```
fileobj = open("abc.txt","w")
fileobj.write("computer science\n is good")
fileobj.close()

fileobj = open("abc.txt",r)
#read ()
str1 = fileobj.read()
print("The output of read method is", str1)
fileobj.close()

#readline()
fileobj = open("abc.txt",r+)
str2 = fileobj.readline()
print("The output of readline method is",str2)

#readlines()
fileobj = open("abc.txt",r+)
str3 = fileobj.readlines()
print("The output of readlines is ", str 3)
#fileattribute()
a = fileobj.name
print("The name of file is ",a)

# b = fileobj.closed
print("close attribute : ", b)
```

Output:-

>>> The output of read method is computer science is good

>>> The output of readline is computer science

>>> The output of readlines is computer science is good

>>> The name of file is abc.txt

>>> close attribute : TRUE

26/11/19

PRACTICAL - I

25

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

Step1:- Create a file object using the open method and use the write access mode to feed some data in the file.

Step2:- Now open the file in read mode and then use read(), readline(), readlines() and store the output in the variable.

Step3:- Now use the file object for finding the name of the file, the file mode in which it is operated whether it is open or close and finally the output of the softspace attribute.

Step 4:- Now open the contents in wr
mode and then change the
content ie. data of the file.

Step 5:- Open file in read mode displ
the update written content of
the file using print functio

Step 6:- Now open the file in app
mode and edit the data in
file and subsequently use th
read mode to display the
of file using print function

```
#tell
fileobj = open("abc.text","r")
pos = fileobj.tell()
print("tell() :", pos)
>>> ('tell(): ', 0L)

# seek
fileobj = open("abc.text","r")
st = fileobj.seek(0,0)
print("seek(0,0) is ", st)
>>> seek(0,0) is 0
fileobj = open("abc.txt","r")
st1 = fileobj.seek(0,1)
print("seek(0) is ", st1)
>>> seek(1,0) is 4
fileobj = open("abc.txt","r")
st2 = fileobj.seek(2,0)
print("seek(2,0) is", st2)
>>> seek(2,0) is 14

# Finding the length of the words in the string of file.
fileobj = open("abc.txt","r")
stat = fileobj.readlines()
print("output is :" stat)
for line in stat:
    print(len(line))
fileobj.close()
>>> output is CSA Institute
14
```

27

Step7: Open fileobj in read mode, declare a variable and perform file object dot tell method and store the output in variable and then display it using print function.

Step8: Use the seek method with the arguements with opening the file obj in read mode and closing subsequently.

Step9: Open fileobj in read mode also use the readlines method and store it in variable. display the output of stored variable using print function After that use for loop and use the len() method to find length of the words in the file. After this print the output in the loop using print function and at last close the file.

5/12/19

PRACTICAL-2

Objective:- Iterators.

Step1:- create a tuple with elements we need to iterate using the output can be displayed using the for loop.

Step2:- store it in the variable and run the program using for loop

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

Step4:- call the declared function using function call.

Step5:- using for conditional statement specify the range use the type casting with map method. declare the λ and print the same

```
#iter() and next()
my_tuple1 = ("banana", "Orange", "apple")
myiter1 = iter(my_tuple1)
for i in my_tuple1:
    print(i)
```

output:-

```
>>> banana
    orange
    apple
```

```
#for loop:
mytuple1 = ("Kevin", "Stuart", "Bob")
for x in mytuple1:
    print(x)
```

```
>>> Kevin
    Stuart
    Bob
```

```
# Square and cube:-
def sq(x):
    y = x * x
    return y
def cube(x):
    z = x * x * x
    return z

funct = [square, cube]
```

```
for x in range(5):
    value = list(map(lambda x: x(2)), funcs)
    print(value)

>>> [0, 0]
    [1, 1]
    [4, 8]
    [9, 27]
    [16, 64]

# map()
list num = [0, 4, 5, 7, 9, 11]
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"
a = list(map(even, list num))
print(a)
>>> ['odd', 'even', 'odd', 'odd', 'odd', 'odd']

# 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
```

B) Factorial of a number using map function.

Step1:- Define a function which will accept one arguement.

Step2:- use if conditional statement and use the else condition by declaring the factorial eg. expression.

Step3:- Take input from the user and use the type casting method to integer.

Step4:- Create a blank list and append the input value to the list.

Step 5:- use the map method and map the factorial and list.

Step5:- After that use print statement and print the factorial of the number.

```
myobj = odd()
myiter = iter(myobj)
x = int(input("enter number"))
for i in myiter:
    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 number :"))
list1 = [ ]
list1.append(n)
a = map (fact, list1)
print("The factorial is ", a)
```

Output:-

```
>>> Enter number : 5
The factorial is 120
```

Exception Handling in file:-

```
try:
    fo = open("abc.txt", "w")   #R
    fo.write("Python")
except IOError:
    print("Enter appropiate mode")
else:
    print("Operation is successful!")
```

>>> Operation is successful
#R output.
>>> Enter appropiate mode

Value Error:-

```
try:
    x = int(input("Enter a statement: "))
except ValueError:
    print("Arithmetic Error!")
else:
    print("Operation successful")
```

