Prac - 4

Topic: Regular expression

Step 1: Import re module define pattern method with define arguments matched then print the same otherwise print pattern not found.

Step 2: Import re module define pattern with lateral and multi character define string values with arguments and print the same

Step 3: Import re module define pattern with multi character use the split () and print the output.

Step 4: Import the re module. define string and accurately define pattern split the blank space with no-space use sub () with 3 arguments and print the string without space.

Scanned with CamScanner

Scanned with CamScanner

No space.

```
import re
string = 'abc def ghi'
pattern = r'\s+'
replace = ''
vi = re.sub(pattern, replace, string)
print(vi)
```

output

```
>>> abcdefghi
```

group

```
import re
sequence = 'python is very good language'
v = re.search('python', sequence)
print(v)
vi = v.groups()
print(vi)
```

output

```
<_sre.SRE_match object; span=(0,6),
match='python'>
python.
```

Scanned with CamScanner

Scanned with CamScanner

Step 5: Import the re-module define a seq are search method for finding subsequence are take the group (s) with dot operator as search (s)

Step 6: Import re module define test with number here we use the up for the condition sum use of counter is either 8 or 9 and result is either is sum of 0 to 9 and check whether the entered number are equal to 10 if criteria matches print the number or print fail

Step 7: Import re module define a string use the module with finalies for finding the value.

Scanned with CamScanner

vowels.

```
import re
str = 'plants are life saver'
output = re.findall(r'\b[aeiouAEIOU]\w+', str)
print(output)
```

```
>>> output
>>> 'are'
```

host and domain

```
import re
seq = 'abc.tcs@edu.com, xyz@gmail.com
output = re.findall(pattern, seq)
print(output)
```

```
>>> ['abc.tcs', 'edu.com', 'xyz', 'gmail.com
```

Scanned with CamScanner

Scanned with CamScanner

counting of first two last

```
import re
s = 'mr.a, ms.b, ms.c, ms.d
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
```

Scanned with CamScanner

Step 8: Import re module declare the host and domain name declare pattern for separate the host and domain name and the function and print the output

Step 9: Import re module enter a string use pattern to display only two elements of the particular string use function declare variables with initial value as zero use for condition and subsequently use the if condition call up the re here increment value and display the values subsequently.

17/9

Scanned with CamScanner

Scanned with CamScanner

Prac: 5

Aim: Gui components.

Step 1: use the tkinter lib for importing the feature
Step 2: create a variable from text method and position it on parent window
Step 3: use the pack method along with the obj created from text method
Step 4: create a variable from text method and position it on the parent window.
Step 5: use the pack method along with the obj created from text method and use the parameters
Step 6: text "left" pady = 20
text "left" pady = 30
text "Top" pady = 40
text "Top" pady = 50

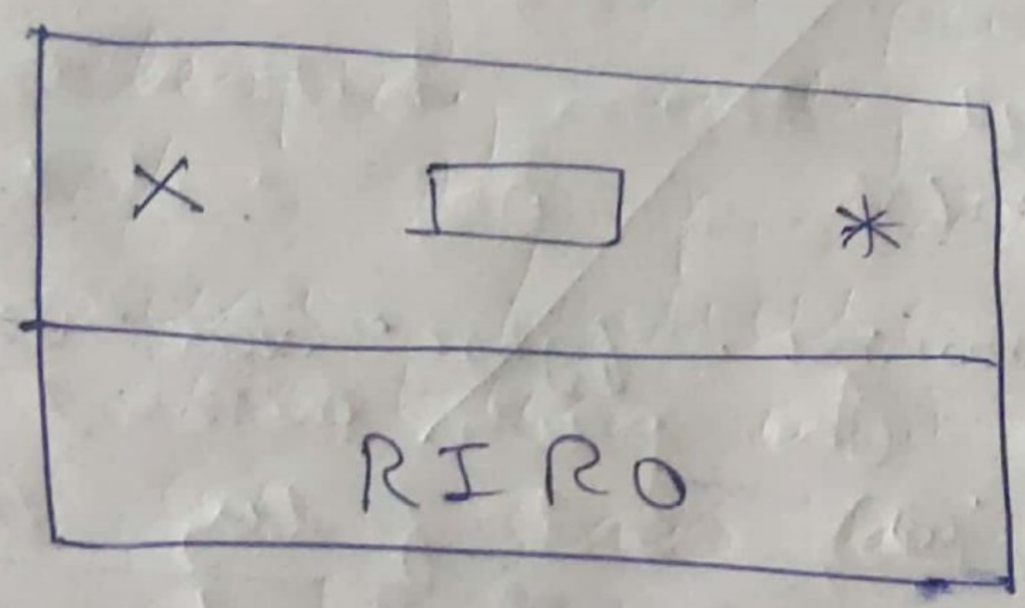
Step 7: now repeat the above step 1 with the table method when takes the following.
1. Text attribute 2) Bg
3. pg

Scanned with CamScanner

1. code

```
from tkinter import *
root = tk()
l = label (root, text = "RIRO")
l.pack()
root.mainloop()
```

output



2. code:

```
from tkinter import *
root = tk()
l = label (root, text = "Python")
l.pack (side = left pady = 30)
l1 = label (root, text "R2RO", bg = "blue", fg = "blue
l1.pack (side = top, ipady = 40)
m = label (root, text = "(S")
m.pack (side = left, padx = 20)
m2.pack (side = left, ipadx = 50)
root mainloop()
```

Scanned with CamScanner

Radio button

code

```
from tk *
root = tk()
def sel():
    selection = "you selected the option" + str(var.get())
    label.config(text = selection)
root = tk()
var = intvar()
r1 = Radiobutton(root, text = "fybsc", var = var, value = 1, command = sel)
r1.pack(anchor = w)
```

```
from tkinter import *
def sel()
    selection = "you selected the option" + str(var.get())
    label.config(text = selection)
root = Tk()
var = intvar()
R1 = Radiobutton(root, text = "FyBSC CS", var = var, value = 1
R1.pack(anchor = w)

R2 = Radiobutton(root, text = "SyBSC CS", var = var, value = 2
R2.pack(anchor = w)
R3 = Radiobutton = (root, text = "TyBSC CS", var = var, value = 3)
R3.pack(anchor = w)
label = label root
root . mainloop
```

Scanned with CamScanner

Radio button:-

Step 1: use the tkinter module to import module.
Step 2: def a function which tells about it and gives a selection.
Step 3: use the config method along with label obj and call it as a arguments
Step 4: now def a parent window.
Step 5: now create an obj from radio button

1. position on parent window
2. Def text variable [1, 2, 3, 4]

Step 6: pack method for the corresponding radio obj to specify the attribute

Step 7: def the label obj for corresponding the data.

Step 8: Def the obj corresponding to the obj window size of parent window in terms of no of pixels.

Step 9: now def the frame obj from the method and place in parent window.

Scanned with CamScanner

Scanned with CamScanner

II

Step 10: Simply clip the right from and by symmetry clip the bottom and now the gun from with the assistance as shown below by and by

Step 11: Now use the paint method along with it

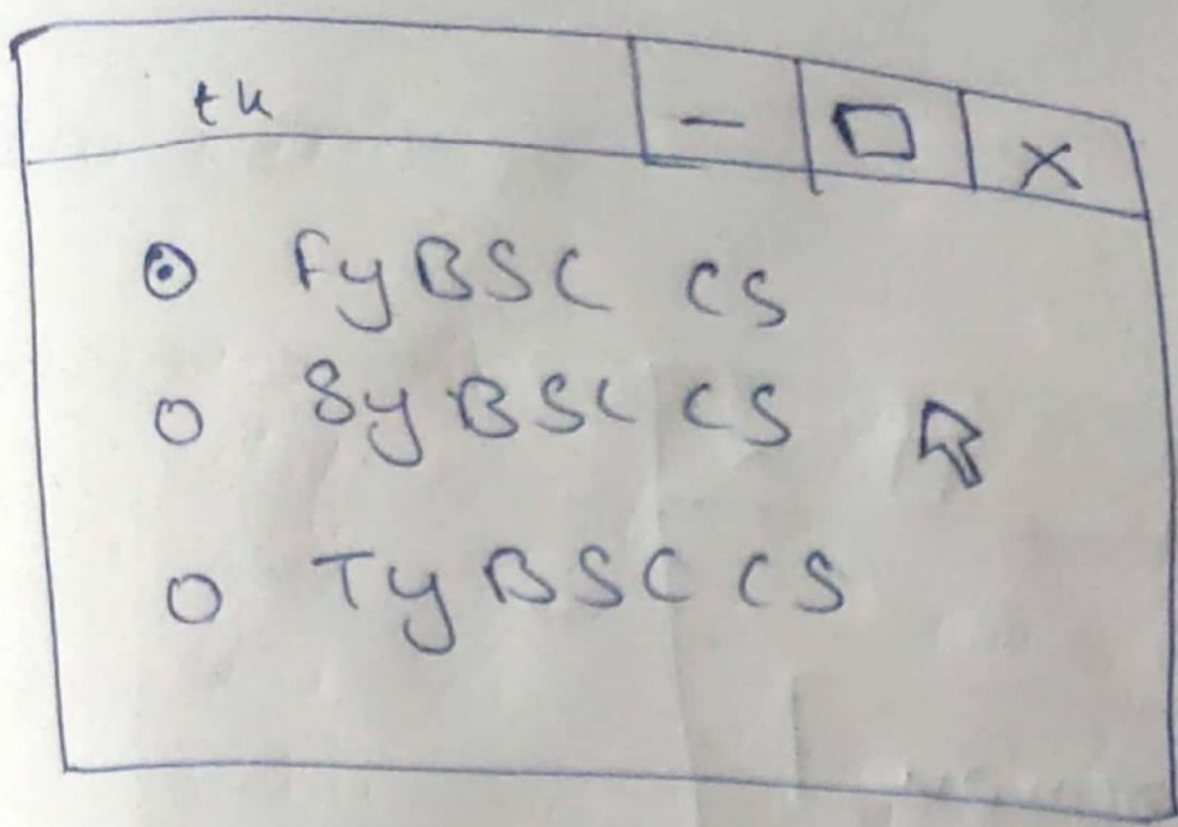
Step 12: add another button and put it on right from obj and turn it as

Step 13: use the paint method for all the obj and finally use the main loop method.

