26/11/19

Practical - 1

Aim : Demonstrate the use of different file accessing modes different attribute read methods

Step1 : Create a file object using open method and use the write access mode followed by writing some contents onto the file and thas closing the file.

Step2 : Now open the file is read mode and thas use read(), readline() and readlines() and store the output is variable and finally display the contents of variable

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

```
fileobj = open("XYZ.txt", "w") # file open (write mode)
fileobj.write("Computer science subjects"+"\n")
fileobj.write("stats\n DS \n Python\n") # file write
fileobj.close() # file close

fileobj = open("XYZ.txt","r") # read mode
# read()
str1 = fileobj.read()
print("The output of read method: ", str1)
fileobj.close()
>>>('The output of read method: ', 'Computer science subjects\n
        stats\n DS\n Python\n')
# readline()
fileobj = open("XYZ.txt","r")
str2 = fileobj.readline()
print("The output of readline method: ",str2)
fileobj.close()
>>>('The output of readline method: ', 'computer science
        subjects\n')
# readlines()
fileobj = open("XYZ.txt","r")
str3 = fileobj.readlines()
print("The output of readlines method: ", str3)
fileobj.close()
>>>('The output of readlines method: ', ['Computer science
    subjects\n', 'stats\n', 'DS\n', 'Python\n']
# file attributes
a = fileobj.name
print("name of file (name attribute): ", a)
>>>('name of file (name attribute)', 'XYZ.txt')

b = fileobj.closed
print("(close) attribute: ", b)
>>>('(close) attribute: ', True)
```

```
file = open("xyz.txt","r")
n = file.readline()
print(n)
for line in n:
    print(len(line))
```

>>> computer science subject

```
c = file.mode
print(c)
```

>>> r

```
file = open("xyz.txt","w+")
file.write("lava")
file.close()
file = open("xyz.txt","r")
a = file.read()
print(a)
```

>>> lava

```
file = open("xyz.txt","a")
file.write("goku")
file.close()
file = open("xyz.txt","r")
b = file.read()
print(b)
file.close()
```

>>> lavagoku

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

Step5 : open fileobj in read mode display the update written contents and close open again in 'r+' mode with parameter passed and display the output subsequently

Step6: Now open fileobj in append mode open write method write contents close the fileobj again open the fileobj in read mode and display the appending output

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

Step 8: Use the seek method with the arguments with opening the file obj is read mode and closing subsequently.

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

```
file = open("XYZ.txt","r")
pos = file.tell()
print("tell(): ",pos)
file.close()
```

```
>>> tell(): 0
```

```
file = open("XYZ.txt","r")
sk = file.seek(0,0)
print(sk)
file.close()
```

```
>>> 0
```

```
file = open("XYZ.txt","r
sk1 = file.seek(0,1)
print(sk1)
file.close()
```

```
>>> 0
```

3 in 1 ?

```
with open("XYZ.txt","r") as g:
    s=1
    c=g.read(s)
    while len(c)>0:
        print(c,end='$')
        c=g.read(s)
```

```
>>> I$a$v$a$g$o$k$u$
```

3 in 1.

```
file = open("XYZ.txt","r")
sk2 = file.seek(0,2)
print(sk2)
file.close()
```

```
>>> 8
```

iter() and next()

```
mytuple = ("goku", "Vegeta", "gohan")
myiter1 = iter(mytuple)
print(next(myiter1))
myiter2 = iter(mytuple))
print(next(myiter1))
myiter3 = iter(mytuple)
print(next(myiter1))
```

```
>>> goku
    Vegeta
    gohan
```

for loop

```
mytuple1 = ("Gaurav", "Daksh", "Aditya")
for x in mytuple1:
    print(x)
```

```
>> Gaurav
   Daksh
   Aditya
```

square and cube

```
def square(x):
    y = x*x
    return y
def cube(x):
    z = x*x*x*x
    return z
functions = [square, cube]
```

Practical-2

Objective : Iterators

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

Step 2 : The similar output can be obtained by using for conditional statement. An iterable variable is to be declared in for loop ~~wh~~ which will iterate

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

Step 4 : Call the declared function using function call

Step5: Using for conditional statement specifying the range use the list typecasting with map method declare a 'lambda' i.e anonymus function and print the same.

Step6: Declare a list num variable and declare some elements then use the map method with help of lambda function give two argument display the output.

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

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

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

```
for r in range(5):
    value=list(map lambda x: x(r), funct()))
    print(value)
>>> [0,0]
    [1,1]
    [4,8]
    [9,27]
    [16,64]
# map()
listnum=[0,4,5,7,9,11,13,15,20,19,25]
listnum=list(map(lambda x: x%.5, listnum))
print(listnum)
def even(x):
    if (x%.2==0):
        return "Even"
    else:
        return "ODD"
list(map(even, listnum))
>>> [0,4,5,7,9,11,13,15,20,19,25]
>>> [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
```

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

>>> Enter a number: 15

1
3
5
7
9
11
13

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

Output:

>>> Enter a number: 7
('The factorial is: ', [5040])

Step 10 : Define an object of a class.

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

Practical - 3

Topic : Exceptions

Step 1 : In the try block open an file in the open mode with write mode. Write some contents in file.

Step 2 : In except (IOError) block use the error in IOError use the appropriate message to display.

Step 3 : Else display the operation is successful

Step 4 : In try block accept an input from the user.

Step 5 : In except block use valueerror and print the some message.

Step 6 : Else display the operation is successful!

IO Error :

```
try: fo = open ("python.txt", "w")
     fo.write ("python is an intendent 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 Value Error:
     print ("Arithmetic Error!")
else:
     print ("operation is successful!")
```

>>> Enter a statement : abc
Arithmetic Error!

>>> Enter a statement : 123
operation is successful!

Type error, zero division error

```
a = t
b = input("Enter: ")
try:
    print(a/b)
except TypeError:
    print("Incompatible Value")
except ZeroDivisionError:
    print("Denominator is zero so operation cannot be performed")
```

>>> Enter: t
Incompatible Value
>>> Enter: 2
0.5
>>> Enter: 0
Denominator is zero so operation cannot be performed.

Multiple exception

```
a,b = 1,0
try:
    print(1|0)      -> 1
    print('10' + 10)  -> '10'
except (TypeError, ZeroDivisionError):
    print("Invalid Input!")
```

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

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

Step 8: For the exception to be raised use the except keyword (and) i e TypeError print Incompatible values

Step 9: Use except with error of zero division Error and print the message according that is if entered number is zero not able to perform operation!

Step 10: Declare static variable and values.

Step 11: For Multiline exception use the error types by separating them with a ~~comma~~ comma.

Step12: Use try block open a file in write mode and subsequently enter value in the file.

Step13: Use the IOError and display appropriate message.

Step14: Define a function with empty list and calculate the length of the list

Step15: Define another function & initialize or declare some elements in list and calculate the length of the same and display the same.

step16: In try block accept input from the user and if the user enters Character values raise an Error that is saying up Enter integer values.

Using except Keyword

```
try:
    a = open("python.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 Values!

4/12

```
# match()
import re
pattern = r"FYCS"
sequence = "FYCS represents Computer Science Stream"
if re.match (pattern, sequence):
    print("Matched pattern found!")
else:
    print ("Not FOUND!")
>>> matched pattern found!
```

~~# numerical pattern found!~~

```
# numerical values (seperation)
import re
pattern = r'\d+'
string = 'hello123, howdy789, 45howru'
output = re.findall(pattern, string)
print(output)
>>> ['123', '789', '45']
# split()
import re
pattern = r'\d+'
string = 'hello123, howdy789, 45howru'
output = re.split(pattern, string)
print(output)
>>> ['hello', 'howdy', ', ', 'howru']
```

Practical-4

Topic : Regular expression.

Step 1: Import re module declare pattern and declare sequence use match method with declare arguments if arguments matched then print the same otherwise print pattern NOT FOUND!

Step 2: Import re module declare pattern with literal and meta character. Declare string value. Use the findall() with arguments and print the same

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

Step4 : Import re module declare string and accordingly declare pattern replace the blank space with no_space. Use sub() with 3 arguments and print the string without spaces.

Step5 : 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 will show up the matched string.

Step 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 and rest number are in range of 0 to 9 and check whether the entered numbers are equal to 10. if criteria matches print all 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 a programming language'
v = re.search('\Apython', sequence)
print(v)
v1 = v.group()
print(v1)
>>> <_sre.SRE_Match object at 0X0281DF00>
    python

# Verifying the given set of phone. numbers
import re
list1 = ['8004658719', '9146537120', '7864523891',
         '9876542310']
for value in list1:
    if re.match(r'[8-9]{1}[0-9]{9}',
    value or len(value)==10):
        print("Criteria matched for cell number!")
    else:
        print("Criteria failed!")
```

>>> Criteria matched for cell number
Criteria matched for cell number
&
Criteria failed!
Criteria matched for cell number

Vowels

```
import re
str1 = 'plant is life overall'
output = re.findall (r'\b[aeiouAEIOU]\w', str1)
print(output)
```

>>> Host & domain

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

>>> ['abc.tcsc', 'edu.com', 'xyz', 'gmail.com']

counting of first 2 letters:

```
import re
s = 'mr.a, ms.b, ms.c, mr.t'
p = r'[ms|mr]+'
o = re.findall(p, s)
print(o)
n = 0
f = 0
for v in o:
```

Step 7 : Import re module declare a string use the module with findall() for finding the vowels in the string and declare the same.

