Practical - 1

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

Step 1 :- Create a file object using open method and use the write access mode followed by writing some contents onto the file and than closing the file.

Step 2 :- Now open the file in read mode and than use read(), readline() and readlines() and store the output in vairable and finally display the contents of vairable.

Step 3 :- Now use the file object for finding the name of the file, the file mode in which it is opened whether the file is still open or close and finally the output of the soft space attribute

step4: Now open the file obj in write mode. write some another content close subsequently then again open the file obj in 'w+' mode that is the update modes and write contents.

step5: Open file obj is 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 file obj in append mode open write method write contents ~~close write method writer~~ to close the file obj again open the file obj in read mode and display the 'appending' output.

```
file obj = open("abc.txt", "w") # file open (write mode)
file obj .write("Subjects in FYJC science" + "\n")
file obj .write("Physics\n chemistry\n maths\n") # file write
fileobj .close() # file close
file obj = open("abc.txt", "r") # read mode
str1 = fileobj .read()
print = ("The output of read method=", str1)
print ("The output of read
fileobj .close()
>>> ("The outputs of read method:", 'subjects in FYJC science
    \n Physics\n Chemistry\n maths\n')

file obj = open("abc.txt", "r")
    str2 = file obj .readlines()
  print("The output of readlines method:", str2)
file obj .close()
>>> ['The output of readline method', 'Subjects in
    FYJC Science\n']
file obj = open("abc.txt", "r")
str3 = file obj .readline()
print("The output of readline method:", str3)
file obj .close()
>>> ('The output of readline method:', ['Subjects.
    in FYJC science', 'Physics\n', 'Chemistry\n', 'Maths\n'])
a = file obj .name
print("name of line (name.attribute):", a)
>>> (name of file (name attribute)', 'abc.txt')
b = file obj .closed
print("(close) attribute:", b)
>>> ('(close) attribute:', 'True')
```

```
c=file obj . mode
Print ("file mode", c)
>>> ("file mode, 'r')
d=file obj . softspace
print ("softspace", d)
>>> ("softspace :', 0)
```

```
# w+ mode
file obj = open ("abc.txt", "w+")
file obj . write (" Python")
file obj . close ()
```

```
file obj = open ("abc.txt", "w")
file obj . write ("stats")
file obj . close ()
```

```
# read mode
file obj = open ("abc.txt", "r")
S=file obj . read ()
print ("output of read
    mode!", S)
>>> ('output of read
    mode!', 'Python')
```

```
# r+ mode
file obj = open ("obc.txt", "r")
S1 = file obj . read (6)
Print ("obtput of r+", S1)
file obj . close ()
>>> ('output of r+', 'stats')
```

```
# append mode
file obj = open ("abc.txt", "a")
file obj = write ("data structure")
file obj . close ():
file obj = open ("abc.txt", "r")
S3 = file obj . read ()
Print ("output of appent mode:", S3)
file obj . cbse ()
>>> ('output of append mode :', 'Python data structure')
```

Step 7:- Open the file obj in read mode, declare a vairable and perform file object dot tell method and store the output consequently in vairable.

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

Step 9: Open file obj 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

```
# tell()
fileobj = open("abc.txt", "r")
pos = fileobj.tell()
print("tell():", pos)
fileobj.close()
>>> ('tell():', pos)
# seek()
fileobj = open("abc.txt", "r")
st = fileobj.seek(0,0)
print("seek(0,0) is:", st)
fileobj.close()
>>> ('seek(0,0) is:', none)
fileobj = open("abc.txt", "r")
S1 = fileobj.seek(0, 1)
print("seek(0, 1) is:", S1)
fileobj.close()
>>> ('seek(0, 1) is:', none)
fileobj = open("abc.txt", "r")
S2 = fileobj.seek(0, 2)
print("seek(0, 2) is:, S2)
fileobj -close()
>>>('seek(0,2) is:', none)
# finding length of different lines exists within lines.
fileobj = open("abc.txt", "r")
S = fileobj.readlines()
print("output:", S)
for line in S:
print(len(line))
fileobj.close()
>>> ('output:', ['Python data structure'])
```

Practical -2

① Iterators

Step 1 :- Define a iter method with an argument and initialize it to the 1st value

Step 2 :- For extracting the next element from the container, use the next method with an argument and compare the no. of elements required in a container by using the Conditional statement.

Step 3 : Now Create an object from the given class and pass this object as an argument to the iter method.

Step 4 :- Now using the conditional statement display all the values from the given Container.

② Step 5 :- Power Iter

Step 5 :- Define a function name Square with a parameter which will obtain output of square values of the given number.

Step 6 : Call the declared function using Function Call.

55

```
Class myrange:
    def __iter__(self):
        self.a = 1
        return self
    def __next__(self):
        if self.a < 30:
            x = self.a
            self.a += 1
            return x
        else:
            raise StopIteration
myclass = myrange()
mydata = iter(myclass)
for x in mydata:
    print(x)
```

Output

1	16	30
2	17	
3	18	
4	19	
5	20	
6	21	
7	22	
8	23	
9	24	
10	25	
11	26	
12	27	
13	28	
14	29	
15		

Step 7 :- Declare a list num vairable and declare one element then use the map

Step 7 :- For extracting next value from container use the next method with an argumen and compare the container by using conditional statement.

Step 8 :- Now create an object from the given class and pass this object as an Argument to the iter method.

Step 9 :- Now using the conditional statement display all the values from the given container.

3) Factorial

Step 10 - Define an object of a class

Step 11 - Accept an number from the user till which we want to display the factorial of the number

Step 12 - Define a function with if conditional, if exactly input is exactly equal to 1 then return one else parameters.

```
class myiter:
    def __iter__(pow):
        pow.no =1
        pow.nos = int(input("Enter a number:"))
        return pow
    def __next__(pow):
        if pow.no<=10:
            num = pow.nos ** pow.no
            pow.no +=1
            return num
        else:
            raise StopIteration

x=iter(myiter())
while True:
    print(next(x))
```

Output

```
Enter a number : 3
3
9
27
81
243
729
2187
6561
19683
59049

raise StopIteration
StopIteration.
```

25

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

```
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", m.value, "is", facto)
        m.value += 1
        if (m.value == 7):
            raise StopIteration
y = iter(fact())
while True:
    next(y)
```

Output

~~Enter a number:~~

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

Step 13 :- Enter a input from the declare an empty list use append method, for appending the input value.