Scanned with CamScanner

Scanned with CamScanner

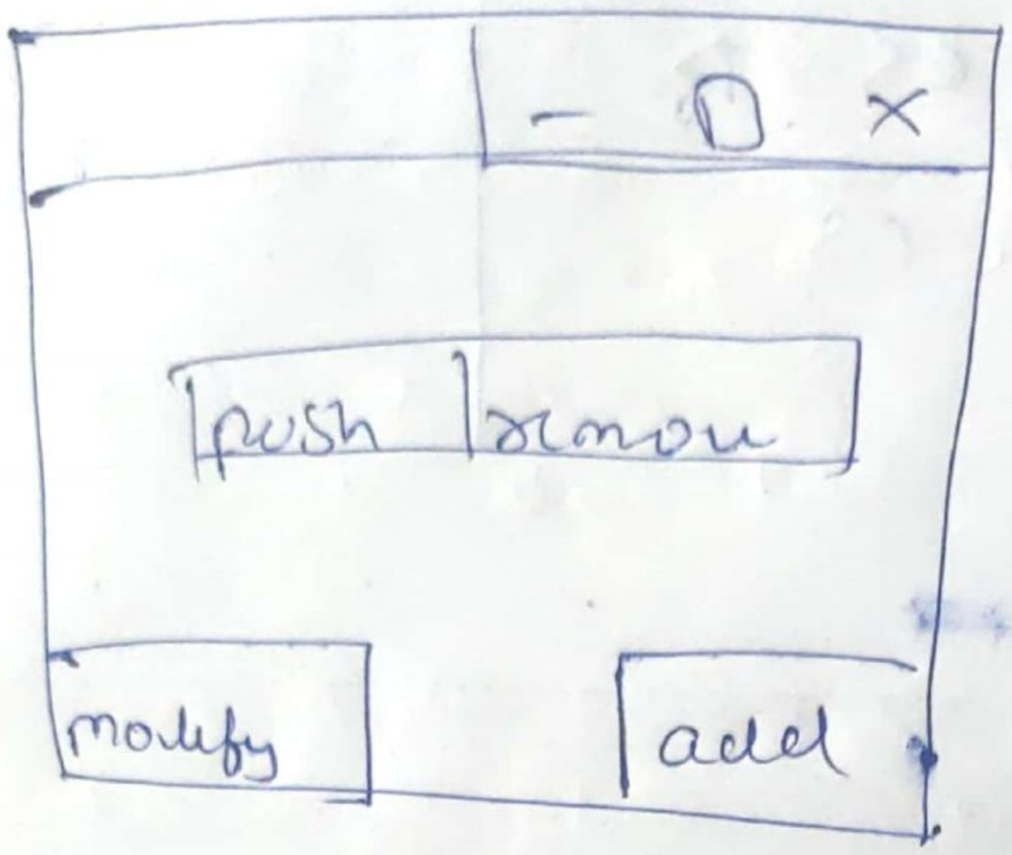
Radio button output




```
# from obj
from tkinter import *
root Tk()
frame = Frame(root)
frame.pack(padx=20, pady=50, side = Top)
left frame = Frame(root)
left frame.pack(side=left)
right frame = Frame(root)
right Frame.pack( side = RIGHT)
buttonpush = Button (frame, text="Push", fg="blue", bg="red")
buttonpush pack(side=left)
button remove = Button (frame, text="Remove" fg=green
                                bg= yellow")
button call = Button (right frame, text="call")
button call.pack(side=left)
button modify.pack(side=Right)
root.mainloop()
```

Scanned with CamScanner

Output



Scanned with CamScanner

* message box

I. Import the relevant Tk method.
II Def a function, and use the msg box
III Thus diff options are available are
showing warning(); showerror(), ask yes/no()
ask question
IV create obj from button method and place
it onto the parent window

V use a pack method to display the
button widget and fully on mainloop
method.

* Relief methods / style.

I use the button with the following attributes
1. parent window
2 Text
3. Relief.

II use the corresponding pack method for
displaying button

III. Fully use the mainloop

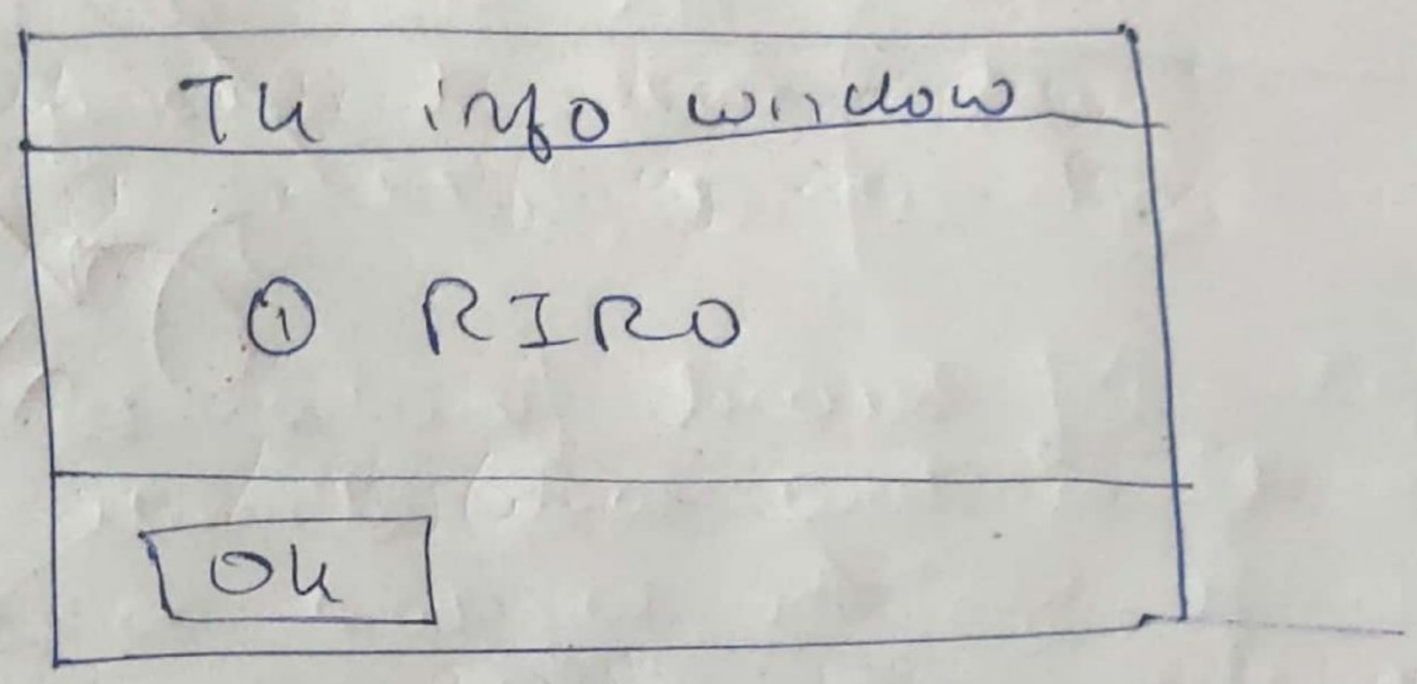
Scanned with CamScanner

Scanned with CamScanner

code

```
from Tkinter import *
import tk messagebox
root = Tk()
def fun()
    Tkmessagebox.showinfo("info window", "RIRO")
b1 = Button(root, text = "click", command = fun)
b1.pack
root.mainloop()
```

output



code

```
from Tkinter import *
root = Tk()
b1. Button(root, text = ("FyBSC", command = FyBSC)
b1.pack()
b2. Button(root, text ("SyBSC", command = SyBSC")

b2.pack()
b3. Button(root, text ("TyBSC", command = TyBSC")
b2.pack()
root.mainloop.
```

P.T.O

No. 6

TU

FYBSC

SYBSC

TYBSC

7/2

Scanned with CamScanner

II. transversing.

Step 1: Def the function and create an obj of the gun window by using the three methods namely config def and mainloop.

Step 2: Create an button obj and use the text and command attribute for triggering the gun event and use grid method along with inter and external padding.

Step 3: Def second function corresponding to second window with attribute config and mainloop for the window object and def one button obj which will shift the focus onto the third window.

Step 4: Create third window and in this create two button obj for moving onto first window for restarting the process and second button for termination.

Step 5: Def a function for terminating and call the first function create and trigger mainloop method.