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

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

```
if (v == 'ms'):
    f = f+1
else:
    m = m+1
print ("No. of males is : ", m)
print (" No. of females is : ", f)
```

```
>>> ['msi', 'ms', 'ms', 'msi']
('No. of males is :' 2)
('No. of females is : ', 2)
```

Result
Dr. 2.11

Practical - 5

Topic: Gui components

Step 1: Use the tkinter libraries for importing the features of the text widgets.

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

Step 3: Create a variable using the widget label and use the text method.

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

#2:

Step1: Use the tkinter library for importing the features of the text widgets.

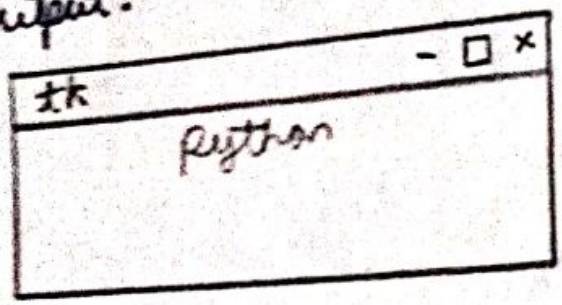
Step2: Create a variable from the text method and position it on the parent window.

Creation of parent window:

1.

```
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 = "gray", fg = "black", font = "10")
l1.pack(side = LEFT, padx = 20)
l2 = Label(root, text = "CS", bg = "light blue", fg = "black", font = "10")
l2.pack(side = LEFT, pady = "30")
l3 = Label(root, text = "CS!", bg = "yellow",
           fg = "black", font = "10")
l3.pack(side = TOP, ipadx = 40)
```

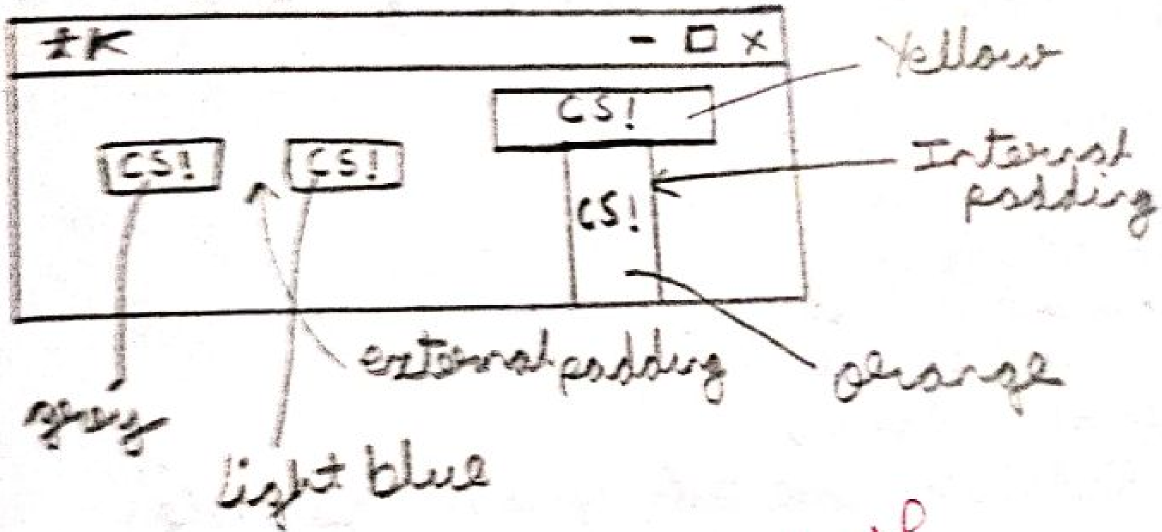
Step 3: Use the pack() along with the object created from the text() and use the parameter.
1) side = LEFT, padx = 20
2) side = LEFT, pady = 30
3) side = TOP, ipadx = 40
4) side = TOP, ipady = 50

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

Step 5: Now repeat above steps with the label() which takes the following arguments
1) Name of the parent window.
2) Text attribute which defines the string.
3) The background color (bg)
4) The foreground (fg) and then use the pack() with a relevant padding attribute.

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

output:



latt
2/11

Radio button

```
from Tkinter import *
root = Tk()
root.geometry("500x500")
def Select():
    Selection = "You just selected" + str(var.get())
    t1 = Label(Text = Selection, bg = "white",
        fg = "green")
    t1.pack(side = TOP)
var = StringVar()
l1 = listbox()
l1.insert(1, "List1")
l2.insert(2, "List2")
l1.pack(anchor = N)
r1 = RadioButton(root, text="option 1", variable
    = var, value="option1", command = Select)
r1.pack(anchor=N)
r2 = RadiusButton(root, text="option 2", variable = var,
    value="option 2", command = select)
r2 = pack(anchor = N)
root.mainloop()
```

Practical 5-B

Aim: GUI Components

#1

step 1: Import the relevant methods from the tkinter library create an object within the parent window

step 2: Use the parent window object along with the geometry() declaring specific pixel size of the parent window.

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

step 4: Now define the parentwindow and define the option with control variable

step 5: Use the listbox() and insert options on the parent window along with the pack() with specifying anchor attribute

step 6: Create an object from radio button which will take following arguments & parentwindow object, text variable which will take the values option no 1, 2, 3.... variable arguments, corresponding values &

Step 1: Bugger the function declared

Step 2: Now call the pack() for radiobject so created the specify the argument using anchor attribute

