```
# read() and write():
fo = open("abc.txt","w")
fo.write("Computer Science subject"+"\n")
fo.write("DBMS\n Python \n DS \n")
fo.close()
fo = open("abc.txt","r")
str1 = fo.read()
print("The output of read method:",
      str1)
fo.close()
>>> The output of read method:
    computer science subject
    DBMS
    Python
    DS
# readline():
fo = open("abc.txt","r")
fo.readline()
str2 = fo.readline()
print("The output of readline():",str2)
fo.close()
>>> The output of readline():
    computer science subject
```

m

Practical 7:

Aim: Demonstrate the use of different file accessing mode, different attributes and read method.

Algorithm:

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

Step 2: Now open the file in read mode and then 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 softspace attribute.

Step 4: Now open the file object in write mode write some another content close subseq[illegible] entry then again open the file object in 'w+' mode [illegible] is the update mode and write contents.

Step 5: Open file object in read mode display the update written contents and close, open again in 'r+' mode with parameters passed and display the output subsequently.

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

Step 7: Open the fileobject in read mode, declare a variable & perform fileobject dot tell

```
# readlines():
  fo = open("abc.txt", "r")
  str3 = fo.readlines()
  print("The output of readlines():", str3)
>> The output of readlines():
  computer science subject
  DBMS
  Python
  OS
# file attributes:
  a = fo.name
  print("Name of file (name attribute):", a)
  b = fo.closed
  print("closed attribute:", b)
  c = fo.mode
  print("File mode:", c)
  d = fo.softspace
  print("softspace:", d)
>> Name of file (name attribute): abc.txt
>> close attribute: True
>> File mode: r
>> softspace: 0
# w+ mode: & r+ mode:
  fo = open("abc.txt", "w+")
  fo.write("ABC")
  fo.close()
```

```
fo = open("abc.txt", "r+")
str1 = fo.read(3)
print("output of r+:", str1)
fo.close()
>>> output of r+: ABC
```

~~write~~ append and read mode:

```
fo = open("abc.txt", "a")
fo.write("Data structure")
fo.close()
fo = open("abc.txt", "r")
str2 = fo.read()
print("output of append mode:", str2)
fo.close()
>>> output of append mode: ABC, Data
                           structure
```

tell(): & seek():

```
fo = open("abc.txt", "r")
pos = fo.tell()
print("tell():", pos)
fo.close()
fo = open("abc.txt", "r")
str4 = fo.seek(0,0)
str5 = fo.read(10)
print("the begining of line:", str5)
>>> tell(): 0
>>> the begining of line:
    computer sc
```

method and store the output conse-
quently in variable

Step8: Use the seek method with the arguments with opening the fileobject in read mode and closing subsequently.

Step9: Open fileobject with read mode also use the readlines method and store the output consequently in & print the same for counting the length use the for condition statement and display the length.

Practical 2:

- Aim: Demonstrate the use of iterables & iterators.

- Theory: In python, iterator is an object which implements iterator class which has 2 methods namely __iter__() and next(). list, tuple, dictionary & the set all represents a iterable object.

Q.1) 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 value and return that value.

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

```
# CODE:
class odd:
    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
```

* code:
```
class power:
    def __iter__(self):
        self.num=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
```

```
>>> p=power()
>>> x=iter(p)
>>> x.next()
2**0 = 1
>>> x.next()
2**1=2
>>> x.next()
2**2=4
>>> x.next()
2**3=8
```

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

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 & pass this object in the iter method

Q-3] write a program using iterable con
to find factorial of number in
range 1 to 10:

Algorithm:

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

Step 2: define the next() with an arg
ument and compare the upper
limit by using a conditional
statement.

Step 3: Now create an object of the give
class & pass this object in the
iter method

Q-4] write a program using iterable con
pt to display multiple of 2 in
range 1 to 10.