Scanned with CamScanner

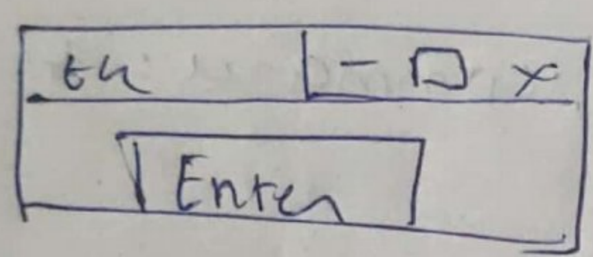
```
code:
from tkinter import *
root = tk()
def main():
    rt = Tk()
    rt.config(bg="blue")
    rt.title("main")
    rt.minsize(300,300)
    l = Label(rt, text="edm")
    l.pack()
    l1 = Label(rt, text="edm gives more fun")
    l1.pack()
    B1 = Button(rt, text="next", command = sl)
    B1.pack(side = Right)
    B2 = Button(rt, text="terminate", command = tr)
    B2.pack(side=Bottom)
    rt.mainloop()

def sl()
    master = Tk()
    master.config(bg="Blue")
    master.title("page 2")
    master.minsize(400,200)
    l2.Label(master, text="workout")
    l2.pack()
    l3 = Label(master, text="workout is good for health")
    l3.pack()
    B3 = Button(master, text="prev", command = main)
    B3.pack(side=LEFT)
```

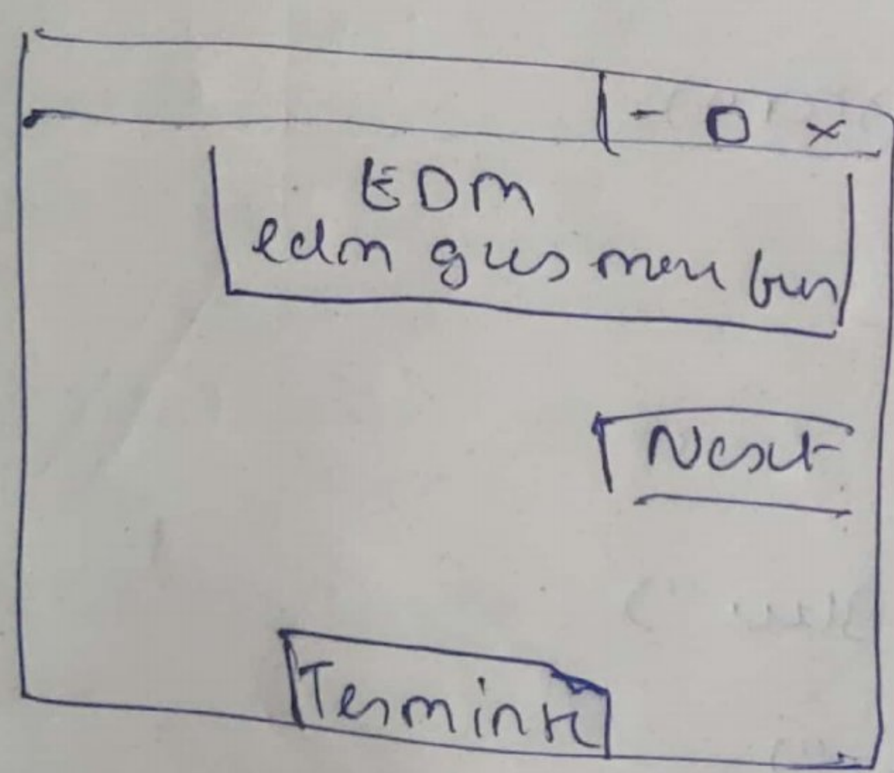
Scanned with CamScanner

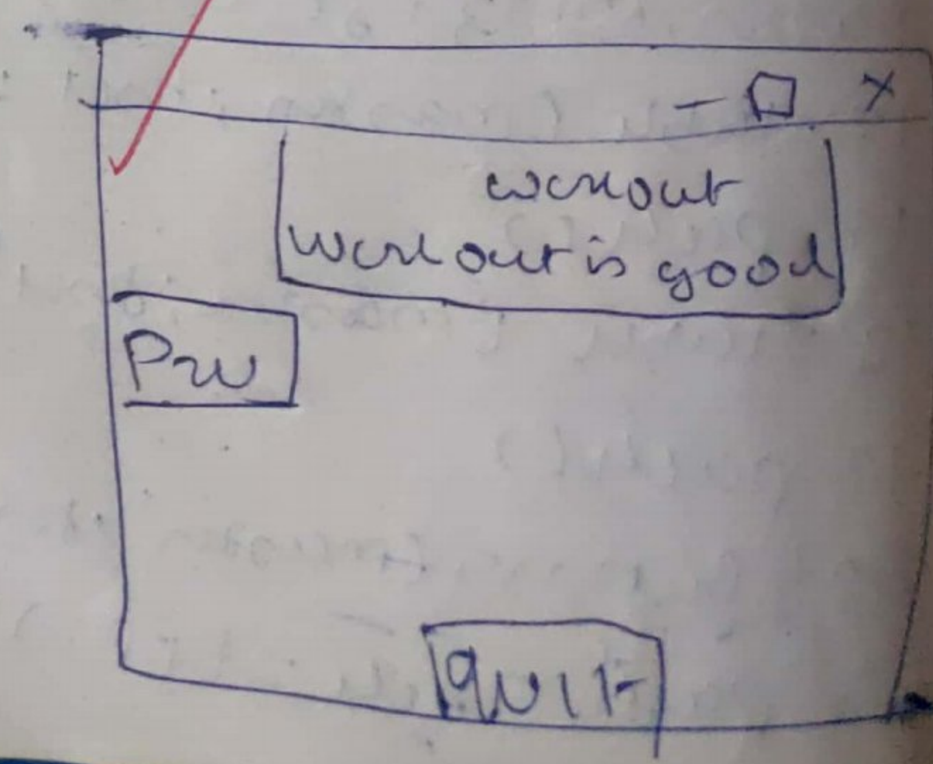
Scanned with CamScanner

```
B42 Button(master, text="quit", command=
B4.pack(side = BOTTOM)
master.mainloop()
def ex():
    quit()
B5= Button(root, text="Enter", command=
B5.pack()
root.mainloop()
```



Step I





Scanned with CamScanner

Spinbox.

Step 1: use the Tkinter lib to import method

Step 2: create the parent window obj

Step 3: create an obj from Spinbox method place it on to the parent window with the option. Spinbox.

Step 4: now use the pack method to make the obj visible to the parent window and call the mainloop method

Scanned with CamScanner

Scanned with CamScanner

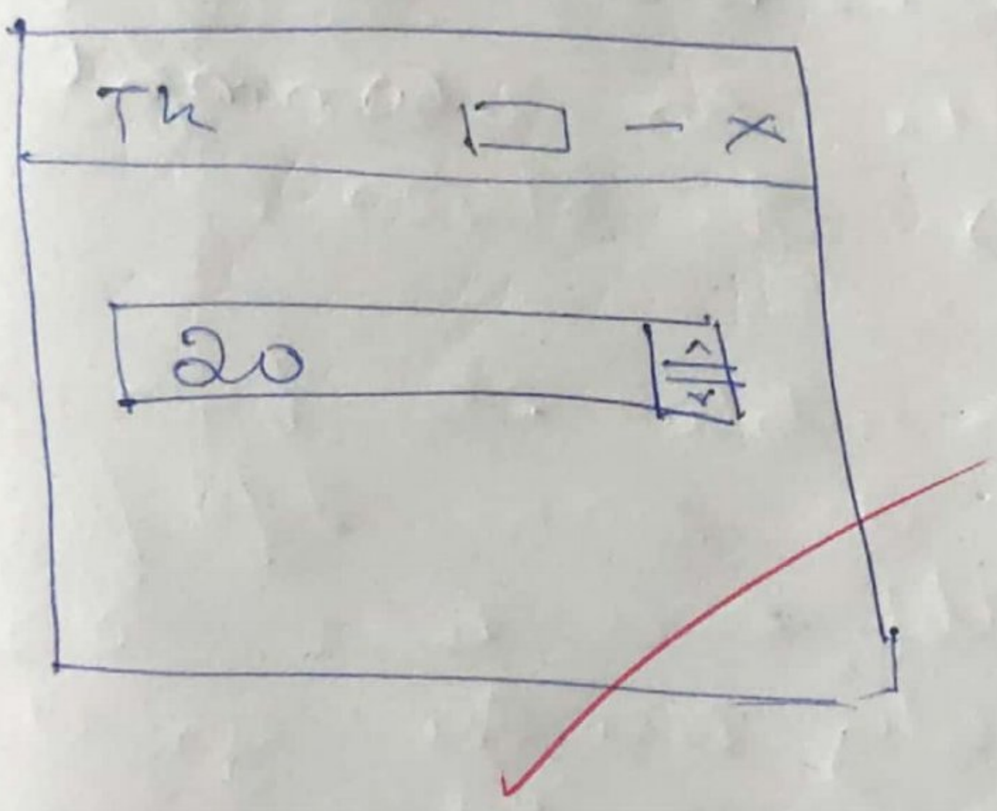
spinbox

```
from tkinter import *
top = Tk()

s1 = Spinbox(top, from = 0, to = 20)
s1.pack()

top.mainloop()
```

