Practical-1

OBJECTIVE:- Demonstrate the use of different file accessing modes, different attributes, read methods.

step 1: Create a file object using open method and use the write accessing modes different followed by writing some contents onto the file and than closing the file.

step 2: Now open the file in read mode and then use the read(), readline() and readlines() and store the output in variable and finally display the contents of variable.

step 3: Now use the file object for finding the name of the file, the file mode in which it's opened whether the file is still open or close and finally the output of the softspace attribute.

```
fileobj = open("abc.txt","w") #file open(write mode)
fileobj.write("computer science subjects"+"\n")
fileobj.write("C\nPython\nDS\n") #file write
fileobj.close() #file close

fileobj = open("abc.txt","r") #read mode

str1 = fileobj.read()
print("The output of read method :",str1)
fileobj.close()
>>>("The outputs of read method :","computer science subjects\nC\nPython\nDS\n")
#readline()
fileobj = open("abc.txt","r")
str2 = fileobj.readline()
print("The output of readline method :",str2)
fileobj.close()
>>>("The output of readline method :","computer science subject\n")
#readlines()
fileobj = open("abc.txt","r")
str3 = fileobj.readlines()
print("The output of readlines method =",str3)
fileobj.close()
>>>("The output of readlines method :",['computer science subjects\n','C\n','Python\n','DS\n'])
# file attributes
a = fileobj.name
print("name of file (name attribute):",a)
>>>('name of file (name attribute)','abc.txt')
b = fileobj.closed
print("(close) attribute :",b)
>>>('(close) attribute :','True')
```

022

Scanned with CamScanner

Step 7:- open the fileobj in read mod, declare a variable and perform file object dot tell method and store the output in variable

Step 8: use the seek method with the arguments with opening the objects in read mode and closing subsiquently.

Step 9: Open file obj with read mode also use the readlines method and store the output consiquently in and print the same for counting the length use the for conditional statement and display the lengths.

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

```
# 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")
st1 = fileobj.seek(0,1)
print("seek(0,1) is:", st1)
file obj.close()
>>>("seek(0,1) is:", None)
fileobj = open("abc.txt","r")
st2 = fileobj.seek(0,2)
print("seek(0,2) is:", st2)
fileobj.close()
>>>("seek(0,2) is:", None).
```

```
# finding length of different lines within lines.
fileobj = open("abc.txt","r")
stat = fileobj.readlines()
print("output:", stat)
for line in stat:
    print(len(line))
fileobj.close()
>>>("output:", ['I eat data structure']).
```

Scanned with CamScanner

iter() and next():-

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

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

for loop:-

```
mytuple1 = ("Tom", "Stuart", "Jerry")
for x in mytuple1:
    print(x)
```

```
>>> Tom
    Stuart
    Jerry
```

square and cube:-

```
def square(x):
    y = x * x
    return y
def cube(x):
    z = x * x * x
    return z
count1 = [square, cube]
```


Practical - 2.

OBJECTIVE : Iterators.

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

step 2: The similar output can be obtained by using for conditional statement. An iterable variable is to be declared in for loop 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 way declare cube to get the value raised 3 and return the same.

step 4: Call the declared function using function call.

Scanned with CamScanner

step 5: using for conditional statement specifying the range use the list typecasting with map method, declare a 'lambda' i.e annonymas function and print the same.

step 6: Declare a listnum variable and declare some elements then use the map method with help of lambda function give two argument display the output.

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

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

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

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

026

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

```
# map()
listnum = [0,4,5,7,9,11,13,15,20,19,25]
listnum = list(map(lambda x: x % 5, listnum))
print(listnum)
def even(x):
    if(x%2==0):
        return "EVEN"
    else:
        return "ODD".

list(map(even, listnum))
>>> [0,4,0,2,4,1,3,0,0,4,0]  o/p ?
```

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
```

Scanned with CamScanner

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

```
>>> enter a number : 15
1
3
5
7
9
11
13.
```

factorial:

```
def f(n):
    if (n==1):
        return 1
    else:
        return (n*f(n-1))
n = int(input("enter a number :"))
lst = []
lst.append(n)
a = map(f, lst)
print("the factorial is:", a)
```

```
>>> enter a number : 5
('the factorial is: ', [120])
```


step 10: Define an object of a class.

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

step 12: Define a function with if condition, if exactly equal to one return else on parameters.

step 13: Enter a input from the user are an empty list use append method. for appending the input value.
Print the output variable.

Scanned with CamScanner

10/12/19
Tuesday

FSC

Practical-3

Aim:- Exceptions

step1: In any try block open an file in the open mode with write mode, write some content in the file.

step 2: In except (IO error) block use the error in IOerror. Use the appropiate message display.

step 3: wise display the operation is successful.

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

step 5: In except block use value error and print the same message.

step 6: wise display the operation is successfull

try: # I/O Error:-
```
    fo=open("yashvi","w")
    fo.write("my name is yashvi")
except:
    print("error")
else:
    print("operation successfull")
```
>>> error

Value Error:-
```
try:
    x=int(input("enter a statement"))
except ValueError:
    print("error")
try:
    a=open("abc.txt","w")
    a.write("Mars")
except IOError:
    print("I live on Mars")
else
    print("operation successfull")
```
>>> Enter: 1
Successful
>>> Enter: abc
I live on Mars.

Scanned with CamScanner

TypeError, ZeroDivision Error

```
try: Input/Output exception.
    print(file)
    print('10' + 23)
except (TypeError, ZeroDivisionError):
    print ("Invalid).
>>> 1-0
>>> 1@10
>>> Invalid
```

Zero Division Error

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

Output

```
>>> Enter: 2
    0.5
>>> Enter: 0
    Denominator is zero
>>> Enter: C
    Incompatible value.
```


step 1: Accept any integer value from the user. In the try use the division method.

step 2: For the exception to be raised use the except keyword (one) i.e TypeError prin. Incompatible value.

step 3: Use except with typeError if zero divis error and print the message accordingly to that if entered number is zero, then it not able to perform operation.

step 10: Declare static variables and values

step 11: For multiple exception use the erro types by seperating them with a comm

10/12/11 Tuesday

DSC

Step 12: use try block open a file in write mode and subsequently enter values in the file.

Step 13: use the IOError and display appropiate message.

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

Step 15: Define another function y. Initialize or declare some elements in list and calculate the length of the same and display the same.

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

using except keyword..

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

OUTPUT:

```
>>> successful
0
None
4
None
```

031

Raise Keyword:

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

OUTPUT:

```
>>> enter: x y z
enter integer value!
```

Scanned with CamScanner

→ **Resplit :-**

```
import re
sequence = 'hallo123, howday456,789how r u')
pattern = r'\d+'
output = re.split(pattern, sequence)
print(output)
```

>>>['123', '456', '789']

→ **Find all :**

```
import re
sequence = 'hello123, howday456, 789Hii')
pattern = r'\d+'
output = re.findall(pattern, sequence)
print(output)
```

>>> ['hello', 'howday', 'Hii']

→ **Removing wide spaces.**

```
import re
s = 'ab  c  d ef')
pattern = r'\s+')
r = ''
output = re.sub(pattern, r, s)
print(output)
```

>>> abcdef

29/12/19

diff after pack ... ghsa

Practical 4 :

Aim : Regular expression.

Step 1 : :-

- Import re module in python environment

Step 2 : :-

- Initialize a variable to store a string. Use appropriate pattern to split the numbers. Now use the split () method with the pattern and string as its arguements.

Step 3 : :-

- Print the result.

Step 4 : :-

- Again import re module.. Initialize a variable with string of your choice. An appropiate pattern should be used for seperating the characters. Use the split () method with pattern and string as its arguement

→ Alpha numeric character :-

```
import re
seq = 'xyzhsc@gmail.com, abchsc@gmail.com'
pattern = '[\w\.-]+[\w\.-]+'
output = re.findall(pattern, seq)
print(output)
```

```
>>> ['xyzhsc', 'gmail.com', 'abchsc', 'gmail.com']
```

→ Research Method :-

```
import re
s = 'python is an indented Language'
output = re.search('\Apython', s)
print(output)
v = output.group()
print(v)
```

```
>>> _sre.SRE_Match object at 0x030F94B8>
python
```

→ Number start from 8 or 9 :-

```
import re
i = ['9930400224', '9819386123', '9308133355']
for v in i:
    if re.match(r'[8-9][0-9]{9}$', v) or len(v) == 10:
```

Scanned with CamScanner

10/12/ Tues

```
else:
    print("call number matches")
else:
    print("call number does not match")
```

```
>>> call number matches
call number matches
call values does not match.
```

⇒ string starts with vowel :-

```
import re
s = 'cats are very Lazy'
p = re.findall(r'[aeiou]\w+',s)
print(p)
>>> ['ats', 'are', 'ery', 'azy']
```

⇒ import re

```
s = ['mr.a', 'ms.e', 'mr.i', 'ms.o', 'ms.u']
p = r'[ms|mr]+'
o = re.findall(p,s)
print(o)
m = 0
f = 0
for v in o:
    if v == 'ms':
        m = m+1
    else:
        f = m+1
print("No of males", m)
print("No of females", f)
```


Step 8:

- Import re module. Initialize a sequence use findall() method with raw string and list of vowels i.e pattern along with the sequence. Print the output.

Step 9:

- Import re module. Initialize a ~~string with list of mails~~ sequence with desired string use search() method and the arguements should be the word to search along with the sequence

Step 10:

Print the result. Now use the group method and then display the output.

Step 11:

Import the re-module from the library Initialize a variable with some call number use for loop and then if loop with match() method and the arguements should be the range also check the range.

Step 12:-

- If it satisfies the condition print "Call number matches else print "Call no does not match"

Step 13:

- Import re module. Initialize a string with list of males and females, from a pattern beginning with raw string. Use find all() method with two arguements i.e pattern and sequence.

Step 14:

- Print the output Initialize a variable for counting mans and females. Use for loop to chek condition if it is 'ms' than count of female will be incremented else count of male will be incremented.

Step 15:

- Display the result with exact count of males and females.

>>> ['mr', 'ms', 'ms', 'mr', 'ms']
No of males : 2
No of females : 3

7/1/2020

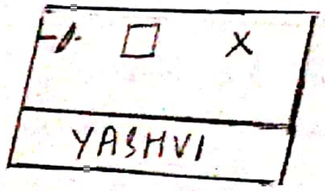
Scanned with CamScanner

• TEXT Label

CODE:-

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

OUTPUT:-



7/1/20
Tuesday

Practical 5 :-

AIM: GUI components

Step 1 :

Use the tkinter Library for importing the feature of text widget

Step 2 :

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

Step 3 :

Use the pack method along with the object created from text method.

Step 4 :

Use the main loop method for triggering of corresponding events.

Step 5 :

Use the tkinter library for importing the feature of text widget.

Scanned with CamScanner

Step 6: Create a variable from text method and position it onto patent window

Step 7: use the pack method along with the object created from text method and use the parameter.

Step 8: side = LEFT , padx = 20
side = LEFT , pady = 30
side = TOP , ipadx = 40
side = TOP , ipady = 50

Step 9: use the main loop method for triggering of corresponding events.

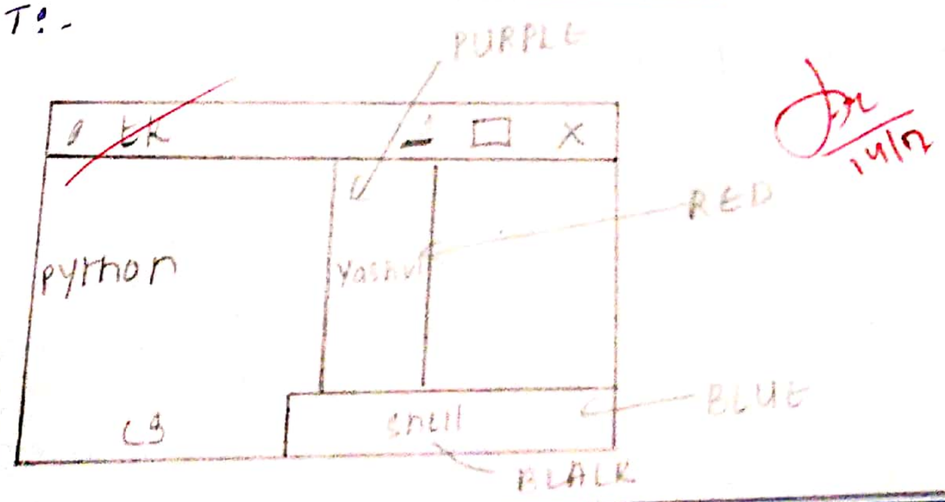
Step 10: Now repeat the above steps with the label method which takes the following arguements:
i) Text attribute which definis string
2) Bg (background) color
3) Fg (foreground) color
4) Name of the parent window
5) use pack method with the reluvant padding attribute

.Label

CODE:

```
from tkinter import *
root = TK()
l = Label (root, text = "python")
l.pack (side = LEFT, pady = 30)
ll = Label (root, text = "yashu", bg = "purple",
                                  fg = "red")
ll.pack (side = TOP, ipady = 40)
m = Label (root, text = "cs")
m.pack (side = LEFT, padx = 20)
m2 = Label (root, text = "shell", bg = "blue",
                                  fg = "black")
m2.pack (side = LEFT, ipadx = 50)
root.mainloop ()
```

OUTPUT:-



1/11/22

C Radio button :

• SOURCE CODE :-

```
from tkinter import *
root = TK()
def sel():
    selection = "you selected the option" + str(var.get())
    L = label.config(root, text = "selection", justify=LEFT)
    L.pack(anchor = S)
    V = IntVar()
r1 = Radiobutton(text="option 1", variable = "V",
    value = '1', command = sel)
r1.pack()
r2 = Radiobutton(text ="option 2", variable = "V",
    value = "2", command = sel)
r2.pack()
r3 = Radiobutton(text="option 3", variable = "V")
    value = "3", command = sel)
r3.pack()
root.mainloop()
```


Radio button

W.A.P making use of the control variable and button widget for selection of the given option.

Step 1: Use the tkinter to import relevant method.

Step 2:
→ Define a function which tells the user about the given selection need of the multiple options avaible.

Step 3:
→ Use the configuration method along with the label object and call the variable as an arguemint within the method.

Step 4:
→ Now define the parent window and define the option using control variable.

Step 5:
Now create an object from the radio button method which will take the foll arguments
(1) Positioning on parent window
(2) Defining the text variable L 1, 2, 3, 4].

13) Define the variable argument.
14) corresponding value and triggiy the given function.

Step 6:
→ Pack method for the corresponding radio. objects so created and specify the attribute as an anchor attribute.

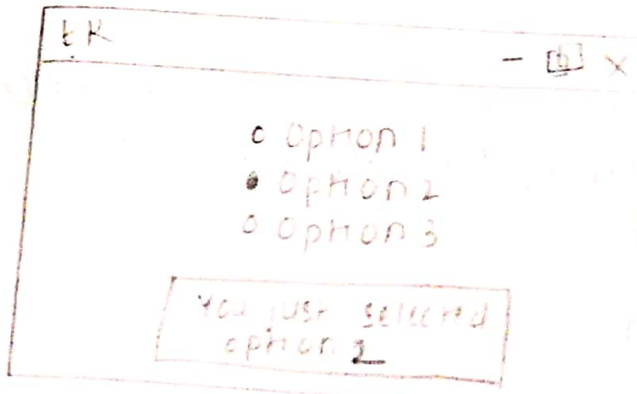
Step 7:
→ Now define the label object from the corresponding method and place it on parent window.
Subsequently use pack method for this widget and make use of the main loop method.

Step 8:
→ Import the relevant method from tkinter Library.

Step 9:
→ Define the object corresponding to the object window and define the size of parent window in terms of no of pixels.

Step 10:
→ Now define the frame object from the method and place it onto the parent window.

- OUTPUT:-

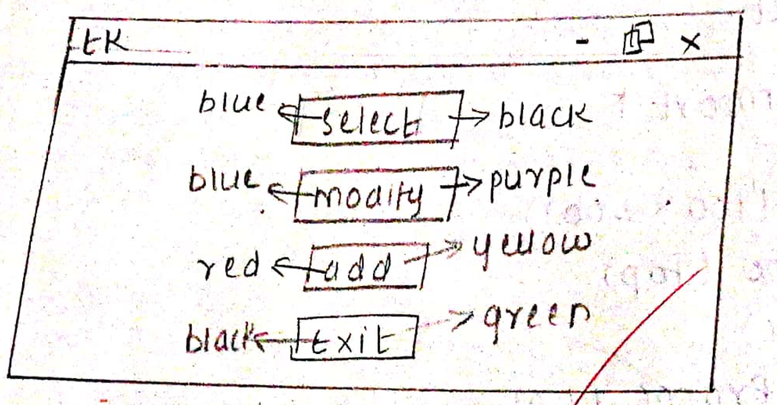


- SOURCE CODE:-

```
# Frame object:

from tkinter import *
top TK()
top.geometry('100x200')
frame = Frame (top)
frame.pack()
left frame = Frame (top)
left frame.pack (side = LEFT)
b1 = Button (frame, text = "select", active background
            = "red", fg = "black")
b1.pack()
b2 = Button (frame, text = "modify", active background
            = "blue", fg = purple)
b2.pack()
```