Algorithm:

Step 1: define a iter() with argument
& initialise the value and
return the value

code:

```
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
```

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

code:

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

>>> 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 2: Define the next() with an assignment and compare the upper limit by using a conditional statement

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

Practical 3:

Aim: Demonstrate the use of Exception Handling

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

1] Write a program to check the range of the age of the students in given class & if 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 <16:
        raise ValueError
    else:
        print ("your age is", age)
valid = False
while not valid:
    try:
        age = accept_age()
        valid = True
    except ValueError:
        print ("your age is not in
                range")
```

```
>>> Enter your age: 15
    your age is not in range
    Enter your age: 32
    your age is not in range
    Enter your age: 17
    your age is 17
```

code:

```
while True:
    try:
        a = int(input("Enter a number"))
        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
```

Step 2: Use if conditional to check whether the input age fall in range & to return the age else use ValueError exception

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.

Q3 Write a program to check whether the number is given class & if the number is a floating point use ValueError 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

Step2: use the except block with exception as valueerror & display appropriate message. to which code is part of try block

3. write a program to demonstrate use of zerodivision error.

Algorithm:

Step1: use the try block & accept the input using input() & then convert it into integer datatype.

Step2: define a function which with 2 parameter to divide the no-s given by user

Step3: define while loop to check whether the boolean expression holds true

Step4: use except with zerodivision 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!
```

```
# CODE 1:
  import re
  string = "hello1234 abc4567"
  result = re.findall("\d+", string)
  result1 = re.findall("\D+", string)
print(result)
print(result1)
# output:
>>> ['1234', '4567']
>>> ['hello', 'abc']
```

Aim: Demonstrate the use of regular expression.

Theory: Regular expression represents the sequence of characters which is mainly used for finding & replacing the given pattern in a string and for this we import re module and common usage of regular expression involve following functionalities:
- Searching a given string
- Finding a string
- Breaking a string into smaller substring
- Replacing part of string

Q1] Write a regular expression segregading numeric and alphabetic values from a given string

Algorithm:

Step1: Now apply string & pattern in findall() and display the output

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

write a regular expression for finding the match string at the beginning of given sequence.

Algorithm:

Step 1: import re module and apply a string

Step 2: use search() with "\APython" and 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("\APython", string)
print(result)
if result:
      print("Match found")
else:
      print("Match not found")
```

Output:

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

```
# CODE 5:
  import re
  li = ["9876543210", "8765432109",
       "7654321098", "65432109867"]
  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
    9867654321
    correct mobile no
    8765432109
    incorrect mobile no
    incorrect mobile no
```

[5] write a regular expression to check whether the given mobile number starts with 8 or 9 & the total length of digit should be atmost 10

Algorithm:

step1: import re module and apply a string of mobile no's

step2: Now use for conditional statement to find if the number starts with 8 or 9 and the total number should length of 10. use match() inside for statement to find the match in given string

step3: Use if conditional statement to know whether we have a match or not. if we have use group() to display the output and if we don't display incorrect mobile no

Q4] Write a regular expression for extracting a word from given string along with space character in between the word and subsequently extract the word without space character

Algorithm:

Step1: Import re module and apply a string

Step2: Use findall () to extract a word from given string

Step3: Use "\w*" to extract word along with space & use "\w+" to extract word without space

Step4: Now display the output

Q5] Write a regular expression for extracting first and last word from a string

```
# CODE 4 :
import re
string = " Python is important"
result1 = re.findall ("\w*", string)
result2 = re.findall ("\w+", string)
print (result1)
print (result2)
```

```
# output :
>>> ['Python', '', 'is', '', 'important', '']
['Python', 'is', 'important']
```

```
# CODE 5:
import re
string = "Python is important"
result = re.findall ("^\w+", string)
result1 = re.findall ("\w+$", string)
print (result)
print (result1)

