```
file obj = open("abc.txt","w")# file open (write mode)
fileobj = write("Computer science subjects"+"\n")
fileobj.write("DBMS\n Python\n DS\n")
file obj.close()
fileobj = open("abc.txt","r")
Str1 = fileobj.read()
print("The output of readmethod:" Str1)
file obj.close()
```

>>> The output of read method: Computer Science
Subject
DBMS
Python
DS

```
# readlines()
file obj = open("abc.txt","r")
str2 = fileobj.readline()
Print("The output of readline method:" str2)
file obj.close()
```

>>> The output of readline method Computer
science subject

DBMS
Python
DS

Practical - 01

* Aim :- Demonstrate the use of different file accessing mode, different attributes and Read method.

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

Step 2 :- Now open the file in read mode and than use read(), readline(), 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 file, the file mode in which it is opened whether the file is still open or close and finally the ~~obj~~ output of the softspace attribute.

Step 4: Now open the file object in write mode write some another Content close subsequently — then again open the file object in 'wt' mode that is the update mode and ~~display then append~~ output

Step 5: Open file object in read mode, display the update written Contents and close, open again in 'rt' mode with paramater passed and display the output Subsequently

Step 6: Now open file object in append mode open write method. write content close the file object again: Open the file object in read mode and display the append output.

file attribute.

```
a = file obj. name
Print("Name of file (name attribute:", a)
>>> (Name of file (name. attribut, abc, txt)
b = file obj. (closed
print ("(close) attribute:" b)
>>> (close) attribut: True
c = file obj. mode
Print ("file mode,"(c)
>>> file mode", 'r',
d = file obj. softspace
Print ("softspace, "d)
>>> ("softspace:", 0)
```

wt mode.

```
fileobj = open("abc.txt", "wt")
file obj. write ("Sawasdh")
fib objed. (lose X)
```

rt mode

```
file mode
fileobj = open("abc.txt", "rt")
str1 = file obj. read (r)
Print ("output of rt", str1)
file object. close()
```

>>> ('output of rt', 'sawasdh')

write mode

```
file obj = open('abc.txt", "w")
fileobj = write ("OBMS")
file obj. close()
```

read mode

```
fileobj = open("abc. txt")
str2 = file obj. read().
Print ("output of read
        mode", str2)
```

>>> ('output of read mode 'sawasdh')

append mode.

```
file obj . open ("abc.txt","a")
file obj . write ("Data structure")
file.close()
file object = open ("abc .txt","r")
str3 = file obj . read()
Print ("output of append mode ",str3)
  file obj . (close())
```

>> ("output of append mode ","saurabh","Data structure")

tell()

```
file obj = open ("abc .txt","","r")
 pos = file obj tell ()
Print ("tell();", pos)
Fileobject close()
>> ( tell ();; pos)
```

seek()

```
File obj = open ("abc .txt","r")
 str.4 = file obj . seek (0,0)
 str 8 = Fileobj . read (10)
Print " The begining of the line is = ", str 8)
```

Step 7 :- Open the file object in read mode. declare a variable and perform fileobj dot tell method and store the output Consequently in variable.

Step 8 :- Use the seek method with the argum with opening the file object in read mo and closing subsequently.

Step 9 :- Open file object with read mode also use the readlines method and store th same for Counting the length use th for condition. Statement and display the length.

180 Practical 2:

Aim: Demonstrate the use of Iterable & iterators

Theory: In Python iterator is an object which implements iterator class which has 2 methods namely --iter()--- and --next()---, List, tuple clictionary & the set all represents a iterable object.

Write a program using Iterable objects for displaying the odd numbers in range 1 to 10.

Algorithm:

Step 1: Define a iter() with argument and initialize the vaccum and return that value

Step 2: Define the next() with an argument and compare the upper limit by using a Conditional Statements

Step 3. Now create an object of the given, Class & pass

Code

```
def __iter__(self):
    self.num=1
    return self ....
def next(self):
    if self.num<=10:
        num=self.num
        self.num+=2
        return num
    else:
        raise StopIteration
```

```
>>> y=count()
>>> z=iter(y)
>>> z.next()
1
>>> z.next()
3
>>> z.next()
5
>>> z.next()
7
>>> z.next()
9
>>> z.next()
11
```

Q.7

Coding:

```
class power:
    def __iter__(self)
        self.p = 0
        return self
    def next(self):
        if self.p <= 10
            num = self.p
            self.p += 1
            po = 2**num
            print("2**", self.p-1, "=", po)
            return po
        else raise StopIteration
```

Output:

```
>>> p = power()
>>> x = iter(p)
>>> x.next()
* 0 = 2
>>> x.next()
2**1 = 2
>>> x.next()
2**2 = 2
>>> x.next()
2** 3 = 8
```

Write a program using an iterator for calculating the power of a given number. For instance number entered is 2, then value calculated should be $1, 2^1, 2^2, 2^3, 2^4$

Algorithm:

Step 1: Define iter() with argument and initialize value and return the value.