```
b3 = Button (frame, text = "Add", active background =
                "yellow", fg = "red")
b3.pack()
b4 = Button (frame, text = "EXIT", active background =
                "green", fg = "black")
b4.pack()
top: mainloop()
```

OUTPUT:-



Step 11:

→ Create another frame object the left frame and put it onto parent window on its left side.

Step 12:

→ Similarly define the right frame and subsequently define the button object placed onto the given frame with the attribute as text active bg and fg.

Step 13:

→ Now use the pack method along with the side attribute.

Step 14:

→ Similarly create the button object corresponding to modify option and put it into the frame object with side equal to right attribute set.

Step 15:

→ Add another button and put it on right frame object and term it as EXIT.

Step 16:

Use the palk method for all the object's and finally use the main loop method.

Scanned with CamScanner

080

* Message Box Method:

Step1: Import the relevant method from tkinter library.

Step 2: Define a function & use the message box along with different methods availa which contains one or more arguements.

Step 3: Thus different options which are avaible are showinfo(), showwarning(), askyesNo(), askquestion().

Step 4: Create object from button method & place it onto parent window with title of button specified.

Step5: use the pack method to display the button widget & finally use mainloop method.

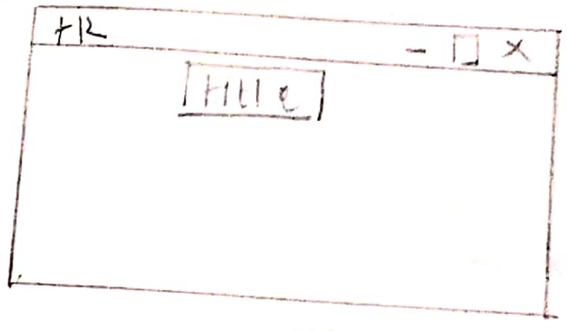
Step6: If the user wants to hide the parent window and only the info window should be visible corresponding to the six option given below.

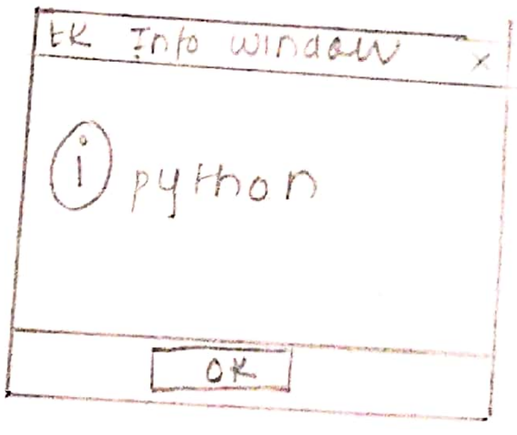
Message Box

040

```
from Tkinter import *.
import tkMessageBox
root = Tk()
def fun():
    tkMessageBox.showinfo("info window", "Python")
b1 = Button(root, text="Hello", command = fun)
b1.pack()
root.mainloop().
```

OUTPUT:-

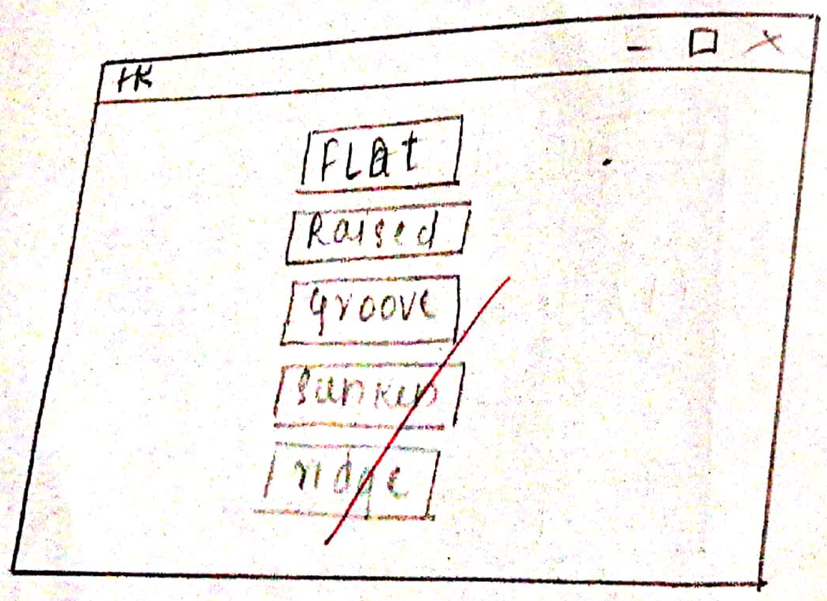




Scanned with CamScanner

```
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:-



* Relief style:

Step 1: Use the button with the following attributes
1. The parent window
2. Text attribute
3. Relief

Step 2: Use the corresponding pack method for the respective button objects and trigger the corresponding event

Step 3: Finally use the mainloop method.

* Traversing and moving use of geometry layout manager method.

Step 1: Define a function & create a object of the given window by using the three methods namely config, title, minsize

Step 2: Create a button object and use the text and command attribute for triggering the given event and use grid method along with internal and external padding. Hence button object will allow application to terminate.

Step 3: Define second function corresponding to second window with attributes config, title, minsize for the window object and shift the focus onto third window

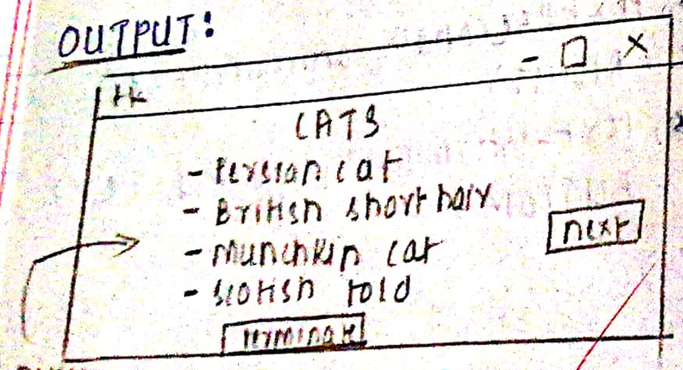
Step 4: Create third window object and in this create two button object for moving on to first window for restarting the process and second button for terminating

