☆☆ INDEX ☆☆

Scanned with CamScanner

7)	GUI Components (panel window, Canvas)	49	13/2/20	
8)	Database connectivity	51	20/2/20	
9)	GUI and Database	52	27/2/20	

Scanned with CamScanner

Practical - 1

Aim: Programs to read and write files, attributes, read operations and accessing mode of a file.

Algo 1:

1) Open a file with write accessing mode and assign it to a variable. (File object)
2) Write a content in opened file and close it afterwards
3) Open the same file with append accessing mode, assigning it to a variable. (File object)
4) Write or append new content in opened file and close it
5) Open the file in read accessing mode and read the content of the specified file.
6) Open the file in read accessing mode and use read, readline, readlines method and print their outputs.
7) Use attributes of a file and print them.

Algo 2:

1) Use raw input which will accept input as a string.
2) Use input method which will take expression and will return valve of it.

Algo 3:

1) Create a file object and open the file from algo 1 in rt accessing mode. or r accessing mode.
2) Use read method to read the 5 character from the file and display the content
3) Use tell method along the file object and display it.
4) Now use the seek method with the two arguments

Code:

```
# writing, reading, reading methods.
f = open("abc.txt", "w")
f.write("my name is sugash" + "\n")
f.close()
f = open("abc.txt", "a")
f.write("studying a professional course.")
f.close()
f = open("abc.txt", "r")
print(f.read(), "\n")
f = open("abc.txt", "r")
print(f.readline())
f = open("abc.txt", "r")
print(f.readlines())
f.close()

# attributes of a file.
print(f.closed)
print(f.mode)
print(f.name)
print(f.softspace)

# input methods.
>>> a = raw_input("Enter string:")
    Enter string: Hello
>>> a
    'Hello'.
>>> b = input("Enter expression:")
    Enter expression: 8*9
>>> b
    72
```

output:

```
my name is sugash
studying a professional course.

my name is sugash

['my name is sugash\n', 'studying a
 professional course']
False
True
r
abc.txt
0
```

Scanned with CamScanner

as zero and then read certain number of characters and display the read characters. similarly use the seek method with zero and one value indicating the reference position of a cursor as the current location, while zero and two indicates the referenced position of the cursor from the end of the file.

Q program to put '*' after every letter:

Algo:

1) Create a file object and open a file in read mode.
2) read one character from the file and store it in a variable
3) use while conditional statement till the length of variable in step 2 is zero.
4) print the variable from step 2 and use 'end' method to print '*' and increase the counter by reading the content of file once again.

Q program to find the length of each line of the content in a file

Algo:

1) Create a file object and open a file in r+ accessing mode.
2) write a content in a file along with file object.
3) use readlines method and print it; also assign it to variable.
4) use for conditional statement till ~~range~~ variable in step 2, subsequently print length ~~for~~ of line in each iteration.

Scanned with CamScanner

seek and tell

```
f = open ("abcd.txt", w)
f.write (" Hello world")
f.close ( )
f = open ("abcd.txt", r)
print ( f.read (5))
print ("current position:", f.tell ())
print (" position (referenced) changed to beginning:", f.seek (0,0))
print ( f.read (7))
print (" reference position:", f.seek (0,1), f.read ())
print (" position changed to end:", f.seek (0,2))
print ( f.read())
```

Output:

```
Hello
current position: 5
current position changed to beginning: 0
Hello W
reference position: 7
orld
position changed to end: 11
```

Scanned with CamScanner

AS

program for adding '*' after every letter

```
g = open (~~info~~"abcd.txt", "r")
c = g.read (1)
while len(c) > 0:
        print( c, end = " * ")
        c = g.read(1)
```

output:

h * e * L * L * O * * W * o * r * L * d *

program for counting length of a line in a file.

```
f = open ("abc.txt", "r")
text = f. realines()
i = 0
for l in text:
    print (text[i], len[l])
    i += 1
```

output:

my name is suyash 17

studying a professional course 30.

Suyash

Scanned with CamScanner

Practical - 2

Aim: Demonstrate the use of iterators and iterables.

Program 1:

Algorithm -

① Define a variable of list data type containing names of fruit.

② Now iterate over that variable and print each element using next method or some conditional loops statements.

Program 2: To display odd no. till 16.

Algorithm:

① Create a class within that define a iter method with an argument and initialize the valve and return that valve.

② Define the next method with an arguement.

③ use if conditional statement to check wethere the variable given in 1st step arguement is smaller than 16.

④ Increase the arguem variable's valve by 2. and return it.

⑤ use else to stop iteration.

⑥ Create an object of given class and pass this object in iter method.

⑦ use while conditional loop to print the next method.

Scanned with CamScanner

Program 3: Factorial of number till 10.

Algorithm:-

① Create a class within that define a iter method with an arguement, initialize a value and return it.

② Define a next method with an arguement

③ Create a variable facto with ~~varia~~ value 1 and use for conditional loop to calculate factorial till ~~range~~ value of variable declared in step 1.

④ Print the facto's value and increase the value of variable in step 1 by 1.

⑤ ~~If~~ use if conditional statement to raise stopiteration if the range of variable in step 1 is bigger than 10.

⑤ Create a object this class and pass it to iter method and use while loop to use next method.

Program 4: Power of number taken from user.

Algorithm:

① Create a class and define a iter method with an arguement; initialize a value and return it. also take a input from user.

② Define a next method with an arguement

③ If the initialized value or input variable is bigger than 5 then raise stop Iteration or else return the multiplication of arguement by itself.

④ ~~use while~~ create a object of given class and pass it to iter method and use while loop to use next method.

Scanned with CamScanner

1] Code:

```
i = ["Apple", "Banana"]
myiter = iter(i)
for K in i:
    print(K)
```

output:

Apple
Banana

2] Code:

```
class odd:
    def __iter__(m):
        m.value=1
        return m
    def __next__(m):
        if m.value < 15:
            m.value+=2
            return m.value
        else:
            raise StopIteration
y=iter(odd())
While True:
    print(next(y))
```

output:
1
3
5
7
9
11
13
15

3] Code:

```
Class fact:
    def __iter__(m):
        m.value=1
        return m
    def __next__(m):
        facto=1
        for i in range(1,m.value+1):
            facto = facto * (i)
        print("The factorial of",m.value,"is",facto)
        m.value+=1
        if (m.value==7):
            raise STOPIteration
```

Scanned with CamScanner

Program 5: To display the number in range 10.

Algorithm:

① Create a class and define iter method with an orguement, initialize a value and return it.

② Now define next method check wether the initialized value is smaller than 10, print that value.