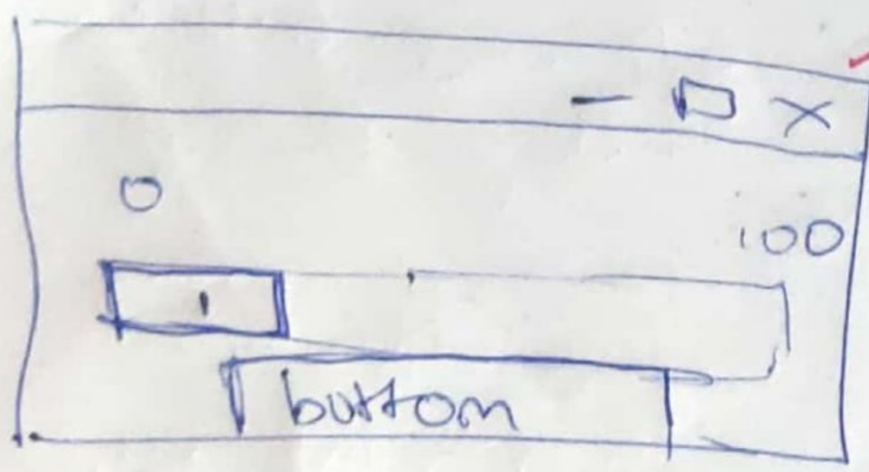
Output



Scanned with CamScanner

```
from tkinter import *
m1 = PanedWindow()
m1.pack(fill = BOTH, expand = 10)
e = Entry(m1, text = "left pane", bd = 10)
m2 = PanedWindow(m1, orient = Vertical)
m1.add(m2)
top = Scale(m2, orient = Horizontal)
m2.add(top)
bottom = Button(m2, text = "bottom pane",
                                  command = top)
m2.add(bottom)
m2.pack()
m2.mainloop()
```

output



Scanned with CamScanner

panel window

Step 1: create an obj from panel window method and in the pack method

Step 2: So make the object visible use pack method.

Step 3: now create an obj from entry widget and panel window one use the add method

Step 4: create an ob of the panel window and add it on the entry window.

Step 5: create an obj from Scale method and place it on the panel window and use add method.

Step 6: create button widget and place it on the panel window and def the function

Step 7: use pack and main loop method for it.

Scanned with CamScanner

Scanned with CamScanner

Displaying the img:

Step 1: Create an obj crossplatform parent window and use the following methods.

Step 2: Create the left frame obj and from the frame method and place it on to the parent window with height width

Step 3: now create an image frame obj as above the steps.

Step 4: Create an label obj from label and from it onto left frame and on the left and use grid. and set sticky use the grid method with padding

Step 5: now use the photo img from input file.

Step 6: use the Simple file and the img give the x, y, coordinate value.

Step 7: use the label method and position it onto the left frame and display

Scanned with CamScanner

Scanned with CamScanner

```
from tkinter import *
root = Tk()
root = title("Python")
root = minisize(1000, 2000)
root.config(bg = "black")
left frame = frame(root, bg: blue, height = 200, width = 200).grid
right frame = frame(root, bg = "green" height = 200, width = 200).grid
label(label, text = "photo", height = 2 width = 20); grid
image 1 = Photo Img (file = "1111.gif")
ori-img = img.subsample(1,2)
image 2 = Photo img (file = "iphone12.gif")
ori-img = image 2. subsample(3,4)
label(left frame, width = 200), height = 100 bg = width) grid
def name():
    print("name RishiRohan")
def hobby():
    print("hobbies: swimming")
def add
    print("add: navagaon")
def dob
    print("Dob = 10/11/1999)
```

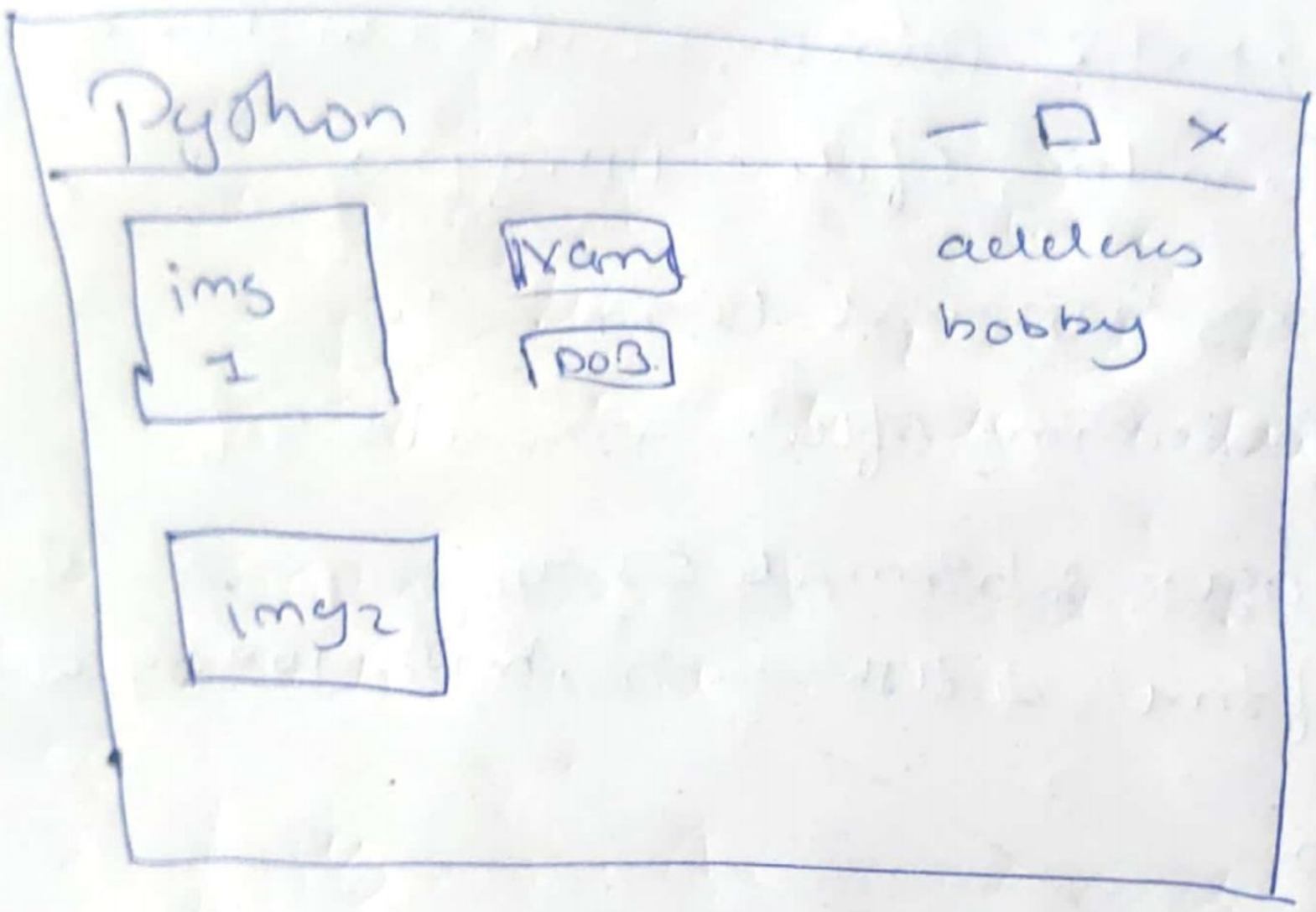
Scanned with CamScanner

S40

Button (root, text="Name", row=1, column=15, command=name)

Button (root, text="DOB", row=1, column=15, command=dob)

Output



Scanned with CamScanner

Scanned with CamScanner

Step 8: create a another obj positioning it on the img.

Step 9: now create a toolbox obj from the form method and position it onto the left form with the height

Step 10: now def various function of toolbox option

Step 11: now use the mainloop method.

Scanned with CamScanner

Scanned with CamScanner

* Canvas widget.

Step1: create an obj from the canvas widget by using the attribute height, width, bg colour, the parent window obj

Step 2: use the corresponding method for drawing the simple geometric shape like, arc, oval, line and specific co-ordinate value

Step 3 finally use the create line and create oval.