Step 14 :- Print the output vairable.

Practical - 3

Topic :- Exceptions & assertion.

Steps

1. In the try block open on file in the Open mode with write mode write some content in file.
2. In except (IO Error) block use the error in IO Error use the appropriate message to display.
3. Else display the Operation is Successful!
4. In try block ~~use Value er~~ - Accept an input from the user.
5. In except block use Value error and print Some message.
6. Else display the Operation is Successful!

IO Error :

```
try:
    fo = Open ("abc.txt", "w")        # R
    fo.write (" python is an intendent language ")
except IoError:
    print (" Enter Oppropriate mode file operation !, ")
else:
    print (" Operation is Successfull ")
```

>>> Operation is Successfull

Value Error :

```
try:
    x = int (input (" Enter a statement : "))
except Value Error:
    Print (" Arithmatic Error ! ")
else:
    print (" Operation is Successful ! ")
```

>>> Enter a ~~set~~ statement : abc
Arithmetic Error !

>>> Enter a statement : 123
Operation is Successful !

Type error, zero division error

```
a=1
b=input("Enter:")
try:
    print(a/b)
except TypeError:
    print("Incompatible value")
except zeroDisision Error:
    print("Denominator is zero so operation
        cannot be performed")
```

```
>>> Enter:2
    0.5
>>> Enter:0
    Denominator is zero so operation cannot
     be performed.
```

Multiline exception.

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

```
>>> 1.0
>>> 10
>>> Invalid Input!
```

7. Accept an integer Value from the user. In the try use the division method.

8. For the exception to be raised use the except keyword (ord) ; e type error print In Compatible. Value.

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

10. Declare static Variables & Values

11. For Multiple exception use the error type by seporating them with Comma.

12. Use try block open a file in write mode & & Subesequently enter value in the file.

13. Use the IO Error & display appropriate message.

14. Define a function with empty list & Calculate the length of the lists.

15 Define another function if initialise or declare some elements in list & Calculate the length of some & display the Some.

16. In try block accept input from the user & if the user enters character values raise an error that is saying up enter integer values.

Using except Keyword

```
try:
    a = open ("abc.txt", "w")
    a.write("python")
except IOError:
        print ("Error!.")
else:
    print (" successful ")
def x();
        l = [ ]
        print (len (l) )
def y();
        li = [2, 4, 4, 1 ]
        print (len (li))
print (x())
print (y())
```

Output: Successful
0
None
4
None.

raise Keyword

```
try:
    a = int (input(" Enter a number:"))
    raise ValueError
except ValueError:
        print(" Enter integer value !")
```

>>> Enter: xyz
Enter integer value !

Practical - 4

Topic : Regular Expression

Step 1 : Import re module declare pattern and declare sequence use match method with declare arguments if arguments matched than 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.

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 values (segregation)

```
import re
pattern = r'\d+'
String = 'hellow 123, howdy 789, 45 how ru'
output = re.findall(pattern, string)
print(output)
```

>>> ['123', '789', '45']

split()

```
import re
pattern = r'\d+'
String = "hellow 123, howdy 789, 45 how r u'
Output = re.split(pattern, String)
Print(output)
```

>>> ['hello', 'howdy', 'how ru']

es

Step 4: 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.

Step 5: import re module declare a sequence use search method for finding subsequently use the group() with dot operator as search() gives memory ~~to~~ 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 ~~step~~ up the for condition statement. use if condition for checking first number is either 8 or 9 and next 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 an interesting language'
V = re.search ('\Apython', sequence)
print (v)
V1 = V.group()
print (v1)
>>> <_sre.SRE_Match object at 0x02810F00 >
    Python
```

verifying the given set of phone numbers #.

```
Import re
list 1 = ['8004567891', '9145673210', '7865432981',
          '9876543201']
for value in list 1:
        if re.match(r'[8-9]{1}[0-9]{9}', value or len
                                          (value)== 10):
                print("Criteria matched!")
        else:
                print ("Criteria Failed")
```

>>> criteria matched ~~for cell number~~

Criteria matched f

Criteria failed !

Criteria matched

```
# vowels
import re
str1 = 'plant is life overall'
output = re.findall (r'\b[aeiou AEIOU]\w+', str1)
print (output)
>>> ['is', 'overall']
```

```
# 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.
importre
s = 'mr.a ;ms.b, ms.c, mr.t'
p = r'[/ms/mr/.]+
o = re.findall (p, s)
print (o)
m = 0
f = 0
for v in o:
    if (v == 'ms'):
        f = f + 1
    else :
```

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 ~~partition~~ particular string use ~~findall~~ findall() declare two vairables 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.

16/07

```
        m=m+1
print ("No. of males is : " m)
print ("No. of females is :", f)
>>> ['m', 'm', 'm', 'm']
    ('No. of males is :', 2)
    ('No. of females is :', 2)
```

16/05

Practical - 5

Topic : Gui Components.

1) Step 1 : Use the tk inter library for importing the features of the text widget.

Step 2 : Create an object using the Tk()

Step 3 : Create a vairable using the widget Label and use the text method.

Step 4 :- Use the mainloop() for triggering of the coresponding about mention events

2:

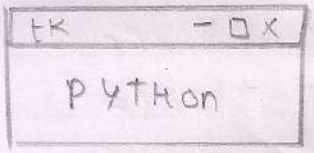
Step 1 : Use the tk inter library for importing the features of the text widget

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

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"
l4. pack (side = TOP, ipady = 30)
root. mainloop ()
```

Step 3: Use the pack() along with the object created from the text () and use the parameter