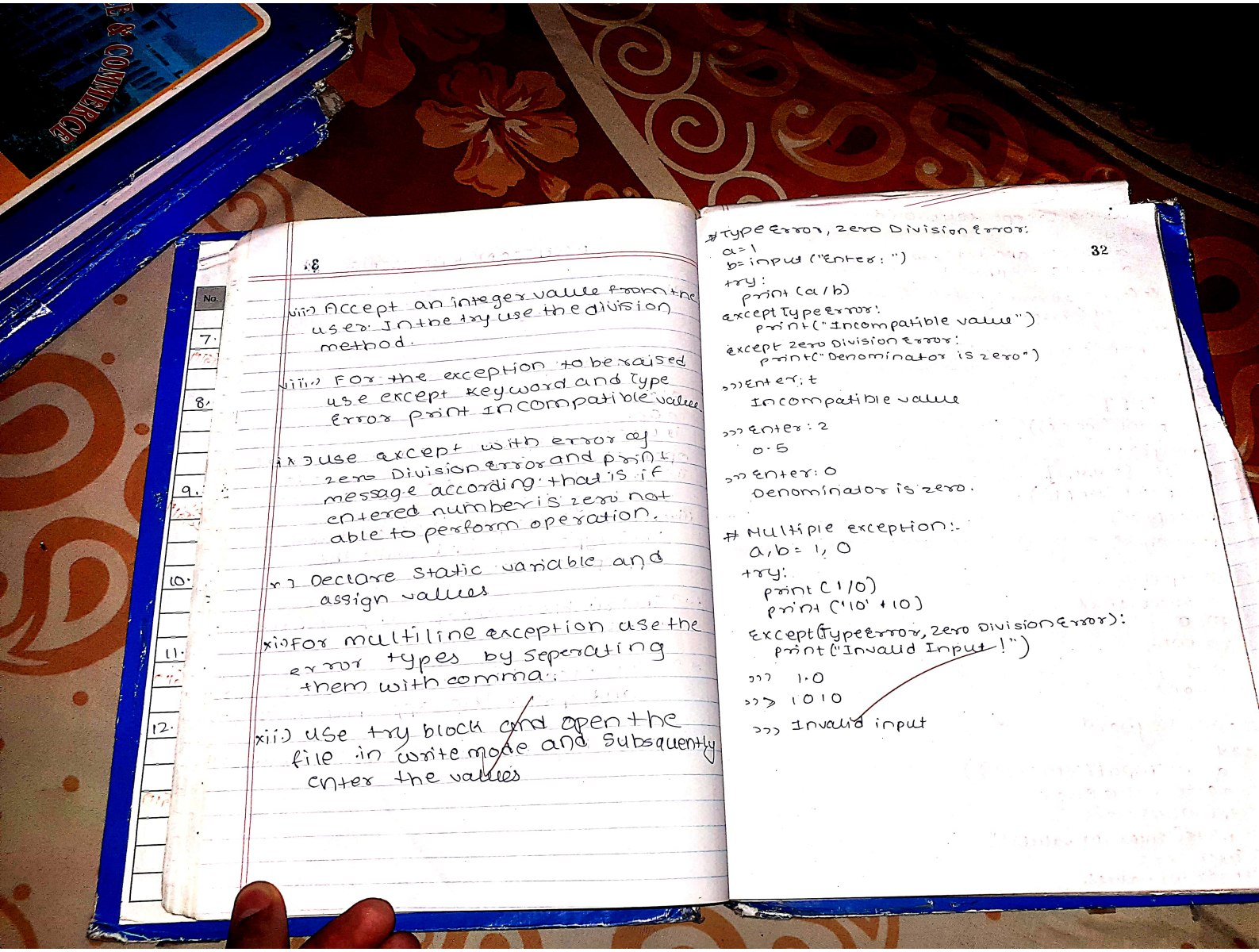
>>> Enter statement: abc
Arithmetic Error!
>>> Enter statement: 123
Operation is successful!

Aim:- To demonstrate the use of exception handling.

~~Exception handling in file:-~~

Algorithms:-

i) Use the try block and inside that open the file in the write mode
ii) After that use the syntax for reading the file.
iii) Use the except block and write the type error and write the appropiate message.
iv) In try block accept input from the user.
v) In except block use value error and print the appropiate message.
vi) else display operation is successful



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

viii) For the exception to be raised use except keyword and type Error print Incompatible values

ix) use except with error of zero Division error and print message according that is if entered number is zero not able to perform operation.

x) Declare static variable and assign values

xi) For multiline exception use the error types by seperating them with comma.

xii) use try block and open the file in write mode and subsequently enter the values

Type error, zero Division Error:

```
a=1
b=input("enter:")
try:
  print(a/b)
except TypeError:
  print("Incompatible value")
except zero DivisionError:
  print("Denominator is zero")
```

>>> Enter: t
Incompatible value

>>> Enter: 2
0.5

>>> Enter: 0
Denominator is zero.