③ Increase the value by 1; if the initialized value is bigger than 10 stop the iteration.

Scanned with CamScanner

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

output:

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

4) Code:

```
class power:
    def __iter__(pow):
        pow.num=1
        pow.nums=int(input("enter number:"))
        return pow
    def __next__(pow):
        if pow.num<=6:
            nu=pow.nums**pow.num
            pow.num+=1
            return nu
        else:
            raise StopIteration.
y=iter(power())
while True:
    print(next(y))
```

output:

Enter number: 2
2
4
8
16
32
64

Scanned with CamScanner

5) code:

```
class range1:
    def __iter__(o):
        o.num = 1
        return o
    def __next__(o):
        if o.num <= 10:
            nu = o.num
            o.num += 1
            return nu
        else:
            raise StopIteration
y = iter(range1())
while True:
    print(next(y))
```

output:

1
2
3
4
5
6
7
8
9
10

19/12/19

Scanned with CamScanner

Aim: Program to demonstrate exception handling

1) use try block, create a file object write some content in that file.
2) Use except IO error and print the errors accordingly
3) use else statement if there is no error

1) take a input from the user and initialize it in a variable in try block.
2) use except statement for value error and print subsequent message
3) use else statement and print operation is successful.

1) Initialize a variable with some value.
2) Take a input from user as a integer data type.
3) use try block and perform division operation with both the variables.
4) use except block for Type block and display appropriate messages.
5) use except block for ZeroDivisionError and display appropriate messages.

Scanned with CamScanner

* 1) Define a function with an argument.
2) use Assert statement to check whether the length of argument is equal to zero.
3) print list is empty if assertion statement is True.
4) Initialize a variable with empty list
5) call the function defined in first step.

* 1) Define a function with an argument and initialize a empty list
2) print the last element in a try block.
3) use except for Index error and print appropriate message. like List is empty else perform something
4) Define a Function with an argument and initialize
5) print the last element in a try block.
6) use except for Index error and print appropriate message.
7) else print some operation on the list
8) call both the function defined.

Scanned with CamScanner

IO error

```
try:
    Fo = open("abc.txt", "w")
    Fo.write("python is an interpreter language")
except IoError:
    print("I/O error is there, fix it")
else:
    print("Operation is successful")
```

>>> Operation is successful!

Value error

```
try:
    x = int(input("Enter a statement:"))
except ValueError:
    print("Nope, Enter integer no.")
else:
    print("Operation is successful!")
```

>>> Enter a statement: XY
Nope, Enter integer no.

Type error, zero division error

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

> Enter: XYZ
Incompatible value

> Enter: 0
Denominator is zero.

Scanned with CamScanner

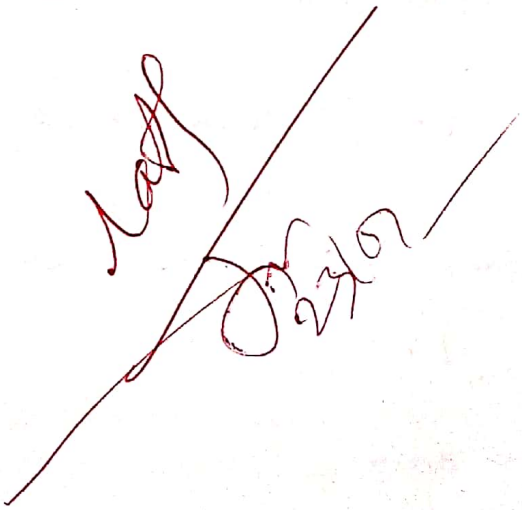
assert

```
def assertion(m):
    assert (len(m) == 0)
    print ("list is empty")
var1 = []
print (assert_(var1))
```

```
def list1(a):
    a = []
    try:
        print (a[-1])
    except IndexError:
        print ("List is empty")
    else:
        print (a * 2)

def list2(b):
    b = [1,2,3,4]
    try:
        print (b[-1])
    except IndexError:
        print ("list is not empty")
    else:
        print (b * 2)
list1("abc")
list2("xyz")
```

```
>>> List is empty.
>>> [2,4,6,8]
```



Scanned with CamScanner

Aim: program to demonstrate the use of regular expression.

1) Algo:
* Import re module
* assign a variable with a string
* match using method first word and subsequently print it.
* ~~as match~~ search using method last word and subsequently print it

2) Algo:
* Import re module
* assign a variable with a string.
* declare pattern with litered and meta characters.
* Use the findall() with arguements and print the same

3) Algo:
* Import re module.
* declare list with number use
* use the conditional statement.
* used for conditional statment inside it use if conditional statement for checking first number is either 8 or 9 and next number are in range of 0 to 9 and check wether the entered number are equal to 10.
* if criteria matches print succesfull match of no. otherwise print failed.

Scanned with CamScanner

assert

def assertion(

assert (

print

var1 = [

print (

which declare metacharacters and without blank space

findall is true then print the

print no blank space.

import re module

use findall method inside which use metacharacter

to get first word of a declared string after it

print the result.

subsequently do the same for last word and print the result

```
import re
data = 'python is an indentated language'
a = re.match (r"python", data)
print ('result:' + a.group())
b = re.search (r'language', data)
print ('result:' + b.group())
```

```
import re
string = 'ABCD 123 abc'
result = re.findall (r'\d+', string)
print (result)
```

```
import re
li = ['8894123446', '4912312345', '9147126341']
for val in li:
    if (re.match(r'[8-9]{1}[0-9]{9}', val) and len(val) == 10):
        print ('correct', val)
    else:
        print ('not correct:', val)
```

Scanned with CamScanner

```
import re
result = re.findall (r'\w+', "g tsg  gtg  g")
if result is Twe:
    print (result)
else:
    print ("no blank space")
```

```
import re
string = "python is indentated language"
result = re.findall (r'^\w+', string)
print (" first word", result)
result = re.findall (r'\w+$', string)
print ("last word", result)
```

Scanned with CamScanner

6) Algo:

* import re module
* use findall() method and use metacharater to get date from the declared string after it
* print the result of findall.

7) Algo:

* import re module
* use findall() method to extract the username from email id by declare after it by appropriate meta-characters.
* subsequently use findall() to extract hostname from email-id by appropriate meta-characters.