Step 4: finally use the pack and mainloop

Scanned with CamScanner

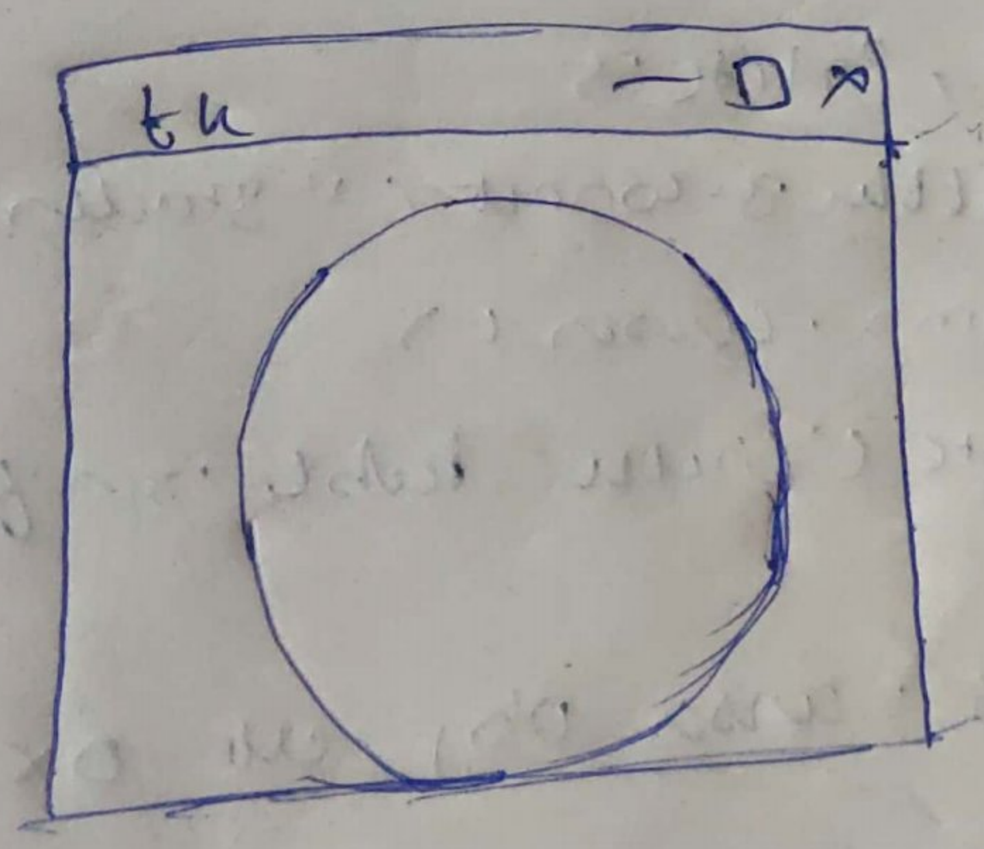
Scanned with CamScanner

code

```
Q2 oval

from Tk import *
master = Tk()
c = Canvas(master bg="white", height=1000,
           width=1500)
c.pack()
c.create_oval(50, 50, 400, 400)
c.mainloop()
```

output



Scanned with CamScanner

~~II program~~

~~>>> Import, sqlite3~~

~~>>> conn = sqlite3 connect ("Student.db")~~

~~>>> cur = conn.cursor()~~

~~>>> cur.execute ("create table into Rno, int, name text, DOB, date~~

~~> sqlite3.cursor, obj at 0x02F62F0>~~

~~>>> cur.execute (insert into values (01, "Bruce", "06-7-2001"), (02, "suresh", "06,04,2000)~~

~~>>> cur.execute (update, info set DOB=03-05,20 where R no 03'~~

code

>>> import, sqlite3

conn = sqlite3.connect ("Student.db")

cur = conn.cursor()

cur.execute ("create table student (Rno int, name text DOB date))

sqlite3.cursor obj at 0x02F62F20>

>>> cur.execute ("insert into values (01, "Jimmy", "2-2-

(02, "Jack", "3-7-99)

(03, "John", "4-7-99)

Scanned with CamScanner

Scanned with CamScanner

practical no 6.

Topic: Database connectivity.

write a program

Step1: Import the lib of database and the OS function.

Step2: now create an obj for making the connection to the gen database

Step3: Future create obj corresponding to the cursor on of execution of the diff query statement.

Step4: use the cursor object created for implementing the structure of the database and the value inserts the DB.

Step5: now use the fetch() along with the cursor obj for displaying the DB value onto the screen.

Scanned with CamScanner

Scanned with CamScanner

>> cur.execute("select DOB from info")
<sqlite3.Cursor object at 0x02F62F20>
[('01-2-99'), ('02-7-99'), ('03-7-99')]

>> cur.execute("update info set DOB = '03-01-99' where RNO=03")
<sqlite3.Cursor object at 0x02F62F20>

>> cur.execute("select DOB from info")
<sqlite3.Cursor object at 0x02F62F20>

>> cur.fetchall()

[('02-9-2000'), ('19-7-99')]

>> cur.execute("drop table info and all values")
<sqlite3.Cursor object at 0x02F62F20>

>> cur.close()

Scanned with CamScanner

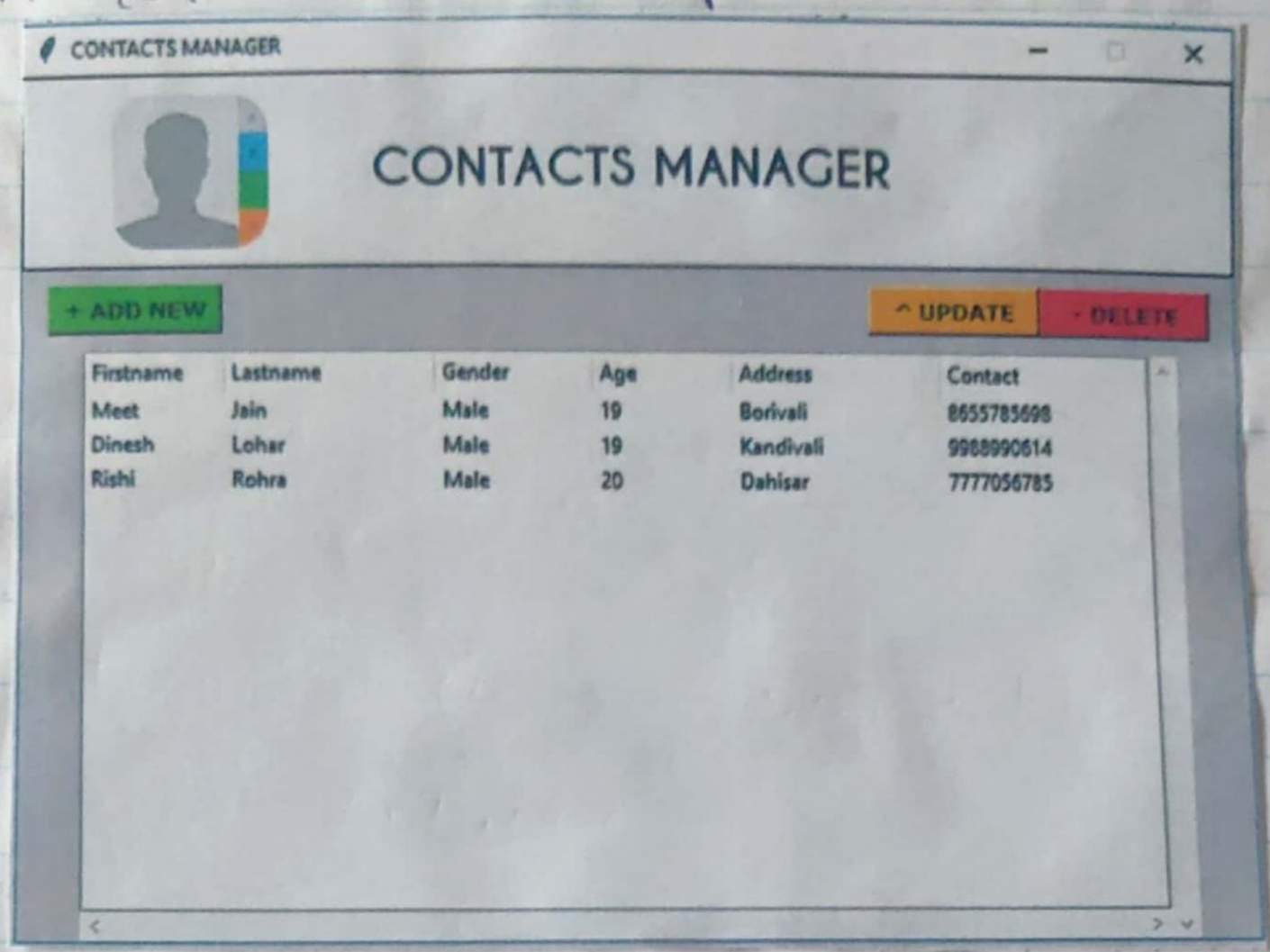
Scanned with CamScanner

Practical - 7

Project -

Aim: contact manager.

When you "After update contact"