Multiple exception:-

```
a,b = 1, 0
try:
  print(1/0)
  print('10' + 10)
except(TypeError, zero DivisionError):
  print("Invalid Input!")
```

>>> 1.0
>>> 1010
>>> Invalid input

```
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:"))
    raise valueerror
except valueerror:
    print("enter int values")
>>> enter: xy2
    enter int values.
```


xiii) Use IO error and display the message.

xiv) Define a function with empty list and calculate the length.

xv) Define another function & initialize or declare some ~~variable~~ elements in the list and calculate the length and display the same.

xvi) In try block accept input from the user and if user enter character raise an Error ~~and~~ and display appropiate message.

25/11/19

PRACTICAL-4

Regular Expressions

Topic :- Regular Expression

Algorithms:-

Step1:- Import the re module declare pattern and declare sequence use match method with declare agruem if argument match then print the same.

Step2:- Import re module declare pattern with literals and meta characters. Declare string value use the find all with arguement

Step3:- Import re module declare pattern with meta character use the split() and print the output.

```
# match()
import re
pattern = r"FYCS"
sequence = "FYCS in TCSC"
if re.match(pattern, sequence):
    print("Matched")
else:
    print("Not matched")
>>> Matched
```

```
# numericalvalues:-
import re
pattern = r'\d+'
string = 'Hello123, Howdy23, 46Hru'
o = re.findall(pattern, string)
print(o)
>>> ['123', '23', '46']
```

```
# split:-
import re
pattern = r'\d+'
string = 'Hello1', Howdy2'
o = re.split(pattern, string)
>>> ['Hello', 'Howdy']
```

```
# no space:
import re
str = 'abc def ghi'
pattern = r'\s+'
replace = ''
v1 = re.sub(pattern, replace, str)
print(v1)
>>> abcdefghi

# group:
import re
seq = 'Python is OOP'
v = re.search('\A Python', seq)
print(v)
v1 = v.group()
print(v1)
>>> python.

# verifying given set of phone numbers
import re.
l1 = ['9768341229', '9421137891', '8004567891']
for value in l1:
    if re.match(r'[8-9]{}[0-9]
        value or len(value) == 10):
        print("criteria matched")

    else:
        print("Not matched")

Output:
>>> criteria matched
>>> criteria matched
>>> criteria matched
```

Step4: import re module declare string and accordingly declare pattern replace the blank space with no space. use sub with 3 arguements and print string without spaces.

Step5: import re module declare a sequence use search method for finding the subsequently word.

Step6: import re module declare list with members. use conditional statements here we have to use up the condition statements. if criteria matches print cell number matched else print not matched.

Step7: Import re module declare a string use the module with findall () for finding the vowels and declare the same.

Step8:- Import re module declare the host and domain name declare pattern for seperating the host & domain name. Use the findall and print the output.

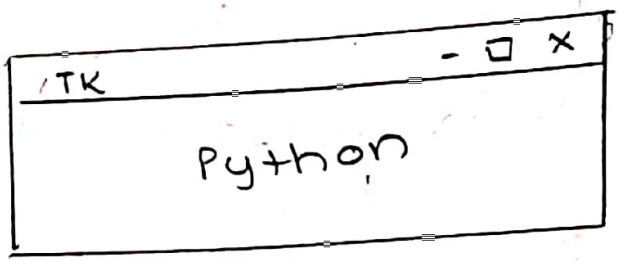
Step9:- Import re module enter a string use pattern to display the only two elements of the particular string use findall declare two variables with initial value as zero. and display the values subsequently

```
# vowels:.
import re
str1 = 'Python support OOP'
o = re.findall (r'\b[aeiouAEIOU]\wt', str1)
print(o)
>>> [object
>>> [OOP]

# host domain
import re
seq = 'abc.tcsc@gmail.com,
       xy.@gmail.com'
pattern = r'[\w\-]+[\w\.-]'
o = re.findall (pattern, seq)
print(o)
['abc.tcss', '@gmail.com', 'xy', '@gmail.com']

#count first two letters:-
import re
s = 'Mr.a, Ms.b, Ms.c, Mr.t'
p = r'[Mrs/Mr/Ms/t]'
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 female is :', f)
print('No of male is:', m)
```

Output:-
>>> ['Mr', 'Mr', 'Ms', 'Mr']
>>> No of female is : 2
>>> No of male is : 2

Creation of parent window

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



#2

```
from tkinter import *
root= TK()
l= label(root, text="Python")
l1= label(root, text="CS", bg="yellow", fg='red'
                font="10")
l1.pack(side=LEFT, padx=20)
l2= label(root, text="CS", bg="yellow", fg='black'
                font="10")
l2.pack(side=LEFT, pady="30")
```

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

Step 5: Now repeat the above step with label which takes the following arguements

i) name of the parent window.
ii) Text attribute.
iii) Background color.
iv) The foreground color and padding attributes

```
# Radio button
from tkinter import *
root = TK()
root.geometry("600x600")
def select():
    selection = "you just selected" + str(var.
    l1 = Label(text=selection", bg="white", fg=
    l1.pack(side=TOP)
var = stringvar()
l1 = Listbox()
l1.insert(1, "List1")
l2.insert(2, "List 2")
l1.pack(anchor=N)
r1 = Radiobutton(root, text="option 1", variab
                 =var, value="Option1", comma
r1.pack(anchor=N)
r2 = Radiobutton(root, text="option2", vari
                 ,value="option2", command=sel
r2.pack(anchor N)
root.mainloop()
```


Aim:- GUI Components

#1:-

Step1:- Import the relevant methods from the tkinter library create an object with the parent window

Step2:- use the parent window object along with the geometry() declaring specific pixel size of parent window.

