20/11/18

Practical - 1

Aim : Demonstrate the use of different file access modes different attributes read methods.

Algorithm :

Step 1 : Create a file object using open method and use the write access mode followed by writing some contents & then closing file

Step 2 : Open the file in read mode and then use read, readline and readlines method and store the output in variable & finally display the contents of variable.

Step 3 : Now use the fileobject for finding the name of file, and finally the output of the softspace attribute.

Step 4 : 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 the content.

```
fileobj = open ("my.txt", "w")
fileobj.write ("Python is indented language" + "\n")
fileobj.write ("It exhibits \n Interpretative \n form \n")
fileobj.close ()

fileobj = open ("my.txt", "r")   # read ()
str1 = fileobj.read ()
print ("output of read method:", str1)
fileobj.close ()
>>> ('output of read method:', 'Python is indented language \n
    It exhibits \n Interpretative \n form \n')

fileobj = open ("my.txt", "r")   # readline ()
str2 = fileobj.readline ()
Print ("output of readline method:", str2)
fileobj.close ()
>>> ('output of readline method:' 'Python is indented language \n')

fileobj = open ("my.txt" "r")    # readlines ()
str3 = fileobj.readlines ()
Print ("output of readlines method:", str3)
fileobj.close ()
>>> ('output of readlines method:', ['Python is indented language
    \n', 'It exhibits \n', 'Interpretative \n', 'form \n'])

x = fileobj.name                 # File attributes.
Print ("name of file:", x)
>>> ('name of file', 'my.txt')

4 = fileobj.closed
Print ("closed attribute:", 4)
>>> ('closed attribute:', 'True')
```

```
print ("file mode", a)
>>> ('file mode', 'a')

z = filobj.seek(0)
print ("seek(0)", z)
>>> ('seek(0) :', 0)

filobj = open("my.txt", "w+")
filobj.write ("Data types")
filobj.close()

#r+ mode
file obj = open ("my.txt", "r+")
s = filobj.read (4)
print ("output of r+", s)
filobj.close()
>>> ('output of r+', 'Data')

# append mode
filobj = open ("my.txt", "a")
filobj.write ("various")
filobj.close()
filobj = open ("my.txt", "r")
s1 = filobj.read()
print ("output of append:", s1)
file obj.close()
>>> ('output of append:', 'Data types various')
```

```
filobj = open ("my.txt", "w")
filobj.write ("Python")
filobj.close()

# Read mode
filobj = open ("my.txt", "r")
m = filobj.read()
print ("output of read:", m)
>>('output of read:', 'Data type')
```

Step 5 : Open filobj in read mode display the update written content and close. Open again in 'r+' mode with parameter passed and display the output subsequently

Step 6 : Now open filobj in append mode open write method write contents close the filobj again open the filobj in read mode and display the corresponding output

Step 7 :- Open the filedobj in read mode declare variable and perform file obj. Use tell method & store the output consequently in variable

Step 8 :- Use the seek method with the argu with opening the file obj in read mode & closing subsequently

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

```
# Tell()
filedobj = open("may.txt", "r")
cob = filedobj.tell()
print("o/p of tell:", cob)
filedobj.close()
>>> (o/p of tell:, 132)
```

```
# seek()
filedobj = open("may.txt", "r")
s = filedobj.seek(0,0)
print("seek(0,0) is", s)
filedobj.close()
>>> ('seek(0,0) is', 0)
file obj = open("may.txt", "r")
ss = filedobj.seek(0,1)
print("seek(0,1) is", ss)
filedobj.close()
>>> ('seek(0,1) is', 0)
filedobj = open("may.txt", "r")
sss = filedobj.seek(0,2)
print("seek(0,2) is", sss)
filedobj.close()
>>> ('seek(0,2) is', 0)
```


finding length of diffrent lines

```
filedobj = open("may.txt", "r")
str9 = filedobj.readlines()
print("output is", str9)
for line in str9:
    print(len(line))
file obj.close()
>>> Output is(['datatypes vali
19
```

3/12/19

iter() and next()

```
mytuple1 = ("banana", "orange", "apple")
myiter1 = iter(mytuple1)
print(next(myiter1))
myiter2 = iter(mytuple)
print(next(myiter1))
myiter3 = iter(mytuple)
print(next(myiter1)
```

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

for loop

```
mytuple1 = ("Raman", "Daniel", "Black")
for x in mytuple1:
    print(x)
```

```
>>> Raman
    Daniel
    Black
```

square & cube

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

l = [square, cube]
```


Practical - 2

Objective : Iterators

Algorithm :

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 & 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 for loop 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.

Step 5 :- Using for conditional statement specify the range use the list type casting with map method declare a 'lambda' i.e anonymous function & print the same.

Step 6 :- Declare a listnum variable & define some elements then use the map method with help of lambda defination give two argument display the output.

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

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

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