# output :
>>> ['python']
>>> ['important']

# CODE 6:
import re
string = "Amit 201 24-12-2019"
result = re.findall ("\d{2}-\d{2}
        -\d{4}", string )
print( result)
output :
>> ['24-12-2019']
```

Algorithm :

Step1: import re module and apply a string

Step2: 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

Step3: Now display the result

Q] write a regular expression for extracting the date in format dd-mm-yyyy by using the findall() where the string has following format Amit 201 24-12-2019

Algorithm:

Step1: import re module and apply string

Step2: use findall method and use '\d{2}-\d{2}-\d{4}' as an parameter

Step3: Now display the output

Q.7] write a re for extracting the
① username from email id
② hostname from email id
③ Both username & hostname from email id

Algorithm:

step1: import re module and 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 and use "[\w\.-]+" for both as parameter in findall ()

step4: Display the output

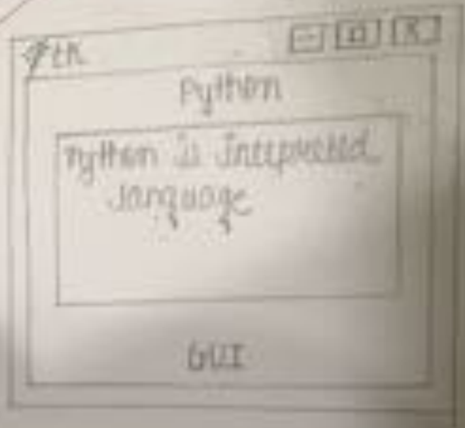
```
# CODE 7:
import re
string = "abc@tcsc.edu"
result1 = re.findall ("^\w+", string)
result2 = re.findall ("@\w+.\w+$",
                         string)

result3 = re.findall ("[\w\.-]+",
                         string )

print (result1)
print (result2)
print (result3)

# output:
>>> ['abc']
>>> ['tcsc.edu']
>>> ['abc', 'tcsc.edu']
```

[illegible]

# CODE:

```
from tkinter import*
root=Tk()
l1=Label(root, text="Python")
l1.pack(padx=20, pady=50, side=
                              TOP)

l2=Label(root, text="GUI")
l2.pack(ipadx=50, ipady=80,
              side=BOTTOM)

T1=Text(root)
quote="Python is interpreted
            language"

T1.insert(END,quote)
T1.pack(padx=70, pady=90, side=
                   RIGHT)

root.mainloop()
```

# output:



Practical 5:

Aim: Demonstrate the use of GUI

write a program to explain types of padding

Algorithm:

Step1: Import the relevant methods from tkinter library

Step2: Create an object corresponding parent window from the Tk()

Step3: Create an object from Label() and place it onto parent window with text attribute as python

Step4: Now use pack() along with argument padx, pady & side

Step5: Now again create a object from Label() and place it onto parent window with text attribute as GUI

Step6: Use pack() along with arguments ipadx, ipady, side

Step7: Now create a object from text() and place it onto parent window

Step8: Now declare a string variable with a sentence in it

Step9: Now use insert () with arguments END & string variable to it

Step10: Now use pack() along with attributes padx, pady, side

Step11: Finally use the mainloop()

Q2] Write a program making use of radiobutton & control variable for selection of given choice.

Step1: Import the relevant methods from tkinter library

Step2: Define a function and define a variable which will keep track of options selected

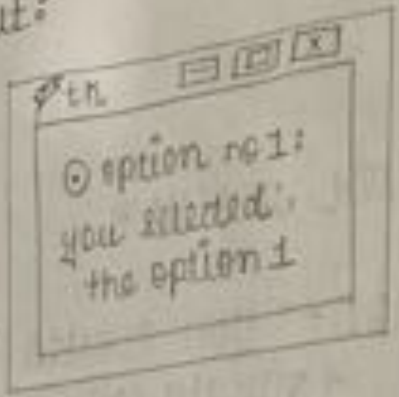
Step3: Now use config() along with label object and use that attribute for displaying the selection made

Step4: Now define the parent window object outside the function definition and define control variable

#CODE:

```
from tkinter import*
def sel():
    selection = "You selected the option"
          + var.str(var.get())
    label.config(text = selection,
              justify = LEFT)

root = Tk()
var = IntVar()
R1 = Radiobutton(root, text = "Option1",
    variable = var, value = 1, command
          = sel)
R1.pack(anchor = W)
label = Label(root)
label.pack()
root.mainloop()
```

output:



code:

```
from tkinter import*
root=Tk()
s=scrollbar(root)
t=Text(root, height=1, width=20)
t.pack(side=RIGHT, fill=Y)
s.pack(side=RIGHT, fill=Y)
s.config(command=t.yview)
t.config(yscrollcommand=s.set)
t.insert(END, para)
para="The mainloop in the GUI
 based application make the given
 widget available"
root.mainloop()
```

Step5: Now define an object corresponding to radiobutton with the following attributes parent window, textvariable, variable, value, command

Step6: Likewise define use pack method simultaneously by defining anchor attribute.

Step7: Now define label object & put it onto the parent window

Step8: finally use the mainloop()

Q3 Write a programme implementing a scrolling feature using the relevant python method.

Step1: Import the relevant methods from tkinter library

Step2: Create an object corresponding parent window from the Tk()

Step3: Create an object from scrollbar() & place it onto the parent window

Step4: Create an object from text() placing it onto the same parent window with height & width attribute specified

Step5: Use the pack() with argument side & fill

Step6: Create an object from the scrollbar() & use the pack() & config() & use the command & config() along with

Step7: Now use config() object & use the scrollbar the command attribute.

Step8: Similarly use the config() with text object and use the y scroll command attribute

Step9: Now define a string variable & use the insert() along with text object & finally use the mainloop()

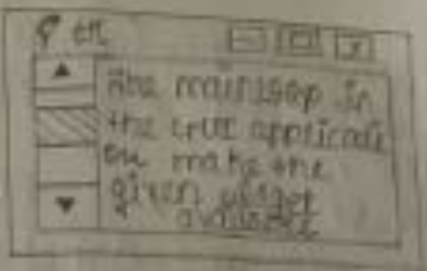
Q.4] Write a program for implementing the frame widget & explain its relevance.

Step1: Import the relevant methods from tkinter library

Step2: Create an object corresponding to the parent window from the Tk()

Step3: Create an object from frame() & place such an object onto the parent window so created

output:

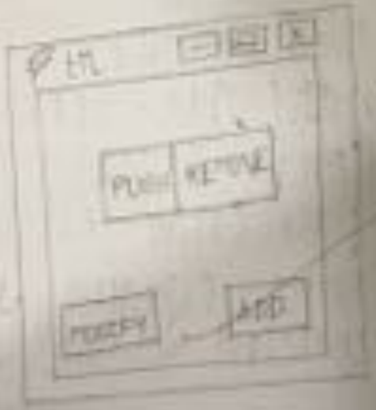


tk
the mainloop in the true application make the given widget available

CODE:

```
from tkinter import *
root = Tk()
frame = Frame (root)
frame . pack (padx = 20, pady = 50, side
              = TOP)
leftframe = Frame (root)
leftframe . pack ( side = LEFT)
rightframe = Frame (root)
rightframe . pack (side = RIGHT)
buttonpush = Button ( frame , text = "PUSH",
    activebackground = "red", fg = "blue",
  bg = "red")
buttonpush. pack (side = LEFT)
```