Step 3: Finally make use of the mainloop() along with parent object

=2

Step 1: Import relevant methods from the tkinter library

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

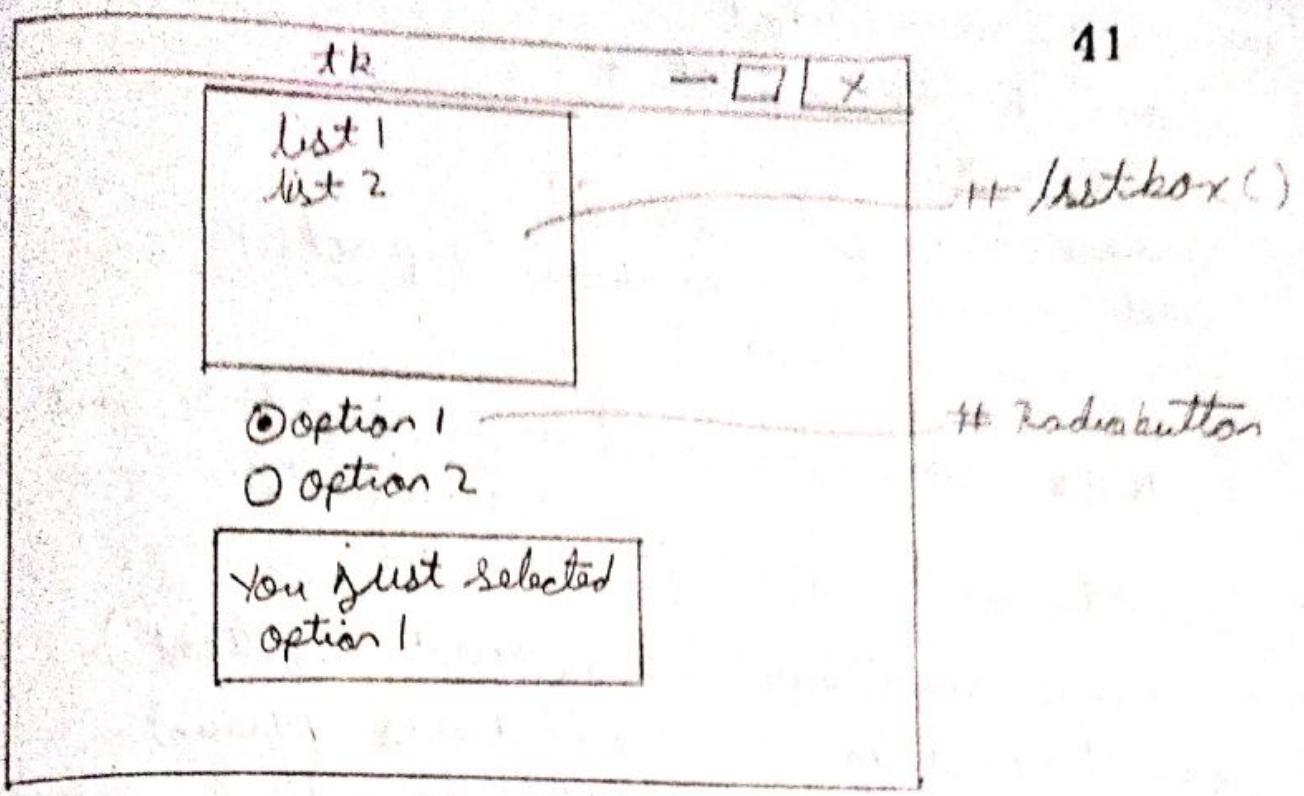
Step 3: Use the geometry() for layout of the widget

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

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

Step 6: Use the mainloop with the parent object

output:

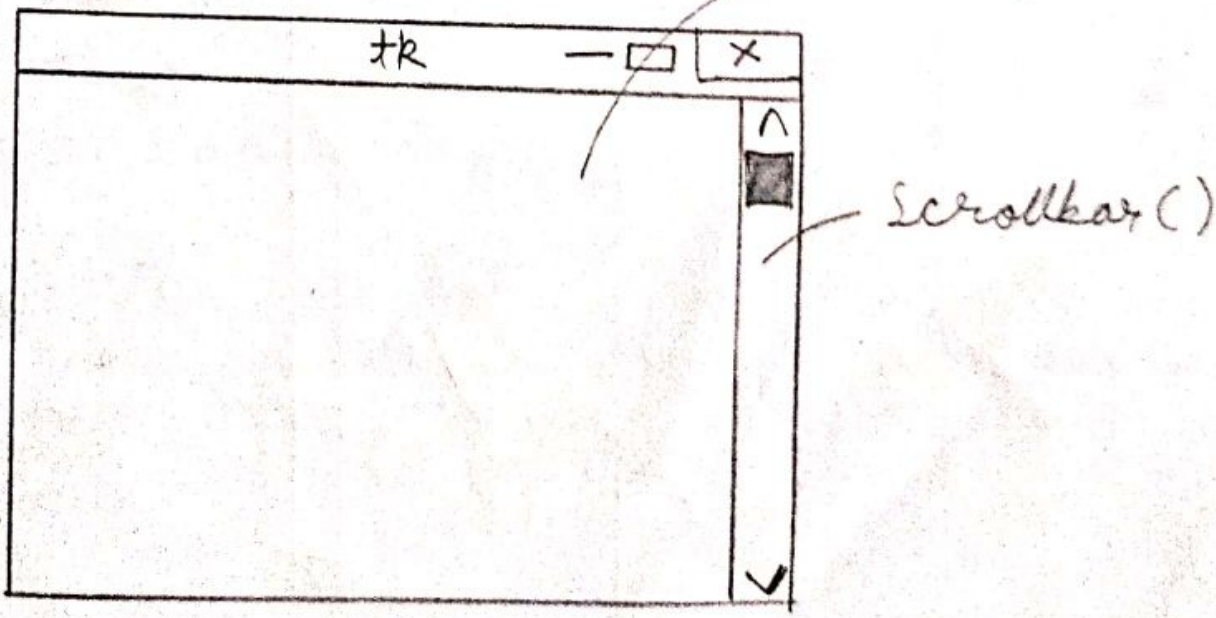


#2:

scrollbar()

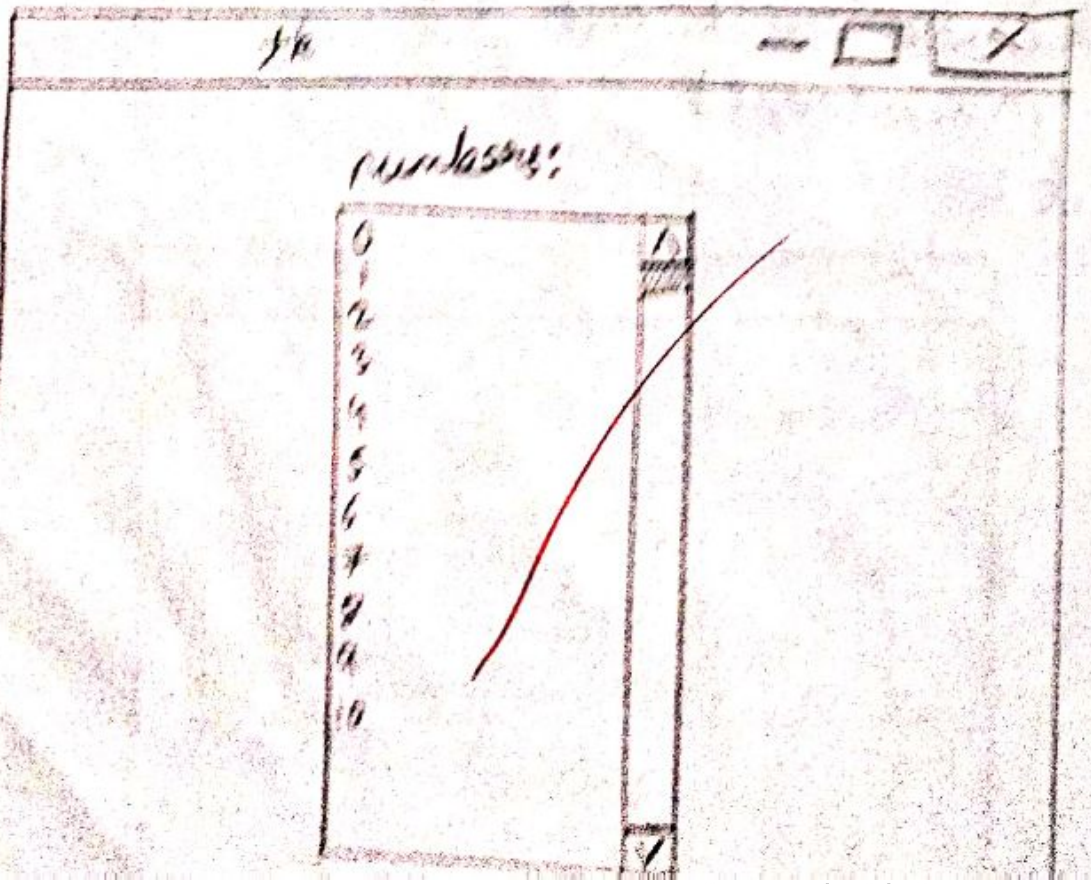
```
from tkinter import *
root = TK()
root.geometry("500x500")
s = scrollbar()
s.pack(side="right", fill="y")
root.mainloop()
```

output:



```
#11 21
#A= Using frame widget
from tkinter import *
window = Tk()
window.geometry("600x500")
label (window, text = "Welcome"). pack()
frame = Frame (window)
frame.pack()
listNodes = Listbox (frame, width = 20, height = 20, font =
                     "Times New Roman", 10)
listNodes.pack(side = "left", fill = "y")
scrollbar = Scrollbar( frame, orient = "vertical")
scrollbar.config (command = listNodes.yview)
scrollbar.pack( side = "right", fill = "y")
for x in range (100):
    listNodes.insert (END, str(x))
window.mainloop ()
```

Output:



#3

Step 1: Import the relevant libraries from the tkinter method

Step 2: Create an corresponding object of the parent window

Step 3: Use the geometry manager with pixel size (680 x 500) or any other suitable pixel value

Step 4: Use the label widget along with the parent object created and subsequently use the pack method

Step 5: Use the frame widget along with the parent object created and use the pack method

Step 6: Use the listbox method along with the attributes like width height font. Do a create a listbox methods object use pack() for the same.