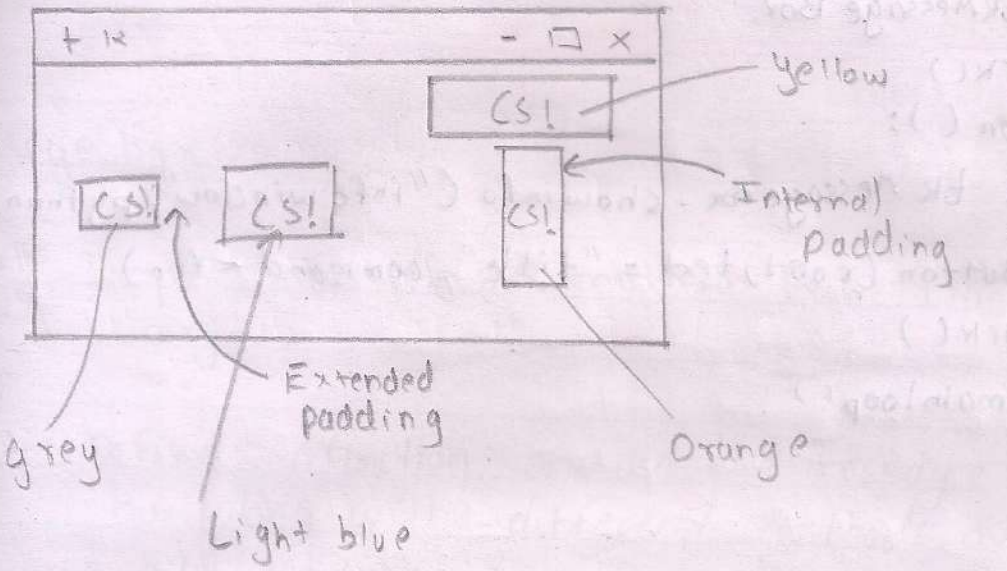
1) side = LEFT, padx = 20
2) side = LEFT, pady = 30
3) side = TOP, ipadx = 40
4) side = TOP, ipady = 50

Step 4: Use the mainloop() 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 than use the pack() with a relevant padding attributes.

Output :


CSS
CSS
CSS
CSS
Yellow
Internal padding
Extended padding
grey
Light blue
Orange

6) Message box

Step 1: Import the relevant method from tkinter library.

Step 2: Define a function and use the message box along with different methods available which contains one or more arguments

Step 3: Thus different options which are available are showinfo(), showwarning(), showerror() askyesno(), askquestion(), askokcancel()

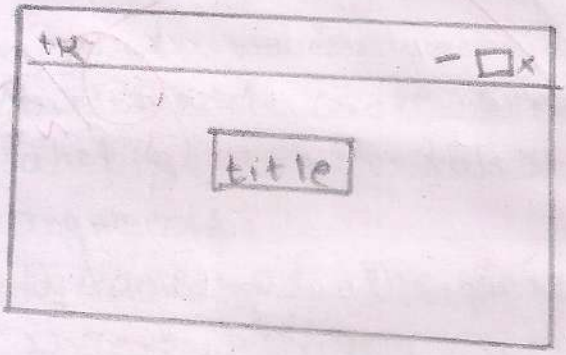
Step 4: Create object from button method and place it onto the parent window with the title of the corresponding event called for triggering.

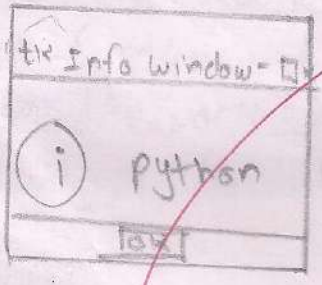
Step 5: Use the pack method to display the button widget and finally use mainloop method.

Step 6: If the user wants to hide the parent window and only the info window should be visible corresponding to the six options given above the withdraw method is used.

```
from Tkinter import *
import tkMessage Box
root = TK()
def fun ():
        tk Message Box . showinfo ("info window", python )
b1 = Button (root, text = "title", command = fun)
b1.pack()
root.mainloop()
```

Output





c) Relief style.

Step1: Use the button object with the following attributes.

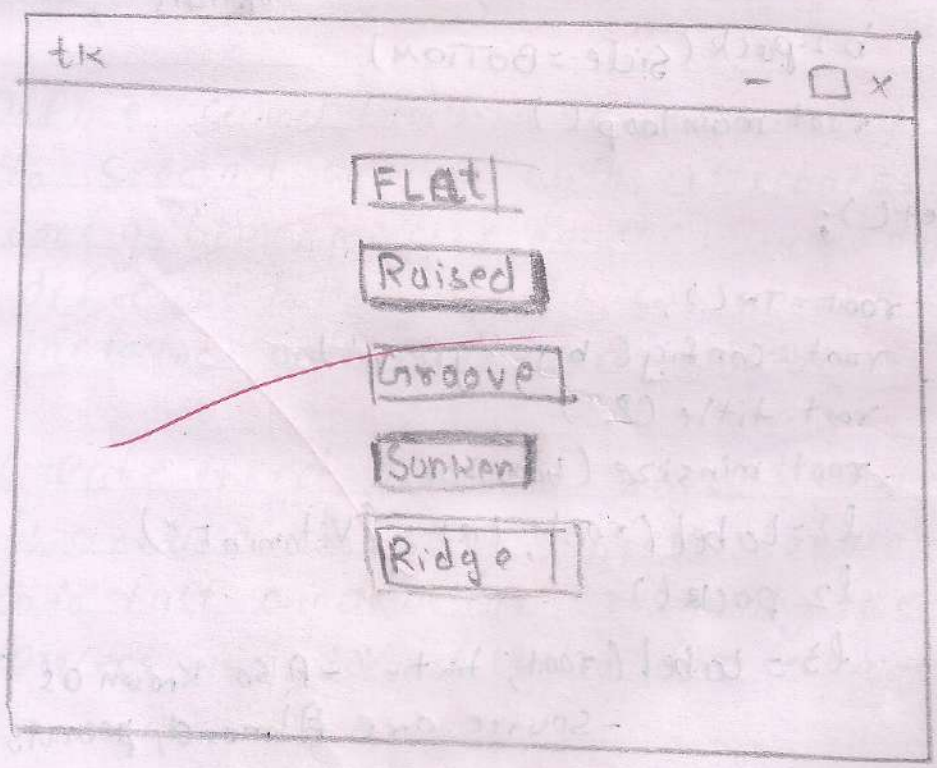
1. The parent window
2. Text attribute
3. Relief

Step2: Use the corresponding pack method for the respective button objects and trigger the corresponding event.