Traversing:-

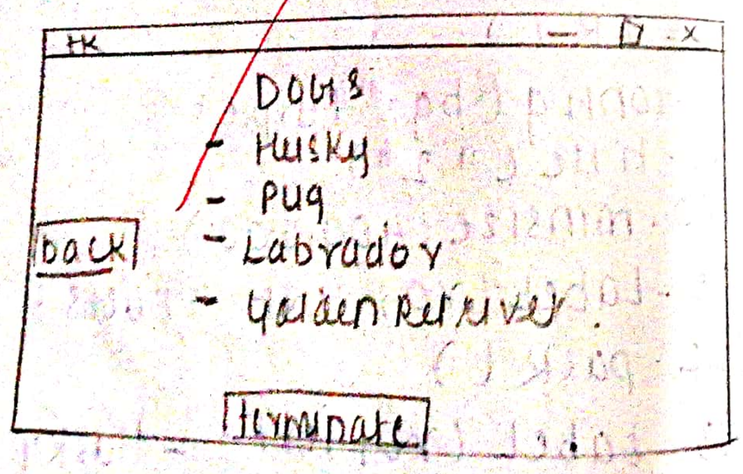
```
from tkinter import *.
root = TK()
def main():
    root = TK()
    roo.config(bg="PINK")
    roo.title("main")
    roo.minsize(200,200)
    L = Label(roo, text="(ATS)")
    L.pack()
    L1 = Label(roo, text="-Persian cat \n- British
                shorthair \n- Munchkin cat \n-
                Scottish Fold")
    L1.pack()
    b1=Button(root, text="second", command = se)
    b1.pack(side = RIGHT)
    b2=Button(roob, text="Terminate", command = ter)
    b2.pack(side = BUTTOM)
    roo.mainloop()
def se():
    ro = TK()
    ro.config(bg="Purple")
    ro.title("2")
    ro.minsize(400,200)
    L2 = Label(ro, text="DOGS")
    L2.pack()
    L3 = Label(ro, text="- Husky \n- Pug \n-
                - Labrador \n - Golden Retriver")
```

```
b2 = Button(r0, text="BACK", command
b2.pack(side=LEFT)
b3 = Button(r0, text="terminate", command
b3.pack(side=BUTTOM)
r00.mainloop()

def key():
    quit()

b4= Button(root, text="GET CATS AND DOGS",
                command = main)

b4.pack()
root.mainloop()
```

OUTPUT:



pink.



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

12/12.

Scanned with CamScanner

- Displaying the image:

ALGORITHM:-

Step 1: Create an object corresponding to the parent window and use the following 3 methods
• Title • Maxsize • config

Step 2: Create a leftframe object from the frame method and place it onto the parent window with the height, width and bg specified. Subsequently use the grid method with the row, column, padx, pady specified.

Step 3: Now create a rightframe object from the frame method with the width, height specified and the row and column value should be specified.

Step 4: Create a label object from the Label method and place it onto the left frame with text attribute denoting the original image with relief attribute used as 'RAISED' value and subsequently use grid method with row, column value specified as (0,0) with some external padding value.

```
from tkinter import *
root = TK()
root.title("Python")
root.maxsize(1000, 9000)
root.config(bg = "purple")
leftframe = Frame(root, bg = "red", height = "400",
    width = "200")
leftframe.grid(row = 0, column = 0)
rightframe = Frame(root, bg = "pink", height = "400",
    width = "250")
rightframe.grid(row = 0, column = 0)
Label(leftframe, text = "Photo", height = 2, width = 20)
    grid(row = 0, column = 0)
image1 = PhotoImage(file = "cat.gif")
image1.subsample(1, 2)
image2 = PhotoImage(file = "cat  gif")
image2.subsample(3, 4)
Label(leftframe, image = image1).grid(row = 0, column = 0,
    padx = 20, pady = 10)
Label(rightframe, image = image2).grid(row = 0,
    column = 1, padx = 10, pady = 10)
toolbar = Frame(leftframe, width = 200, height = 400,
    bg = "white").grid(row = 2, column = 0)
Label(toolbar, text = "Personal Info", height = 2,
    width = 20, relief = RAISED).grid(row = 0, column = 0,
    padx = 20, pady = 20)
```

```
def name():
    print("Name: Yashvi")

def hob():
    print("Hobby: Swimming")

def dob():
    print("DOB: 14/06/2000")

def add():
    print("Address: Mumbai")

Button(toolbar, text="Name", height=1, width=16,
    command=name).grid(row=1, column=0)

Button(toolbar, text="Hobby", height=1, width=16,
    command=hob).grid(row=1, column=1)

Button(toolbar, text="DOB", height=1, width=16,
    command=dob).grid(row=2, column=0)

Button(toolbar, text="Add", height=1, width=16,
    command=add).grid(row=2, column=1)

root.mainloop().
```


step 5: Now use the photo image method with the file attribute specified

step 6: Use the sub sample method with the object of the image and give x, y co-ordinates values

step 7: Use the label method and position it onto the left frame and placing the image after the sampling and use the grid method for the positioning in the first row.

step 8: Create another label object positioning it onto the right frame and specifying the image and background attribute with row and column attribute specify it as (0,0)

step 9: Now create a tool bar object from the Frame method and position it onto the left frame with height and width specified and position it onto the second row.

step 10: Now define the various function for different tool bar options provided in the left frame.