Step 7: Use the scrollbar() with an object use the attribute of vertical then configure the same with object created from the scrollbar() and use pack()

Step 8: Trigger the events using mainloop

#4:

Step 1: Import relevant methods from tkinter library

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

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

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

Step 5: Similarly, define the RIGHT frame and subsequently define the button object placed onto the given frame with the attribute as text, activebackground and foreground.

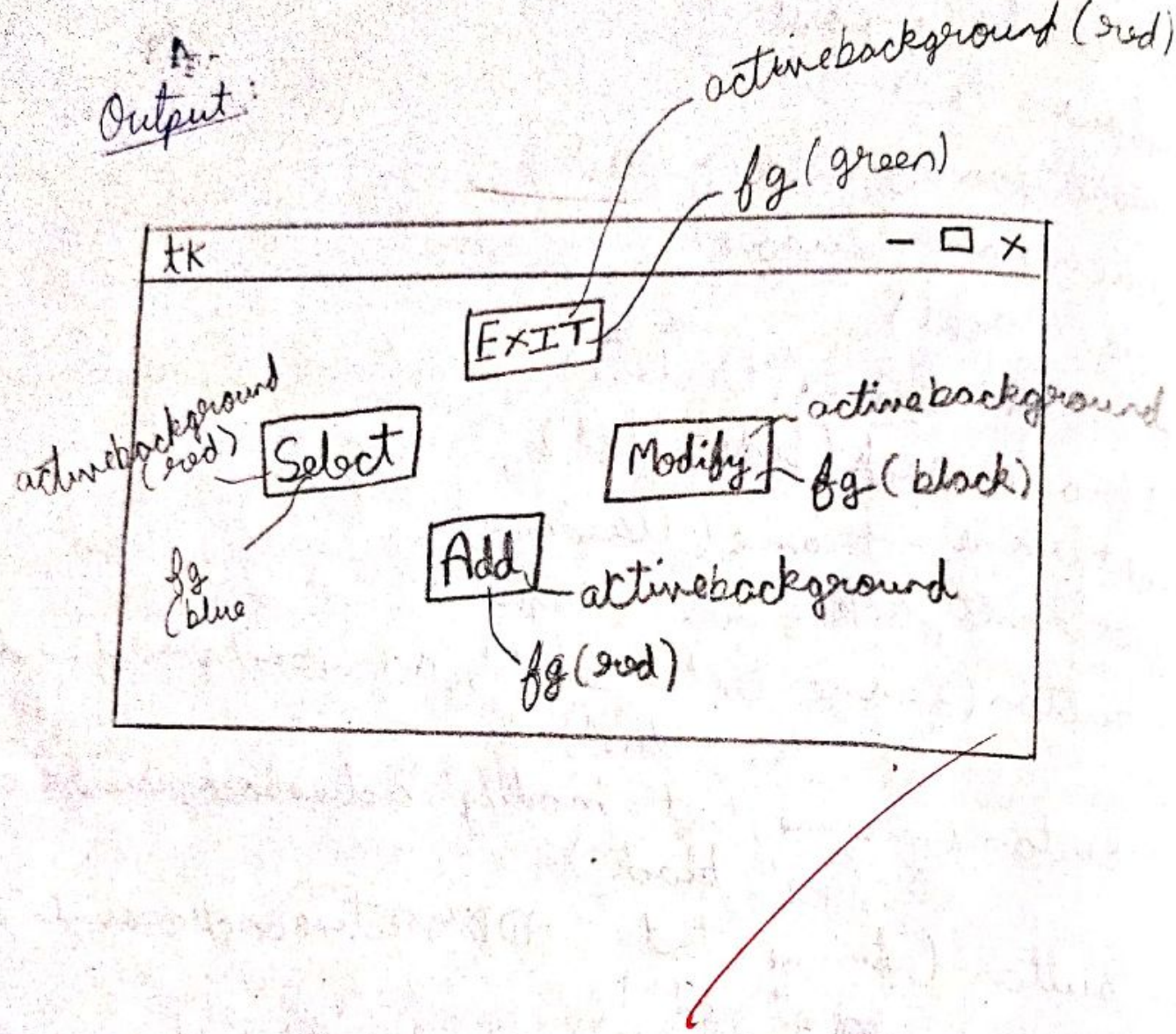
Step 6: Now use the pack() along with the side attribute.

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

#11

```
from tkinter import *
window = Tk()
window.geometry("650x500")
frame = Frame(Window)
frame.pack()
leftframe = Frame(Window)
leftframe.pack(side="left")
rightframe = Frame(Window)
rightframe.pack(side="right")
b1 = button(frame, text="Select", activebackground="red",
            fg="blue")
b2 = Button(frame, text="modify", activebackground="blue",
            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="button", pady=20)
b4.pack(side="top")
```

Output :



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

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

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

Practical - 5C

Aim: GUI components

Step 1: Import the relevant methods from tkinter library

Step 2: Import the messagebox.

Step 3: Define a Parent window object along with the Parent window.

Step 4: Define a function which will use the message box with showinfo method along with info window attribute.

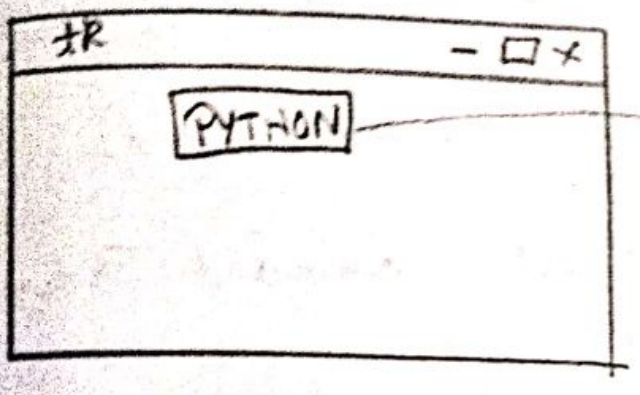
Step 5: Declare Declare a button with parent window object along with the command attribute.

Step 6: Place the button widget onto the parent window and finally call mainloop() for triggering of the events called above

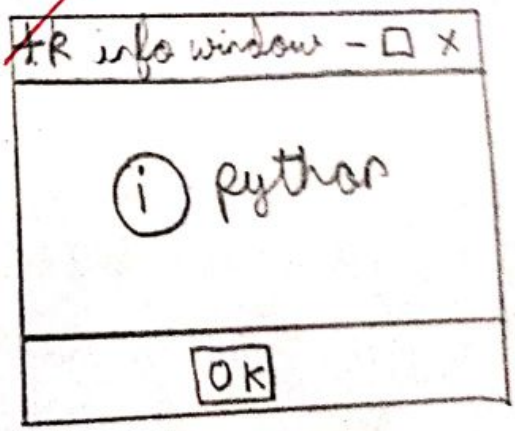
message box

```
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:



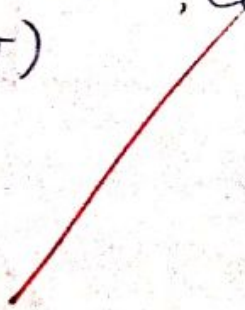
Click Here this window will pop up



```
## Multiple windows
## 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(300,300)
    L = label(top, text="SAN FRANCISCO
In Places of Intrest: \n Golden gate Bridge \n Lombard
street \n Chinatown \n Coit tower")
    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()
```



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

Step 2: Use parentwindow object along with minsize function for window size

Step 3: Define a function main, declare parent window object and use config(), title(), minsize(), label() as well as button as use pack() & mainloop() simultaneously.

Step 4: Similarly define the function second as use the attribute accordingly.

Step 5: Declare ~~another~~ button along with parent object and declare button with attributes like FLAT, RIDGE, GROOVE, RAISED, SUNKEN along with the relief widget.

Step 6: Finally called the mainloop() for event driven programming