Step3:- Now define a function which tells the user about the given selection mode from multiple option available.

Step4:- Now define parent window and define the option with the control variable.

Step5:- use the listbox() and insert options on the parent window along with the pack with specifying anchor attribute.

Step6:- use the pack and mainloop method.

#2 - Scrollbar:-

Step 1:- Import relevant methods from tkinter library.

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

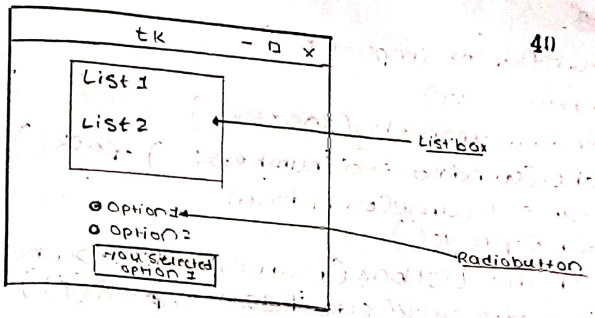
Step 3:- Use the geometry() for laying the window.

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

Step 5:- use the pack() along with the scroll bar object with side and fill attributes

Step 6:- use the mainloop with parent object.

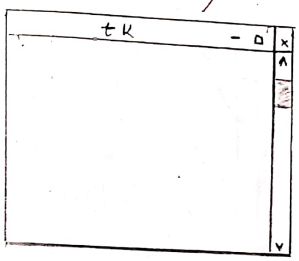
output:-



#2 Scrollbar:

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

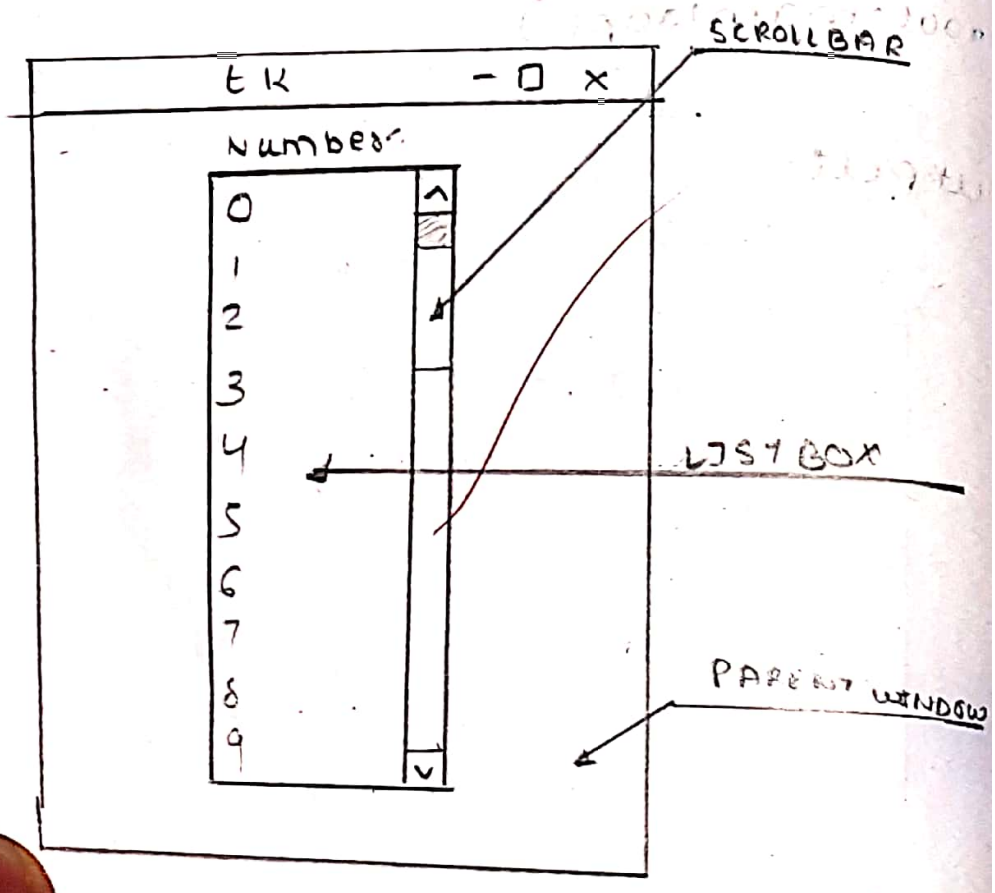
output:



#3

```
from tkinter import *
window = Tk()
window.geometry("500x500")
label(window, text="numbers:").pack()
frame = Frame(window)
frame.pack()
listNodes = Listbox(frame, width=20, height
listNodes.pack(side="RIGHT", fill="Y")

for x in range(100):
    listnode.insert(END, str(x))

window.mainloop()
```

output:



#4:-

```
from tkinter import *
window=Tk()
window.geometry("500x500")
frame=Frame(window)
frame.pack(side="LEFT")
rightframe=Frame(window)
rightframe(side=LEFT)
rightframe.pack(side='RIGHT')
leftframe=frame(window)
leftframe.pack(side="LEFT")
b1=Button(frame, text="select", activebackground=
, fg="black")
b2=Button(frame, text="modify", activebackground="ye
, fg="green")
b3=Button(frame, text="ADD", activebackground="re
fg="yellow")
b4=Button(frame, text="EXIT", activebackground="re
fg="green")
```

Step 8:- Create another button object and place in on to the right frame and label it as add.