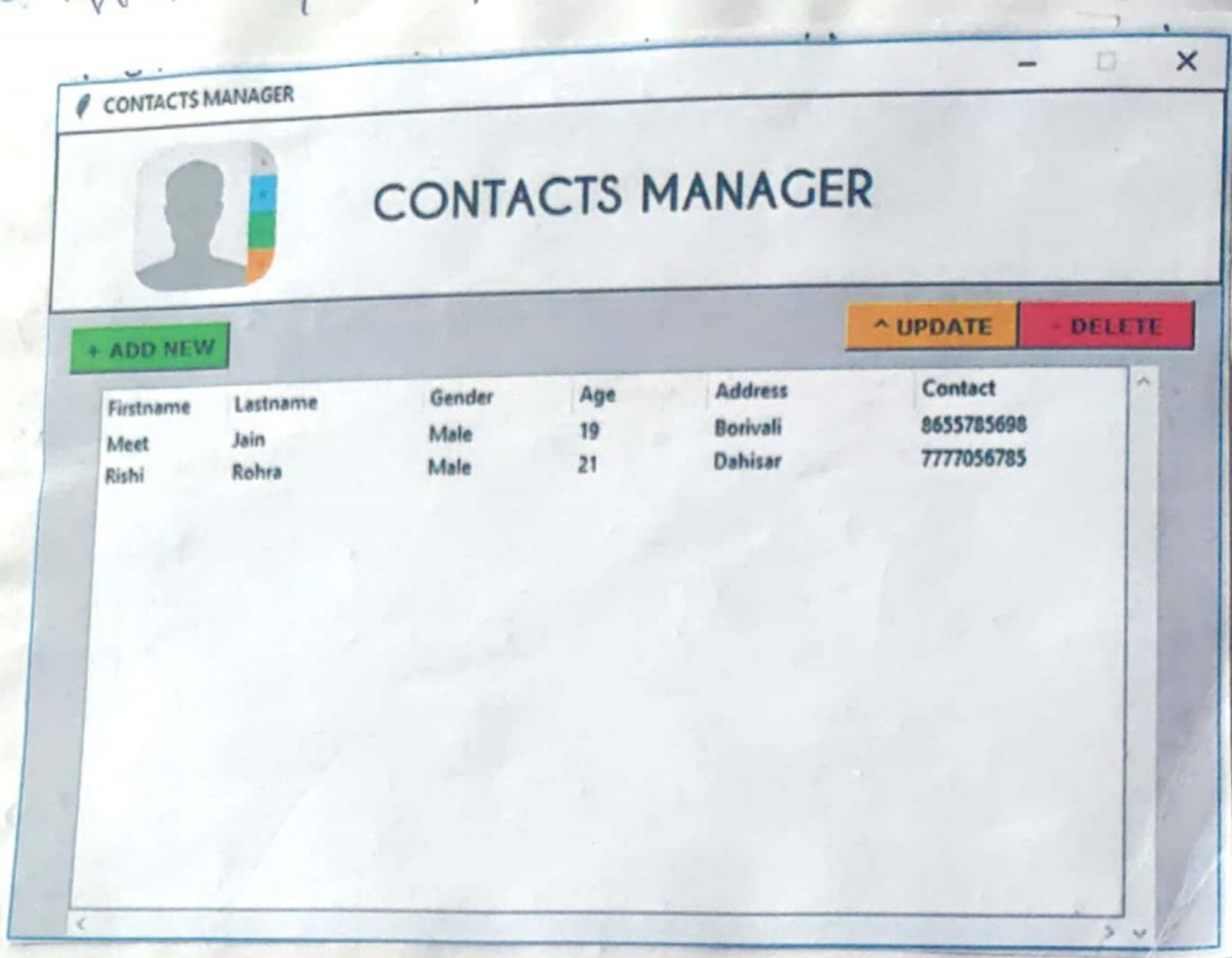


When you click on " ^ update " Button

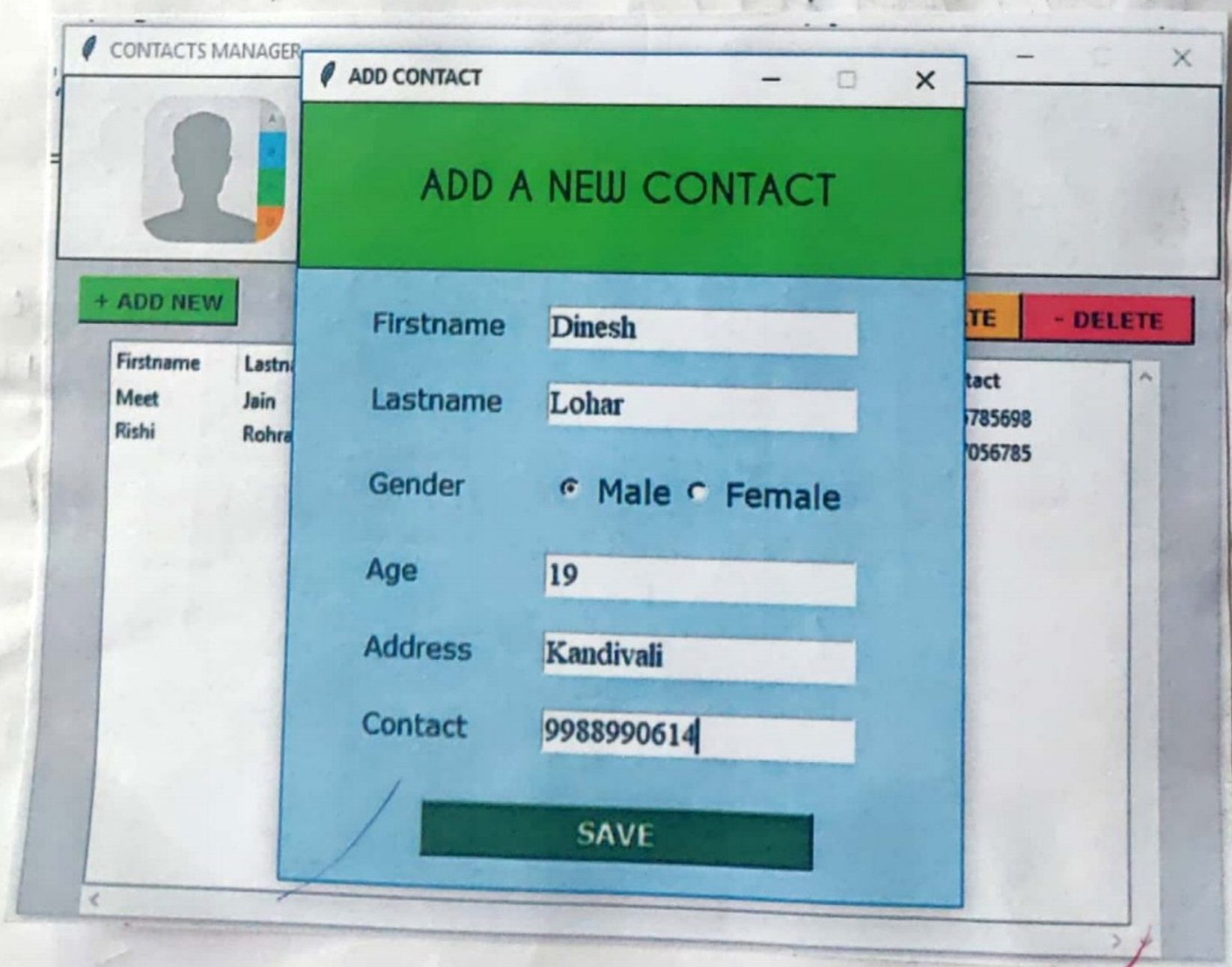
Scanned with CamScanner

Scanned with CamScanner

When you open the program (Run)



When you click on "+ADD NEW" Button



Scanned with CamScanner

```
from tkinter import *
import sqlite3
import tkinter.ttk as ttk
import tkinter.messagebox as tkMessageBox

root = Tk()
root.title("CONTACTS MANAGER")
width = 700
height = 500
screen_width = root.winfo_screenwidth()
screen_height = root.winfo_screenheight()
x = (screen_width/2) - (width/2)
y = (screen_height/2) - (height/2)
root.geometry("%dx%d+%d+%d" % (width, height, x, y))
root.resizable(0, 0)
root.config(bg="#c4c4c4")

#VARIABLES

FIRSTNAME = StringVar()
LASTNAME = StringVar()
GENDER = StringVar()
AGE = StringVar()
ADDRESS = StringVar()
CONTACT = StringVar()

#METHODS

def Database():
    conn = sqlite3.connect("pythontut.db")
    cursor = conn.cursor()
    cursor.execute("CREATE TABLE IF NOT EXISTS `member` (mem_id INTEGER NOT NULL  PRIMARY KEY AUTOINCREMENT, firstname TEXT,
lastname TEXT, gender TEXT, age TEXT, address TEXT, contact TEXT)")
    cursor.execute("SELECT * FROM `member` ORDER BY `lastname` ASC")
    fetch = cursor.fetchall()
    for data in fetch:
        tree.insert('', 'end', values=(data))
    cursor.close()
    conn.close()

def SubmitData():
    if  FIRSTNAME.get() == "" or LASTNAME.get() == "" or GENDER.get() == "" or AGE.get() == "" or ADDRESS.get() == "" or
CONTACT.get() == "":
        result = tkMessageBox.showwarning('Warning !', 'Please Complete The Required Field', icon="warning")
    else:
        tree.delete(*tree.get_children())
        conn = sqlite3.connect("pythontut.db")
        cursor = conn.cursor()
        cursor.execute("INSERT INTO `member` (firstname, lastname, gender, age, address, contact) VALUES(?, ?, ?, ?, ?, ?)",
(str(FIRSTNAME.get()), str(LASTNAME.get()), str(GENDER.get()), int(AGE.get()), str(ADDRESS.get()), str(CONTACT.get())))
        conn.commit()
        cursor.execute("SELECT * FROM `member` ORDER BY `lastname` ASC")
        fetch = cursor.fetchall()
        for data in fetch:
            tree.insert('', 'end', values=(data))
        cursor.close()
        conn.close()
        FIRSTNAME.set("")
        LASTNAME.set("")
        GENDER.set("")
        AGE.set("")
        ADDRESS.set("")
        CONTACT.set("")

def UpdateData():
    if GENDER.get() == "":
        result = tkMessageBox.showwarning('Warning !', 'Please Complete The Required Field', icon="warning")
    else:
        tree.delete(*tree.get_children())
        conn = sqlite3.connect("pythontut.db")
        cursor = conn.cursor()
        cursor.execute("UPDATE `member` SET `firstname` = ?, `lastname` = ?, `gender` =?, `age` = ?,  `address` = ?, `contact` = ?
WHERE `mem_id` = ?", (str(FIRSTNAME.get()), str(LASTNAME.get()), str(GENDER.get()), str(AGE.get()), str(ADDRESS.get()),
str(CONTACT.get()), int(mem_id)))
        conn.commit()
        cursor.execute("SELECT * FROM `member` ORDER BY `lastname` ASC")
        fetch = cursor.fetchall()
        for data in fetch:
            tree.insert('', 'end', values=(data))
        cursor.close()
        conn.close()
        FIRSTNAME.set("")
        LASTNAME.set("")
        GENDER.set("")
        AGE.set("")
        ADDRESS.set("")
        CONTACT.set("")

def OnSelected():
    global mem_id, UpdateWindow
    curItem = tree.focus()
    contents =(tree.item(curItem))
    selecteditem = contents['values']
    if not tree.selection():
        result = tkMessageBox.showwarning('Warning !', 'Please Select A Contact First!', icon="warning")
    else:
        result = tkMessageBox.askquestion('Confirmation !', 'Are you sure you want to Update this record?')
        if result=='yes':
            mem_id = selecteditem[0]
            FIRSTNAME.set("")
            LASTNAME.set("")
            GENDER.set("")
            AGE.set("")
```