Scanned with CamScanner

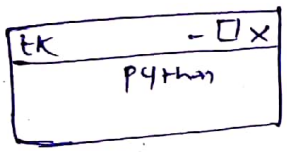
```
import re
string = "amit 201 24-12-2019
date = re.Findall (r'\d{2}-\d{2}-\d{4}', string)
print (date)
```

```
import re
string = 'yashdubey9630@gmail.com'
email = re.findall (r'(\w+)@', string)
print ('username:', email)
email = re.findall (r'(\w|-]', string)
]
print ('hostname:', email)
```

creation of parent window

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

Output:



#2

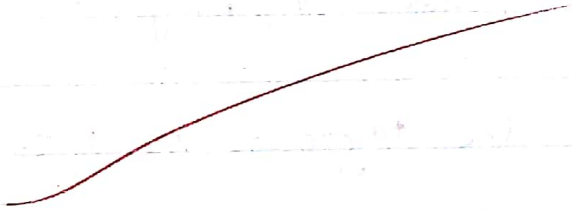
```
from TKinter import *
root = TK()
l = Label(root, text="python")
l.pack()
l1 = Label(root, text="cs", bg="grey", fg="black", font="10")
l1.pack(side=LEFT, padx=20)
l2 = Label(root, text="cs", bg="light blue", fg="black", font="20")
l2.pack(side=LEFT, pady="30")
l3. = Label(root, text="cs", bg="yellow", fg="black", font="10")
l3.pack(side=Top, ipadx=40)
l4 = Label(root, text="cs", bg="orange", fg="black", font="10")
l4.pack(side=Top, ipady=50)
root.mainloop()
```

Scanned with CamScanner

GUI components

① Algo:

1) use the tkintor library for importing the features of the text widget
2) create an object using the TK()
3) create a variable using the widget label and use the text method.
4) use the mainloop() for triggering of the corresponding above mentioned events.



Scanned with CamScanner

(2) Algo:

1) Use the tkinter library for importing the features of the text widget

2) Create a variable from the text method and position it on the parent window.

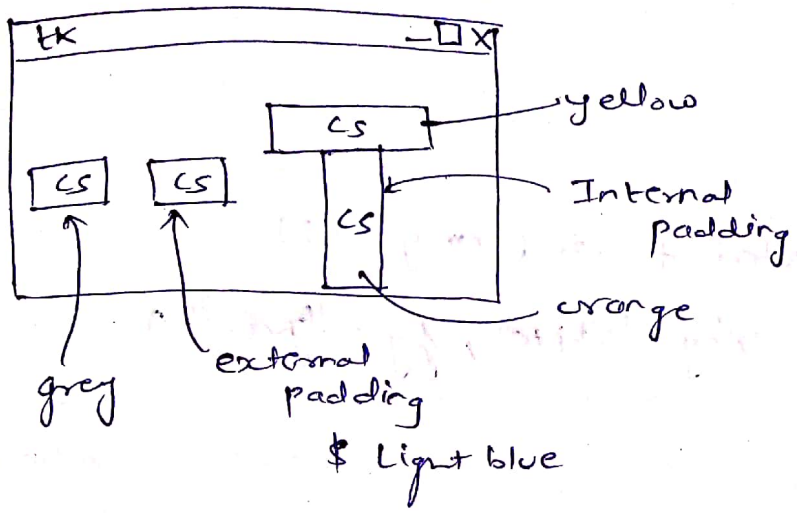
3) Use the pack() along with the object created from the text() and use the parameter
side = left, padx = 20 ; side = left, pady = 30
side = Top, ipadx = 40 ; side = Top, pady = 50

4) Use the mainloop() for the triggering of the corresponding events

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

Scanned with CamScanner

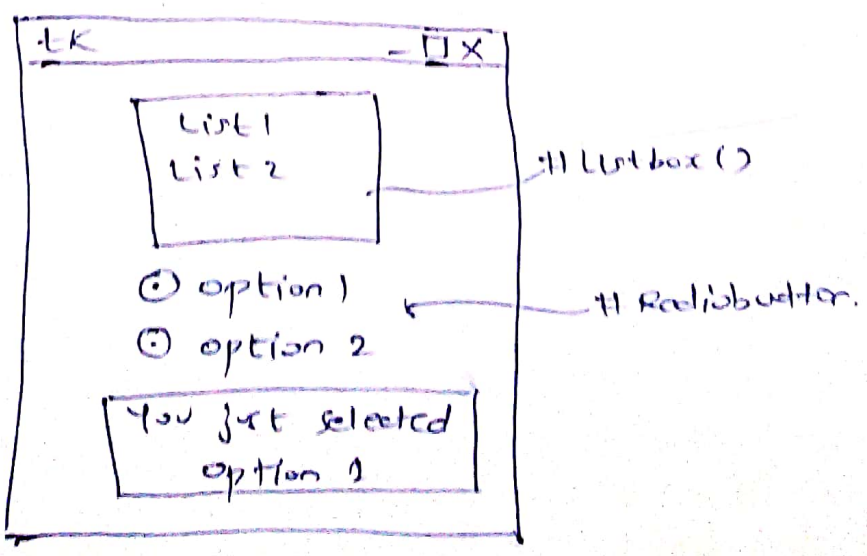
output:


tk
cs
cs
cs
cs
yellow
Internal padding
orange
grey
external padding
$ Light blue

#1

```
from tkinter import *
root = Tk()
root.geometry("500x500")
def select():
    selection = "You just selected" + str(var.get())
    l1 = Label(text = selection, bg = "white", fg = "green")
    l1.pack(side = Top)
var = StringVar()
l1 = Listbox()
l1.insert(1, "List 1")
l2.insert(2, "list 2")
l1.pack(anchor = N)
r1 = Radiobutton(root, text = "option 1", variable = var, value =
    "option 1", command = select)
r1.pack(anchor = N)
r2 = Radiobutton(root, text = "option 2", variable = var, value =
    "option 2", command = select)
r2.pack(anchor = N)
root.mainloop()
```

Output:



Scanned with CamScanner

Practical-5 (b)

#1

Algo:

1) Import the relevant methods from the tkinter library create an object with the parent window.
2) Use the parent window object along with the geometry() declaring specific pixel size of the parent window.
3) Now define a function which tells the user about the given selection made from multiple option available
4) Now define the parent window and define the option with control variable
5) Use the listbox() and input insert option on the parent window along with the pack() with specifying anchor attribute
6) Create an object from radio button which will take following arguements & parent windows object, text variable which will take the values option no 1, 2, 3, ... variable argument, corresponding value & trigger the function declared
7) Now call the pack() for radio object so created and specify the argument using anchor attribute.
8) Finally make use of the mainloop() along with parent object

Scanned with CamScanner

#2

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

#3

1) Import the relevance libraries from the tkinter method.
2) Create an corresponding object of the parent window.
3) use the geometry manager with pixel size (680*600) or any other suitable pixel value.
4) use the label widget along with the parent object created and subsequently use the pack method.
5) use the frame widget along with the parent objet ~~of~~ created and use the pack method.
6) use the listbox method along with the attributes like width, height font. Do create a listbox method's object use pack() for the same
7) Use the scrollbar() with an object use the attribute of vertical then configure the same with objects created from the scrollbar() and use pack()
8) Trigger the events using mainloop.