```
for x in range (5):
    value = list (map (lambda x: x(x), (x*x 1))
    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*1.5, listnum))
print (listnum)
def even (x):
    if (x*1.1 == 0):
        return "even"
    else:
        return "odd"
list(map (even, listnum))
```

```
# 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
```

Practical - 3

Topic : Exceptions

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

Step 2 : In except (IO error) block use the error in IO error 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 & print the same message.

Step 6 : Else display the operation is successful!

IO error :

```
try:
    fo = open("may.txt", "w")
    fo.write("Ethical Hacking")
except IOError:
    print("Enter appropriate mode for file operation!")
else:
    print("operation is correct")
>>> operation is correct
```

R output :

```
>>> Enter appropriate mode for file operation!
```

Value Error :

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

>>> Enter a statement: May
    Inappropriate value!

>>> Enter a statement :123
    Operation successful!
```

```
x,y=
y = input("value: ")

try:
    print(x/y)
except ValueError:
    print("Incompatible value")
except ZeroDivisionError:
    print("Denominator is zero")
```

```
>>> enter : 5
Incompatible value
>>> enter : 16
0.25
>>> enter : 0
Denominator is zero
```

Multiline exception :

```
x,y = 1,0
try:
    print(1/0)
    print('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 ~~path~~ division method

Step 8 : Use the exception to be raised use the except keyword (one) i.e TypeError print Incompatible value

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

Step 10 : Declare static variable & value

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

Step 12 : Use try block open a file in write mode & subsequently enter value in the file.

Step 13 : use the I o error and display the appropriate message

Step 14 : Define a function with empty list and calculate the length of list.

Step 15 : define another function & initialise or declare some elements in list and calculate the length of the same & display the same.

Step 16 : In try block accept input from the user and if the user enter character value raise an error that is saying up enter integer values.

Using Except keyword

```
try:
    x = open("my.txt", "w")
    x.write("Hacking")
except IO error:
    Print("Error!")
else:
    Print("Successful")

def a():
    l = []
    Print(len(l))
def b():
    li = [2,4,4,1]
    Print(len(li))
Print(x())
Print(y())
```

output: Successful
0
None
4
None

Raise Keyword:

```
try:
  x = int(input("Enter a number"))
  raise value error
except value error:
  Print("Enter integer value!")
```

>>> Enter: May
Enter integer value!

Match ()

```
import re
Pattern = r"Brock"
sequence = "Brock is the champion"
if re.match (pattern, sequence):
        Print ("Pattern found")
else:
        Print ("NOT found")
>>> Pattern found
```

Numerical Value (Segregation)

```
import re
Pattern = r'\d+'
String = "MSD 007, Brock869, bobby 456'
output = re.findall (pattern, string)
Print (output)
>>> ['007', '869', '456']
```

Split ()

```
import re
pattern = r'\d+'
String = "MSD 007, Brock 869, bobby 456'
Output = re.split (pattern, string)
Print (output)
>>> ['MSD', 'Brock', 'bobby']
```

Practical : 4

Topic : Regular Expression

Step 1 : Import re module, declare sequence and pattern and then use match method with declare argument, if argument matched then print the same otherwise print pattern not found.

Step 2 : Import re module, declare pattern literal & meta character, declare string value using the argument & print the same.

Step 3 : Import re module, declare meta character use the split () & print the output.

Step 4 : import re module declare string & accordingly declare pattern replace the blank space with no. space Use sub() with 3 arguments and print the string without spaces.

Step 5 : import re module declare a sequence use search method for finding substring use the group() with dot operator as search() finds many location using group() it will show up the matched string.

Step 6 :- import re module declare list with number use the conditional statement how we have used up for condition statement use it condition for checking first number is either 8 or 9 & the next number nine numbers are in range of 0-9 & check whether the entered number are equal to 10. If the above criteria matches print (all number matches otherwise print fail.

```
# no - space
import re
seq = "may week have"
pattern = r"\s+"
replace = ""
output = re.sub(pattern, replace, seq)
print(output)

>>> mayweekhave
# group()
import re
str = "line is platinum"
out = re.search('IA line', str)
print(out)
output = out.group()
print(output)
>>> <_sre.SRE_Match object at 0x02810F00>
    line
# validating the given set of phone numbers
import re
list1 = ["9090343414", "8849649961", "9999666333"]
for value in list1:
    if re.match(r'[8-9][0-9]{9}', value) and
                len(value) == 10:
        print("all no. match")
    else:
        print("no. not match")
```

