Practical

* **Objective** : Demonstrate the uses of different file accessing mode, different attributes of read method.

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

Step 2 : Now open the file in read mode and than use read ( ), readline ( ), 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 output of the softspace attribute

Step 4 : Now open the file object in the write mode write some another content close subsequently then again open the file object in 'wt' mode that is the update mode and write contents

Step 5: open file object in read mode, display the updated written contents and close, ~~subsequently then again open~~ again open the file object in 'r+' with parameter passed & 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 & display the append output.

file attributes

```
a= file obj.name
print ("Name of file (name attribute): ", a)
>>> (Name of file (name of attribute, abc.txt)
b= file.obj.closed
print ((close) attribute : ", b)
>>> (close) attribute : True)
c= file obj.mode
print ("file mode, "c)
>>> (file mode, 'r')
d = file obj.softspace
print ("softspace, "d,)
>>> ("softspace : "0)
# wt mode                # write mode
file obj = open ("abc.txt", "wt")      file obj=open(abc.txt","w")
file obj.write ("Saurav")              file obj.write("...
file object.close ()                   file obj.close()

# rt Mode
file obj=open ("abc.txt", "rt")        # read mode
str1 = file obj.read (r)               file obj=open("abc.txt","r")
print ("output of rt," str1)           str2=file obj.read(
file.object.close()
>>> ('output of rt', 'saurav')
                                       print ("output of read mode)", str
                                       >>> (output of read mode', 'saurav')
```

```
# append mode
file obj = open("abc.txt", "a")
file obj.write("data structure")
file.close()
bin object = open('abc.txt', 'r')
str 3 = file obj.read()
print("output of append mode", str 3)
file obj.close()
```

>>> ("output of append mode:", 'samar' 'data structure')

```
# tell()
file obj = open("abc.txt", 'r')
pos = file obj.tell()
print("tell():", pos)
file object.close()
```

>>> ("tell():", pos)

```
# seek()
file obj = open("abc.txt", 'r')
start = file obj.seek(0,0)
str 8 = file obj.read(10)
print("the beginning of the line is:", str 8)
```


Step 7 : open the file object in read mode. declare a variable and perform file object dot tell. method and store the output consequently in variable

Step 8 : use the seek method with the argument with the opening the file object in read mode and closing subsequently

Step 9 : open file object with read mode also use the readline method & store the output consequently in and print the same four counting the length use the for condition statement and display the length

Prac : 2

Aim : Demonstrate the use of iteration & iterators

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

Q1) write a program using iterable objects to displaying the odd nos in range 1 to 10.

Algorithm :

Step1 : Define a iter() with argument and initialize the value and return that value.

Step2 : 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
```

```
>>> it = count(1)
>>> z = iter(it)
>>> 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.n = 0
        return self
    def next(self):
        if self.n <= 10:
            num = self.n
            self.n += 1
            p = 2 ** num
            print("2 **", self.n-1, "=", p)
            return p
        else:
            raise StopIteration
```

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

Q) Write a program using an iterator for calculating the power of a given no. for instance number entered is 2 then value calculated should be $2^1, 2^2, 2^3, 2^4$.

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.2] Write a program using iterable concept to find factorial of no. in range 1 to 10.

Algorithm:

Step 1: Define a iter() with argument & initialize 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 & pass this object in the iter method.

[Q1] Write a program using iterable concept to display multiple of 2 in range 1 to 10.

Algo:

Step 1: Define a iter() with argument and initialize 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.n = 1
        return self
    def next(self):
        if self.n < 10:
            num = self.n
            self.n += 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 argument 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

Prac-03

Aim: Demonstrate the use of expecting on handling.

Theory: An exception is an event which occurs during exception of program which distrupt the normal flow of program. Thus, a exception represents object which represents an error. This object is derived from given class and when the python script raises an exception it must be handled immediately: otherwise it'll terminate and close the program.

Q.1] Write a program to chick the range of the age of the students in given class and if age does not fall in given range use value error exception otherwise return the valid no.

Algo:

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 falls in range and so return the age else use value error or 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.

Q2] Write a program to check whether the number is given class & if the number is a floating point use value error as exception for the given input.

Algorithm

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

use the except block with exception as ValueError and display appropriate message if its suspicious code is part of try block

Q.3] Write a program to demonstrate use of zero division error.

Algorithm :

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

Step 2 : Define a function with 2 parameters to divide the nos given by user

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

Step 4 : use except block with zerodivision error and print 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:",
        break
    except ZeroDivisionError:
        print("Error!")
```

>>> Enter first number : 1
>>> Enter second number : 1
Division of 1 and 1 is 1

>>> Enter ~~your~~ first number : 1
>>> Enter second number : 0
Error!