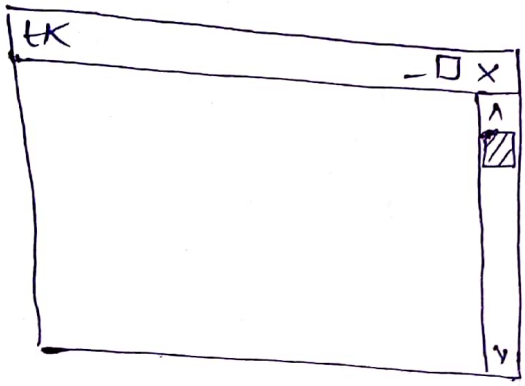
Scanned with CamScanner

#2

Scrollbar()

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

Output:

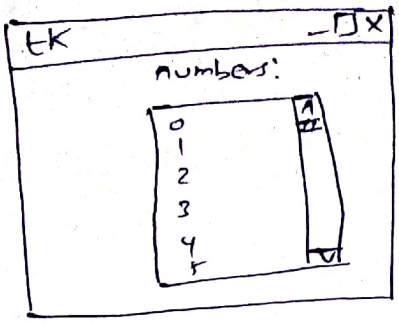


#3

```
from tkinter import *
window = TK()
window.geometry("680 x 800")
Label(window, text="numbers:").pack()
frame = Frame(window)
frame.pack()
listnodes = Listbox(frame, width=20, height=20, font=("Times New Roman", 10))
Listnodes.pack(side="left", fill='y')
scrollbar = scrollbar(frame, orient="vertical")
scrollbar.config(command = listNodes.yview)
scrollbar.pack(side="right", fill="y")
for x in range(100):
    listNodes.insert(END, str(x))
window.mainloop()
```

Scanned with CamScanner

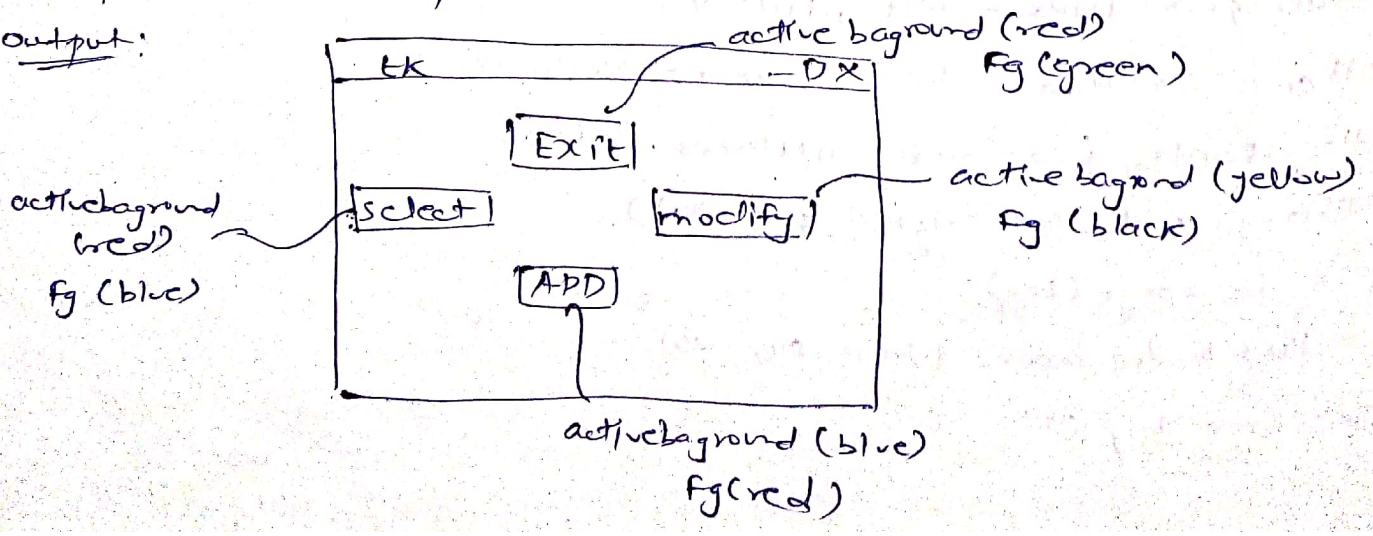
Output:



#4

```
from tkinter import *
window = TK()
window.geometry("680x500")
frame = Frame(window)
frame.pack()
leftframe = Frame(window)
leftframe.pack(side="left")
rightframe = Frame(window)
rightframe.pack(side="right")
b1 = Button(frame, text="select", activebackground="red", fg="blue")
b2 = Button(frame, text="modify", activebackground="yellow", fg="black")
b3 = Button(frame, text="ADD", activebackground="blue", fg="red")
b4 = Button(frame, text="exit", activebaground="red", fg="green")
b1.pack(side="left", padx=20)
b2.pack(side="Right", padx=30)
b3.pack(side="Bottom", pady=20)
b4.pack(side="Top")
```

Output:



Scanned with CamScanner

4.

1) Import relevant methods from tkinter library
2) Define the object corresponding to parent window and define the size of parent window in terms of no of pixels.
3) Now define the frame object from the method and place it on the parent window
4) Create another frame object termed as the left frame and put it on the parent window on its left side
5) Similarly define the Right frame and subsequently define the button object placed onto the given frame with the attribute as text, active background and foreground.
6) Now use the pack() along with the side attribute.
7) Similarly create the button object corresponding to the Modify operation put it into frame object on side = "right"
8) Create another button object & place it on to the Right frame & label the button as ADD.
9) Add another button & put it on the top of frame and label it as exit
10) Use the pack() simultaneously for all the objects & finally use the mainloop().

practical 5(c)

Algo:

1) Import the relevant method from tkinter library.

2) Import tkMessagebox

3) Define a parent window object along with the parent window.

4) Define a function which will use tkMessagebox with showinfo method along with info window attribute.