```
>>> all no match
    no no match
    no no match

# vowels
import re
txt = "All is well"
output = re.findall (r'\b[aeiouAEIOU]\w+', txt)
print (output)
>>> ['All', 'is']

# host & domain
import re
txt = 'abc.test@edu.com, xyz@gmail.com'
pattern = r'[\w.-]+[\w.-]'

output = re.findall (pattern, txt)
print (output)

>>> ['abc.test', 'edu.com', 'xyz', 'gmail.com']
```

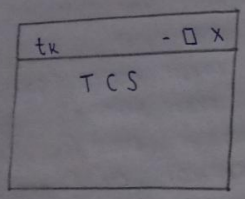
step 7 : import re module
string
findall () for
vowels in the
during the

step 8 : import re
host and
pattern for
& domain
findall ()
respectively

creating a

```
from tkinter import *
root = Tk()
l = Label(root, text = "TCS")
l.pack()
root.mainloop()
```

output:



#2:

```
from tkinter import *
root = Tk()
l = Label(root, text = "TCSC")
l.pack()
l1 = Label(root, text = "CEH", bg = "Red", fg = "Black",
    font = "10")
l1.pack(side = LEFT, padx = 20)
l2 = Label(root, text = "TCS", bg = "Red", fg = "Black",
    font = "20")
l2.pack(side = LEFT, pady = "30")
l3 = Label(root, text = "TCSC", bg = "Red", fg = "Black",
    font = "10")
l3.pack(side = Top, ipadx = 40)
```


Topic : GUI components

Step 1: Use the Tkinter library for importing the features of the text widget.

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 words.

#2:

Step 1: Use the Tkinter library for importing the features of the text widget.

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

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 , pad y = 30

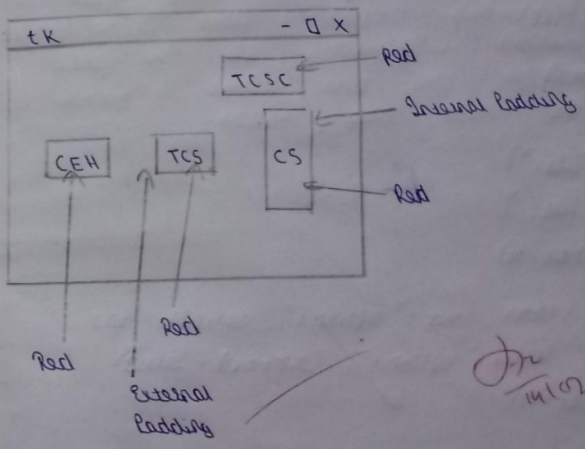
3) side = TOP , ipad x = 40
4) side = TOP , ipad y = 50

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

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

```
l4 = Label (root, text = "CS", bg = "Red", fg = "Black",
     (font = "ARIAL BOLD", 10))
l4.pack (side = TOP, ipady = 50)
root.mainloop()
```

Output :



Radio Button

```
from Tkinter import *
root = Tk()
root.geometry("500x500")
def select():
    selection = "You just selected" + str(var.get())
    l1 = label(root, text = selection, bg = "white", fg = "red")
    l1.pack(Top)