```
def second():
    top2 = TK()
    top2.config(bg="orange")
    top2.title("About us!")
    top2.minsize(300,300)
    L = Label(top2, text="Created by: Gaurav Singh \n
    For more details contact to our official account")
    L.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("300x300")
    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="Sunkenbutton", relief=SUNKEN)
    b4.pack()
```

```
b5=Button(top3, text="ridge button",
 relief=RIDGE)
top3.mainloop()
def terminate():
    quit()
b5=Button(root, text="TOUR DETAILS", command=
b5.pack()
b6=Button(root, text="BUTTON DETAILS",
        command=button
b6.pack()
root.mainloop()
```

Practical 5-D

Aim: GUI components

Step 1: Create a parent window. Use methodes: title, config, label, minsize, geometry

Step 2: Create a left frame object using frame method and place it on the parent window, bg, width, height, grid, sticky, padding.

Step 3: Similarly create the right frame with the row = 0, column = 1

Step 4: Create the label object and place it onto the leftframe with text, relief, row = 0, column = 0

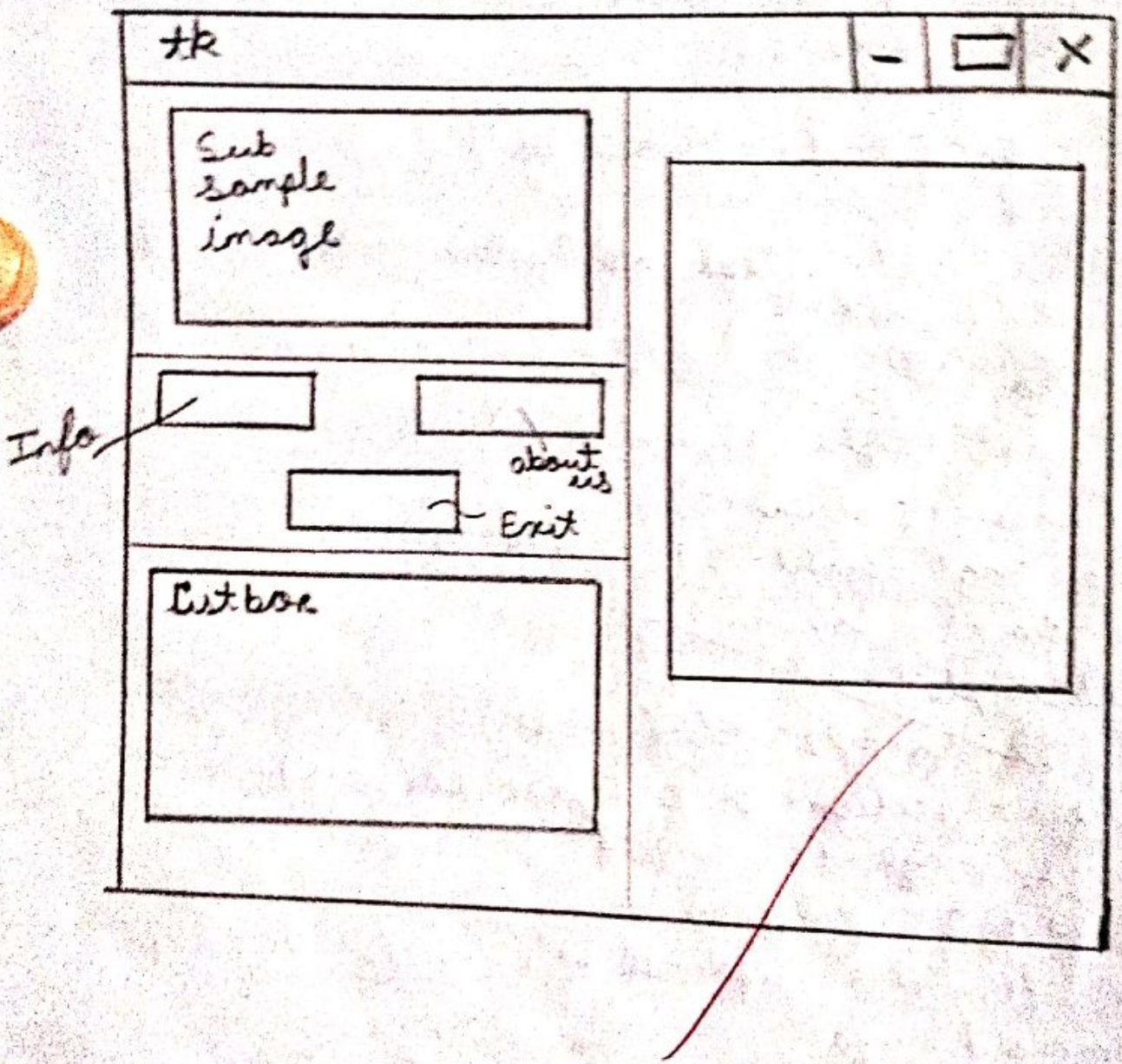
Step 5: Create label corresponding to the right frame with row = 0, column = 1

Step 6: Now create an image from the photoimage and use the file attribute

Step 7: Create another image object with the subsample method from the photoimage

```
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(row1, column=0)
```

```
b2 = Button(f1, text = "About us", relief = SUNKEN,
        command = aboutus)
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()
```



Step 8: Now place the image onto the leftframe with the objectimage 2 and the rightframe will carry the image object with the corresponding object

Step 9: Create a toolbox from the frame method and put it onto the leftframe with the height and width specified with row = 2, column = 0.

Step 10: Define a function which will display a message onto the shell environment.

Step 11: Use the label method and place it corresponding to the toolbox object with text and relief attribute specified and with corresponding row and the column attribute.

Step 12: Now use the button method and place it onto the toolbox object with text and command attribute specified. Use the grid method with the corresponding attributes and call the mainloop method.

Practical - 5

Aim: Demonstrate the use of Paned Window and spinbox and canvas

Paned Window

Step I: Create an object from Paned Window and use the pack() with the attribute fill and expand.

Step II: Create an object from the table 0 and put it onto the Paned Window with the text attribute and use the add() to enked the new object.

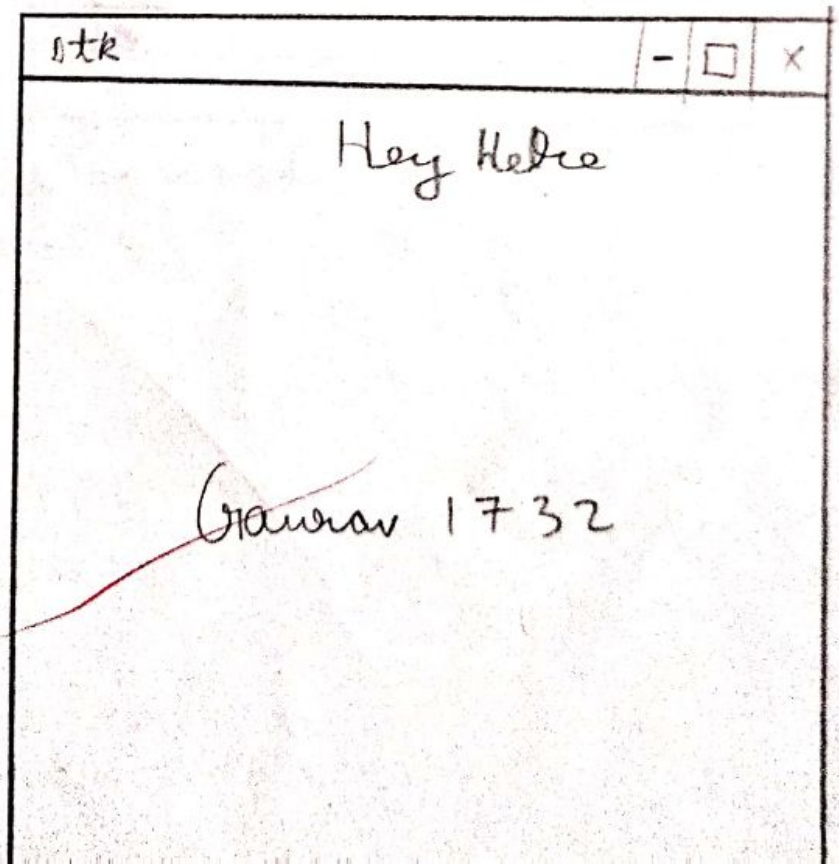
Step III: Similarly, create a second Paned Window object and add it on to the first Paned Window with orientation specified.

Step IV: Now, create another label object and place it onto the second Paned window object and add to onto the second Paned Window created so

Step V: Now use the mainloop() to trigger the events.