5) Declare a button with parent window object along with the command attribute

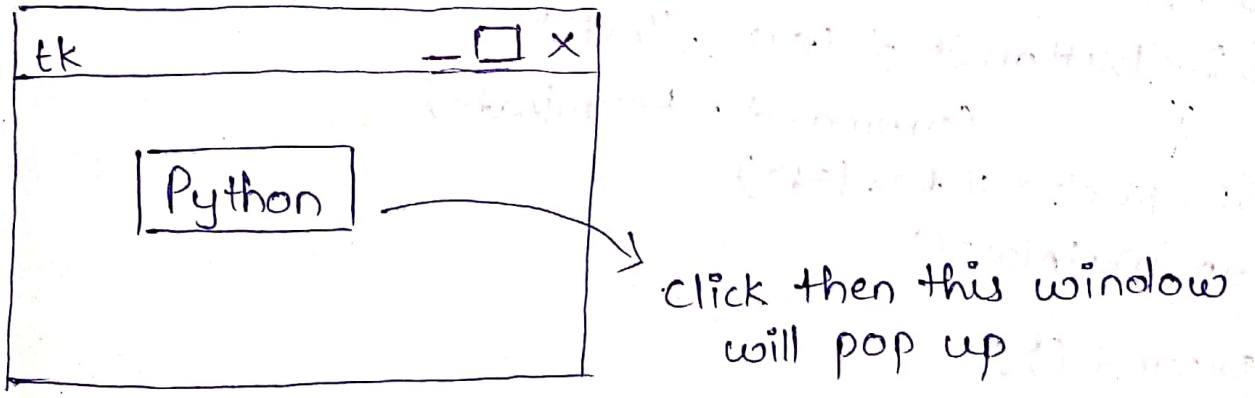
6) Place the button widget onto the parent window & finally call mainloop for triggering of the events called above.

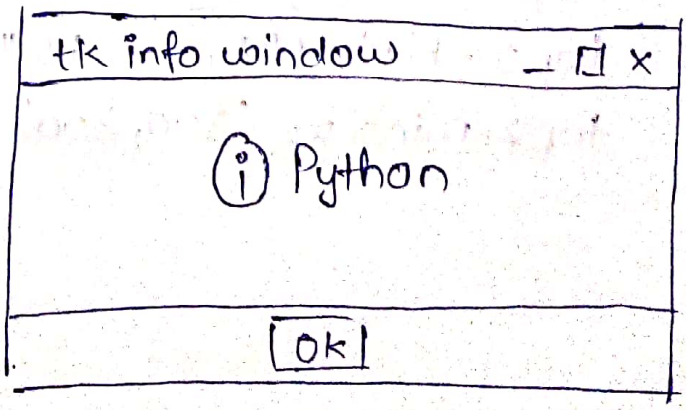
Scanned with CamScanner

Message Box

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

OUTPUT :-





Scanned with CamScanner

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

from tkinter import *
root = Tk()
root minsize (300, 300)
def main():
        top = tk()
        top.config (bg = "black")
        top.title ("HOME")
        top.minsize (200,200)
        L = label (top, text = "ARIBA
                In Name = In DRISHTI In UNNATI
                In SAURAV")
        L.pack()
        b1 = Button (top, text = "next",
                command = second')
        b1.pack()
        b2 = Button (top) text = "exit",
                command = terminate)
        b2.pack (side = LEFT)
        top.mainloop()

def second():
        top2 = Tk()
        top2.config (bg = "orange")
        top2.title ("About US!")
        top2.minsize (300, 300)
```

Scanned with CamScanner

#2 Algo:

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

2) Use parent window object along with minsize function for window size

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

4) Similarly, define the function second & use the attribute accordingly

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

Scanned with CamScanner

```
bf = Button (top3, text = "ridge button", relief = RIDGE
  bf.pack()
top3.mainloop ()
def  terminate ()
      bf. quit ()
      b5 = Button (root, text = "Nomec", command = main)
      b5.pack ()
      b6 = Button (root, text = "Button details", command = Button)
      b6.pack()
root.mainloop()
```

Scanned with CamScanner

```
l = Label (top2, text = "Created by:
AripplesoftAll In for more detail =
Contact to our official site")
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 ("250x250")
    b1 = Button (top3, text="flat button", relief = FLAT)
    b1.pack()
    b2 = Button (top3, text = "raised button", relief = RAISED)
    b2.pack()
    b3 = Button (top3, text = "grave button", relief = GROOVE)
    b3.pack()
    b4 = Button (top3, text = "sunken button",
                 relief = SUNKEN)
    b4.pack()
```

Scanned with CamScanner

```
from Tkinter import *
root = Tk()
root
root.maxsize(1000, 900)
root.config(bg = "black")
leftframe = Frame(root, bg = "pink", height = "400",
                  width = "300")
leftframe.grid(row = 0, column = 0)
rightframe = Frame(root, bg = "green", height = "300",
                   width = "250")
rightframe.grid(row = 0, column = 0)
Label(leftframe, text = "photo", height = 2, width = 20).
                  grid()
Image1 = PhotoImage(file = "dance.gif")
Image1.subsample(1,2)
Image2 = PhotoImage(file = "dance.gif").
Image2.Subsample(3,4)
Label(leftframe, image = image1).grid
      (row = 0, column = 0)
Label(rightframe, image = image2).grid
      (row = 0, column = 1)
toolbar = Frame(leftframe, width = 200,
          height = 400, bg = "white").
          grid(row = 2, column = 0).
```

Scanned with CamScanner

Aim: Displaying the image, spinbox

Algo:

1) Create an object corresponding to the parent window & use the following 3 methods :- Title, config, maxsize

2) Create a leftframe object from the frame method & place it onto the parent window with the height, width & the bg specified

3) Now create rightframe object from the frame method with the width, height specified & the row & the column value should be specified.

4) Create a label object from the label method & place it onto the leftframe with the text attribute denoting the original image with relief attribute used as the RAISED & subsequently use grid method with row, column value specified as (0,0) with same external padding view

5) Now use the photo image method with the file attribute specified

6) Use the subsample method with the object of the image & give the x, y coordinate value.

Scanned with CamScanner

7) Use the label method & position it onto the leftframe & placing the image after the sampling & use the grid method for the positioning in the first row.

8) Create another label object positioning it onto the leftframe & specifying the image & background attribute with row and column attribute specify it as (0,0)

9) Now create a toolbar object from the frame method & position it onto the leftframe with the height & width specified & position it onto the second row.

10) Now define the various function for different toolbar options provided in the leftframe.

11) From the label method position the text on the toolbar use the relief attribute & corresponding grid value & incorporate the internal padding as (0,1)

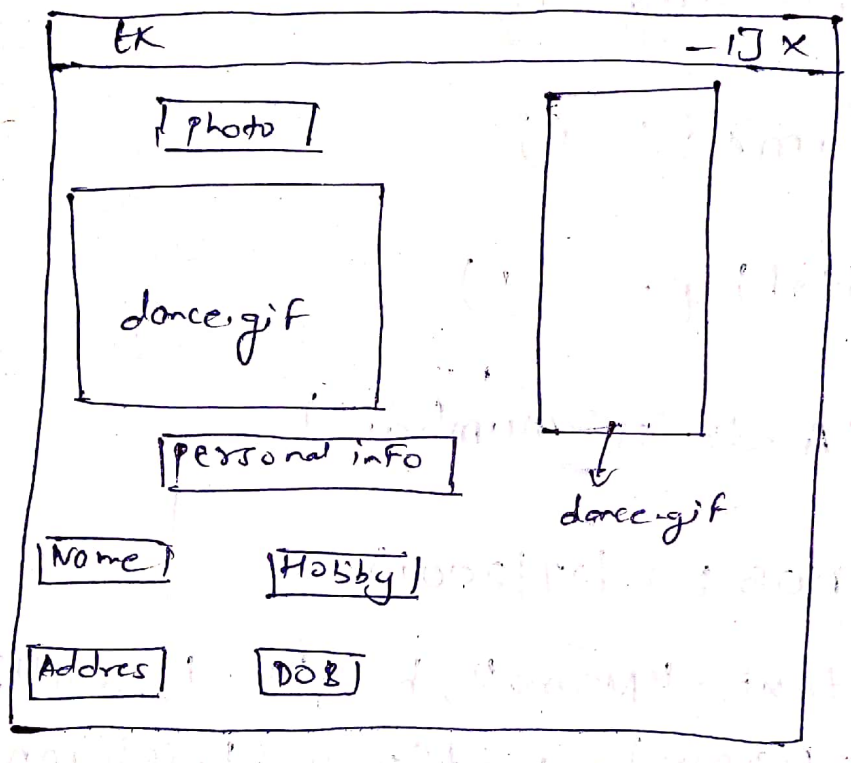
12) Create the label method position it on to the toolbar with the next title as personal information & position it on same row but next column

13) Now make use of mainloop method.

Scanned with CamScanner