```
buttonremove = Button (frame, text
    = "remove", fg= "green", bg=
    "yellow")
buttonremove.pack(side= BOTTOM)
buttonadd = Button (rightframe,
        text= "add")
buttonadd .pack (side= LEFT)
buttonmodify= Button ( leftframe, .
    text= "Modify")
buttonmodify.pack (side= RIGHT)
root mainloop()
```
output :



47

step4: use the pack() for positioning the widget onto the parent window.

step5: Create an object formed as left frame and position it onto left side of parent window. similarly create right frame object and position on right side of parent window.

step6: Now create a button object and place it onto the frame widget with text attribute foreground colour & background colour position the button onto left side.

step7: Similarly create another button object names as remove and put it onto frame and position it onto right side. Further create add button & put it onto right frame positioning onto right side. and create modify button & put it onto left frame & position onto left frame.

step8: Finally use the mainloop().

Practical 6

Aim: Demonstrate the use of GUI by creating a human face and converting celsius into farenheit.

Q.1] write a program to draw human face using GUI.

Algorithm:

Step1: import relevant methods from tkinter library

Step2: create an object corresponding to the parent window from Tk()

Step3: create an object from Canvas () & place it onto parent window along with height & width

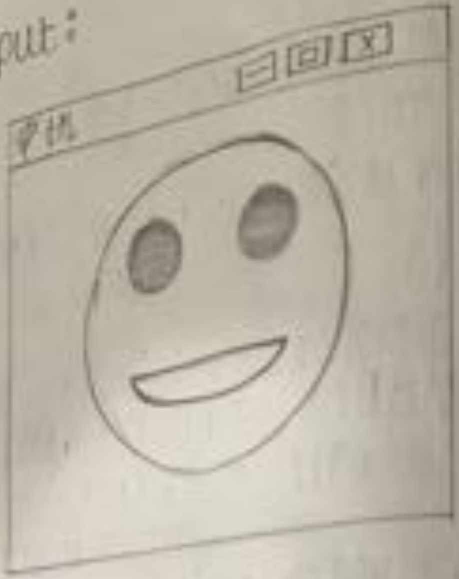
Step4: Now use pack() for positioning of widget onto the parent window

Step5: Now create an object face & use object create_oval () with coordinates 50,50,350,350 & outline="black", fill=" yellow

CODE:

```
from tkinter import*
root = Tk()
C = Canvas (root, width=500, height =
                                  500)
C.pack()
face= C.create_oval(50,50,350,350,
        outline = "black", fill="yellow")
eye1 = C.create_oval(125,125,175,175,
                      fill=" black")
eye2 = C.create_oval(225,125,275,175,
                      fill=" black")
mouth= C.create_arc(125,225,275,275,
     start = 0, extent= -180, width=5,
                    fill="red")
root.mainloop()
```

#output:



as attribute to create face

Step 6: Now create eye1 object & again use object.create_oval() with appropriate coordinates along with fill as 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 appropriate coordinates, start = 0, extent = -180 & fill = "red", width = 5 as attribute to create mouth

Step 9: Finally use the mainloop()

Q2] Write a program to convert celsius into fahrenheit using GUI

Algorithm:

Step1: Import all the relevant methods in tkinter library

Step2: Create object corresponding to the parent window from Tk()

Step3: Now initialise fahrenheit as DoubleVar() & set it to 32.0

Step4: Now define a function convert with argument celsius & to convert celsius into fahrenheit using .set()

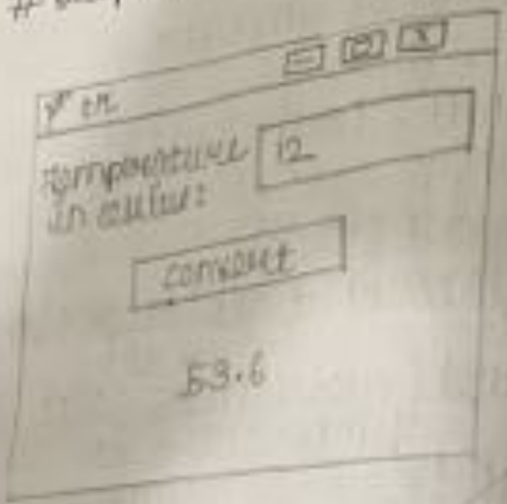
Step5: Now create an object l2 using Label() & place it onto parent window & use text attribute as enter a no:

Step6: Now use grid() for position the object onto the parent window.

CODE:

```
from tkinter import*
window = Tk()
fahrenheit = DoubleVar()
fahrenheit.set(32.0)
def convert (celsius):
    fahrenheit.set((9.0/5.0)* celsius
                  + 32)
l1 = Label( ~~root~~ 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, column
                          span = 2)
B = Button(window, text="calculate",
     command=lambda : convert
            (celsius.get()))
B.grid (row=1, column=0, columnspan=2
~~root~~.mainloop()
window
```

#output:



step 6: Initialize celsius as integer using IntVar()

step 7: Create another object & use entry widget to enter the input and place it onto the parent window

step 8: Now use grid() for positioning the object onto parent window with [illegible]

step 9: Now again use label () along with textvariable attribute & to display output & use grid () for positioning

step 10: Finally use mainloop()

Practical 7

Aim: write a program to find factorial of number & use arithmetic operations on two numbers using GUI

Q.I] write a program to find factorial of number using GUI

Algorithm:

step1: import relevant methods from tkinter library.

step2: Now define a function factorial to calculate factorial using recursive function

step3: define another function calculate to call factorial function

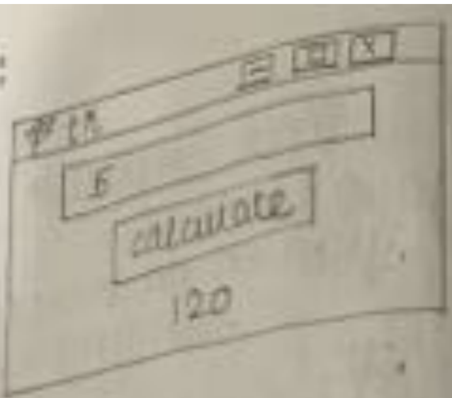
step4: Now create an object with label entry() and use pack() for positioning on parent window

step5: Now create and object with button() along with command = attribute to calculate

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(entrytext
                          .get()))
    info.config(text= result)
root = Tk()
entrytext = Entry(root)
entrytext. pack()
btn = Button(root, text = "calculate",
             command = calculate)
btn pack()
info = Label (root, text = "factorial"
info. pack()
root mainloop()
```

output:



#CODE:
```
from tkinter import *
def calculate():
    if int(v.get()) == 1:
        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)
```

Step6: Now again create an object with label() to show output

Step7: Finally use the mainloop()

8] write a program to perform ~~attribute~~ arithmetic operation on 2 numbers using GUI

Algorithm:

Step1: Import relevant methods from tkinter library

Step2: Now create an object corresponding to parent window

Step3: Now define a function calculate to carry out arithmetic operations on 2 numbers.

Step4: Now create object with label () as num1 & num2 and use grid () to place it onto parent window

Step5: create objects with entry () to take input from user ()

Step 6: Now initialize r as integer using IntVar()

Step 7: Now create an objects with Radiobutton() to choose any one of arithmetic operators & use grid() for positioning onto parent window

Step 8: Now create a object with command button() along with command attribute to carry out the arithmetic operation of user's choice

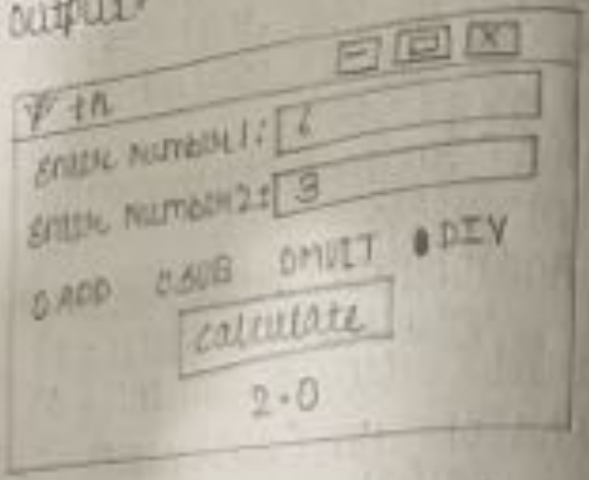
Step 9: Now create a object with label() to show output