24/12/24

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']
```

Practical : 04

Aim : Demonstrate the use of regular expression

Theory : Regular expression represents the sequence of characters which is mainly used for finding and replacing the given pattern in string and for this we import re module and common usage of regular expression involved following functionalities :

- Searching a given string
- finding a string
- Breaking a string into smaller substrin
- Replacing part of string.

Q.1] Write a regular expression regcreating numeri and 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 whereas \D is used to match non decimal digits.

Q.2] 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 3

```
import re
li = ["9876543210", 876542102";
      "7921801221", "6543210998" ]
for element in li:
    result = re.match ["[8-9]{1}
                       [0-9]{9}", element)
if result:
    Print("Correct mobile no")
    Print ((result.group(1))
else:
    Print ("Incorrect mobile no")
```

output:

```
>>> correct mob no
    9876543210
    correct mob no
    8765432109
    Incorrect mobile no
    Incorrect mob no.
```


Q3] write a regular expression to check whether the given mobile no starts with 8 or 9 & the local length of digit should be almost 10.

Algorithm

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

Step 2: Now use for conditional statement to find if the no starts 8 or 9 and total no should length of 10. use match & () inside for statement to find the match in given string.

Step 3: use if conditional statement to know whether we have a match or not. if we have use group 1) to display the out-put and if we dont displays incorrect mobile no.

Q.4] Write a regular expression for extracting a word from given string along with space characters in between word and subsequently extract the word without space character.

Algorithm

Step 1 : Import re module and apply a string

Step 2 : Use find.all () .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.

Q.5] write a regular expression for extracting first and last word from a string

code 4

```
import re
string = " python is important"
result 1 - re. findall ("|w*", string)
result 2 = re. findall ("|w+", string)
print (result 1)
print (result 2)
```

output

```
>>> ['python', ' ', 'is', ' ', 'important', '']
    ['python', 'is', 'important']
```

Q.7) Write a re. for extracting the
(A) username from email id
(B) hostname from email id
(C) Both username and hostname from email id

Algorithm

Step 1 : Import re module and apply a string.
Step 2 : Use findall() to find username, hostname and both of email id.

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

Step 4 : Display the output.

Code 7

```
import re
string = "abc@tesc.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']
>>> ['@tesc.edu']
>>> ['abc', 'tesc.edu']
```

07/01/2020

Practical : 05

Topic : Gui components

Step 1 : use the tkinter library for importing the features of the text widgets.

Step 2 : Create an object using the TK()

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

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

~~Step 5~~ #2 :-

Step 1 : use the tkinter library for importing the features of the text widgets.

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

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

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

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

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

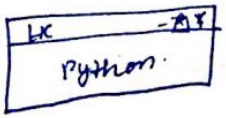
Step 5: Now repeat above steps with the label() which takes the following arguments.

1) Name of the parent window
2) Text attribute which defines the string
3) The background colour.
4) The font and then use the pack() with relevant padding attributes

creation of parent window :-

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

output:

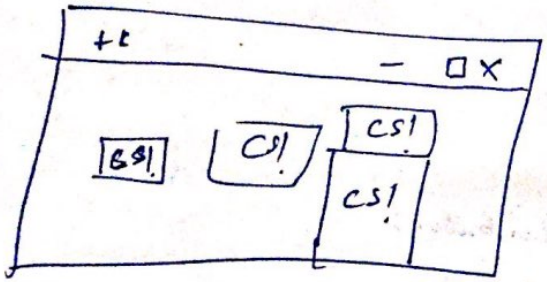


2: label, attributes :-

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

```
l4=Label(root, text="CSI", bg="orange",
         fg="black", font="10")
l4.pack(side=TOP, ipady=50)
root.mainloop()
```

output



Practical - 5(B)

AIM : GUI components

#1

Step 1 : Import the relevent methods from the tkinter library create an object within 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 above given selection made from multiple option available.

Step 4 : Now define the p.w and define the option with control variable

Step 5 : Use the listbox() and insert options on the p.w along with the pack() with specific anchor attribute.

Step 6: create an object from radiobutton which will take following arguments & p.w object, text variables which will take the values option no 1, 2, 3 variable argument, corresponding value &

Step 7: trigger the function declared.

Step 8: Now call the pack() for radio object so created and specify the argument using anchor attribute.

Step 9: Finally make use of the mainloop() along with p.w.

#radiobutton

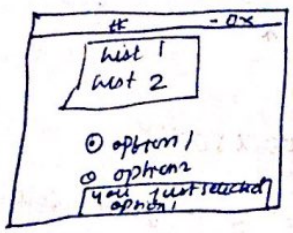