Step 9:- Add another button and put it on the top of frame and label it as exit.

Step 10:- use pack and mainloop method subsquently.

Source Code:-

```
from tkinter import *

def btn():
    B1 = Button(text = "Next",
```

in the command attribute.

step5:- Finally use the mainloop method for trivasing from one window to another.

Source Code:-

```
from tkinter import *
root= TK( )
root.geometry ('500x500')
def finish( ):
    quit( )
def w1( ):
    s=tk( )
    s.geometry ("500x500")
    Label(s, text = "Jhisisssecond window", bg="
        fg="black").pack(side = TOP)
def withdraw( ):
    s.withdraw( )
Button(s; text="BACK", relief = RIDGE).pack(side
Button(s, text="Quit", relief= sunken, command=fi
.pack(side= TOP)
s.mainloop( )
def msg( )
    message box.ask yes no ("warning", Are you su
m= msg( )
Label (text = "This is I window).pack(side= TOP)
Button (text = "Click Here", relief= RIDGE).pack (side=
Button(text= "Quit", command= finish, relief= SUNKEN
root.mainloop( )
```

Step 7:- Define a function of back and add the function of command Add some attributes to it.

Step 8:- Define another window and place it onto the second window. Call the finish function to quit the program on tapped.

Step 9:- Define a message box to ask yes or no and give title and appropiate message to it.

Step 10:- Call the message box functi Now create a label to give heading as the first window.

Step 11:- Create a button such that on tapped it will go onto the second window.

Step 12:- Create another button of qui and again call the finish button to perform quit operation. Final call the main loop method along with parent window.

Source code:

```
from tkinter import *
root = TK()
root.config(bg = "grey")

def finish():
    Messagebox.askokcancel("warning", This will
                                      Program
    .quit()

def info():
    list1 = Listbox()
    list1.insert(1, "CO.Name : TATA")
    list1.insert(2, "Partner : None")
    list1.insert(3, "Owner: RATAN TATA")

def about us():
    list2 = Label(text = "About us")
    list2.grid(ipadx = 30)
    list3 = Label(text = "upcoming project")
    list3.grid(ipadx = 24)

p1 = PhotoImage(file = "abc.gif")
f1 = Frame(root, height = 350 width = 250)
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 = 5)
l2 = Label(f2, image = p1, relief = SUNKEN)
l2.grid(padx = 25, pady = 10)
```

Step 7:- Create a frame obj along with the Frame() and place it on the parent window obj - Specify height and width and subsequently use the grid() with row & column attribute.

Step 8:- Similarly create another frame as declared by step 7.

Step 9:- create another object and use subsample method with x and y co-ordinates

Step 10:- Use label widget along with frame obj, relief attribute, image attribute and subsequently use grid()

Step 11:- Now create button object dealing with different sections of frame,

Step 12:- Finally use the main loop method for triggering the events.

02

```
# define function programme
from tkinter import *
def add():
    selection = int(input("Enter the 1st no"))
    selection1 = int(input("Enter little 2nd no"))
    add = selection + selection1
    label.config(text = selection)

def sub():
    selection = 88-4
    label.config(text = selection)

def mult():
    selection = 3*2
    label.config(text = selections)

root = Tk()
R1 = Radiobutton(root, text="add", value=1, command=add)
R1.pack(anchor=W)

R2 = Radiobutton(root, text="88-4", value=2, command=sub)
R2.pack(anchor=W)

R3 = Radiobutton(root, text="3*2", value=3, command=mult)
R3.pack(anchor=W)

label = Label(root)
label.pack()
root.mainloop()
```

5.

Image processing

S1:- Import the library

S2:- create the window use photoimage method the image of given file

S3:- display the image on parent window

S4:- use sample=n, sub sample (3,3) to making the

S5:- create label and use label root, gives image with padx method

S6:- label, and given the size attribute on the side

S7:- create object a= and write frame text "This image and"

S8:- create object where justify = the window left, with padx c10, text = a. with pady side left.

S9:- terminate the loop with root.mainloop method.

Output :-

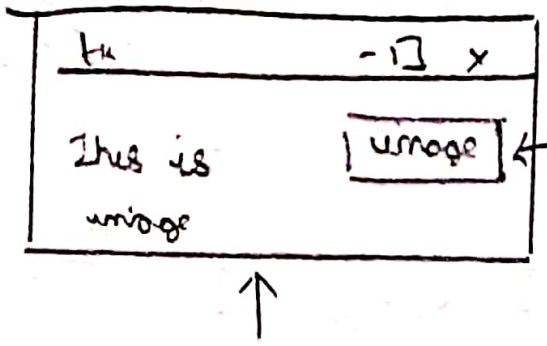


image in gif format

load to browse.