Step3: Finally use the mainloop method.

```
from Tkinter import *
root = Tk()
b1 = Button(root, text = "Flat", relief = FLAT)
b1.pack()
b2 = Button(root, text = "Raised", relief = RAISED)
b2.pack()
b3 = Button(root, text = "Groove", relief = GROOVE)
b3.pack()
b4 = Button(root, text = "Sunken", relief = SUNKEN)
b4.pack()
b5 = Button(root, text = "Ridge", relief = RIDGE)
b5.pack()
root.mainloop()
```

Output



D) Traversing.

Step1: Define a function and create a objet of thegiven window by using the three methods namely config. title and minsize

Step 2: Create a button object and use the text and commanc attribute for triggering the given event and used grid method abng with internal and external padding. Specified. and create anothe button object which will allow application to terminate.

Step 3: Define second function corresponding to second window with attributes config, title, minsize for the window object define one button object which will shift the focus onto the third window

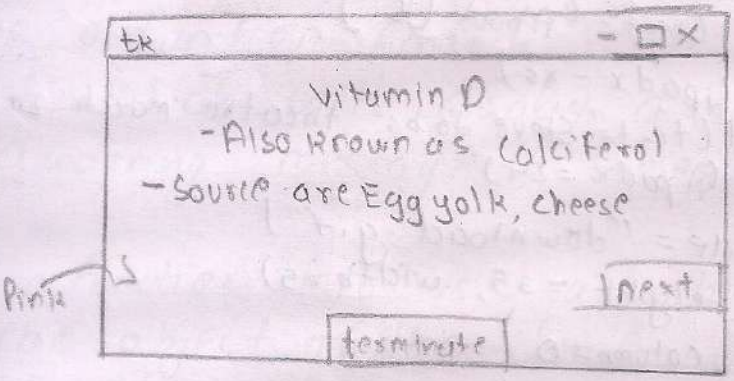
Step 4: Create third window object and in this create two button object for moving on to first window for restarting the process and second button for terminating

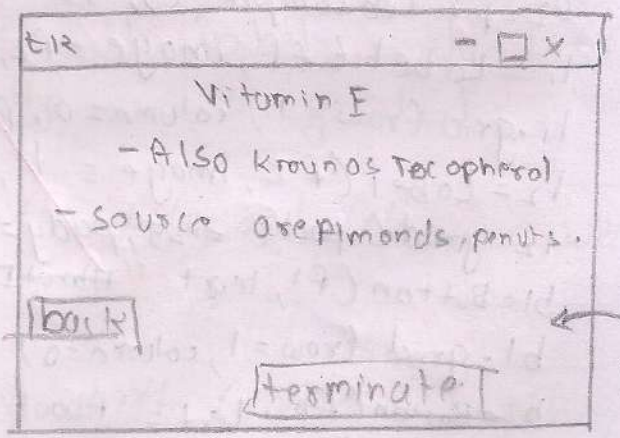
```
from tkinter import *
root=TK()
def main():
    root=TK()
    root.config(bg="pink")
    root.title("main")
    root.minsize(200,200)
    l=label(root,text="Vitamin D")
    d=pack()
    l1= label(root,text="-Also known as Calciferol\n
        -Sources are Egg yolk, cheese, etc")
    l1.pack()
    b1=Button(root,text="next", command=set)
    b1.pack(side=RIGHT)
    b2=Button(root, text="Terminate", command=ter)
    b2.pack(side=BOTTOM)
    root.mainloop()

def set();
    root=TK(),
    root.config(bg="green")
    root.title("2")
    root.minsize(400,200)
    l2=Label(root, text="Vitamin E")
    l2.pack()
    l3=label(root, text="-Also Known as Tocopherol\n
        -source are Almond, peanuts etc")
    l3.pack()
    b3=Button(root, text="back", command=main)
    b3=pack(side=LEFT)
```

Step 5: Define a function for termination and call the quit method and finally call the first function created and trigger. mainloop method.

```
b4 =    b4 = Button (root, text = "Terminate", command = ter)
        b4 = pack (side = BOTTOM)
        root.mainloop()
def ter():
    quit()

b5 = Button (root, text = "Know your vitamins",
             command = main)

b5.pack()
root.mainloop.()
```

Output





E) Image

1. Import relevant method from the tkinter library

2. Create parent window object & use the config method along with background colour attribute specified.

3. Define a function finish with the message box widget which will display a message a message i.e a warning message & subsequently terminate.

4. Define a function info use a listbox widget along with the object of the same. Use the listbox object along with insert method & insert the same & finally use the grid() with ipadx attribute.

5. Define a function about us with label widget & text attribute & subsequently use the grid().

6. Use photoimage widget with file and filename with gif attribute.

7. Create a frame object along with frame() along with parent window object height & width specified & subsequently use the grid() with row & column attribute specified.