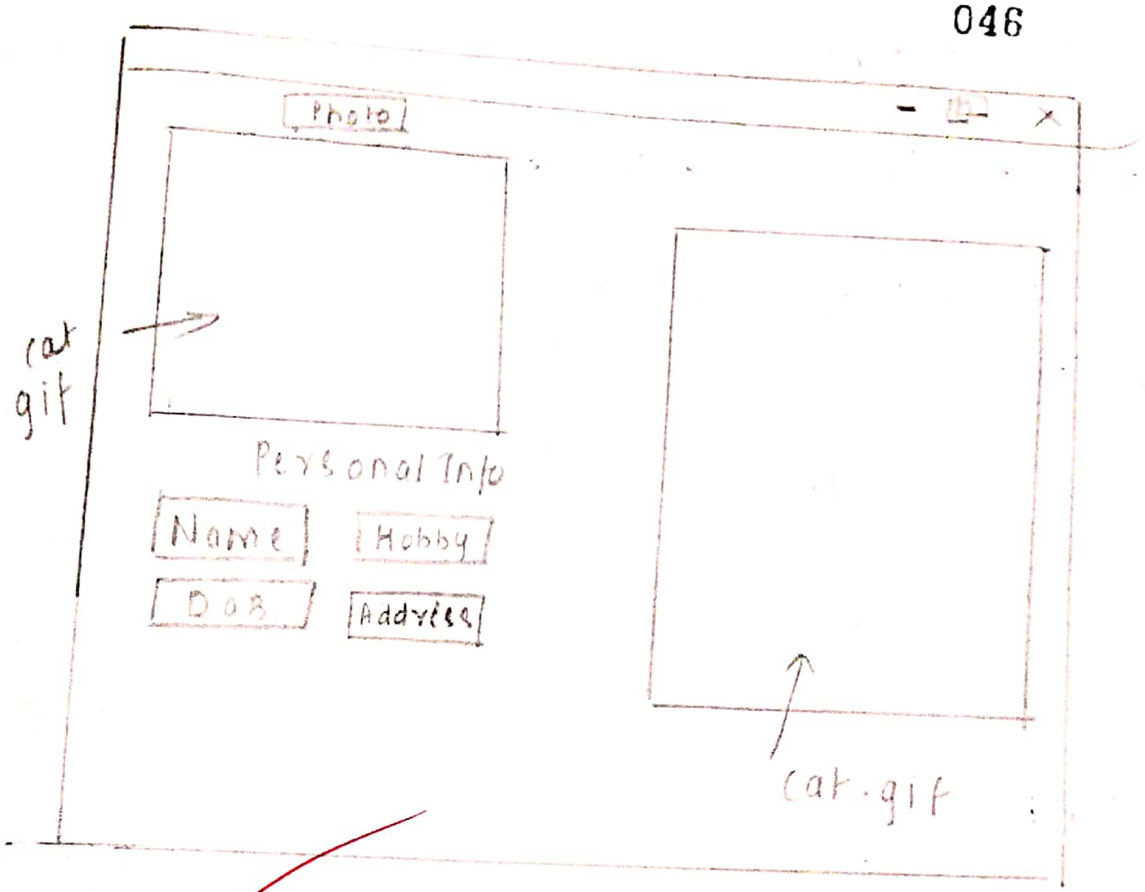
Scanned with CamScanner

Step11 : From the label method position the text on the toolbar, use the relief attribute and corresponding grid value and incorporate the internal padding as well

Step12: Create the label method position it on the toolbar with the next title as personal information and position it on same row but next column.

Step 13 : Now make use of mainloop method

OUTPUT:-



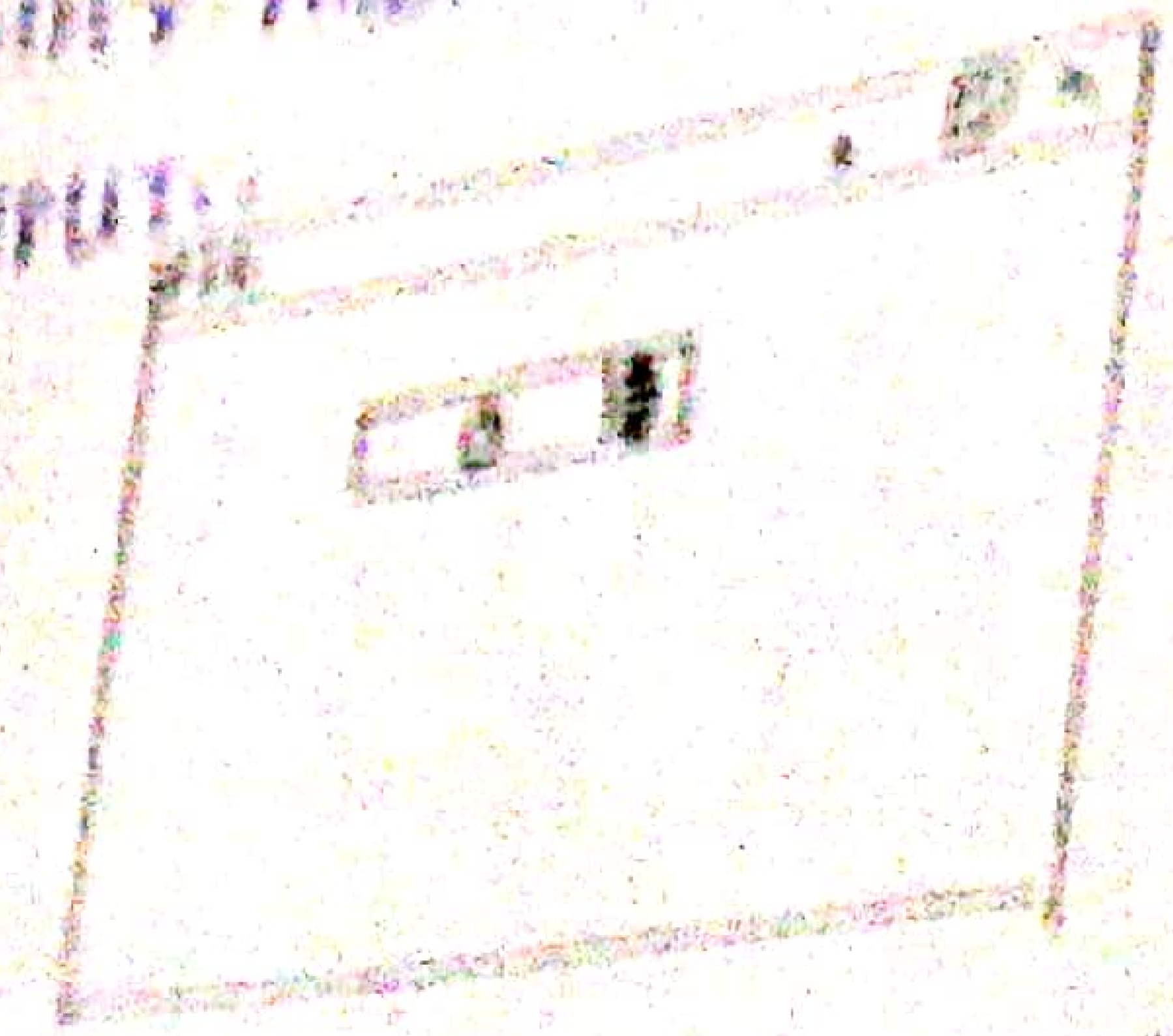
Scanned with CamScanner

• Spinbox

CODE:

```
from tkinter import *
master = Tk()
s1 = Spinbox(master, from_=0, to=10)
s1.pack()
master.mainloop()
```

OUTPUT:



• PANNED WINDOW

CODE:

```
from tkinter import *
root = Tk()
p = PanedWindow()
p.pack(fill=BOTH, expand=10)
L = Label(p, text="This is python")
p.add(L)
p1 = PanedWindow(p, orient=VERTICAL)
p.add(p1)
L1 = Label(p1, text="python")
p1.add(L1)
root.mainloop()
```

OUTPUT:

• Spinbox:

Algorithm

• Step 1: Create an object from the Tk method and subsequently create an object from Spinbox method.

• Step 2: Make the object to extend on parent window and trigger the corresponding events.

• PANNED WINDOW:

• Step 1: Create an object from paned window() and use the pack() with attribute fill and expand.