```
# Pared Windows ()
from tkinter import *
root = Tk()
t = label
P = Pared Window (root, orient = VERTICAL)
P.pack (fill = BOTH, expand = 10)
L1 = Label (P, text = "Hey Here")
P.add (L1)
P1 = Pared Window (P, Orient : HORIZONTAL pack fill =
                                    BOTH, expand)
P.add (P1)
L2 = Label (P1, text, " ~~Ubuntu~~ Gaurav 1732")
P1.add (L2)
mainloop ()
```

Output

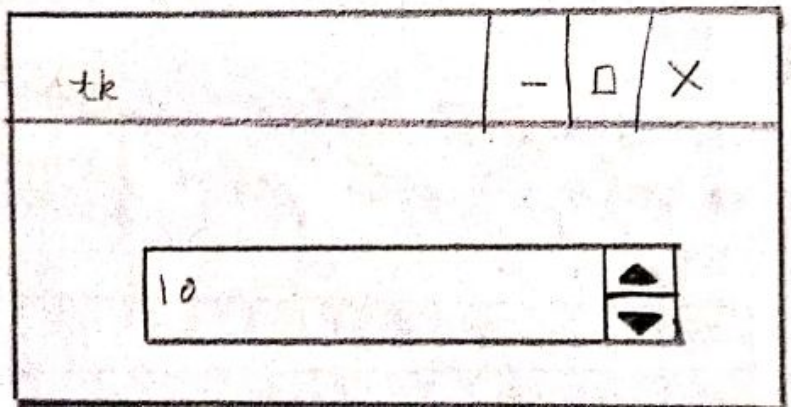


2.

SpinBox

```
from tkinter import *
master = TK()
s = spinbox(master, from=0, to=10)
s.pack()
mainloop()
```

Output :



Spin Box

Step I :- Create an object from the Tk() and subsequently create an object from the Spinbox()

Step II :- Make the object so created onto the parent window and trigger the corresponding events

Step III :- Use the pack() to provide the direction using anchor attribute.

Step IV :- Use the mainloop() to terminate and trigger the event.

Canvas:

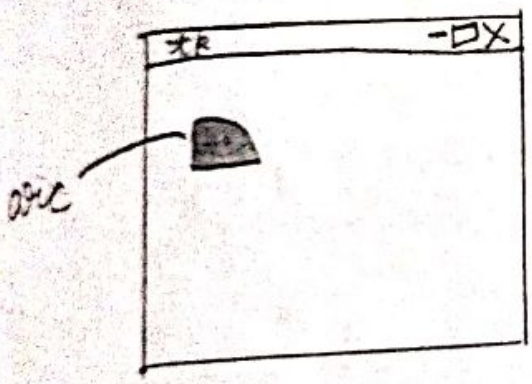
Step 1: Import relevant methods from tkinter library declare a parent window object

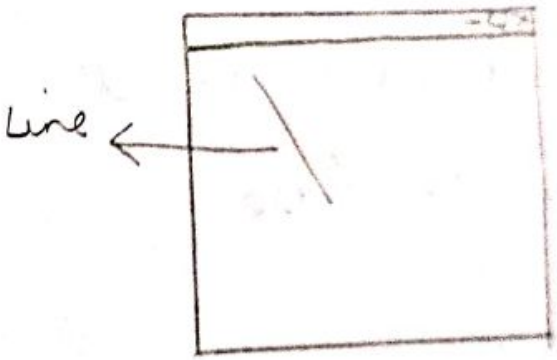
Step 2: Create a canvas object along with the canvas method with attributes such as height, width, background colour.

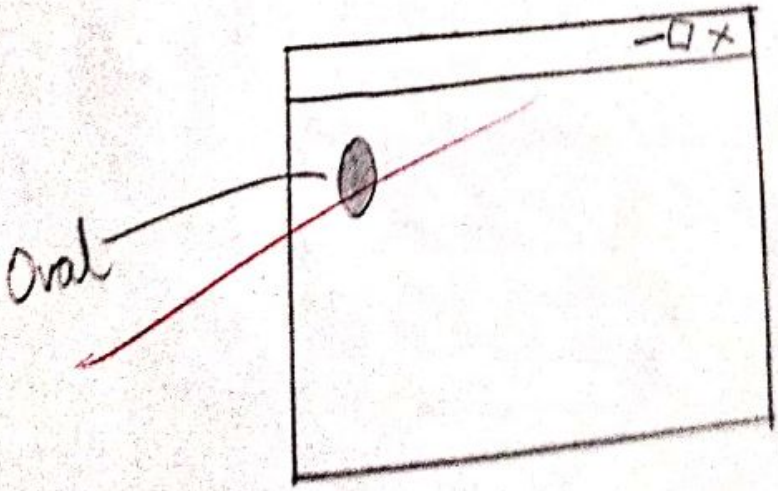
Step 3: Use create_arc() along with canvas object declared along with start and extend coordinates.

Step 4: Similarly for oval and line. Use the pack method and finally call the mainloop method.

```
from tkinter import *
canvas_obj = Tk()
c = canvas (root, height = 100, width = 200, bg = "red")
carc = c.create_arc(10,20,30,40, start = 10, extend = 50,
                    fill = "black")
coval = c.create_oval(20,31,45,50, fill = "orange")
cline = c.create_line(10,15,30,20, fill = "blue")
c.pack()
canvas_obj.mainloop()
```







Practical VI

Aim :- Database Connectively

Algorithm

Step I: Import the dbm library use the open for creating the database by specifying the name of the database along with the corresponding flag

Step II: Use the object so created for accessing the given website and corresponding regular name for the website

Step III: Check whether the given url address matches with the regular name of the page is not equal to none the display the message that particular found / match or else not found / unmatched

Step IV: Use the close() to terminate the database library

Database Connectivity

```
>>> import dbm
>>> db = dbm.open("database", 'c')
>>> db["name"] = "name"
>>> if db["name"] == None:
        print("database empty")
    else:
        print("Match found")
Match found
>>> db.close
```

Algorithm

Step I :- Import corresponding library to make database connection, os and sqlite 3.

Step II :- Now create the connection object using sqlite 3 library and the connect() for creating new database.

Step III :- Now create cursor object using the cursor() and from the connection object created.

Step IV :- Now use the execute() for creating the table with the colums name and respective datatype.

Step V :- Now with cursor object use the insert statement for entering the values corresponding to different field, corresponding the datatype.

Step VI :- Use the commit() to complete the transaction using the connection object.

Source Code

```
import os, sqlite3
con = sqlite3.connect("Student.db")
c = con.cursor()
c.execute("Create Table student(roll INTEGER, name
                                TEXT)")
c.execute("Insert INTO student values (1732,'Aparna'),
          (1729,'Vikram'),(1734,'Sachin')")
c.execute("SELECT * ~~from~~ FROM student WHERE
                    roll = 1729")
print(c.fetchall())
con.commit()
c.close()
```

output :-

```
[(1729, "Vikram")]
```

Step VII :- Use the execute statement along with cursor object for accessing the values from the database using the select from where close

Step VIII :- Finally use the fetch() of fetchall() and print the value from the table using cursor object

Step IX :- Finally close() the cursor object