Step 2: Now define next() with an argument and compare the upper limit by using conditional statement.

Step 3: Now create an object of the given class and pass this object in the iter method.

Algorithm:-

Step 1:- Define a iter() with argument & initialise the value and return the value.

Step 2:- Define the next() with an argument and compare the upper limit by using a conditional statement

Step 3:- Now create an object of the given class and pass this object in the iter method.

Write a program using iterable concept to display multiple of 2 in range 1 to 10

Algorithm:

Step 1:- Define a iter() with argument and Initialise the value and return the value.

Coding:-

```
class fact:
    def __iter__(self):
        self.f = 1
        return self
    def __next__(self):
        if self.f <= 10:
            num = self.f
            self.f += 1
            fac = 1
            for i in range(1, num+1):
                fac = fac*i
            print(self.f-1, "! =", fac)
        else:
            raise StopIteration.
```

>>> output:

```
>>> f = fact()
>>> x = iter(f)
>>> x.next()
    1! = 1
>>> x.next()
    2! = 2
>>> x.next()
    3! = 6.
```

```
def next(self):
    if self.num<10:
        num=self.num
        self.num+=1
        table=2*num
        print("2 *",num,"=",table)
    else:
        raise StopIteration.
```

Output::

```
>>> m=mult()
>>> x=iter(m)
    x.next()
    2*1=2
>>> x.next()
    2*2=4
>>> x.next()
    2*3=6
>>> x.next()
    2*4=8
```

Step 3: Now create an object of the given class and pass this object in the iter method.

7&8 Practical 8:

Aim: Demonitrate the use of exception Handling

Theory: An exception is an event which occurs during execution of program which distrupt the normal flow of program Thus a exception represents object which represents object which represents ~~object~~ error which This object is derived from given class & when the python script raises an exception it must be handled immediately. Otherwise it will terminate & close the program.

Write a program to check the range of the age of the students. In given class & in age does not fall in given range use value error exception otherwise return the valid no.

Algorithm:

Step 1: Define a function which will accept the age of the student from standard input.

code.

```
def accept_age():
    age = int(input("enter your age:"))
    if age>30 or age<18:
        raise value error.
    else:
        Print("your age is",age)
valid=false
while not valid:
    try:
        age = accept_age()
        valid = True
    except value error:
        Print("your age is not in ~~range~~
```

>> Enter your age: 15
your age is not in range
Enter your age:2
your age is not in age
Enter your age:= 77
your age is TT

print ("valid number")
break
except ValueError:
print ("Not a valid number ! try again")

>>> Enter a number : 17.2
Not a valid number ! Try again
Enter a number : 17
Valid number

Q.2)

Step 3: Define the while loop to check whether the boolean expression holds true. Use the try block to accept the age of student & terminate the looping condition

Step 4: Use except with value error & print the message not a valid range.

Write a program to check whether the number in given class & if the number is a floating point use value error as exception for the given input.

Algorithm:

Step 1: Use try block & accept the input using input() & convert it into integer datatype and subsequently terminate the block

Step 2: Use the except block with exception as

Algorithm:

Step 1 : Use the try block & accept the input using input() & then convert it into integer datatype.

Step 2 : Define a function with 2 parameter & divide the no. given by user

Step 3 : Define while loop to check whether the boolean expression holds true

Step 4 : Use except with zero division error & print the message

Code

```
def divide(a,b):
    ans = a/b
    return ans
while True:
    try:
        a = int(input("Enter first number:"))
        b = int(input("Enter second number:"))
        ans = divide(a,b)
        Print("division of", a, "and", b, "is", ans
        break
    except ZeroDivisionError:
        print("Error!")
```

```
>>> Enter first number:1
>>> Enter second number:1
    Division of 1 and 1 is 1
>>> Enter first number:1
>>> Enter second number:0
    Error!
```

24/12/19

result = re.findall( )
print (result)
Print (result)

output:

>>> ('1234', '0567')
>>> ('hello', 'abc')

Practical 4:

Aim: Demonstrate the use of regular expression.

Theory: Regular expression represents the sequence of character which is mainly used for finding & replacing the given pattern in a string & for this we import re module & common usage regular expression involves following functionalities
- Searching a given string
- finding a string
- Breaking a string into small substring

Q.1] Write a regular expression aggregating numeric & alphabetic values from a given string.

Algorithm:

Step 1 : Now apply string & pattern in findall and display the output.

Step 2 : \d is used for matching all decimal digits wheres D is used to match non decimal digits

match string at the beginning of given string

Algorithm:

Step 1 : Import re module & apply a string

Step 2 : Use search() with "\A Python" & string as two parameters.