Step 2: Create an object from the label method and put it onto the paned window with the text 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.

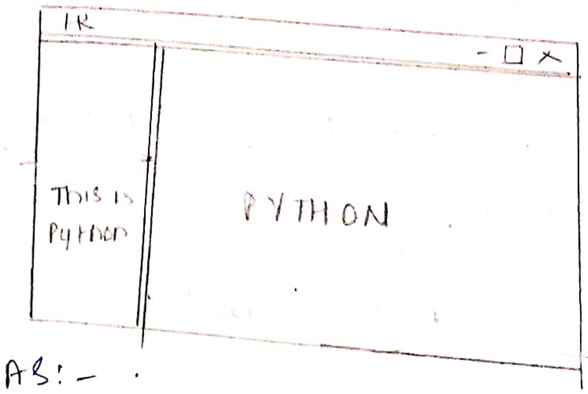
• Step 4:- Now create another label object and place it onto the second paned window object and add it onto the second paned

• CANVAS:-

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

Step 2: Use the method create line, oval and create arc along with canvas object so created as use the cordinate values.

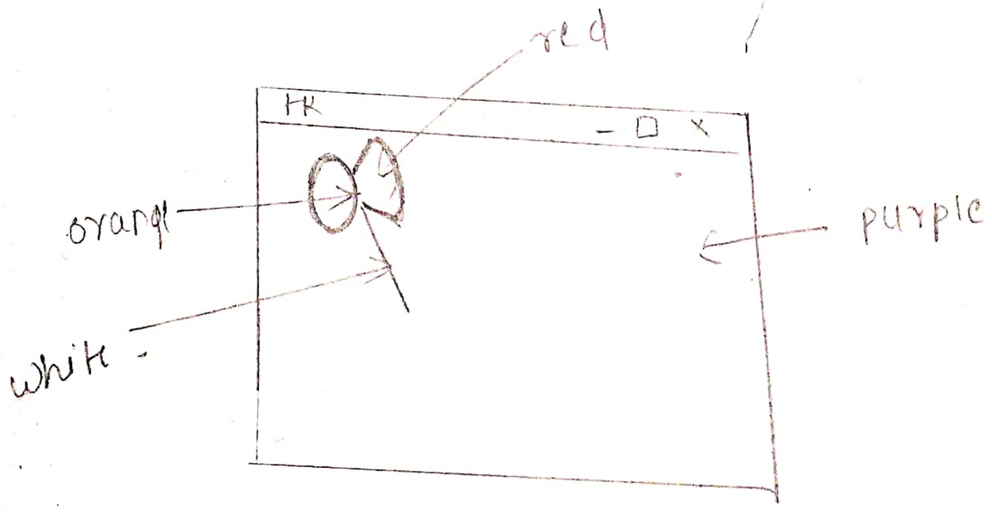
Step 3: Similarly use the other method and call the pack method and mainloop method.



• CANVAS:-

```
from tkinter import *
root = TK()
c = Canvas(root, height=200, width=250, bg="purple")
carc = c.create_arc(10,70,60,10, fill="red", width=4)
cline = c.create_line(35,40,50,55, fill="purple", width=2)
coval = c.create_oval(10,20,35,50, fill="orang",
                      width=3)
c.pack()
root.mainloop()
```

OUTPUT:-



Practical 6

AIM: Database connectivity :

Steps:

Step 1:
- import dbm
- db = dbm.open("database", flag='c', mode=0666)
- db["name"] = "name"
- if db["name"] != None:
  - print("database found!!")
- else:
  - print("database not found!!")
- match
- db.close()

OUTPUT

Database not found

Step 2:
use the object so created for accessing the given website and corresponding regular name for the website.

Step 3:
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 the database library.

Scanned with CamScanner

#2:

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

Step 2:
Now create the connection object using SQL Lite - 3 library and the connect() for creating new database.

Step 3:
Now create cursor object using the cursor and from the connection object created.

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

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

Step 5:
use the commit() to complete the transaction using the connection object.

#1:

```
import os, sqlite3
con = sqlite3.connect("student.db")
cur = con.cursor()
cur.execute("create table dos (Name char,
    Roll No int)")
cur.execute("insert into dos value ('Yashvi',
    1706), ('Sanjana', 1709)")
con.commit()
cur.execute('select * from dos')
print(cur.fetchall())
con.close()
```

OUTPUT:

[('Yashvi', 1706), ('Sanjana', 1709)]

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

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

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

CAT Project:-

```
from tkinter import *
import sqlite3
from time import *

def show():
    def sub():
        name=str(e1.get())
        contact=int(e2.get())
        email=str(e3.get())
        add=str(e4.get())
        cat=str(e5.get())
        doa=strftime('%d-%m-%Y')

        conn=sqlite3.connect('cathouse.db')
        cur=conn.cursor()
        cur.execute('CREATE TABLE if not exists "adoption" ("Customer_No"
INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE, "Name" TEXT NOT NULL,\
    "Contact_No" INTEGER NOT NULL, "Email_ID" TEXT NOT NULL, "Address" TEXT NOT
NULL, "Cat" TEXT NOT NULL, "Date_of_Adoption" TEXT NOT NULL)')

        cur.execute('INSERT INTO adoption
(Name,Contact_No,Email_ID,Address,Cat,Date_of_Adoption) VALUES (?,?,?,?,?,?)'\
                    ,(name,contact,email,add,cat,doa))

        for row in cur.execute('SELECT *FROM adoption'):
            print(row)

        conn.commit()

        close1=Tk()
        close1.title('Sparkle Cat House')
        close1.config(bg='#ce66d4')
        close1.maxsize(height=90, width=400)
        close1.minsize(height=90, width=400)
        Label(close1, text="THANKS FOR ADOPTING\nA CAT !", font='Verdana 12
bold', bg='#ce66d4').pack()

        Button(close1, text="OK", command=close1.destroy, font='Verdana 12
bold', bg='#52004b', fg='white', width=8).pack()

    y=Toplevel()
    y.title('Sparkle Cat House')
    y.geometry('600x600')
    y.maxsize(height=500, width=600)
    y.config(bg='#ce66d4')

    Label(y, text='\nSPARKLE CAT HOUSE\nADOPTION CENTRE\nConfirm The
Adoption\n', font='Times 19 bold', bg="#52004b", fg="white",
width=70).pack(anchor='c')

    pic=PhotoImage(file='cats.png')
    pic.subsample(3,4)
    Label(y, image = pic, bg='#52004b').place(x=10, y=15)
```