var = StringVar()
l1 = Listbox()
l1.insert(1,"list1")
l2.insert(2,"list2")
l1.pack(anchor=N)
r1 = Radiobutton(root, text = "option1", variable = var,
                 value = "option1", command = select)
r1.pack(anchor=N)
r2 = Radio Button(root, text = "option2", variable = var,
                  value = "option2", command = select)
r2.pack(anchor=N)
root.mainloop()
```

Practical - 5 (B) 35

Aim : GUI components

#1 : (Radio Button)

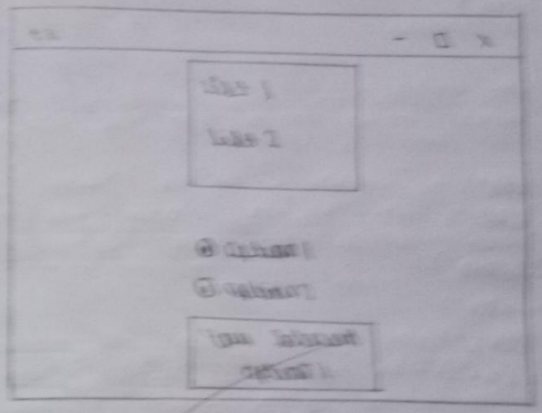
Step 1 : Use Tkinter module to import relevant method

Step 2 : Define a function which tells the user about the given selection made from the multiple options available

Step 3 : Use the config() along with label object & (all the variable as an argument within the method

Step 4 : Now define parent window & define the option using control variable

Step 5 : Create a object from radio button which will take following argument 1] Positioning on the parent window 2] Defining the text variable which will take the value option 1, 2, etc. 3] Define the variable argument as (selection) value & display the given function



24-1-2024

Practical - 5 (c)

Aim : GUI Components

#1 (Message Box)

Step 1 : Import relevant method from Tkinter library

Step 2 : Define a function & use the Message box along with different methods available which contain 1 or more argument

Step 3 : The different option which are available are 1] Show_Info , 2] Warning 3] Show error 4] Ask Yes/No 5] Ask question 6] Ask ok/cancel

Step 4 : Create an object from button method & place it onto the parent window with the title of button specified & corresponding event called for triggering

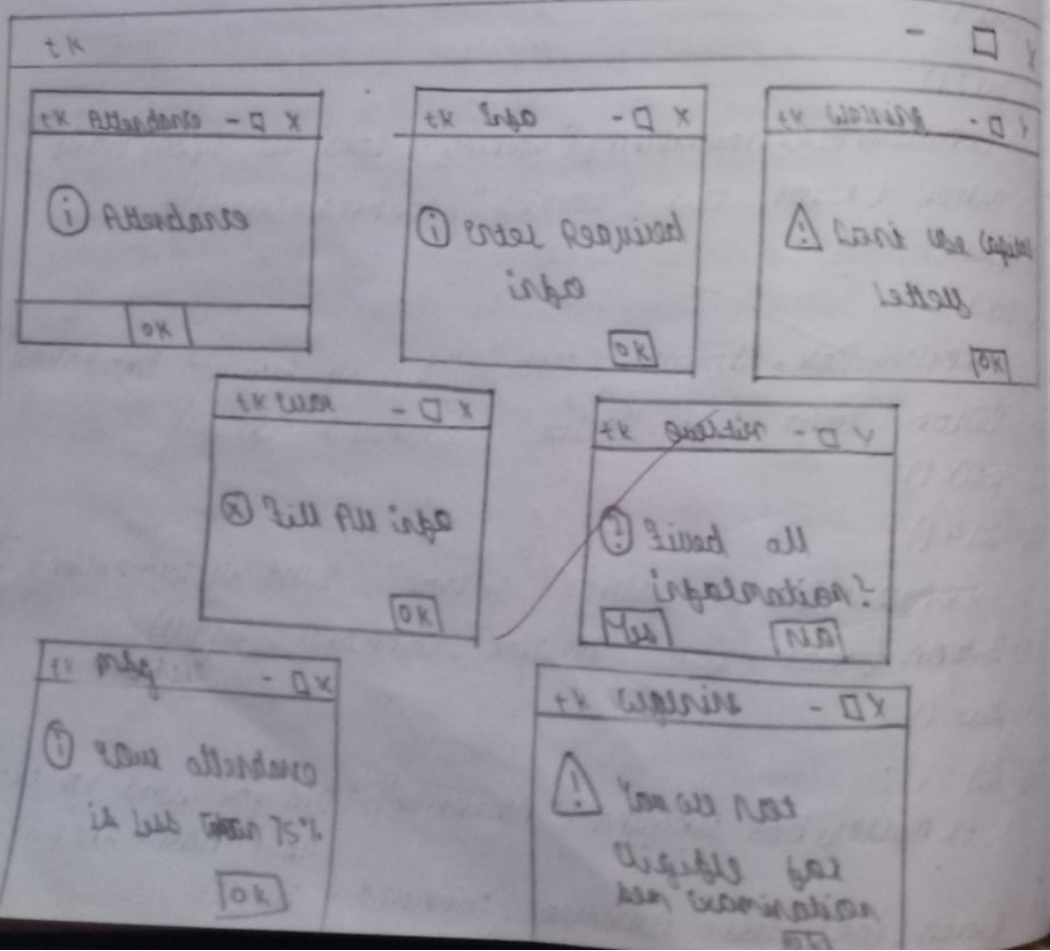
Step 5 : Use the pack method to display the button widget and finally use main loop method.

#1 (Message Box)


```
from Tkinter import *
import tkMessageBox
root = Tk()
def Sol ():
    tkMessageBox.showinfo ("Attendance", "Login")
b = Button (root, text = "Attendance", command = Sol ())
b.pack()
def Sol1 ():
    tkMessageBox.showinfo ("Info", "Enter Required Information")
b1 = Button (root, text = "Info", command = Sol1 ())
b1.pack ()
def Sol2 ():
    tkMessageBox.showwarning ("Warning", "Don't use Capital letters")
b2 = Button (root, text = "Warning", command = Sol2 ())
b2.pack()
def Sol3 ():
    tkMessageBox.showerror ("Error", "Fill all Required Information")
b3 = Button (root, text = "Error", command = Sol3 ())
b3.pack ()
def Sol4 ():
    tkMessageBox.askquestion ("Question", "Filled all information?")
b4 = Button (root , text = "Question", command = Sol4 ())
b4.pack ()
def Sol5 ():
    tkMessageBox.askyesno ("Attendance", "Did you want to che[illegible]
                                      your Attendance ....")
b5 = Button (root , text = "Attendance", command = Sol5 ())
```