Step 10: Finally use the mainloop()

```
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 2nd no:")
l2.grid(row=1, column=0)
e2 = Entry(root)
e2.grid(row=1, column=1)
r = IntVar()
r1 = Radiobutton(root, text="add",
        variable = v, value=1)
r1.grid(row=2, column=0)
r2 = Radiobutton(root, text="sub",
        variable = v, value=2)
r2.grid(row=2, column=1)
r3 = Radiobutton(root, text="Mult",
        variable = v, value=3)
r3.grid(row=2, column=2)
r4 = Radiobutton(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:



```
tk
Enter Number1: 6
Enter Number2: 3
○ADD  ○SUB  ○MULT  ●DIV
    Calculate
      2.0
```

#CODE:

```
import socket
def server_program:
    host = socket.gethostname()
    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))
```

Practical 8:

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

Write a program to demonstrate use of socket module and server client programs

Algorithm:

```
Step1: Import the socket module to
       import relevant methods.
Step2: define a function as server_progr-
       am to get hostname.
Step3: Now get value for port variable
       to initialise port no above 1024
Step4: use .socket() to get instance
Step6: Now use bind() function to bind
       host address and port together
       to configure how many client
       the server can list simultaneously
Step7: Now use accept() to accept new
       connection
```

Step8: Now print the address

Step9: Use while loop as True to receive data stream

Step10: Now close the program

(For socket client program)

Algorithm:

Step1: Import socket module to import methods that are relevant

Step2: Define a function client_program get the host name & give port a value 5000

Step3: Now again initiate by using socket.socket()

Step4: Use connect() to connect the server

Step5: Now take the input ("→")

```
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()
```

(Now run the program & now again write client program)

```
# CODE:
  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 for socket_program
$ python3.6 socket_server.py
connection from : ('127.0.0.1', 57822)
from connected user : Hi
→ Hello
from connected user : How are you?
→ Good
from connected user : Awesome !
→ OK then, bye!
# output for client_program
$ python3.6 socket_client.py
→ Hi
Received from server : Hello
→ How are you?
Received from server : Good
→ Awesome !
Received from server : Ok bye then,
                                  bye!

→ Bye
```

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 by using close()

Practical 9

Aim: Demonstrate the use of database connectivity

Algorithm:

step1: Import sqlite3 module to import relevant methods

step3: Now initialize a variable conn to connect by using connect() to a new database using extension .db .

step3: Now initialize a variable to connect to cursor()

step4: Now use cur.execute() to create a table, insert values into table & use DML, DDL statements to manipulate the data in the database

step5: use fetchall() to show the output

step6: use commit to save all changes

step7: use close() to terminate the program.

CODE IN SHELL ENVIRONMENT

```
>>> import sqlite3
>>> conn = 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 (50), dob
     date)')
<sqlite3.cursor object at 0x0322EBED>
>>> cur.execute ('insert into student values
   (101, "Ankita", "borivali", "FyCs", "12/07/2001")
<sqlite3.cursor object at 0x0322EBED>
>>> cur.execute ('insert into student values
   (102, "Esha", "kandivali", "FyCs", "13/08/2001")
<sqlite3.cursor object at 0x0322EBED>
>>> cur.execute ("select * from student")
<sqlite3.cursor object at 0x0322EBED>
>>> cur.fetchall()
[(101, 'Ankita', 'borivali', 'FyCs', '12/07/2001')
 (102, 'Esha', 'kandivali', 'FyCs', '13/08/2001)]
>>> cur.execute ("update student set
      dob = "13/09/2002" where roll_no = 101')
<sqlite3.cursor object at 0x0322EBED>
```

```
>>>cur.execute('select * from
   student where address = "Kandivali
<sqlite3.cursor object at 0x0322EBED>
>>> ~~fetch~~ cur.fetchall() .
  [(102, 'Esha', 'Kandivali', 'FYCS',
        '13/08/2001')]
>>> cur.execute('commit)
<sqlite3. cursor object at 0x0322EBED
>>> cur.close()
```