PRACTICAL - 5(F):-

GUI components:- Paned windo

Algorithm:

1) Import the relevant libraries from the tkinter module.

2) Define a function, add and ini the function declare two varia getting entries from the use the tkinter window. add the variables and insert it into t left side of paned window.

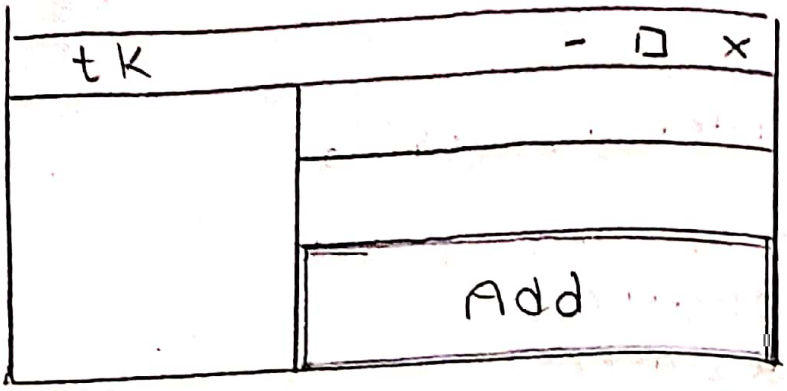
3) Create a paned window w1 and pack it onto the parent window with fill and expand attributes.

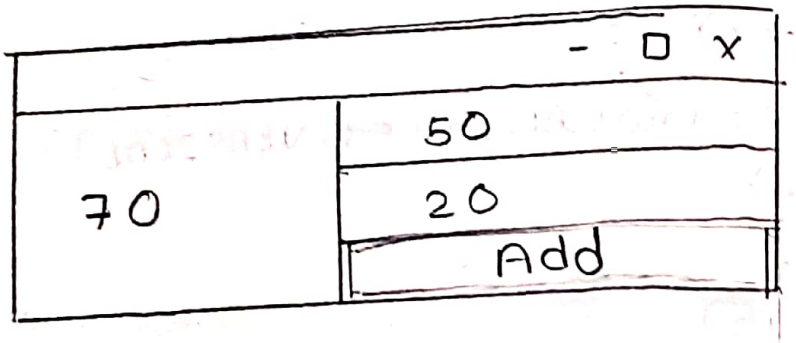
4) Create an object left with Entry widget and add it onto the w1 paned window.

5) Create another paned window an put it onto the right side.

Source Code:.

```
from tkinter import *
def add():
    a= int(e1.get())
    b= int(e2.get())
    leftdata= a+b
    left.insert(1, leftdata)
w1= PanedWindow()
w1.pack(fill= BOTH, expand= 1)
left= Entry(w1, bd= 5)
w1.add(left)
w2= PanedWindow(w1, orient=VERTICAL)
w1.add(w2)
e1= Entry(w2)
e2= Entry(w2)
w2.add(e1)
w2.add(e2)
B= Button(w2, text=" Add", command= add)
w2.add(B)
mainloop()
```


tK
- □ x
Add


- □ x
50
70
20
Add

PRACTICAL-5 (GUI)

Aim:- GUI Components (Canvas, Wi

1.) Creating rectangle:-

Algorithm:-

1.) Import the relevant method from tkinter module.

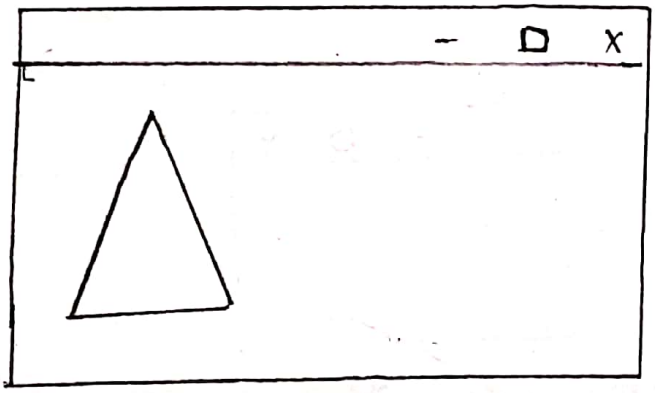
2.) Create an object for parent window.

3.) Create a canvas widget and place it onto the parent wind and add width and height of the canvas widget and pack it.

4.) Create a rectangle by using its co-ordinates and finally use the main loop method.