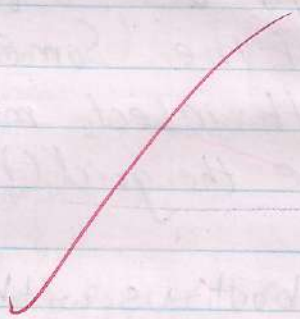
```
#image
from Tkinter import *
root=Tk()
root.config(bg="grey")
def finish():
    messagebox.askokcancel("Warning", "This will end the program")
    quit()
def info():
    list1 = Listbox()
    list1.insert(1, "Co. Name: Apple")
    list1.insert(2, "Product: iphone")
    list1.insert(3, "Language: Swift")
    list1.insert(4, "OS: IOS")
    list1.grid(ipadx=30)
def aboutus():
    list2 = Label(text="About US")
    list2.grid(ipadx=36)
    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=1, relief=SUNKEN)
l2.grid(padx=25, pady=10)
b1=Button(f1, text="AboutInfo", relief=SUNKEN, command=info)
b1.grid(row=1, column=0)
b2=Button(f1, text="About us", relief=SUNKEN, Command=about
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()
```

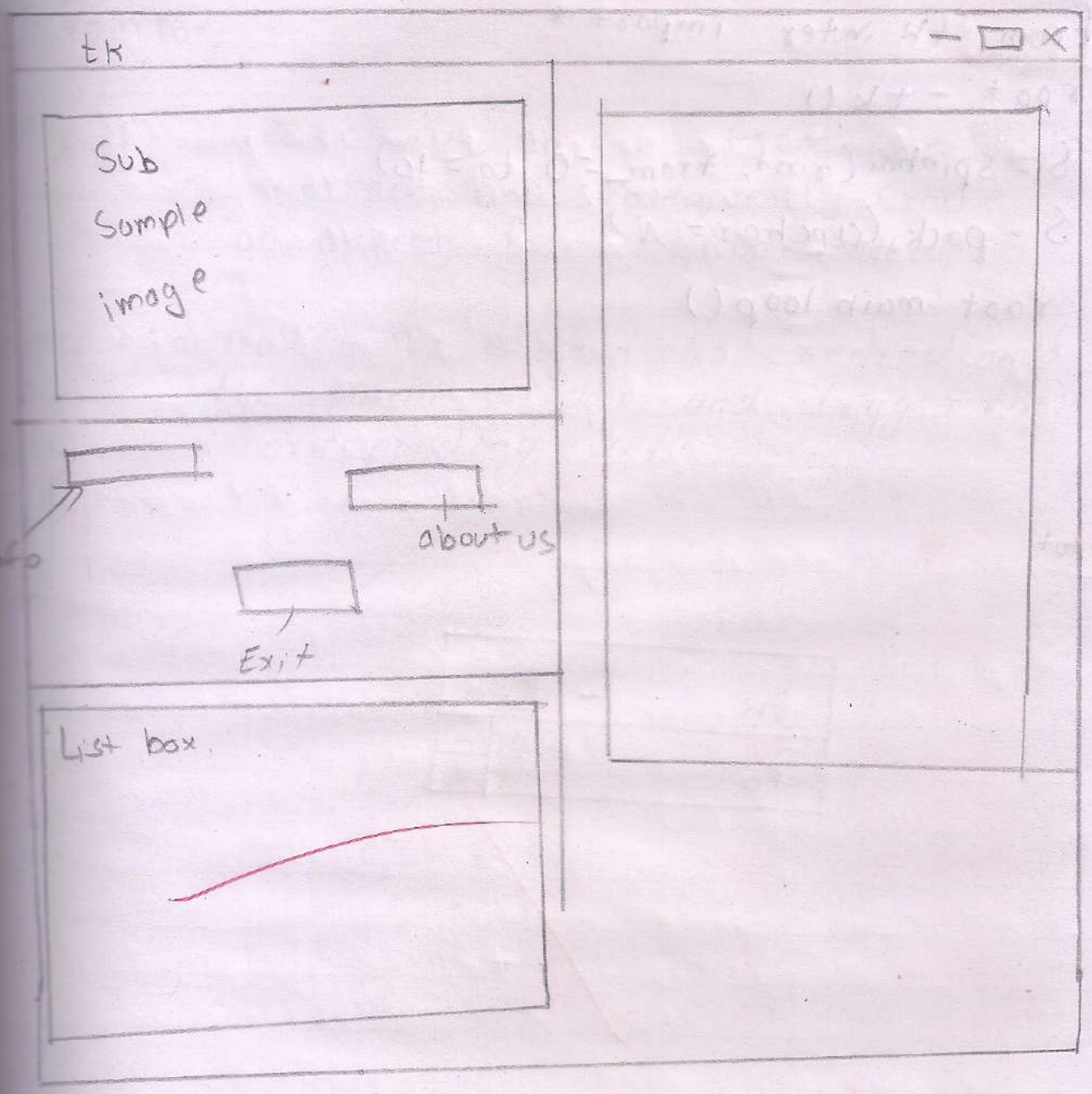
8. Similarly Create another frame object as declared bys tep 7.

9. Create another object & use the subsample (5, 4)

10. Use label widget along with the frame object relief attribute and subsequently use the grid ( )

11. Now Create button object dealing with different Section of frame.




tk
Sub
Sample
image
about us
Exit
List box

F) Spin box.

Step 1 :- Create an object from the tkmethod and subsequently create an object from Spinbox method.

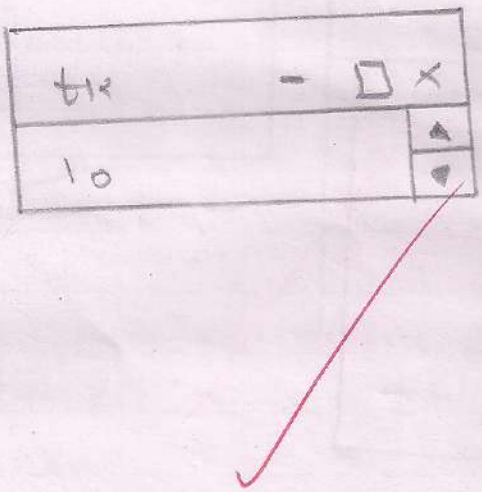
Step 2 :- make the object so created onto the parent window and trigger the corresponding events.

spin box

02

```
from tkinter import *
root = Tk()
S = Spinbox(root, from_=0, to=10)
S.pack(anchor=N)
root.mainloop()
```

Output



6) Paned Window.

Step 1 :- Create an object from the panned window method and use the pack method with the attribute fill and expand.

Step 2 :- Create an object from the label method and put it onto the ~~paro the~~ paned window with the attribute and use the add method to embed the new object.

Step 3 :- Similarly create a second paned window object and add it onto the first paned window with orientation specified.