```
from tkinter import *
root=Tk()
Label(root,text="OLD BOOK
SALE",fg="black",bg="red",font=10,height=4,width=40).pack(side=TOP)
def cakes():
    cakes=Tk()
    var1=StringVar()
    C1=Checkbutton(cakes,text="ENGLISH
TEXTBOOK",variable=var1,onvalue="ENGLISH
TEXTBOOK",offvalue="None",fg="green",bg="yellow",padx=26,pady=10)
    C1.pack(pady=30)
    C1.deselect()
    var2=StringVar()
    C2=Checkbutton(cakes,text="MARATHI
TEXTBOOK",variable=var2,onvalue="MARATHI
TEXTBOOK",offvalue="None",fg="green",bg="yellow",padx=26,pady=10)
    C2.pack()
    C2.deselect()
    var3=StringVar()
    C3=Checkbutton(cakes,text="MATHS TEXTBOOK",variable=var3,onvalue="MATHS
TEXTBOOK",offvalue="None",fg="green",bg="yellow",padx=20,pady=10)
    C3.pack(pady=30)
    C3.deselect()
    var4=StringVar()
    C4=Checkbutton(cakes,text="HINDI TEXTBOOK",variable=var4,onvalue="HINDI
TEXTBOOK",offvalue="None",fg="green",bg="yellow",padx=16,pady=10)
    C4.pack()
    C4.deselect()
    var5=StringVar()
    C5=Checkbutton(cakes,text="SCIENCE
TEXTBOOK",variable=var5,onvalue="SCIENCE
TEXTBOOK",offvalue="None",fg="green",bg="yellow",padx=12,pady=10)
    C5.pack(pady=30)
    C5.deselect()
    var6=StringVar()
    C6=Checkbutton(cakes,text="HISTORY
TEXTBOOK",variable=var6,onvalue="HISTORY
TEXTBOOK",offvalue="None",fg="green",bg="yellow",padx=12,pady=10)
    C6.pack()
    C6.deselect()
    def show():
        show=Tk()
        L=Label(show,text="YOU HAVE SELECTED YOUR BOOKS!!!",font=60).pack()
        L=Label(show,text="\n\n\n\n\n\n\n\n\n TO PURCHASE THE BOOKS YOU HAVE
SELECTED.",font=20).pack()
        L=Label(show,text="CONTACT US : 8850014832.",font=10).pack()
        L1=Label(show,text=var1.get()).pack()
        L2=Label(show,text=var2.get()).pack()
        L3=Label(show,text=var3.get()).pack()
        L4=Label(show,text=var4.get()).pack()
        L5=Label(show,text=var5.get()).pack()
        L6=Label(show,text=var6.get()).pack()


Button(cakes,text="Back",command=cakes.destroy,padx=70,pady=20,fg="white",bg
="blue").pack(pady=40)

Button(cakes,text="Proceed",command=show,padx=70,pady=20,fg="white",bg="blue
").pack()
b1=Button(root,text="10TH STANDARD TEXTBOOKS
```

```
(CBSE)",fg="white",bg="green",padx=10,pady=10,command=cakes)
b1.pack(pady=30)

def cakes1():
    cakes1=Tk()
    var1=StringVar()
    C1=Checkbutton(cakes1,text="PHYSICS
TEXTBOOK",variable=var1,onvalue="PHYSICS
TEXTBOOK",offvalue="None",fg="green",bg="yellow",padx=26,pady=10)
    C1.pack(pady=30)
    C1.deselect()
    var2=StringVar()
    C2=Checkbutton(cakes1,text="CHEMISTRY
TEXTBOOK",variable=var2,onvalue="CHEMISTRY
TEXTBOOK",offvalue="None",fg="green",bg="yellow",padx=26,pady=10)
    C2.pack()
    C2.deselect()
    var3=StringVar()
    C3=Checkbutton(cakes1,text="MATHS TEXTBOOK",variable=var3,onvalue="MATHS
TEXTBOOK",offvalue="None",fg="green",bg="yellow",padx=20,pady=10)
    C3.pack(pady=30)
    C3.deselect()
    var4=StringVar()

C4=Checkbutton(cakes1,text="NCRT",variable=var4,onvalue="NCRT",offvalue="Non
e",fg="green",bg="yellow",padx=16,pady=10)
    C4.pack()
    C4.deselect()
    var5=StringVar()

    def show1():
        show1=Tk()
        L=Label(show1,text="YOU HAVE SELECTED YOUR BOOKS!!!",font=60).pack()
        L=Label(show1,text="\n\n\n\n\n\n\n\n TO PURCHASE THE BOOKS YOU HAVE
SELECTED.",font=20).pack()
        L=Label(show1,text="CONTACT US : 8850014832.",font=10).pack()
        L1=Label(show1,text=var1.get()).pack()
        L2=Label(show1,text=var2.get()).pack()
        L3=Label(show1,text=var3.get()).pack()
        L4=Label(show1,text=var4.get()).pack()


Button(cakes1,text="Back",command=cakes1.destroy,padx=70,pady=20,fg="white",
bg="blue").pack(pady=40)