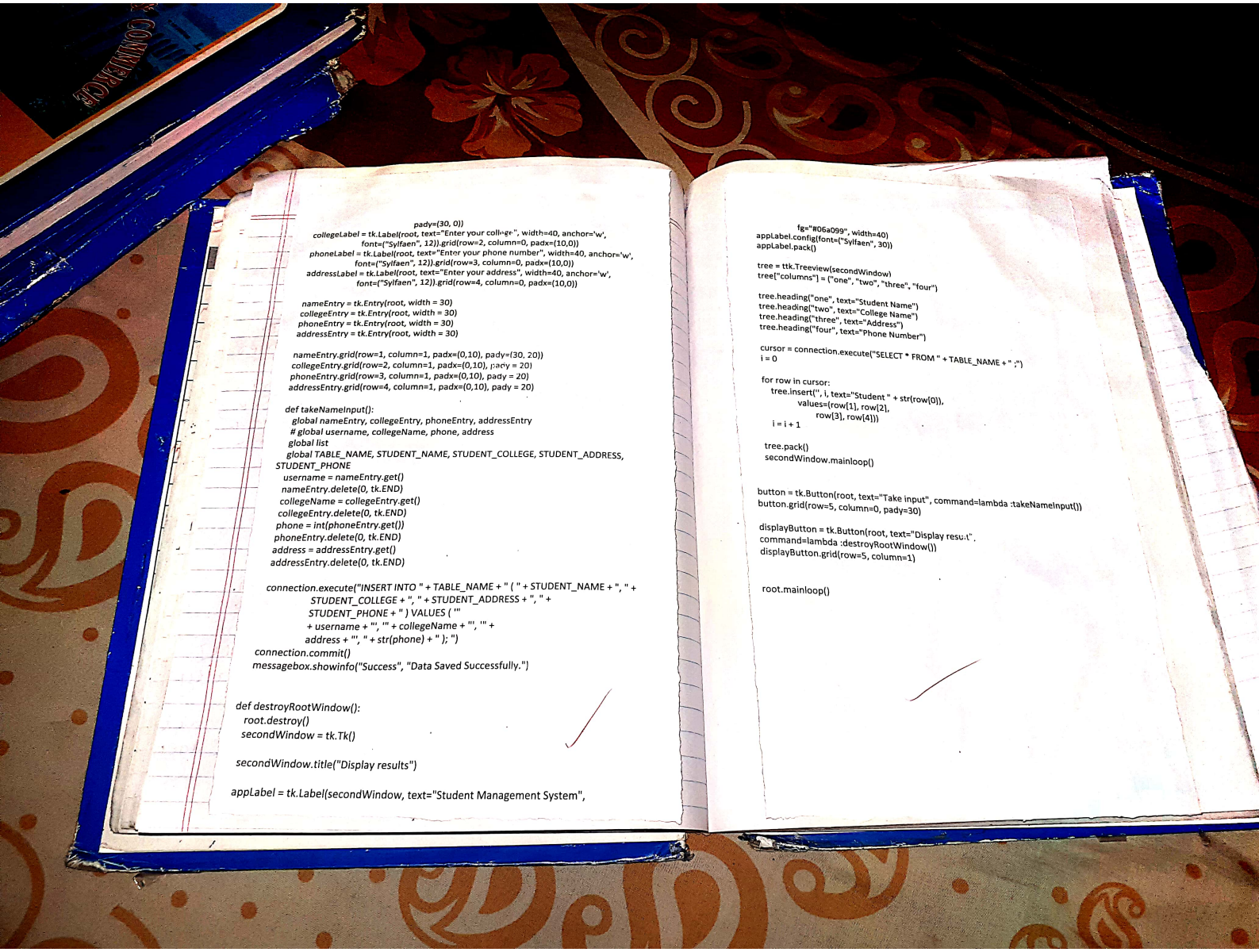
Source Code:

```
from tkinter import *
c=Canvas()
c.pack()
c.create_polygon([20,100,50,0,100,100],fill
mainloop()
```



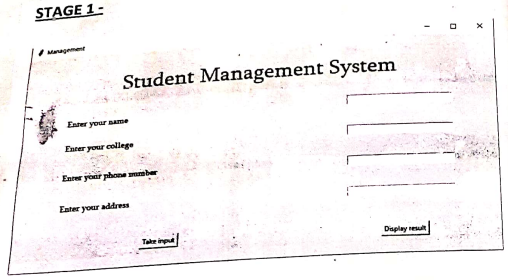
PYTHON PROJECT STUDNET MANAGEMENT SYSTEM
USING TKINTER AND SQLITE3 DATABASE SYSTEM

```
import tkinter as tk
from tkinter import ttk, messagebox
import sqlite3

root = tk.Tk()
root.title("Management")

connection = sqlite3.connect('management.db')

TABLE_NAME = "management_table"
STUDENT_ID = "student_id"
STUDENT_NAME = "student_name"
STUDENT_COLLEGE = "student_college"
STUDENT_ADDRESS = "student_address"
STUDENT_PHONE = "student_phone"

connection.execute(" CREATE TABLE IF NOT EXISTS " + TABLE_NAME + " ( " + STUDENT_ID +
            " INTEGER PRIMARY KEY AUTOINCREMENT, " +
            STUDENT_NAME + " TEXT, " + STUDENT_COLLEGE + " TEXT, " +
            STUDENT_ADDRESS + " TEXT, " + STUDENT_PHONE + " INTEGER);")

appLabel = tk.Label(root, text="Student Management System", fg="#06a099", width=35)
appLabel.config(font=("Sylfaen", 30))
appLabel.grid(row=0, columnspan=2, padx=(10,10), pady=(30, 0))

class Student:
  studentName = ""
  collegeName = ""
  phoneNumber = 0
  address = ""

  def __init__(self, studentName, collegeName, phoneNumber, address):
    self.studentName = studentName
    self.collegeName = collegeName
    self.phoneNumber = phoneNumber
    self.address = address

nameLabel = tk.Label(root, text="Enter your name", width=40, anchor='w',
            font=("Sylfaen", 12)).grid(row=1, column=0, padx=(10,0),
```