Step 3 : Now display the output.

Step 4 : Now use if conditional statement for user to know whether the match is found or not.

Code 2:

```
import re
string = "Python is an important language"
result = re.search("n Python", string)
print(result)
if result:
    print("Match found")
else:
    print("Match not found")
```

Output:

```
<re.Match object; span=(0,6);
    match = "Python">
Match found.
```

Q3)

```
import re
li = ["9876543210", "8765432109", "7654321098",
      "6543210987"]
for element in li
    result = re.match("[8-9]{1}[0-9]{9}", element)
    if result:
        print("Correct mobile no")
        print(result.group())
    else:
        print("Incorrect mobile no")
```

output :

```
Correct mobile no
9876543210
Correct mobile no
8765432109
Incorrect mobile no
Incorrect mobile no
```

given total length of digit should be

Algorithms:

Step 1: Import re module & apply a string of mobile nos.

Step 2: Now use for conditional statement to to find if the number starts with 8 or 9 & the total number should length of 10 use match() inside for statement to find the match in given string.

Step 3: Use if conditinal statement to know whether we have a match or not. if we have use group() to display the output & if we don't display incorrect mobile no.

Write a regular expression to extract the word from given string along with subsequently extract the word with space character.

Algorithm:

Step 1: Import re module & apply a string.

Step 2: We findall() to extract a word from given string.

Step 3: Use "\w*" to extract word along with space & use "\w+" to extract word without space

Step 4: Now display the output

Write a regular expression for extraction of first & last word from a string

```
result = re.
print(result)
print(result2)
```

Output:

```
>>> ('Python', '
    ('python', '
```

CODE 5:

```
import re
string = "python, is important"
result = re.findall ("\not", string)
result = re.findall ("\w + ot", string)
print (result)
print (result)
```

Output

>> ['python']

>> ['important']

CODE 6:

Q.6]

```
Import re
string = "Anand 201 24-12-2014"
result = re.findall ("\d{2} -\d{2} -\d{4}", string)
Print (result)
```

Output:

>> ('12-12-2014')

Algorithm:

Step 1: Import re module & apply a string.

Step 2: Use findall() in which use "\w+" as one parameter to find first word of string then use "\w+$" as parameter to find last word of string

Step 3: Now display the result

Write a regular expression for extracting the da
in format dd-mm-yyyy using the
findall() when the string has following format
201 24-12-2019

Algorithm:

Step 1: Import re module & apply string

Step 2: Use findall method & use '\d{2}3
- \d{4}' as an parameter.

Step 3: Now display the output.

② hostname from email id

③ Both username & hostname from email id

Algorithm:

Step 1: Import re module & apply a string

Step 2: Use findall() to find username, hostname & both of email id

Step 3: Use "\w+" for username use "@\w+", "\w+$" for hostname & use "[\w.-]+" for both as parameter in findall()

Step 4: Display the output.

```
result 1 = re.findall("\w+", string)
result 2 = re.findall("[\w\s-]+", string)
print (result 1)
print (result 2)
print (result 3)
```

output:

```
>>> ['abc']
>>> ['tesc : edu']
>>> ['abc', 'tesc : edu']
```

```
l = label (root, text = py...
l = pack()
root.mainloop()
```

#2: label, attributes

```
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 LEFT, padx = 20)
l2 = Label (root, text = "cs", bg = "light blue", fg = "black", font = "1
l2. pack (side = LEFT, pady = "30")
l3 = Label (root, text = "cs!!", bg = "yellow", fg = "black", font = "10"
l3. pack (side = TOP, ipadx = "40")
```

Step 1: Create an object using T

Step 3: Create a variable using and use the text me

Step 4: Use the mainloop c corresponding above

#2:

Step 1: Use the Tk inter the following. feac widget.

Step 2: Create a variab and position

Step 3: Use

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

Step 5 : Now repeat the above steps with the label which takes the following arguments

1) Name of the parent window
2) Text attribute which defines the string.
3) The background color (bg)
4) The foreground fg and then use the pack() with a relevant padding attributes.

```
l4= label (root, text = "...",
          font = "110")
l4.pack (side = TOP, ipady = 50)
root.mainloop()
```

```
from Tkinter import*
root = Tk()
root.geometry("500x500")
def Select():
    Selection = "you just selected" + str(var.get())
    l1 = Label(Text = Selection, bg = "white",
               fg = "green")
    l1.pack(side = Top)
var = StringVar()
l1 = Listbox()
l1.insert(1, "List 1")
l2.insert(2, "List 2")
l1.pack(anchor = N)
R1 = RadioButton(root, text = "option 1", variable = var, value
                 = "option 1", command = Select)
R1 = pack(anchor = N)
R2 = RadioButton(root, text = "option 2", variable = var, val
                 = "option 2", command = select)
R1.pack(anchor = N)
R2 = RadioButton(root, text = "option 2", variable = var, value =
                 command = select)
R2 = pack(anchor = N)
root.mainloop()
```

Practical - 5 (B)

Aim: GUI components

#1:

Step 1: Import the relevant methods from the Tkinter library. Create an object with the parent window.