Button(cakes1,text="Proceed",command=show1,padx=70,pady=20,fg="white",bg="bl
ue").pack()
b2=Button(root,text="12TH STANDARD BOOKS
(SCIENCE)",fg="white",bg="green",padx=10,pady=10,command=cakes1)
b2.pack(pady=30)
root.mainloop()
```

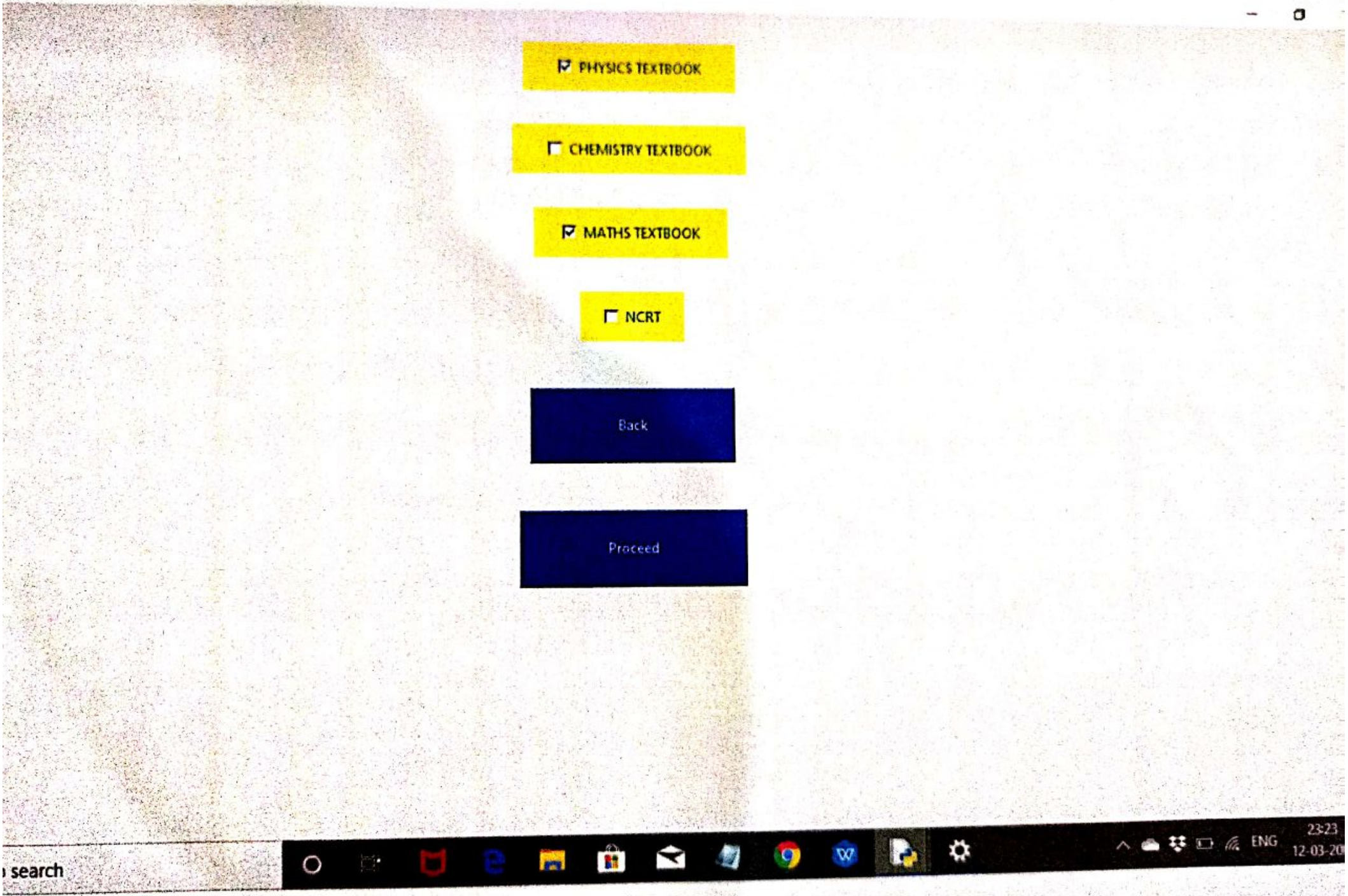
OLD BOOK SALE

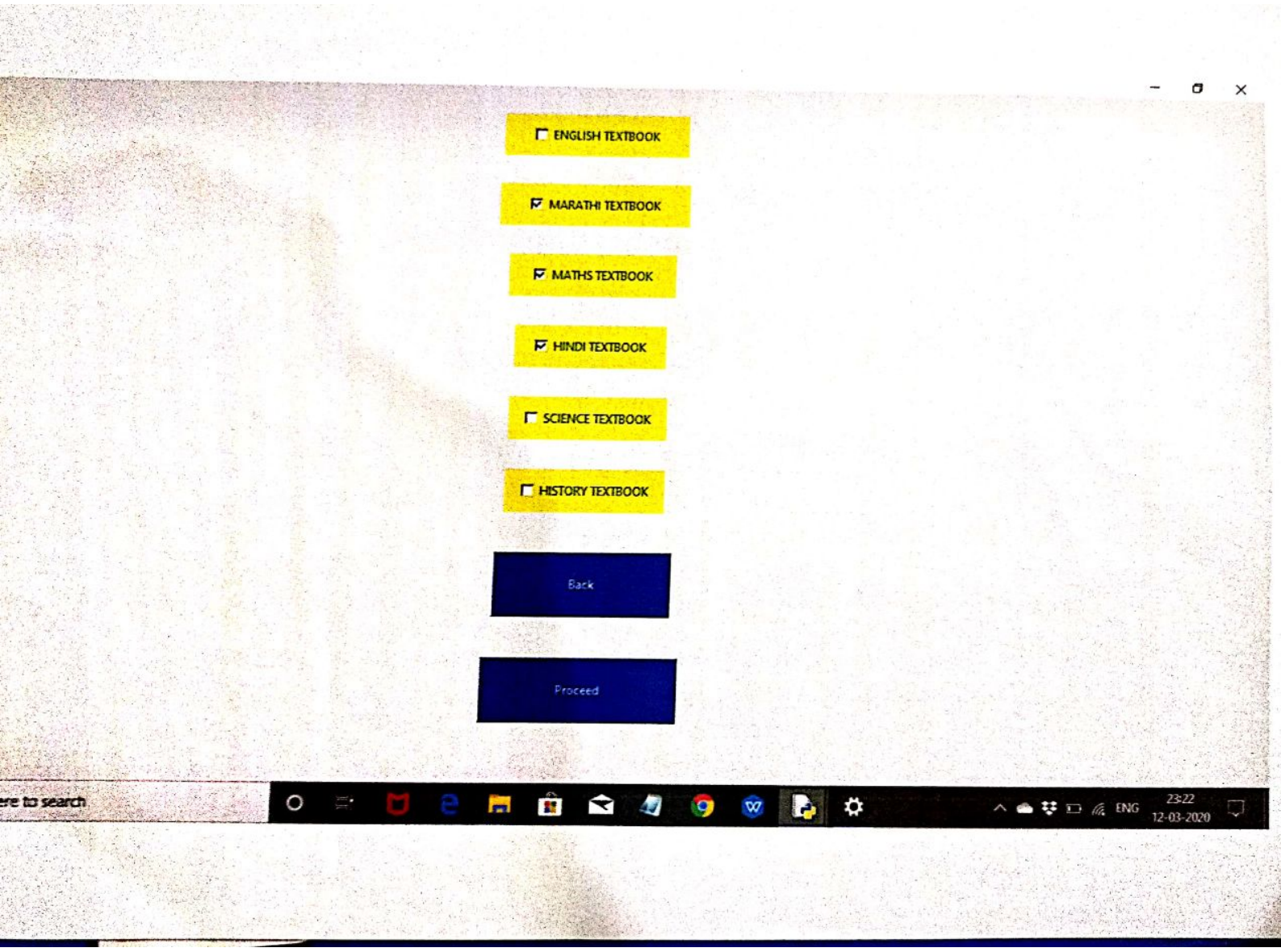
10TH STANDARD TEXTBOOKS (CBSE)

12TH STANDARD BOOKS (SCIENCE)

Type here to search

ENG 23:21 12-03-2020





YOU HAVE SELECTED YOUR BOOKS!!!

TO PURCHASE THE BOOKS YOU HAVE SELECTED.
CONTACT US : 8850014832.

```
import sqlite3

#Creating Connection To Database
con = sqlite3.connect("student.db")

#Creating Cursor Object
c = con.cursor()

#Creating Table Once Only and Then Comment It So As To Avoid Error
'''c.execute("""CREATE TABLE Student(
        name TEXT,
        roll INTEGER,
        class TEXT,
        address TEXT,
        contact INTEGER,
        email TEXT)""")'''

#Asking User To Enter Choice
print("Please Enter Your Choice:")
print("1.Enroll Yourself\t2.See Yourself")
check = int(input())
if(check == 1):
        con = sqlite3.connect("student.db")
        c = con.cursor()

        #Taking Inputs from The User
        name = input("Enter Your Name:")
        roll = int(input("Enter Your Roll No.:"))
        clas = input("Enter Your Class:")
        #clas = clas.upper()
        add = input("Enter your Address:")
        contact = int(input("Enter Your Mobile No.:"))
        email = input("Enter Your Email:")

        try:
                #Taken Inputs are being stored in database
                c.execute("INSERT INTO Student
VALUES(:name,:roll,:class,:address,:contact,:email)",

                {
                    'name':name,
                    'roll':roll,
                    'class':clas,
                    'address':add,
                    'contact':contact,
                    'email':email
                })

                print("You have successfully Enrolled!")
                #Displaying the inputs
                print("Name:  \t"+name)
                print("Roll No.:\t"+str(roll))
                print("Class:  \t"+clas)
                print("Address: \t"+add)
                print("Contact: \t"+str(contact))
                print("Email:  \t"+email)
```

```
        except:
                print("Unknown Error Occurred")
        con.commit()
        c.close()

elif(check == 2):
        con = sqlite3.connect("student.db")
        c = con.cursor()

        #Taking Roll If Already Enrolled
        roll = int(input("Enter Yout Roll No.:"))
        #try:
        c.execute("SELECT * FROM Student WHERE
roll="+"'"+str(roll)+"'")

        record = c.fetchall()

        try:
                #Checking if the roll no. is there in the database or
not
                if(record[0][1] == roll):
                        print("Name: \t"+record[0][0])
                        print("Roll No.:\t"+str(record[0][1]))
                        print("Class:  \t"+record[0][2])
                        print("Address: \t"+record[0][3])
                        print("Contact: \t"+str(record[0][4]))
                        print("Email:  \t"+record[0][5])
                        #print("Here")
                else:
                        print("No Records Found...")
                        con.commit()
                        c.close()
        except:
                print("Please Enter Valid Roll No. Or Else Enroll
Yourself....")
else:
        print("Please Enter Proper Choice")

con.commit()

c.close()
```

Python 3.7.4 Shell

File Edit Shell Debug Options Window Help

```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64 bit
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/vkpan/AppData/Local/Programs/Python/Python37/GauravDB.py =
Please Enter Your Choice:
1.Enroll Yourself       2.See Yourself
1
Enter Your Name:Gaurav Singh
Enter Your Roll No.:1732
Enter Your Class:FYBSC CS
Enter your Address:Mira Road
Enter Your Mobile No.:8850014832
Enter Your Email:gauravpsingh720@gmail.com
You have successfully Enrolled!
Name:           Gaurav Singh
Roll No.:       1732
Class:          FYBSC CS
Address:        Mira Road
Contact:        8850014832
Email:          gauravpsingh720@gmail.com
>>>
= RESTART: C:/Users/vkpan/AppData/Local/Programs/Python/Python37/GauravDB.py =
Please Enter Your Choice:
1.Enroll Yourself       2.See Yourself
2
Enter Yout Roll No.:1732
Name:           Gaurav Singh
Roll No.:       1732
Class:          FYBSC CS
Address:        Mira Road
Contact:        8850014832
Email:          gauravpsingh720@gmail.com
>>>
```

Ln: 33 Col: 4