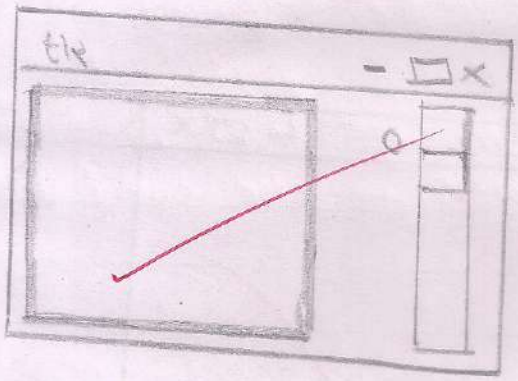
Step 4 :- Now create another label object and place it onto the second paned window object and add it onto the second pane.

Step 5 :- Trigger main loop.

paned window

```
from tkinter import *
root = Tk()
p = PanedWindow()
p.pack(fill=BOTH, expand=1)
e = Entry(p, bd=11, bg="black")
p.add(e)
p1 = PannedWindow(p, orient=VERTICAL)
p.add(p1)
top = Scale(p1, orient=VERTICAL)
p1.add(top)
root.mainloop()
```

Output



4) Canvas widget

Step 1 :- Create an object from the canvas method and use the attribute height, width, bg, and the parent window object.

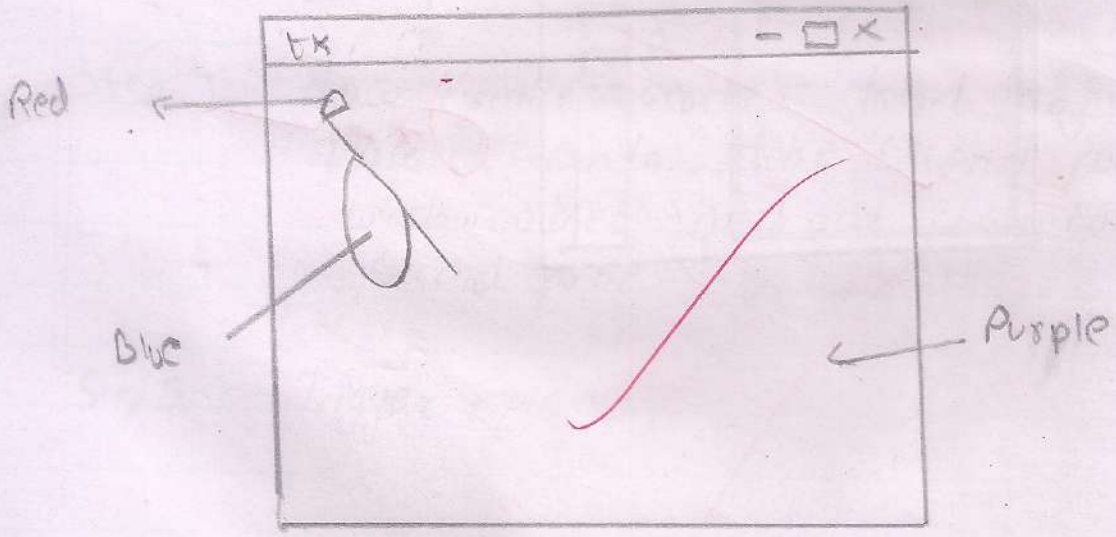
Step 2 :- Use the method creteline, create_oval and create-arc along with the canvas object so created and use the co-ordinate values

Step 3 :- Similarly use the other method and call the pack method and the main loop method.

#Canvas

```
from tkinter import*
root=TK()
C=Canvas(root,height=200,width=250,bg="purple")
Carc=C.create_arc(10,20,30,40,fill="red",start=10,extent=50)
Cline=C.create_line(20,31,45,89,fill="black")
Coval=C.create_oval(15,98,36,45,fill="blue")
C.pack()
root.mainloop()
```

Output:



PRACTICAL - 6

Aim : Database Connectivity.

(A) Step1 : Import re(DBM) dbm library and use the open() for creating the database by specifying the name of the database along with the corresponding flag

Step2 : Use the object so created for accessing the given website and corresponding regular name for the website

Step3 : Check whether the given url address matches with the regular name of the page is not equal to none than display the message that particular found / match or else not found / unmatched.

step 4 : Use the close() to terminate database library.

```
import dbm
db = dbm.open("database", flag='c', mode=438)
db["name"] = "name"
if db["name"] != None:
        print("database not empty // match!")
else:
        print("database empty!" // Not match")
match
db.close()
```

8) Step 1: Import corresponding library to make database connection, os & SQL Lite-3

Step 2: Now Create the connection object using SQL Lite-3 Library & the connect() for creating new database.

step 3: Now create cursor object using the cursor() & from the connection object created.

Step 4: Now use the execute() for creating the table with the column name & respective datatype.

Step 5: Now with cursor-object use the insert stmt for entering the values corresponding to different fields, corresponding the datatype

step6: Use the commit() to complete the transaction using the connection Object.

Step 7: Use the execute stmt along with cursor_ object for accessing the values from the database using the select from where clause.

#2

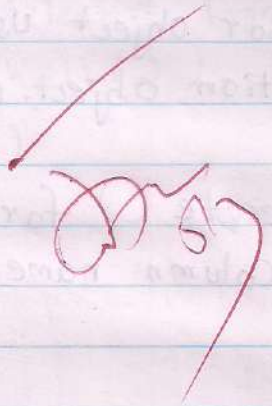
```
import os, sqlite3
conn = sqlite3.connect("employee.db")
cur = conn.cursor()
cur.execute ('create table dos (Name Char, Rollno int)
cur.execute ("insert into dos values ('Saurav', 1790),
('Sagar', 1784)")
conn.commit()
cur.execute ('select * from dos')
print (cur.fetchall())
conn.close()
```

Output

```
[('Saurav', 1790), ('Sagar', 1784)]
```

Step 8: Finally use the fetch() or fetchall() for displaying the values from the table using the cursor_objet.

Step 9: Execute() and drop table Syntax for terminating the database and finally use the close().



GUI Project.