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

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

Step 4: Now define the parents window and define the option with control variable.

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

Step 6: Create an object from radio button which will take following arguments parent window object, text variable which will take the values optain no 1, 2, 3.... variable argument, corresponding value & bigger the function declared.

using anchor attribute

Step 8 : Finally make use of the mainloop() along with parent object

#2 :

Step 1 : Import relevant methods from the Tkinter library

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

Step 3 : Use the geometry() for laying of the windows

Step 4 : Create an object and use the scrollbar

Step 5 : Use the pack() along with the Scrollbar object with side and fill attribute

#2:

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

#3:

```
# using frame widget
from tkinter import *
window = Tk()
window.geometry("680x500")
Label(window, text="numbers:").pack()
frame = frame(window)
frame.pack()
listNodes = Listbox(frame, width=20, height=20, font=
                    ("Time new Romon", 10))
listNodes.pack(side="left", fill='y')
SCrollbar = Scrollbar(frame, orient="Vertical")
Scrollbar.config(command = listNodes.yview)
Scrollbar.pack(side="right", fill="y")
for x in range(100):
    listNodes.insert(END, str(x))
window.mainloop()
```

#3:

Step 1: Import the relevant libraries from the tkinter method

Step 2: Create an corresponding object of the parent window

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

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

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

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

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

Step 8: Toriggle The events using mainloop

#4:

Step 1: Import relevant methods from tkinter library

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

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

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

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

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

Similarly create the button object and the Modify operation

4:

```
from tkinter import*
window = Tk()
window.geometry("680x500")
frame = Frame(window)
frame.pack()
leftframe = frame(window)
leftframe = pack(side = "left")
rightframe = frame(window)
rightframe.pack(side = "right")
b1 = Button(frame, text = "select", activebackground = "red", fg
b2 = Button(frame, text = "Modify", activebackground
   = yellow", fg = "black")
b3 = Button(frame, text = "Exit", activebackground = "red
   , fg = "green")
b4 = Button(frame, text = "Exit", activebackground = "red
                fg = "green")
b1.pack(side = "LEFT", padx = 20)
b2.pack(side = "right", padx = 30).
b3.pack(side = "bottom", pady = 20)
b4.pack(side = "top").
```

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

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

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

Practical - 5(c)

Aim : Gui components

Step 1 : Import the relevant methods from tkinter libraries

Step 2 : Import tkmessage-box.

Step 3 : Define a parent window object along with the parent window

Step 4 : define a function which will use tkmessagebox with show info method attribute along with info window attribute

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

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

```
# message box
from Tkinter import *
import tkMessageBox
root = Tk()
def function():
    tkMessageBox.showinfo("Info window", "python")
    b1 = Button(root, text="python", command = function)
    b1.pack()
root.mainloop()
```

```
from Tkinter import *
root = Tk()
root.minsize(300,300)
def main():
    top = Tk()
    top.config(bg="black")
    top.title("Home")
    top.minisize(300,300)
    L = Label(top, text = "SANFRANCISCO \n places of Interest
            \n Golden Gate Bride \n Lombard street \n Chinatown
            \n Coit Tower")
    L.pack()
    b1 = Button(top, text="next", Command = Second)
    b1.pack(side = RIGHT)
    b2.Button(top, text = "exit", Command = teriminate)
    b2.pack(side = LEFT)
    top.mainloop()
```

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

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

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

Step 4: Similary define the function second and use the attribute accordingly.

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

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

```
def second ():
    top 2 = Tk()
    top 2. config (bg = "orange")
    top 2. title ("About us!")
    top 2 . minsize (300, 300)
    l = Label (top2, text = "Created by: Tanvi More\n for more
     details contact to our official account")
    l . pack()
    b 3 = Button (top2, text = "prev", command = main )
    b3. pack (side = LEFT)
    b2 = Button (top2, text = "exit", command = terminate)
    b2 . pack (side = RIGHT)
    top 2. mainloop ()

def button ():
    top 3 = Tk()
    top 3. geometry ("300x300")
    b1 = Button (top 3, text = "flat button", relief = FLAT)
    b1. pack()
    b2 = Button (top 3, text = "groove button", relief = GROOVE)
    b2. pack()
    b3 = Button (top 3, text = "raised button", relief = RAISED)
    b3. pack()
    b4 = Button (top 3, text = " SUNKEN button", relief = SUNKEN)
    b4. pack()

    b5 = Button (top3 , text = "ridge button", relief = RIDGE)
    relief = RIDGE
    top3 . mainloop()
    def terminate ():
        quit()
```

```
b5= Button (root,text = "TOUR DETAILS", Command = main)
b5.pack()
b6 = Button (root, text = "Button DETAILS", Command = button)
b6.pack()
root.mainloop()
```

Practical - 5 (D)

Aim : Gui Components

Step 1 : Import relevant methods from the tkinter library

Step 2 : Create parent window object and use the config method along with background color attribute specified

Step 3 : define a function finish with the messagebox widget which will display a message i.e a warning message and subsequently terminate the program

Step 4 : define a function info use a listbox widget along with the object of the same use the listbox object along with pack method and insert the same and finally use the grid() with ipadx attribute.