Scanned with CamScanner

Scanned with CamScanner

```
        ADDRESS.set("")
        CONTACT.set("")
        FIRSTNAME.set(selecteditem[1])
        LASTNAME.set(selecteditem[2])
        AGE.set(selecteditem[4])
        ADDRESS.set(selecteditem[5])
        CONTACT.set(selecteditem[6])
        UpdateWindow = Toplevel()
        UpdateWindow.config(bg='light blue')
        UpdateWindow.title("UPDATE CONTACT")
        width = 400
        height = 460
        screen_width = root.winfo_screenwidth()
        screen_height = root.winfo_screenheight()
        x = ((screen_width/2) + 450) - (width/2)
        y = ((screen_height/2) + 20) - (height/2)
        UpdateWindow.resizable(0, 0)
        UpdateWindow.geometry("%dx%d+%d+%d" % (width, height, x, y))
        if 'NewWindow' in globals():
            NewWindow.destroy()

        #FRAMES
        FormTitle = Frame(UpdateWindow, bg='light blue')
        FormTitle.pack(side=TOP)
        ContactForm = Frame(UpdateWindow, bg='light blue')
        ContactForm.pack(side=TOP, pady=10)
        RadioGroup = Frame(ContactForm, bg='light blue')
        Male = Radiobutton(RadioGroup, text="Male", variable=GENDER, value="Male",  font=('Verdana', 14), bg='light
blue').pack(side=LEFT)
        Female = Radiobutton(RadioGroup, text="Female", variable=GENDER, value="Female",  font=('Verdana', 14), bg='light
blue').pack(side=LEFT)

        #LABELS
        lbl_title = Label(FormTitle, text="\nUPDATE A CONTACT\n", font=('Caviar Dreams', 18, 'bold'), bg="#ffa500",  width =
300, relief=SOLID, bd=1)
        lbl_title.pack(fill=X)
        lbl_firstname = Label(ContactForm, text="Firstname", font=('Verdana', 12), bd=5, bg='light blue')
        lbl_firstname.grid(row=0, sticky=W)
        lbl_lastname = Label(ContactForm, text="Lastname", font=('Verdana', 12), bd=5, bg='light blue')
        lbl_lastname.grid(row=1, sticky=W)
        lbl_gender = Label(ContactForm, text="Gender", font=('Verdana', 12), bd=5, bg='light blue')
        lbl_gender.grid(row=2, sticky=W)
        lbl_age = Label(ContactForm, text="Age", font=('Verdana', 12), bd=5, bg='light blue')
        lbl_age.grid(row=3, sticky=W)
        lbl_address = Label(ContactForm, text="Address", font=('Verdana', 12), bd=5, bg='light blue')
        lbl_address.grid(row=4, sticky=W)
        lbl_contact = Label(ContactForm, text="Contact", font=('Verdana', 12), bd=5, bg='light blue')
        lbl_contact.grid(row=5, sticky=W)

        #ENTRY
        firstname = Entry(ContactForm, textvariable=FIRSTNAME, font=('Times', 14))
        firstname.grid(row=0, column=1, padx=20, pady=10)
        lastname = Entry(ContactForm, textvariable=LASTNAME, font=('Times', 14))
        lastname.grid(row=1, column=1, padx=20, pady=10)
        RadioGroup.grid(row=2, column=1, padx=20, pady=10)
        age = Entry(ContactForm, textvariable=AGE, font=('Times', 14))
        age.grid(row=3, column=1, padx=20, pady=10)
        address = Entry(ContactForm, textvariable=ADDRESS, font=('Times', 14))
        address.grid(row=4, column=1, padx=20, pady=10)
        contact = Entry(ContactForm, textvariable=CONTACT, font=('Times', 14))
        contact.grid(row=5, column=1, padx=20, pady=10)


        #BUTTONS
        btn_updatecon = Button(ContactForm, text="UPDATE", font='Verdana 12 bold', width=20, command=UpdateData, bg='dark
blue', fg='white')
        btn_updatecon.grid(row=6, columnspan=2, pady=10)

def DeleteData():
    if not tree.selection():
       result = tkMessageBox.showwarning('Warning !', 'Please Select A Contact First!', icon="warning")
    else:
        result = tkMessageBox.askquestion('Confirmation !', 'Are you sure you want to delete this record?')
        if result == 'yes':
            curItem = tree.focus()
            contents =(tree.item(curItem))
            selecteditem = contents['values']
            tree.delete(curItem)
            conn = sqlite3.connect("pythontut.db")
            cursor = conn.cursor()
            cursor.execute("DELETE FROM `member` WHERE `mem_id` = %d" % selecteditem[0])
            conn.commit()
            cursor.close()
            conn.close()

def AddNewWindow():
    global NewWindow
    FIRSTNAME.set("")
    LASTNAME.set("")
    GENDER.set("")
    AGE.set("")
    ADDRESS.set("")
    CONTACT.set("")
    NewWindow = Toplevel()
    NewWindow.config(bg='light blue')
    NewWindow.title("ADD CONTACT")
    width = 400
    height = 460
    screen_width = root.winfo_screenwidth()
    screen_height = root.winfo_screenheight()
    x = ((screen_width/2) - 455) - (width/2)
    y = ((screen_height/2) + 20) - (height/2)
    NewWindow.resizable(0, 0)
    NewWindow.geometry("%dx%d+%d+%d" % (width, height, x, y))
```