```
from tkinter import *
root = Tk()
root.geometry("500x500")
def select():
    selection = "you just selected" + str
                (var.get())

    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", variable
     = var, value = "option1", command
     = select)
r1.pack(anchor = N)
r2.RadioButton(root, text = "option 2", var
     = var, value = "option2",
     command = select)
r2.pack = pack(anchor = N)
root.mainloop()
```

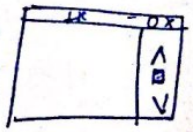
output



2. scrollbar()

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

Output



#3

```
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=("Times New Roman", 0))
```


code 2

step 1 : Import relevant methods from the tkinter library.

step 2 : Create a p.object corresponding to the p.w

step 3 : use the geometry() for saying of the window

step 4 : create an object and use the scrollbar().

step 5 : use the pack() along with the scroll bar object with side and fill attributes.

step 6 : use the mainloop with parent object.

3 using frame widget

step 1: Import the relevant libraries from the tkinter method.

step 2 : create an corresponding object of p.w

step 3 : use the geometry manager with pixel size (680x500) or any other suitable pixel value.

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

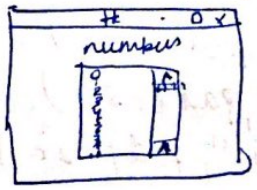
step 5 : use the frame widget along with the parent object created and use the pack method.

step 6 : use the listbox along with the attributes like width, height font. Do create a list-box 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 the scroll bar() and use pack()


```
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(1,10):
    listNodes.insert(END, str(x))
window.mainloop()
```

Output




```
## from tkinter import *
window = Tk()
window.geometry("600x500")
frame = Frame(window)
frame.pack()
leftframe = frame(window)
leftframe.pack(side = "left")
rightframe = frame(window)
rightframe=pack(side= "right")
```

b1 = Button (frame, text = "select", active background = "red", fg = "blue")

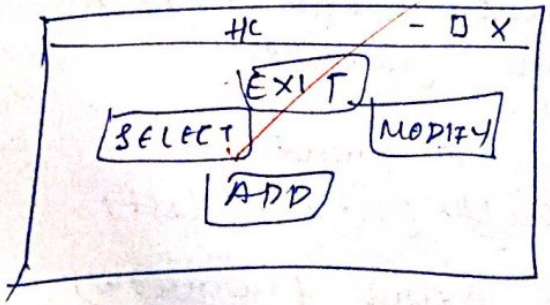
b2 = Button (frame, text = "modify" activeb... = "red", fg = "black")

b3 = Button (frame, text = "ADD", activebackg = "blue", fg = "red")

b4 = Button (frame, text = "Exit", activebg = "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")

output



step 8: Trigger the events using mainloop #4.

step 1: define import relevant methods from tkinter library.

step 2: Define the object corresponding the p.w and define the size of p.w in terms of no. of pixels.

step 3: Now define the frame object-named as the left frame and put it on the p.w on the LEFT side

step 5: similarly define the RIGHT frame and subsequently define the button object-placed onto the given frame with attributes as text, active background and foreground.

step 7: Now use the pack () along with side attribute

step 7: Similarly create the button object corresponding to the MODIFY operation put it into frame object on side = "right".

step 8: create another button object and place is on to RIGHT frame ~~frame~~ and label the button as ADD.

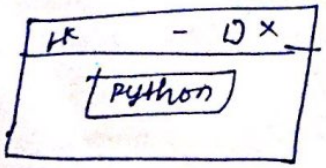
step 9: Add another button and put is on the top of frame and label it as EXIT.

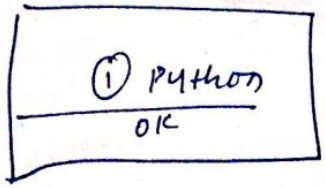
step 10: Use the pack() simultaneously for all the objects & finally use the mainloop()

08 # 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()
```

Output





Practical - 4

Aim : GUI Components

Step 1 : Import the relevant methods from tkinter library.

Step 2 : Import tkmessage box

Step 3 : Define a parent window object along with p.w.

Step 4 : define a function which will use tkmessage box with show info attribute.

Step 5 : Declare a button with p.w 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.

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

Step 2: use pw object along with minsize function for window size.

Step 3: Define a function main, declare parent window object and use config(), title(), minsize(), label() as use as button() and use pack() & mainloop() simultaneously.