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

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

Step 8 : Similarly create another frame object - as done by stp 7.

Stp 9 : Create another object & use the sub sample()

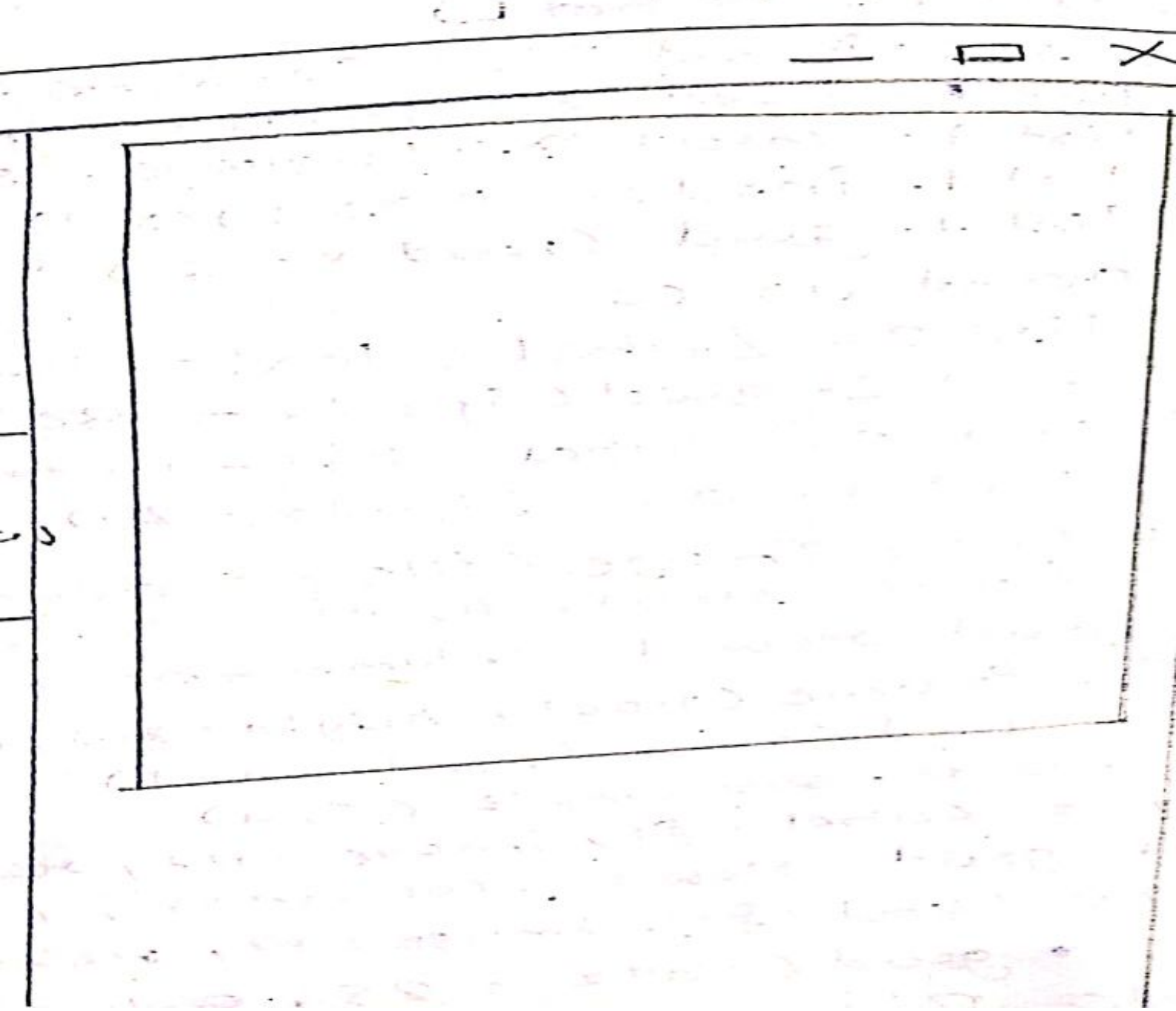
Step 10 : Use label widget along with the frame object relief attribute and subsequently use the grid()