```
Label (toolbar, text = "Personal Info",
      height = 2, width = 20, relief = RAISED)
  grid (row = 0, column = 0)
def name ():
      print ("Name : ")
def lob ():
      print ("Hobby = ")
def add ():
      print ("Address: Mumbai")
def dob ()
      print ("DOB : 07/07/2001)
Button (toolbar, text - "Name", height = 1, width = 16,
      command = name). grid (row = 1, column = 0)
Button (toolbar, text = "Hobby", command = hob).
      grid (row = 0, column = 0)
Button (toolbar, text = "Address", command = add).
      grid (row = 0, column = 0)
root.mainloop()
```

OUTPUT :-



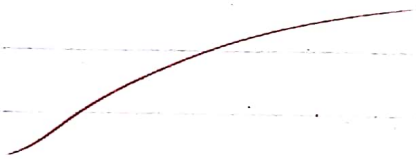
Scanned with CamScanner

Spinbox Widget :-

Algorithm

1) Create an object from the Tk() & subsequently create an object from the spinbox

2) Make the object so created onto the parent window & trigger the corresponding events.

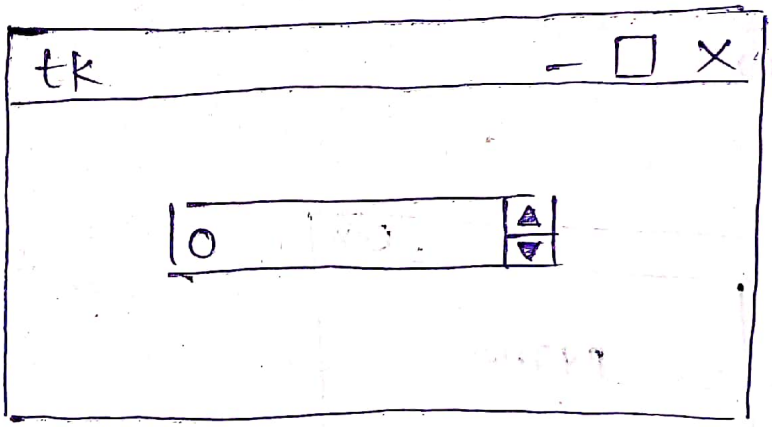


Scanned with CamScanner

CODE :-

```
from Tkinter import *
master = Tk()
S = Spinbox (master, from_=0, to = 10)
S.pack ()
master.mainloop()
```

OUTPUT :-



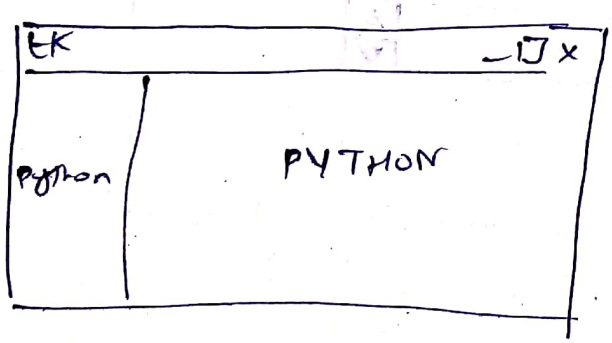
Scanned with CamScanner

8A

Code:

```
from tkinter import *
root = TK()
p = panedWindow (bg ="pink")
p.pack ( Fill =BOTH , expond =10)
l = Label (p, text = "python")
p.add (l)
p1 = Panedwindow (p ,orient =VERTICAL, bg = "white")
p.add (p1)
l1 = Label (p1 ,text= "PYTHON")
p1.add (l1)
root.mainloop ()
```

Output:



Scanned with CamScanner

Aim: GUI components (panedwindow, canvas)

PANED WINDOW WIDGET

Algorithm:

1) Create an object from panedwindow() and use the pack() with attribute fill & expand.

2) Create an object from the label method & put it onto the paned window with the text attribute & use the add method to embed the new object

3) Similarly, create a second paned window object & add it onto the first panned window with orientation specified.

4) Now create another label object & place it onto the second paned window object & add it onto the second paned window. Trigger the mainloop.

Scanned with CamScanner

P.1

CANVAS WIDGET

Algorithm:

1) Create an object from the canvas method and use the attribute height, width, bg & pared window objects.

2) Use the method create line, create oval & create arc along with the canvas object so created & use the Coordinate values.

Scanned with CamScanner

code:

```
from Tkinter import *
root = Tk()
c = Canvas (root, height =200, width =280, bg ="orange")
arc = c.create_arc (10,20,30,40, start=0, extent =100, Fill ="red")
line = c.create_line (50,60,70,80)
oval = c.create_oval (90,100,110,120)
c.pack()
root.mainloop()
```

```
>>>import os.sqlite3
>>>conn=sqlite3.connect("student.db")
>>>cur = conn.cursor().
>>>cur.execute('create table sinfo (RNO int, Name text, DOB date)').
<sqlite3.cursor object at 0x02F62F20>
>>> cur.execute('insert into sinfo values (01,"Anuj","02-09-2001"),
    (02,"Raj","09-02-2001"),(03,"Bhavna","03-05-2000"),
    (04,"Ram","05-06-2000"),(05,"Lavesh","03-09-2002")')
<splite3.cursor object at 0x02F62F20>
>>> cur.execute('select DOB from sinfo')
<splite3.cursor object at 0x02F62F20>
>>> cur.fetchall()
[('02-09-2001'),('09-02-2001'),('03-05-2000'),('05-06-2000'),('03-09-2002
>>> cur.execute(update stu_sinfo set DOB="03-05-2001" where RNO=03')
<splite3.cursor object at 0x02F62F20>
>>> cur.execute('select DOB from sinfo')
<split3.cursor object at 0x02F62F20>
>>> cur.fetchall()
[('02-09-2001'),('09-02-2001'),('03-05-2001'),('05-06-2000'),('03-09-200
>>> cur.execute('alter table sinfo add Address text')
<splite3.cursor object at 0x02F62520>
>>> cur.close()
```

Scanned with CamScanner

Practical-8

Database Connectivity

Algo:

1) Import the relevant libraries for database and the operating system functionality.

2) Now create an object for making the connection to the given database.

3) Further create object corresponding to the cursor area for execution of the different query statement

4) Use the cursor object so created for implementing the structure of the database and the values within the DB.

5) Use the execute() for implementation of the select cause for filtering the information.

6) Now use the Fetchall() along with the cursor object for displaying the value onto the screen

27/12

practical - 9

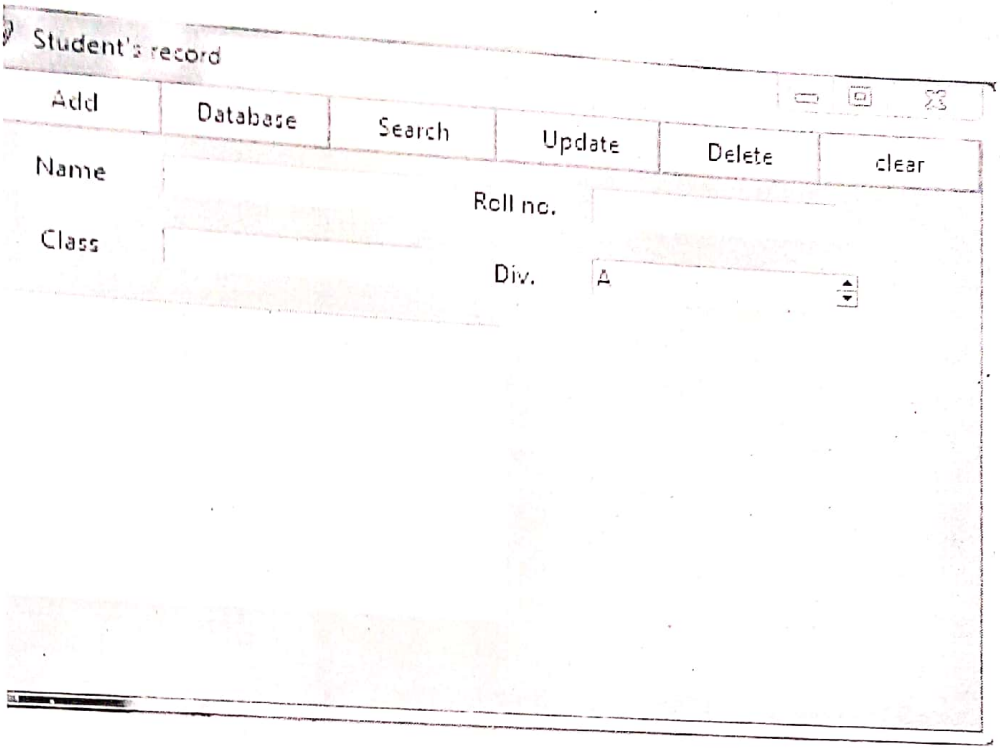
GUI and Database.

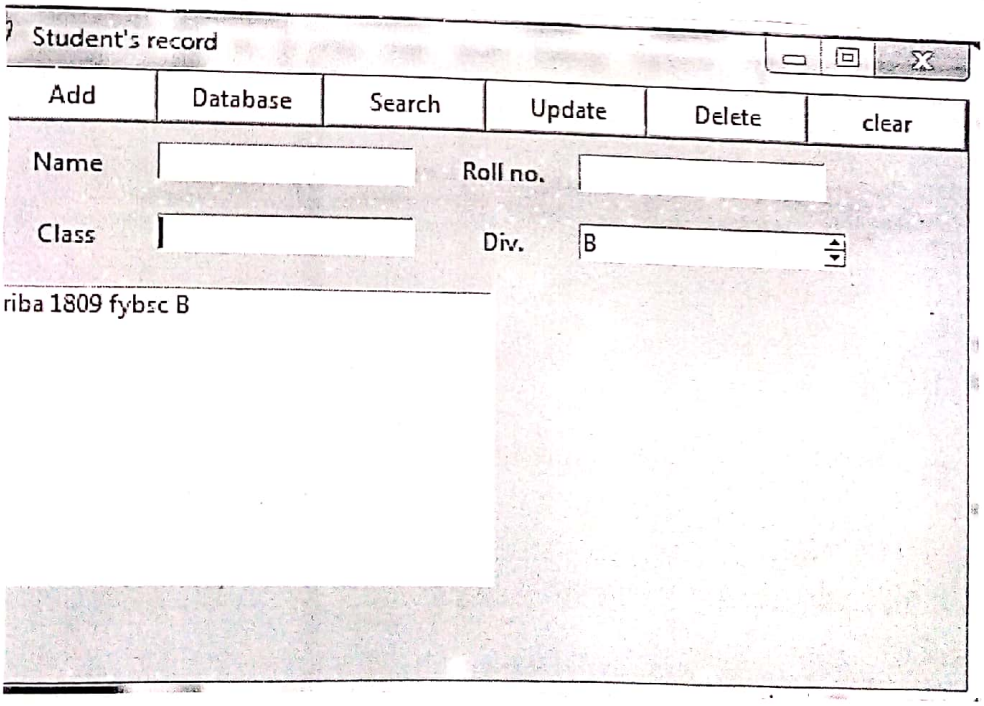
```
from tkinter import *
import sqlite3
conn = sqlite3.connect("student.db")
cur = conn.cursor()
cur.execute("CREATE TABLE IF NOT EXISTS student (id INTEGER PRIMARY
KEY,name TEXT, roll_no  INTEGER, class TEXT, div TEXT)")
conn.commit()
def  clear():
        name.set('')
        class_.set('')
        roll_no.set('')
def insert():
        cur.execute("INSERT INTO student VALUES (NULL,?,?,?,?)",
(name.get(),roll_no.get(),class_.get(),div.get()))
        list_.delete(0, END)
        list_.insert(END,
(name.get(),roll_no.get(),class_.get(),div.get()))
        clear()
        conn.commit()
def update():
        index = list_.curselection()[0]
        sel=list_.get(index)
        cur.execute("UPDATE student SET name=?, roll_no=?, class=?,div=?
WHERE id=?", (name.get(),roll_no.get(),class_.get(),div.get(),sel[0]))
        conn.commit()
def delete():
        index = list_.curselection()[0]
        selected = list_.get(index)
        cur.execute("DELETE FROM student WHERE id=?", (selected[0],))
        conn.commit()
def search():
        cur.execute("SELECT * FROM student WHERE name=? OR roll_no=? OR
class=? OR div=?", (name.get(),roll_no.get(),class_.get(),div.get()))
        rows = cur.fetchall()
        list_.delete(0, END)