```
b5.pack()

def sel6():
    tkMessageBox.showinfo("msg", "your attendance is less than 75%")
b6 = Button(root, text="msg", command = sel6())
b6.pack()
def sel7():
    tk.messagebox.showwarning("warning","you are not eligible for
                                        sem Examination")
b7 = Button(root, text="warning", command = sel7())
b7.pack()
root.mainloop()
```



step 6 :- If the user want to hide the parent window & only the info window show visible corresponding to the user given option above, withdraw method is use

(Relief style for button widget)

step 1 :- Import relevant method from Tkinter library

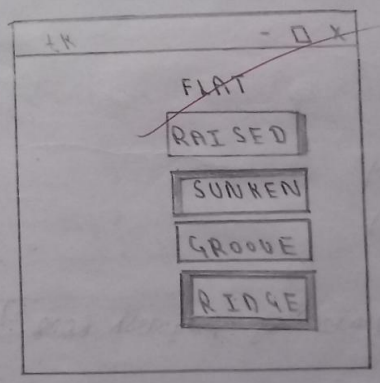
step 2 :- Use the button object with the following attribute 1] Parent window
2] Text attribute
3] Relief

Step 3 :- Use the corresponding pack method for the respective button object.

Step 4 :- Finally used the mainloop method to triggred the corresponding event.

Relief style of button

```
from TKinter import*
root = TK()
B1 = Button (root, text=" FLAT ", relief = FLAT)
B2 = Button (root, text="RAISED", relief = RAISED)
B3 = Button (root, text=" SUNKEN", relief = SUNKEN)
B4 = Button ( root, text="GROOVE", relief = GROOVE)
B5 = Button (root, text="RIDGE", relief = RIDGE)
B1. pack()
B2. pack()
B3. pack()
B4. pack()
B5. pack()
root. mainloop()
```



```
from Tkinter import *
win = Tk()
win.title("Tech")
win.maxsize(300,300)
win.config()
leftframe = Frame(win, bg="Red", height=20, width=20)
leftframe.grid(row=0, column=0, ipadx=10, ipady=20)
Rightframe = Frame(win, bg="Black", height=20, width=20).grid
        (row=0, column=1)
l = Label(win, text="Sports", relief=RAIS).grid
        (row=0, column=0)
image = PhotoImage(file="graph.gif")
ai = image.subsample(10,10)
Label(leftframe, image=a.i).grid(row=0, column=0)
Label(Rightframe, image=image).grid(row=0, column=1)
toolbar = Frame(leftframe, width=20, height=18)
toolbar.grid(row=2, column=0, padx=5, pady=5)
def name():
    Print("Enter your name")
        Iam
def hobbie():
    Print("Enter your hobbies")
            writing
```


Aim : GUI (arguments)

Create Image

Step 1 :- Create an object of tkinter to pass window and use the following 3 methods
1) Title 2) Maxsize 3) Config

Step 2 :- Create a leftframe object from the frame method & place it onto parent window with the height, width & background attribute specified subsequently use the grid method with Row, Column, padX, padY attribute specified

Step 3 :- Now create a Rightframe obj from frame() with the width and height is specified & the row & the column value should be specified.

Step 4 :- Create a label object from label method & place it onto left frame with the text attribute denoting the original image with relief attribute chosen with raised value and subsequently used grid with row, column value specified as 0,0 with some extra padding value.

12.

Step 5 :- now use the photo image method with the file attribute specified

Step 6 :- use the subsample () with the object of the image and give the x & y (co-ordinate value.

Step 7 :- use the label () and position it on left frame & placing the image after the sampling & use the grid method for positioning in the first row.

Step 8 :- Create another label object positioning onto right frame & specifying image & background attribute with row & column attribute specified by 0,0 :-

Step 9 :- Now create toolbar object from frame method & position onto left frame with height & width specified & position it onto the 2 row.

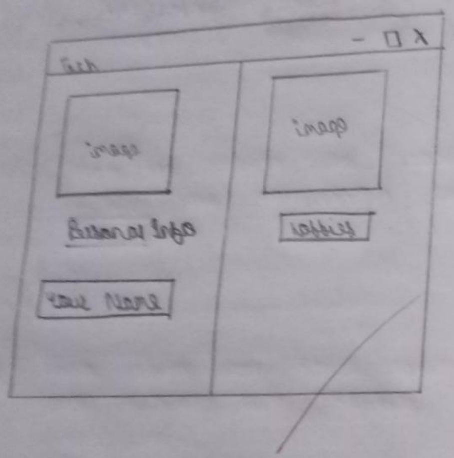
Step 10 :- Define various function for different toolbar option provided in the left fr