Age calculator

Source Code.

```
from tkinter import *
from ~~tint~~ tkinter import messagebox
def clearAll():
        dayField.delete(0, END)
        monthField.delete(0, END)
        yearField.delete(0, END)
        givenDayField.delete(0, END)
        givenMonthField.delete(0, END)
        givenYearField.delete(0, END)
        rsltDayField.delete(0, END)
        rsltmonthField.delete(0, END)
        rsltYearfield.delete(0, END)
def checkError():
        if (dayField.get()=="" or monthField.get()=="" 
            or yearField.get()=="" or givenDayField.get()=="" 
            or givenMonthField.get()=="" or givenYearField.get()==""):
                messagebox.showerror("Input Error")
                clearAll()
                return -1
def CalculateAge():
        value = checkError()
        if value == -1:
                return
        else:
```

```
birth_day = int(dayField.get())
birth_month = int(monthField.get())
birth_year = int(yearField.get())

given_day = int(givenDayField.get())
given_month = int(givenMonthField.get())
given_Year = int(givenYearField.get())

month = [31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]

if (birth_day > given_day):
    given_month = given_month - 1
    given_day = given_day + month[birth_month - 1]

if (birth_month > given_month):
    given_year = given_year - 1
    given_month = given_month + 12

calculated_day = given_day - birth_day
calculated_month = given_month - birth_month
calculated_year = given_year - birth_year

rsltDayField.insert(10, str(calculated_day))
rsltMonthField.insert(10, str(calculated_month))
rsltYearField.insert(10, str(calculated_year))
```

```
if __name__ == "__main__":
    gui = TK()
    gui.config(bg = "orange")
    gui.title("Age Calculator")
    gui.geometry("525x260")
    dob = Label(gui, text = "Date of Birth", bg = "red")
    givenDate = Label(gui, text = "Given Date", bg = "red")
    day = Label(gui, text = "Day", bg = "White")
    month = Label(gui, text = "Month", bg = "White")
    year = Label(gui, text = "Year", bg = "White")
    givenDay = Label(gui, text = "Given Day", bg = "White")
    givenMonth = Label(gui, text = "Given Month", bg = "White")
    givenYear = Label(gui, text = "Given Year", bg = "White")
    rslt Day = Label(gui, text = "Days", bg = "White")
    rslt month = Label(gui, text = "Months", bg = "White")
    rslt Year = Label(gui, text = "Year", bg = "White")
    resultantAge = Button(gui, text = "Resultant Age", fg = "Black",
                          bg = "Red", command = calculateAge)
    ClearAllEntry = Button(gui, text = "Clear All", fg = "Black",
                           bg = "Red", command = clearAll)
    dayField = Entry(gui)
    monthField = Entry(gui)
    yearField = Entry(gui)

    givenDayField = Entry(gui)
    givenMonthField = Entry(gui)
    givenYearField = Entry(gui)
```

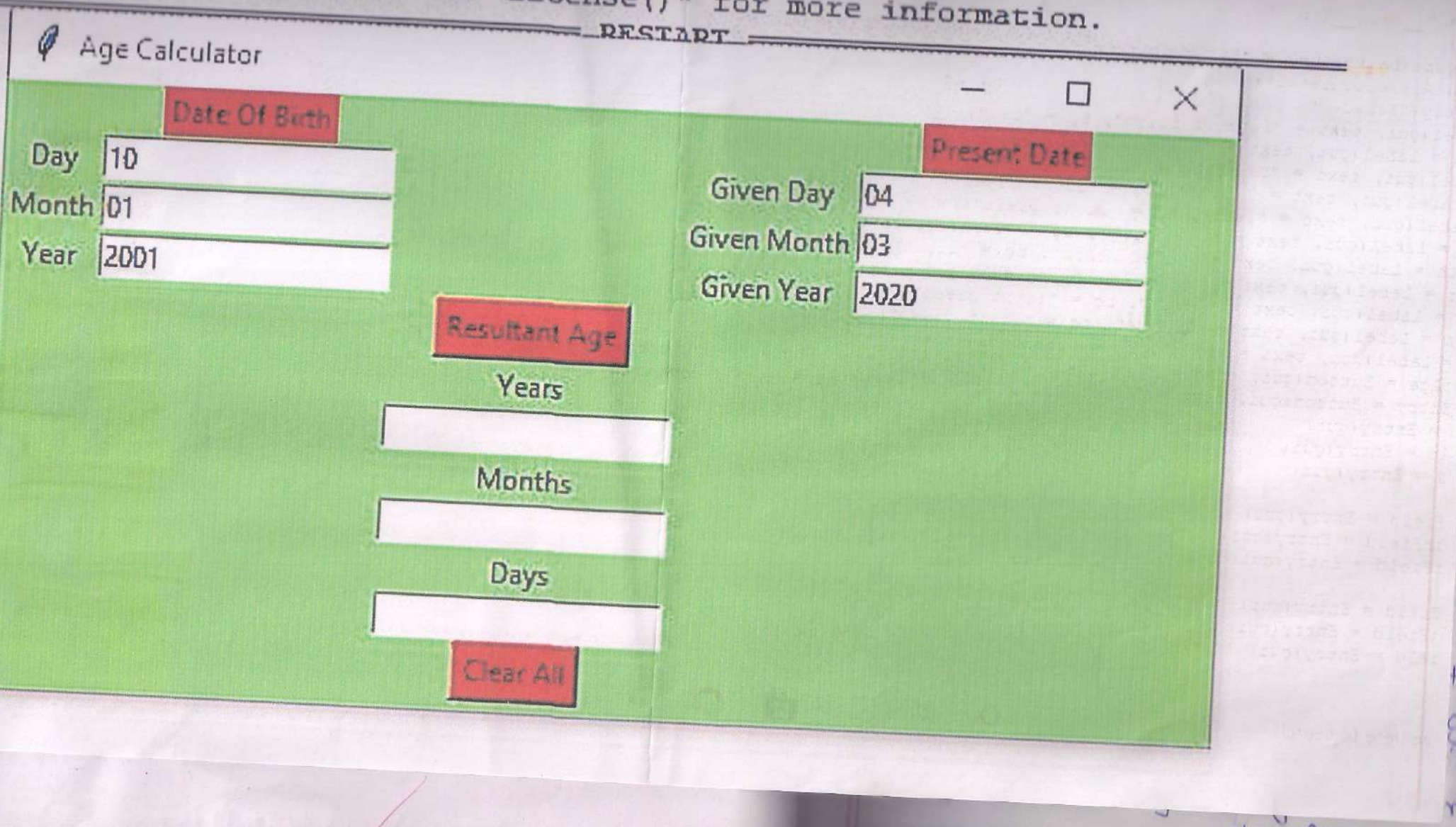
```
rsltDayField = Entry(gui)
rsltmonthField = Entry(gui)
rsltYearField = Entry(gui)

dobgrid (row = 0, column = 1)
day.grid(row=1, column=0)
dayField.grid(row=1, column=1)
month.grid( row= 2, column=0)
monthField.grid(row=2, column=1)
year.grid(row=3, column=0)
yearField.grid(row=3, column=1)

givenDate.grid(row=0, column=4)
givenDay.grid(row=1, column=3)
givenDayField.grid(row=1, column=4)
givenMonth.grid(row=2, column=3)
givenMonthField.grid(row=2, column=4)
givenYear.grid(row=3, column=3)
givenYearField.grid(row=3, column=4)

resultantAge.grid(row=4, column=2)
rsltDay.grid(row=9, column=2)
rsltDayField.grid(row=10, column 2)
rsltmonth.grid(row=7, column=2)
rsltmonthField.grid(row=8, column=2)
rsltYear.grid(row=5, column=2)
rsltYearField.grid{row=6, column=2)
```

```
Clear All Entry.grid(row=12, column=2)
gui.mainloop()
```