Stp 11 : Now create button object dealing with diffrent section of frame.

```
from Tkinter import*
root = Tk()
root = config (bg = "grey")
def finish ():
    messagebox . askokcancel ("Warning", This will End the
                                Program").
    quit()
def info ():
    list 1 = listbox()
    List 1 . insert (1," Co. Name : Apple ")
    list 1 . insert (2," Products : iphone")
    list 1 . insert (3," Language : Swift ")
    list 1 . insert (4," OS : IOS ")
    list 1 . grid (ipad x = 30)
def about us ():
    list 2 = Label ( text = "About us")
    List 2 . grid (ipadx = 30)
    list 3 = Label ( text = "steve jobs theatre March 2020")
    list 3 . grid(ipadx = 24)

P1 = Photo Image ( 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)
```

bout us", relief = SUNK
bout us)
= 2, padx = 5)
, relief = RAISED, command
2, ipadx = 15)



Aim: ... human face and conveying

Write a program to draw human face using GUI

Algorithm:

Step 1: Import relevant methods from tkinter library

Step 2: Create an object corresponding to the parent window from Tk()

Step 3: Create an object from canvas() & place it onto parent window along with height & width

Step 4: Now use pack() for positioning as widget onto the parent window

Step 5: Now create object face & use object create oval() with coordinates 50, 50, 350, 350, outline = 'black', fill = "yellow" as attribute to create face

CODE:

```
from tkinter import*
root = Tk()
C = canvas (root, width = 500, height = 500)
C.patch()
face = C.create_oval (50,50,350,350, outline = "black", fill = "yellow")
eye1 = C.create_oval (125, 125, 275, 175, fill = "black")
eye2 = C.create_oval (125, 125, 275, 175, fill = "black")
mouth = C.create_arc (125, 225, 275, 275, start = 0, extent = -180,
                      width = 30, fill = "red")

root.mainloop()
```

Step 6: Now create eye object & again use object Create_oval() with appropriate co-ordinates along with fill us attribute to create left eye.

Step 7: Now repeat the same step 6 to create right eye

Step 8: Create an object mouth & use object. create arc() with appropiate co-ordinates start = 0 extent = -180 & fill = "red") width = 5 as attribute do create mouth

Step 9: Finally use the mainloop().

W.A.P to convert celsius into farhenheit using GUI

Algorithm

Step 1: Import all relevant methods of tkinter library

Step 2: Create object corresponding to the parent window from TKU

Step 3: Now initialize fahrenheit as DoubleVar() & set it to 320.

Step 4: Now define a function convert with argument celsius & to convert celsius into fahrenheit using set()

Step 5: Now create an object 2 using label() & place it onto parent window & use text attribute as enter a no.

Step 6: Now use grid() for position the object onto the parent window

Step 7: Initialize celsius as integer using IntVar()

#Code:

```
from tkinter import *
window = Tk()
fahrenheit = DoubleVar()
fahrenheit.set(320)
def convert(celsius):
    fahrenheit.set((9.0/5.0) * celsius + 32)
l1 = Label(window, text="Temperature in Celsius")
l1.grid(row=0, column=0)
e = Entry(window, textvariable=celsius)
e.grid(row=0, column=1)
celsius = IntVar()
l2 = Label(window, textvariable=fahrenheit)
l2.grid(row=2, column=0, columnspan=2)
B = Button(window, text="Calculate", command=lambda: conv
                                        [celsius.get()])
B.grid(row=1, column=0, columnspan=2)
window.mainloop()
```

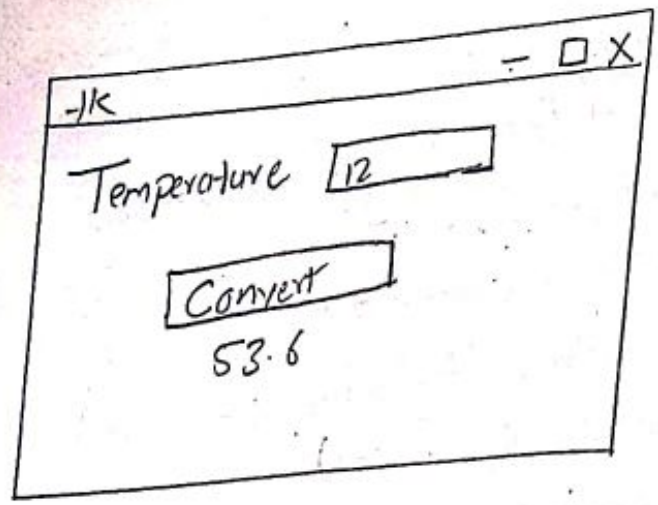
Step 9: Now use grid() for positioning the object onto parent window with textvariable attribute.

Step 10: Now again use label() along with textvariable attribute to display output & use grid() for positioning.