```
l = label ( toolbar , text = "Personal info" , relief = RAISED ).grid
    ( row = 0 , column = 0 , padx = 5 , pady = 5)

B1 = Button ( leftframe , text = "Enter new Name" , command = Name ) . grid
    ( row = 0 , column = 0 )

B2 = Button (Rightframe , text = "Existing Hobbies" , command = hobbies ) . grid
    ( row = 0 , column = 1 )

win. mainloop
```



Step 11 : From label method position text on toolbar use the sticky attribute & corresponding grid value and incorporate internal padding as well

Step 12 : Create the label method position it onto toolbar with next title and position it on to the same row but next column.

Practical 5

Aim : GUI components

Spinbox

Step 1 : Create an object from tK method & subsequently create an object from Spinbox method.

Step 2 : Make the object to created and pass window & trigger the corresponding query

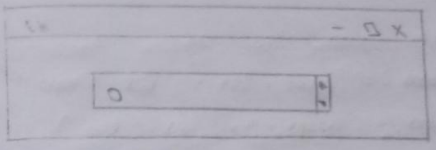
Panedwindow :-

Step 1 : Create object from Panedwindow & use the pack method with attribute fill and expand

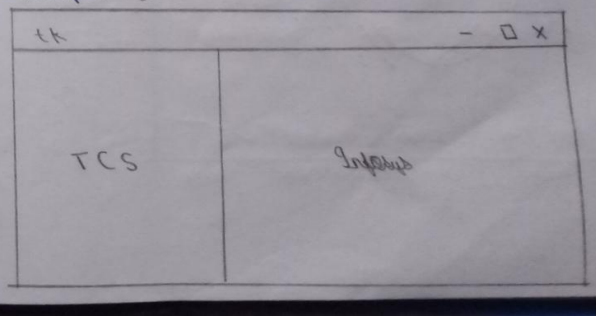
Step 2 : Create an object from Label method & put it on to the panedwindow with the text attribute & use the add method to add the new object.

Step 3 : Similarly create a second paned window object & add it on to the 1st paned window with orientation specified.

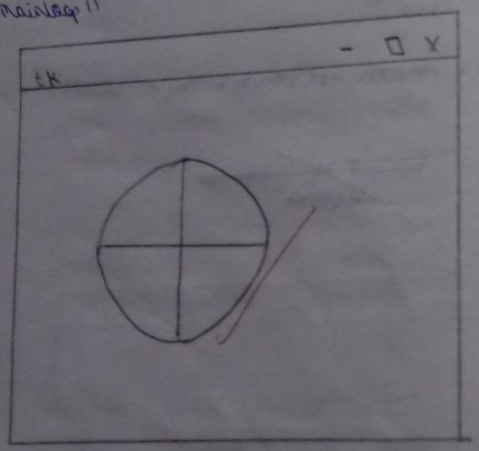
```
from Tkinter import *
root = Tk()
S1 = Spinbox(root, from_ = 0, to = 5)
S1.pack()
root.mainloop()
```



```
from tKinter import *
root = Tk()
P = Panedwindow()
P.pack(fill = both, expand = 1)
l = Label(P, text = "PYTHON")
                    TCS
P.add(l)
P1 = Panedwindow(P, orient = VERTICAL, bg = "black")
P.add(P1)
l1 = label(P1, text = "Programming language")
                        Tutorials
P1.add(l1)
root.mainloop()
```