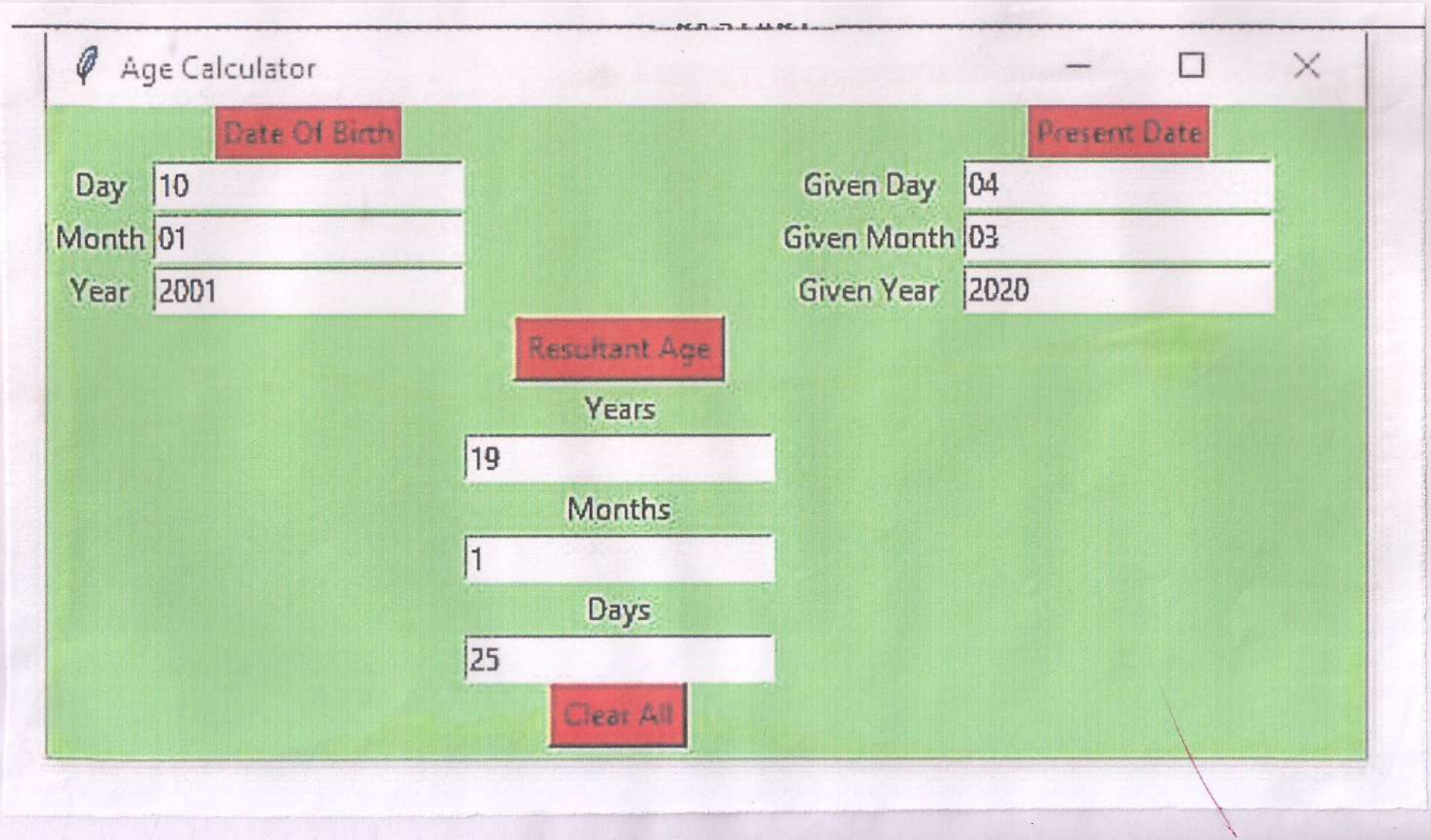
Ans

"copyright", "credits" or "license()" for more information.

========== RESTART ==========




```
        clearAll()
        return -1
def calculateAge():
    value = checkError()
    if value == -1:
        return
    else:
```


Age Calculator
Date Of Birth
Day
10
Month
01
Year
2001
Present Date
Given Day
04
Given Month
03
Given Year
2020
Resultant Age
Years
19
Months
1
Days
25
Clear All