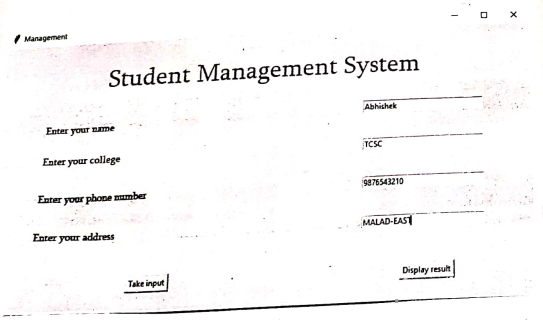


```
                                pady=(30, 0))
        collegeLabel = tk.Label(root, text="Enter your college", width=40, anchor='w',
                                font=("Sylfaen", 12)).grid(row=2, column=0, padx=(10,0))
        phoneLabel = tk.Label(root, text="Enter your phone number", width=40, anchor='w',
                              font=("Sylfaen", 12)).grid(row=3, column=0, padx=(10,0))
        addressLabel = tk.Label(root, text="Enter your address", width=40, anchor='w',
                                font=("Sylfaen", 12)).grid(row=4, column=0, padx=(10,0))

        nameEntry = tk.Entry(root, width = 30)
        collegeEntry = tk.Entry(root, width = 30)
        phoneEntry = tk.Entry(root, width = 30)
        addressEntry = tk.Entry(root, width = 30)

        nameEntry.grid(row=1, column=1, padx=(0,10), pady=(30, 20))
        collegeEntry.grid(row=2, column=1, padx=(0,10), pady = 20)
        phoneEntry.grid(row=3, column=1, padx=(0,10), pady = 20)
        addressEntry.grid(row=4, column=1, padx=(0,10), pady = 20)

        def takeNameInput():
            global nameEntry, collegeEntry, phoneEntry, addressEntry
            # global username, collegeName, phone, address
            global list
            global TABLE_NAME, STUDENT_NAME, STUDENT_COLLEGE, STUDENT_ADDRESS,
        STUDENT_PHONE
            username = nameEntry.get()
            nameEntry.delete(0, tk.END)
            collegeName = collegeEntry.get()
            collegeEntry.delete(0, tk.END)
            phone = int(phoneEntry.get())
            phoneEntry.delete(0, tk.END)
            address = addressEntry.get()
            addressEntry.delete(0, tk.END)

            connection.execute("INSERT INTO " + TABLE_NAME + " ( " + STUDENT_NAME + ", " +
                               STUDENT_COLLEGE + ", " + STUDENT_ADDRESS + ", " +
                               STUDENT_PHONE + " ) VALUES ( '"
                               + username + "', '" + collegeName + "', '" +
                               address + "', " + str(phone) + " ); ")
            connection.commit()
            messagebox.showinfo("Success", "Data Saved Successfully.")


        def destroyRootWindow():
            root.destroy()
            secondWindow = tk.Tk()

        secondWindow.title("Display results")

        appLabel = tk.Label(secondWindow, text="Student Management System",
                            fg="#06a099", width=40)
        appLabel.config(font=("Sylfaen", 30))
        appLabel.pack()

        tree = ttk.Treeview(secondWindow)
        tree["columns"] = ("one", "two", "three", "four")

        tree.heading("one", text="Student Name")
        tree.heading("two", text="College Name")
        tree.heading("three", text="Address")
        tree.heading("four", text="Phone Number")

        cursor = connection.execute("SELECT * FROM " + TABLE_NAME + " ;")
        i = 0

        for row in cursor:
            tree.insert('', i, text="Student " + str(row[0]),
                        values=(row[1], row[2],
                                row[3], row[4]))
            i = i + 1

        tree.pack()
        secondWindow.mainloop()


        button = tk.Button(root, text="Take input", command=lambda :takeNameInput())
        button.grid(row=5, column=0, pady=30)

        displayButton = tk.Button(root, text="Display result",
        command=lambda :destroyRootWindow())
        displayButton.grid(row=5, column=1)


        root.mainloop()
```

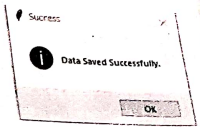
STAGE 1 -



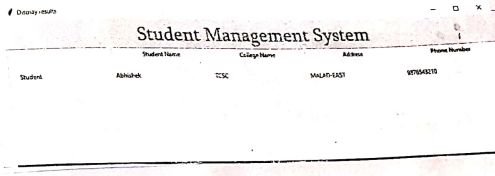
STAGE 2 -



STAGE 3 -



STAGE 4 -



15/9/2022