Scanned with CamScanner

Scanned with CamScanner

```
    if 'UpdateWindow' in globals():
        UpdateWindow.destroy()

    #FRAMES
    FormTitle = Frame(NewWindow, bd=1, relief=SOLID)
    FormTitle.pack(side=TOP)
    ContactForm = Frame(NewWindow, bg='light blue')
    ContactForm.pack(side=TOP, pady=10)
    RadioGroup = Frame(ContactForm, bg='light blue')
    Male = Radiobutton(RadioGroup, text="Male", variable=GENDER, value="Male",  font=('Verdana', 14), bg='light
blue').pack(side=LEFT)
    Female = Radiobutton(RadioGroup, text="Female", variable=GENDER, value="Female",  font=('Verdana', 14), bg='light
blue').pack(side=LEFT)

    #LABELS
    lbl_title = Label(FormTitle, text="\nADD A NEW CONTACT\n", font=('Caviar Dreams', 18,'bold'), bg="#5bff2e",  width = 300)
    lbl_title.pack(fill=X)
    lbl_firstname = Label(ContactForm, text="Firstname", font=('Verdana', 12), bd=5, bg='light blue')
    lbl_firstname.grid(row=0, sticky=W)
    lbl_lastname = Label(ContactForm, text="Lastname", font=('Verdana', 12), bd=5, bg='light blue')
    lbl_lastname.grid(row=1, sticky=W)
    lbl_gender = Label(ContactForm, text="Gender", font=('Verdana', 12), bd=5, bg='light blue')
    lbl_gender.grid(row=2, sticky=W)
    lbl_age = Label(ContactForm, text="Age", font=('Verdana', 12), bd=5, bg='light blue')
    lbl_age.grid(row=3, sticky=W)
    lbl_address = Label(ContactForm, text="Address", font=('Verdana', 12), bd=5, bg='light blue')
    lbl_address.grid(row=4, sticky=W)
    lbl_contact = Label(ContactForm, text="Contact", font=('Verdana', 12), bd=5, bg='light blue')
    lbl_contact.grid(row=5, sticky=W)

    #ENTRY
    firstname = Entry(ContactForm, textvariable=FIRSTNAME, font=('Times', 14))
    firstname.grid(row=0, column=1, padx=20, pady=10)
    lastname = Entry(ContactForm, textvariable=LASTNAME, font=('Times', 14))
    lastname.grid(row=1, column=1, padx=20, pady=10)
    RadioGroup.grid(row=2, column=1, padx=20, pady=10)
    age = Entry(ContactForm, textvariable=AGE,  font=('Times', 14))
    age.grid(row=3, column=1, padx=20, pady=10)
    address = Entry(ContactForm, textvariable=ADDRESS,  font=('Times', 14))
    address.grid(row=4, column=1, padx=20, pady=10)
    contact = Entry(ContactForm, textvariable=CONTACT,  font=('Times', 14))
    contact.grid(row=5, column=1, padx=20, pady=10)


    #BUTTONS
    btn_addcon = Button(ContactForm, text="SAVE", font='Verdana 12 bold', width=20, command=SubmitData, bg='dark green',
fg='white')
    btn_addcon.grid(row=6, columnspan=2, pady=20)


#FRAMES
Top = Frame(root, width=500, bd=1, relief=SOLID)
Top.pack(side=TOP)
Mid = Frame(root, width=500,  bg="#c4c4c4")
Mid.pack(side=TOP)
MidLeft = Frame(Mid, width=100)
MidLeft.pack(side=LEFT, pady=10)
MidLeftPadding = Frame(Mid, width=370, bg="#c4c4c4")
MidLeftPadding.pack(side=LEFT)
MidRight = Frame(Mid, width=100)
MidRight.pack(side=RIGHT, pady=10)
TableMargin = Frame(root, width=500)
TableMargin.pack(side=TOP)
#LABELS
lbl_title = Label(Top, text="\nCONTACTS MANAGER\n", font=('Caviar Dreams', 21,'bold'), width=500, bg='#f0f0f0')
lbl_title.pack(fill=X)

#BUTTONS
btn_add = Button(MidLeft, text="+ ADD NEW", bg="#5bff2e", command=AddNewWindow, width=10, font='Verdana 10 bold')
btn_add.pack()
btn_delete = Button(MidRight, text="- DELETE", bg="#ff2e2e", command=DeleteData, width=10, font='Verdana 10 bold')
btn_delete.pack(side=RIGHT)

btn_update = Button(MidRight, text="^ UPDATE", bg="#ffa500", command=OnSelected, width=10, font='Verdana 10 bold')
btn_update.pack(side=LEFT)

pic=PhotoImage(file='logo.png')
pic.subsample(3,3)
Label(root, image=pic, bg='#f0f0f0').place(x=50, y=10)

#TABLES
scrollbarx = Scrollbar(TableMargin, orient=HORIZONTAL)
scrollbary = Scrollbar(TableMargin, orient=VERTICAL)
tree = ttk.Treeview(TableMargin, columns=("MemberID", "Firstname", "Lastname", "Gender", "Age", "Address", "Contact"),\
                    height=400, selectmode="extended", yscrollcommand=scrollbary.set, xscrollcommand=scrollbarx.set)
scrollbary.config(command=tree.yview)
scrollbary.pack(side=RIGHT, fill=Y)
scrollbarx.config(command=tree.xview)
scrollbarx.pack(side=BOTTOM, fill=X)
tree.heading('MemberID', text="MemberID", anchor=W)
tree.heading('Firstname', text="Firstname", anchor=W)
tree.heading('Lastname', text="Lastname", anchor=W)
tree.heading('Gender', text="Gender", anchor=W)
tree.heading('Age', text="Age", anchor=W)
tree.heading('Address', text="Address", anchor=W)
tree.heading('Contact', text="Contact", anchor=W)
tree.column('#0', stretch=NO, minwidth=0, width=0)
tree.column('#1', stretch=NO, minwidth=0, width=0)
tree.column('#2', stretch=NO, minwidth=0, width=80)
tree.column('#3', stretch=NO, minwidth=0, width=120)
tree.column('#4', stretch=NO, minwidth=0, width=90)
tree.column('#5', stretch=NO, minwidth=0, width=80)
tree.column('#6', stretch=NO, minwidth=0, width=120)
tree.column('#7', stretch=NO, minwidth=0, width=120)
```


Scanned with CamScanner

Scanned with CamScanner

```
tree.pack()
tree.bind('<Double-Button-1>', OnSelected)

#INITIALIZATION
if __name__ == '__main__':
    Database()
    root.mainloop()
```

Contact Manager

Scanned with CamScanner

Scanned with CamScanner

Python Lecture Exp

Button:

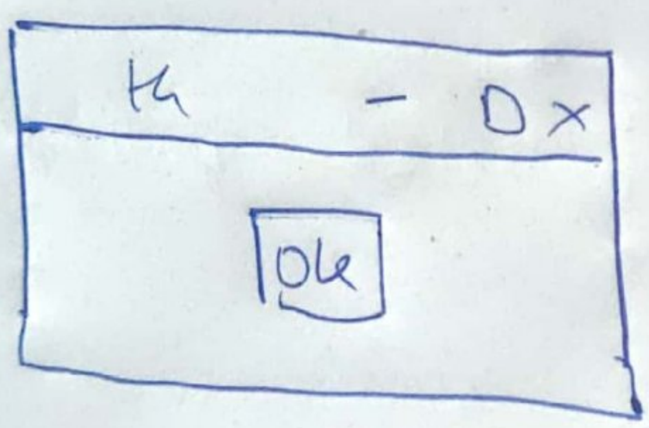
Theory: The button widget is standard tk widget used for various other req.
These function is used to def various task
A button widget is used to interact with user.
button can contain img, text, etc files.

Scanned with CamScanner

* code:

```
from tkinter import *
master = Tk()
def callback():
    print("click")
b= button(master,text="ok", command=c
b.pack
mainloop()
```

output

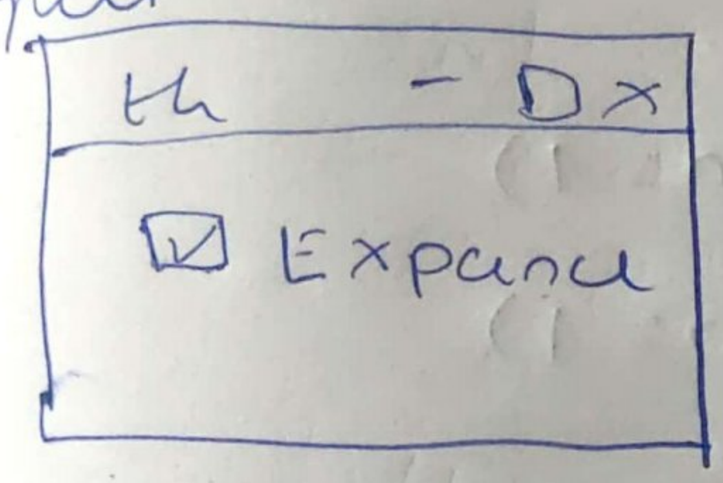


Scanned with CamScanner

```
* from tkinter import *
master = Tk()
var = IntVar()
c = Checkbutton(master, text = "Expand",
                variable = var)
c.pack()

mainloop()
```

output

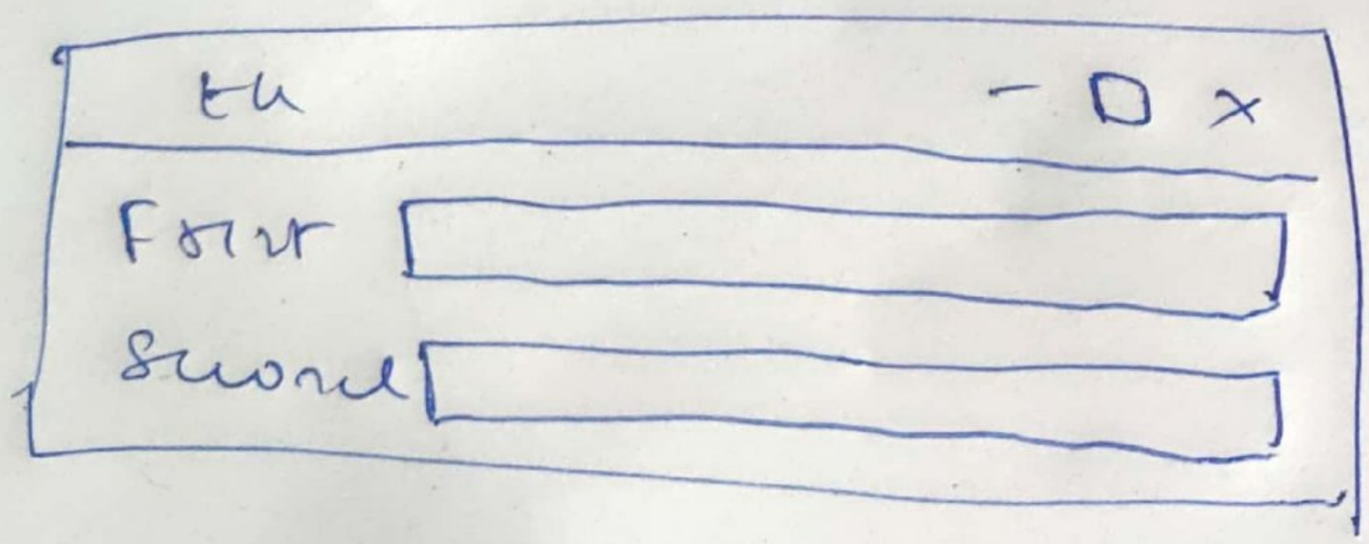


Scanned with CamScanner

* Entery widget

```
from tkinter import *
m = tk()
lable (m, text = "first").grid (row=0, Sh
lable (m, text = "second").grid (row=1, Sh
e1 = Entery (m)
e2 = Entry( m)
e1 = grid (row=0, column=1)
e2 = grid( row=1, column=1)
mainloop ()
```

Output



Scanned with CamScanner