Scanned with CamScanner

```
        pic3=PhotoImage(file='cats3.png')
        pic3.subsample(3,4)
        Label(y, image = pic3, bg='#52004b').place(x=470, y=10)

        sel="\n* Name : "+str(e1.get())
        Label(y, text=sel, font='Verdana 12 bold', bg='#ce66d4').pack(padx=20,
    anchor='w')
        sel="\n* Contact Number : "+str(e2.get())
        Label(y, text=sel, font='Verdana 12 bold', bg='#ce66d4').pack(padx=20,
    anchor='w')
        sel="\n* Email ID : "+str(e3.get())
        Label(y, text=sel, font='Verdana 12 bold', bg='#ce66d4').pack(padx=20,
    anchor='w')
        sel="\n* Address : "+str(e4.get())
        Label(y, text=sel, font='Verdana 12 bold', bg='#ce66d4').pack(padx=20,
    anchor='w')
        sel="\n* Cat : "+str(e5.get())
        Label(y, text=sel, font='Verdana 12 bold', bg='#ce66d4').pack(padx=20,
    anchor='w')
        sel="\n* Date of Adoption : "+strftime('%d-%m-%Y')
        Label(y, text=sel, font='Verdana 12 bold', bg='#ce66d4').pack(padx=20,
    anchor='w')

    Button(y, text="SUBMIT", command=sub, font='Verdana 12 bold', bg='#52004b',
fg='white').pack(anchor='s',side='right', padx=20, pady=20)
    Button(y, text="EDIT", command=y.destroy, font='Verdana 12 bold',
bg='#52004b', fg='white').pack(anchor='s',side='right', padx=20, pady=20)

    y.mainloop()

x=Tk()
x.title('Sparkle Cat House')
x.maxsize(height=700, width=900)
x.config(bg='#ce66d4')

Label(x, text='\nSPARKLE CAT HOUSE\nADOPTION CENTRE\n', font='Times 19 bold',
bg='#52004b', fg='white', width=60).grid(row=0, columnspan=2)

pic=PhotoImage(file='cats.png')
pic.subsample(3,4)
Label(x, image = pic, bg='#52004b').place(x=170, y=5)

pic3=PhotoImage(file='cats3.png')
pic3.subsample(3,4)
Label(x, image = pic3, bg='#52004b').place(x=640, y=0)

e1=Entry(x, width=40, font='Verdana 12 bold')
e2=Entry(x, width=40, font='Verdana 12 bold')
e3=Entry(x, width=40, font='Verdana 12 bold')
e4=Entry(x, width=40, font='Verdana 12 bold')

Label(x, text="Foster's Name", width=20, height=3, font='Verdana 12 bold',
bg='#ce66d4').grid(row=1, column=0)
e1.grid(row=1, column=1, padx=20)
```



```
Label(x, text='Contact Number', width=20, height=3, font='Verdana 12 bold',
bg='#ce66d4').grid(row=2, column=0)
e2.grid(row=2, column=1, padx=20)

Label(x, text='Email_ID', width=20, height=3, font='Verdana 12 bold',
bg='#ce66d4').grid(row=3, column=0)
e3.grid(row=3, column=1, padx=20)

Label(x, text='Address', width=20, height=3, font='Verdana 12 bold',
bg='#ce66d4').grid(row=4, column=0)
e4.grid(row=4, column=1, padx=20)

Label(x, text='Cat', width=20, height=3, font='Verdana 12 bold',
bg='#ce66d4').grid(row=5, column=0)
e5=StringVar()
e5.set("Select A Cat")
list1 = ['Munchkin (5 Months Old)','Siamese (6 Months Old)','Sphynx (3 Months
Old)',\
         'British Shorthair (7 Months Old)','Russian Blue (3 Months Old)',\
         'Persian (6 Months Old)','Bengal Cat (4 Months Old)','Nebelung (4
months Old)']
dl=OptionMenu(x,e5, *list1)
dl.config(width=34, relief='flat', activebackground='light pink', font='Verdana
12 bold', bg='purple', fg='white')
dl.grid(row=5, column=1)

Label(x, text='Date of Adoption', width=20, height=3, font='Verdana 12 bold',
bg='#ce66d4').grid(row=6, column=0)
time = strftime('%d-%m-%Y')
e6=Label(x, text=time, width=20, height=3, font='Verdana 12 bold', bg='#ce66d4')
e6.grid(row=6, column=1)

Button(x, text='Submit', command=show, font='Verdana 12 bold', bg='#52004b',
fg='white').grid(row=8, column=1, pady=20, padx=20, sticky=E)

mainloop()
```

Scanned with CamScanner

```
pic3=PhotoImage(file='cats3.png')
pic3.subsample(3,4)
Label(y, image = pic3, bg='#52004b').place(x=470, y=10)

sel="\n* Name : "
```

Sparkle Cat House

SPARKLE CAT HOUSE ADOPTION CENTRE

Foster's Name	Bhavya
Contact Number	7788554411
Email_ID	bhavya0087@gmail.com
Address	Jogeshwari
Cat	British Shorthair (7 Months Old)
Date of Adoption	04-03-2020

Submit

```
c1.grid(row=1, column=1, padx=20)
```

Scanned with CamScanner

Sparkle Cat House

SPARKLE CAT HOUSE ADOPTION CENTRE

Confirm The Adoption

* Name : Bhavya

* Contact Number : 7788554411

* Email ID : bhavya0087@gmail.com

* Address : Jogeshwari

* Cat : British Shorthair (7 Months Old)

* Date of Adoption : 04-03-2020

EDIT SUBMIT

Subject : Data structu

Sparkle Cat House

SPARKLE CAT HOUSE ADOPTION CENTRE
Confirm The Adoption

*** Name :**

*** Contac**

*** Email II**

*** Address : Jogeshwari**

*** Cat : British Shorthair (7 Months Old)**

*** Date of Adoption : 04-03-2020**

EDIT

SUBMIT

Sparkle Cat House

THANKS FOR ADOPTING A CAT !

OK

Scanned with CamScanner