        for i in rows:
            list_.insert(END,i)

def database():
        def psswd_chk():
                pss=psswd_entry.get()
                if pss=="   ":
                    cur.execute("SELECT * FROM student")
                    rows = cur.fetchall()
                    list_.delete(0, END)
                    for i in rows:
                        list_.insert(END,i)
                else:
                    Label(window,text="invalid").place(x=340,y=210)
                psswd_entry.set('')
        Label(window, text="pswrd").place(x=275,y=150)
        psswd_entry= StringVar()
Entry(window,textvariable=psswd_entry,show='*').place(x=320,y=152)
```

Scanned with CamScanner

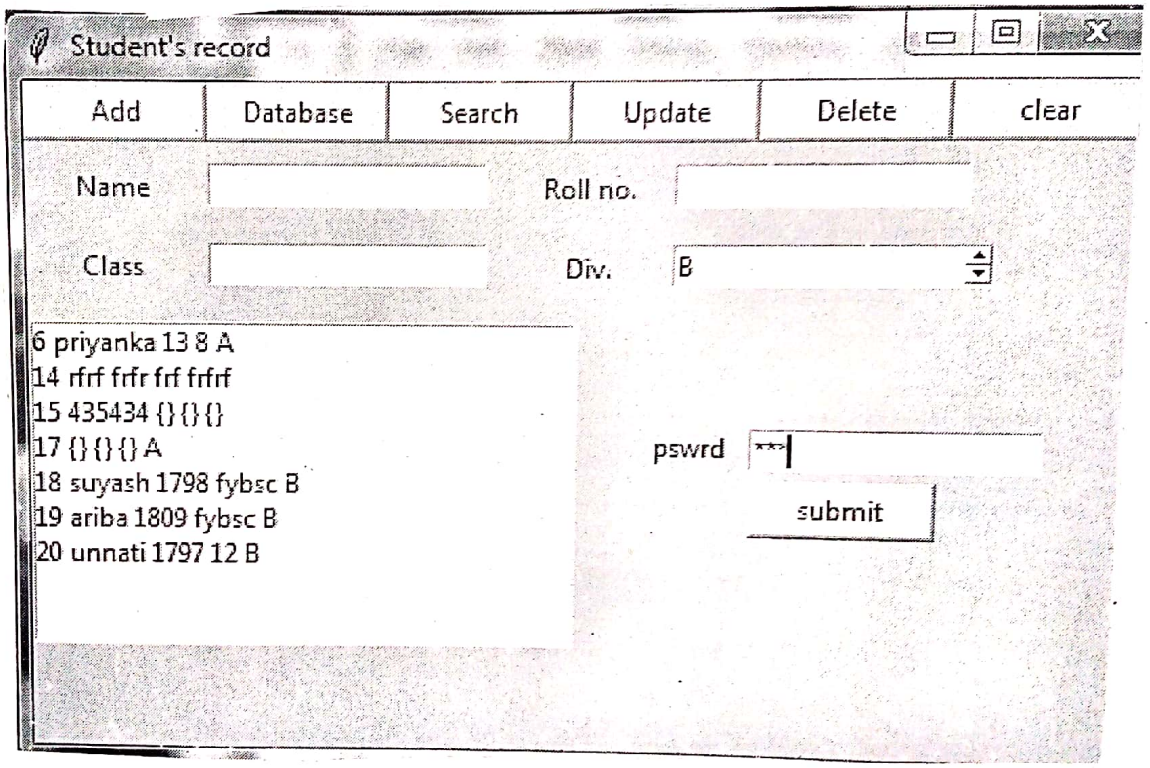




Scanned with CamScanner

```
        Button(window, text="submit", width=10,
command=psswd_chk).place(x=320,y=175)
window=Tk()
window.geometry("480x300")
window.title("Student's record")
Button(window, text="Add",width=10, command=insert).place(x=0,y=0)
Button(window, text="Database",width=10,command=database).place(x=80,y=0)
Button(window, text="Search", width=10,command=search).place(x=160,y=0)
Button(window, text="Update", width=10, command=update).place(x=240,y=0)
Button(window, text="Delete", width=10, command=delete).place(x=320,y=0)
Button(window, text="clear", width=10,command=clear).place(x=400,y=0)
Label(window, text="Name",width=10).place(x=0,y=35)
name= StringVar()
Entry(window, textvariable=name).place(x=80,y=35)
Label(window, text="Roll no.",width=10).place(x=210,y=35)
roll_no=  StringVar()
Entry(window, textvariable=roll_no).place(x=285,y=35)
Label(window, text="Class",width=10).place(x=0,y=70)
class_ = StringVar()
Entry(window, textvariable=class_).place(x=80,y=70)
Label(window, text="Div.",width=10).place(x=210,y=70)
div = StringVar()
e3
=Spinbox(window,textvariable=div,values=('A','B','C','D')).place(x=285,y=
70)
list_ = Listbox(window, height=9, width=40)
list_.place(x=0,y=105)
window.mainloop()
```

Scanned with CamScanner



Scanned with CamScanner