Step 11: Use mainloop()

080

6



Step 9: Now use grid() for positioning the object onto parent window with textvariable attribute

Step 10: Now again use label() along with textvariable attribute to display output & use grid() for positioning

Step 11: Use mainloop().

Aim : With ... arithmetic operation

W.A.P to find factorial of a number using GUI

Algorithm :

Step 1 : Import relevant methods from Tkinter library

Step 2 : Now define a factorial to calculate factorial using recursive function

Step 3 : Define another function calculate to call factorial function

Step 4 : Now create an object with entry() and use pack() for positioning on parent window.

Step 5 : Now create an object with button() along with command = attribute to calculate factorial

Step 6 : Now again create an object with label() to show output

Step 1 : Use the mainloop()

code:

```
from tkinter import *
def factorial(n):
    if n==0 or n==1:
        return 1
    else:
        return n * factorial(n-1)
def calculate():
    result = factorial(int(entry_text.get()))
    info.config(text=result)

root = Tk()
entry_text = Entry(root)
entry_text.pack()
btn = Button(root, text="Calculate", command=calculate)
btn.pack()
info = Label(root, text="Factorial")
info.pack()
root.mainloop()
```

```
        res = int(e1.get()) + int(e2.get())
        l3.config(text=res)
    elif int(v.get())==2:
        res= int(e1.get()) - int(e2.get())
        l3.config(text = res)
    elif int(v.get()) == 3:
        res = int(e1.get()) * int(e2.get()):
        l3.config(text=res)
    else:
        res= int(e1.get()) / int(e2.get())
        l3.config(text = res)

root=Tk()
l1=label(root, text="Enter a no")
l1=grid(row=0, column=0)
e1.Entry(root)
e1.grid(row=0, column=1)
l2=Label(root, text="Enter second no")
l2= grid(row1, column=0)
e2=Entry(root)
e2.grid(row=1, column=1)
```

1. W.A.P to perform arithemetic operation on 2 numbers using GUI:

Algorithm:

Step 1: Import relevant methods from Tkinter library

Step 2: Now create an object corresponding to parent window

Step 3: Now define a function calculate to carryout arithemetic operations on 2 numbers

Step 4: Now create an object with label() as num1 & num2 and use the grid() to placed once parent window

Step 5: Create object with entry() to take input from user()

Step 6: Now Initialize as integer using IntVar()

Step 7: Now create 4 objects with RadioButton() to choose any one of the arithemetic operation, & use grid() for positioning onto parent window

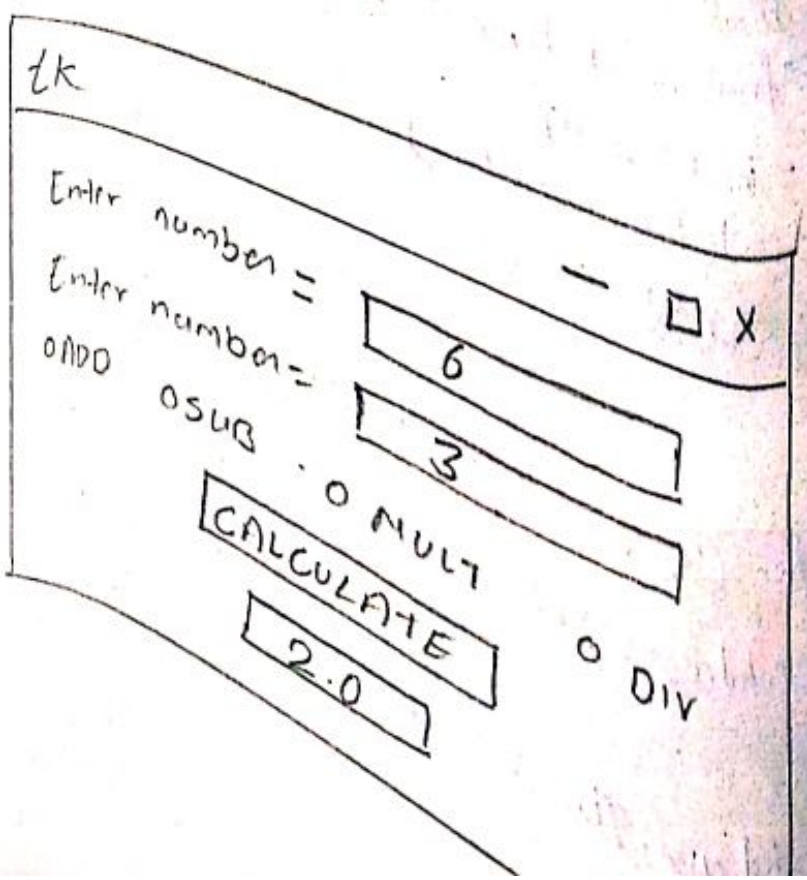
Step 8: Now create a object with button() along with Command & attribute to carryout the arithmetic operation of users choice.