```
from tkinter import *
root = Tk()
C = Canvas(root, height=900, width=1440, bg="white")
co1 = 100, 800, 800, 100
co2 = 50, 825, 825, 50
C.create_arc(co2, start=0, extent=359.9, fill="black")
C.create_arc(co1, start=0, extent=90, fill="blue")
C.create_arc(co1, start=90, extent=90, fill="white")
C.create_arc(co1, start=180, extent=90, fill="blue")
C.create_arc(co1, start=270, extent=90, fill="white")
C.pack()
root.mainloop()
```



step 4: Now create another label object & place it on to the second panel object & add it onto the 2nd panel

step 5: Finally mainloop method

Canvas widget

step 1: Create object from Canvas() and use the attribute height, width, bg & parent window

step 2: Use the method to create line, oval, arc along with canvas object created & the co-ordinates values

step 3: Finally use the last method & call the pack method the mainloop method

15-07-19

Practical - 6

Aim : Database

Step 1 : Import DBM library for making database connection with it os & sqlite3

Step 2 : now create the connection object using sqlite3 library & the correct method for creating the new database.

Step 3 : now create cursor object using cursor method from the connection object created in earlier steps.

Step 4 : now use the execute method for creating table with column name & respective datatype.

Step 5 : now with cursor object use insert statement for storing the value corresponding to different fields considering datatype.

Step 6 : use the commit method to complete the transaction using the connection object.

Code

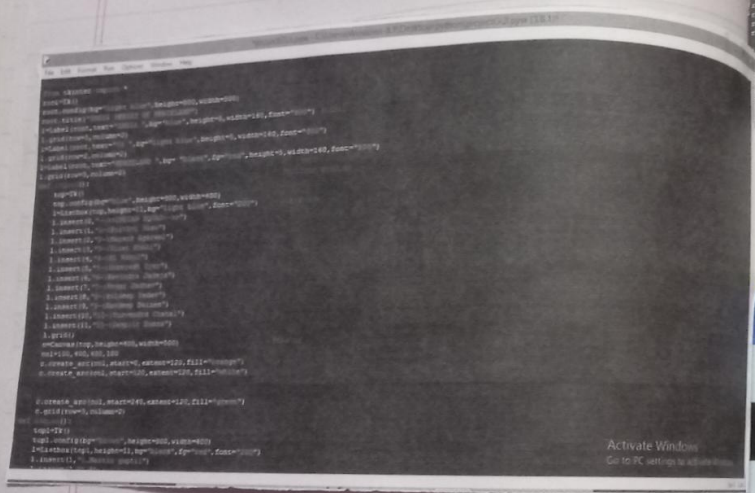
```
import os, sqlite3
connection = sqlite3.connect("Data.db")
cursor1 = connection.cursor()
cursor1.execute("create table menu(menu char(10), Rate int(5))
cursor1.execute("insert into menu values('Rice', 100),
                ('Idli', 110)")
connection.commit()
cursor1.execute('select * from menu')
print(cursor.fetchall())
connection.close()
```

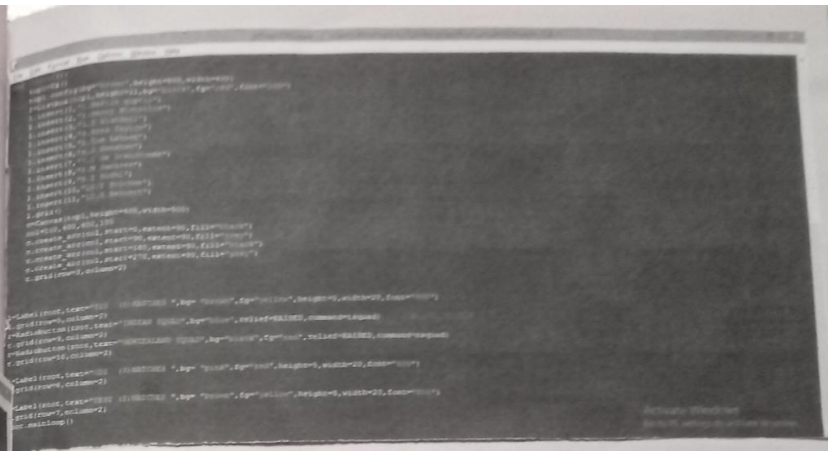
output :

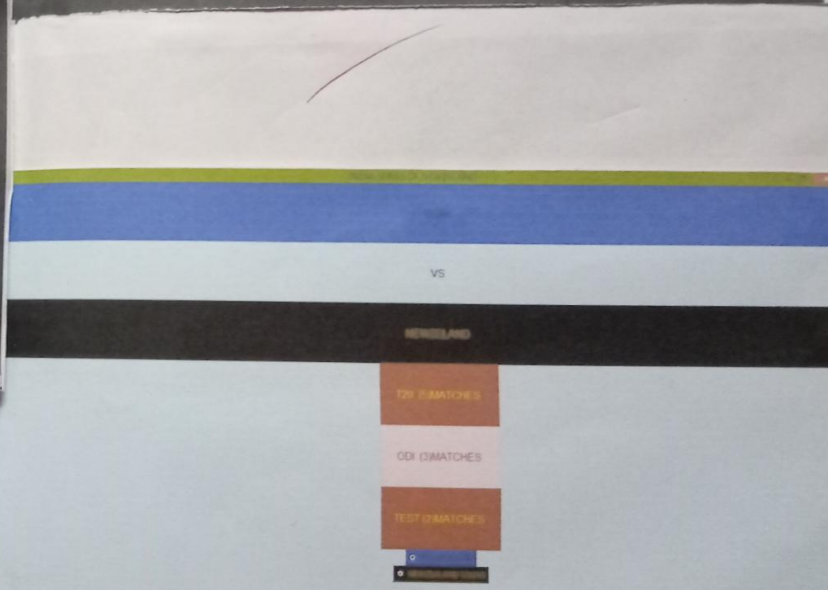
```
[('Rice', 100), ('Idli', 110)]
```

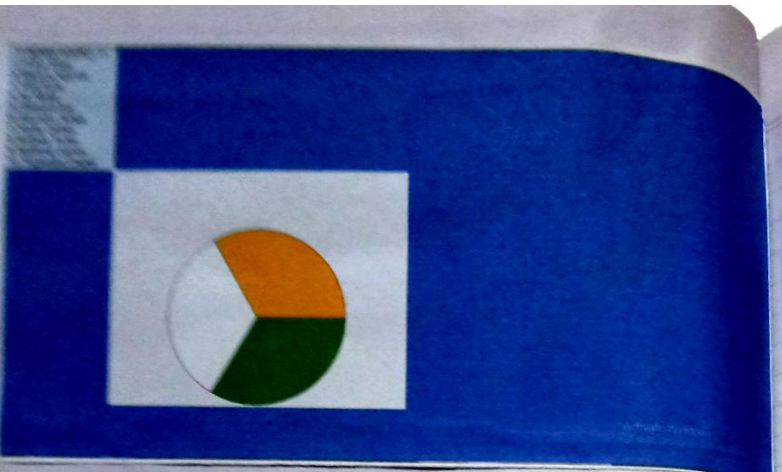
* Project X

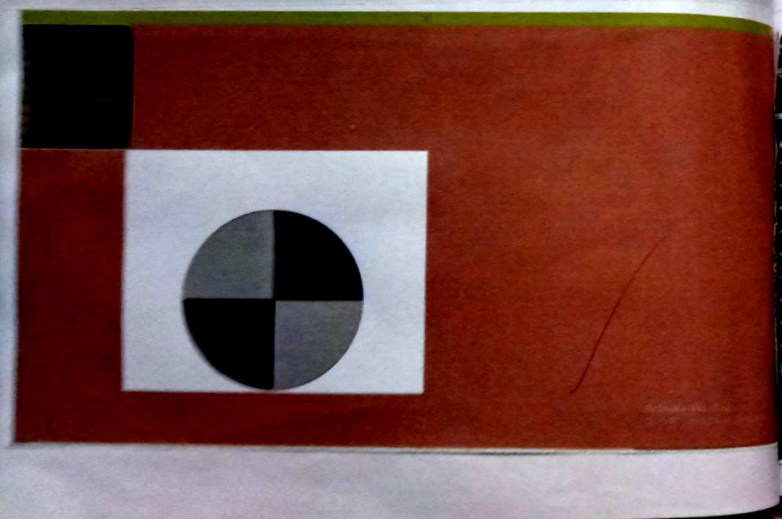
* GUI Components :

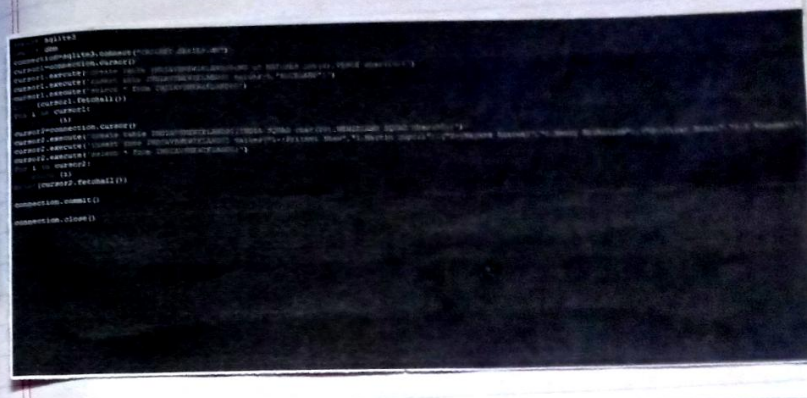












```
hon 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 22:39:24) [MSC v.1916
e "help", "copyright", "credits" or "license()" for more informatio

==== RESTART: C:\Users\Windows-8.1\Desktop\python\projectdatabase.
, 'AUCKLAND')]
-:Prithvi Shaw', '1.Martin guptil')
-:Mayank Agarwal', '2.Henry Nicholles')
-:Virat Kohli', '3.T Blundell')
-:KL Rahul', '4.Ross Taylor')
-:Shreyash Iyer', '5.Tom Latham')
-:Ravindra Jadeja', '6.J Neesham')
-:Kedar Jadhav', '7.C de Grandhomme')
-:kuldeep Yadav', '8.M Santner')
-:Navdeep Sainee', '9.I Sodhi')
0-:Yuzvendra Chahal', '10.T Southee')
1-:Jasprit Bumra', '11.H Bennett')
```