Step 9: Now create a object with label() to show the output.

Step 10 : Use the mainloop()

```
V= IntVar()
r1 = RadioButton(root, text="add", variable=v, value=1)
r1 = grid (row=?, column=0)
r2 = RadioButton(root, text="Sub", variable=v, value=2)
r2 = grid(row=?, column=1)
r3 = RadioButton(root, text="Mult", variable=v, value=3)
r3. gride(row=?, column=2)
r4 = Radi Button (root, text="Div", variable=v, value=4)
r4 grid (row=2, column=3)
B = Button(root, text="Calculate", command=calculate)
B.grid(row=3, column=1, columnspan=2)
l3 = label(root)
l3. grid(row=4, column=1)
root.mainloop
```

output



code.

```
import socket
def server_program:
    host = socket.hostname()
    port = 5000
    server_socket = socket.socket()
    server_socket = bind(host, port)
server_socket.listen(2)
conn, address = server_socket.accept()
print("Connection from:" + str(address))
while True:
    data = conn.recv(1024).decode()
    if not data:
        break
    print("from connected user:" + str(data))
    data = input("->")
    conn.send(data.encode())
    conn.close()
```

output:

```
Connection from: ('127.0.0.1', 57822)
from connected user: Jai
->Hello
from connected user: How are you?
->Good
from connected user: Awesome!
->ok then, bye!
```

Practical - 8

Aim: Demonstrate the use of socket module and server client programs.

WAP to demonstrate the use of socket module program

Algorithm:

Step 1: Import the socket module to import relevant methods

Step 2: Define a function as server-program to get hostname.

Step 3: Now get value for port variable to initialize port not above 1024.

Step 4: Use socket() to get instance.

Step 5: Now we bind() to bind port address and port together to configure how many client and server can list simultaneously.

Step 6: Now use accept() to accept new connection

Step 7: Now print the address.

Step 8: Use while loop to as long as to receive data stream.

Step 9: Now close the program.

Algorithm:

Step 1: Import socket module to import methods that are relevant

Step 2: Define a function client_program to get the hostname give port a value 5000.

Step 3: Now again initiate by using socket.socket()

Step 4: Use connect() to connect the server

Step 5: Now take the input msg("→")

Step 6: Use while conditional loop to send a message

Step 7: Now use decode to receive response

Step 8: Now show the data.

Step 9: Again take input.

Step 10: Close the program.

```
import socket
def client_program():
    host = socket.gethostname()
    port = 5000
client_socket = socket.socket()
client_socket = connect((host, port))
message = input("→")
while message.lower().strip()!='bye':
    client_socket.send(message.encode())
    data = client_socket.recv(1024).decode()
print("Received from server="+data)
message = input("→")
client_socket.close()
```

#output

→ Hi

Received from server = Hello

→ How are you

Received from server: good

→ Username

Received from server: ok-then bye!

#Code IN SHELL ENVIRONMENT

```
>>> import sqlite3
>>> import = sqlite3.connect("Student1.db")
>>> cur = conn.cursor()
>>> cur.execute("Create table student (roll_no int(5) primary key, name varchar
(50) not null, address varchar(50) not null, class varchar(
, dob date)")
<sqlite3.cursor object at 0x0322EBED)
>>> Car.execute("insert into student values (101, "Lalit", " Mira Road", "FYCS"
,"10/08/2001")
<sqlite3.cursor object at 0x0322EBE0>
>>> Car.execute["select * from student")
<sqlite3.cursor Object at 0x 322EBED)
>>> Car.fetchall()
[(101,'lalit','Mira Road','FYCS','10/08/2001'),
(102,'Shubham','Mumbai','FYCS','30/07/2002')]
>>> Cur.execute["update student set dob = "30/08/2002" where
roll_no = 102")
<sqlite3.cursor object at 0x322EBED)
>>> Cur.execute("Select * from student where address = 'Mumbai')
<sqlite3.cursor object at 0x0322EBED>
>>> cur.fetchall()
[('102' 'Shubham', 'Pawai', 'Fycs', '30/08/2002')]
>>> Cur.execute("Commit")
<sqlite3.cursor object at 0x0322EBED)
>>> Cur.close()
```

Practical . 9.

Aim : Demonstrate the use of database Connectivity

Algorithm :

Step 1 : Import sqlite3 module to import relevant methods

Step 2 : Now initialize a variable con to connect by using connect() to a new database using extension.

Step 3 : Now initialize a variable to connect to cursor()

Step 4 : Now use cur = execute() to create a table, insert values into table & use Dml, DDL, Statements to manipulate the data in the database.

Step 5 : Use fetchall() to show the output.

Step 6 : Use commit to save all changes

Step 7 : Use close() to terminate the program.