Step 4: Similarly define the function second and use the attributes accordingly

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

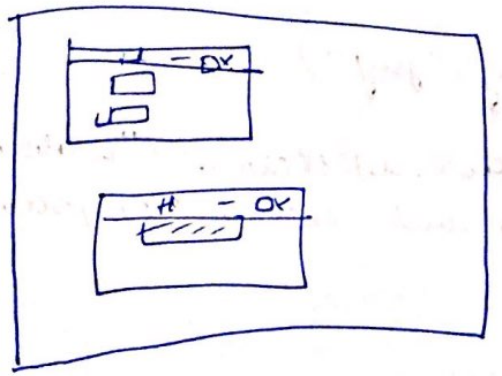
Step 6: finally called the main for event driven program

multiple window

```
from Tkinter import *
root = Tk()
root.minsize (300,300)
def main():
    top = Tk()
    top.config (bg = "black")
    top.title ("HOME")
    top.minsize (300,300)
    L=Label (top, text = "SAN FRANCISCO
        \n Places of Interest: \n golden gate
        Bridge \n Lombard street \n
        Chinatown)
    L.pack()
    b1 = Button (top, text = "next",
        command = second)
    b1.pack (side = RIGHT)
b2 = Button (top, text = "exit", command
        = terminate)
b2.pack (side = LEFT)
top.mainloop ()
```

```
def second():
    top2 = Tk()
    top2.title("About us")
    top2.geometry("300x300")
    l = Label(top2, text = "created by :-shaln for more details contact official mail")
    l.pack()
    b3 = Button(top2, text = "prev", command = main)
    b3.pack(side = LEFT)
    b2 = Button(top2, text = "exit", command = terminator)
    b2.pack(side = RIGHT)
    top2.mainloop()
def button():
    top3 = Tk()
    top3.geometry("300x300")
    b1 = Button(top3, text = "flat button", relief = FLAT)
    b1.pack()
    b2 = Button(top3, text = "ground button", relief = GROOVE)
    b3.pack()
```

output



```
from Tkinter import *
root = Tk()
root.config(bg = "grey")
def finish():
        messagebox.askokcancel("warning",
            "this will end the program")
        quit()
def info():
    list1 = listbox()
    list1.insert(1, "Co.Name : Apple")
    list2.insert(2, "Products : iphone")
    list1.insert(3, "Language : swift")
    list1.grid(ipadx = 30)
def aboutus():
    list2 = Label(text = "About us")
    list2.grid(ipadx = 30)
    list3 = Label(text = "Steve jobs thaven
                  March 2020")
    list3.grid(ipadx = 24)
p1 = PhotoImage(file = "download.gif")
f1 = Frame(root, height = 35, width = 5)
f1.grid(row = 1, column = 0)
f2 = frame(root, height = 250, width = 500)
f2.grid(row = 1, column = )
```


Practical : 08

step 1 : Import relevant methods from the tkinter library.

step 2 : create parent window object and use the config method along with bg color attribute specified.

step 3 : define a function finish with the messagebox widget which will display a message in a warning message and subsequently terminate the program

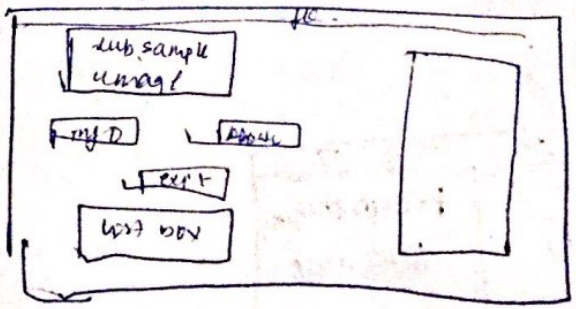
step 4 : define a function info use a list box widget along with object of same. and finally use the grid (i) ipadx attribute

step 5 : Define a function aboutus with label widget. and text attribute and subsequently with grid()

```
l1 - grid (row=1, column=0, padx=20,
           pady=15)
l2 = Label (f2, image=p1, relief=SUNKEN)
l2 - label (f2, image=p7, relief=SUNKEN,
            command=info)
b1 · grid (row1, column=0)

b2 = button (f1, text="About us", relief=SUNKEN,
             command: Aboutus)
b2 grid (row=1, column=2, padx=+)
b3 = Button (f1, text="EXIT", relief
             =RAISED, command=top)
b3·grid (row=2, column=1, padx=15)
root.mainloop()
```

Output



Step 6: Use PhotoImage widget with file and filename with gif attribute.

Step 7: create a frame object along with frame() along with p.w and subsequently use the grid() with row & column.

Step 8: Similarly create another frame object as declared by step 7

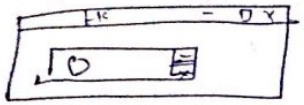
Step 9: Creat another object and use the subsample(5,4)

Step 10: use label widget along with the frame object relief attribute and subsi-quently use the grid()

Step 11: Now creat button object dealing with different attributes of frame.

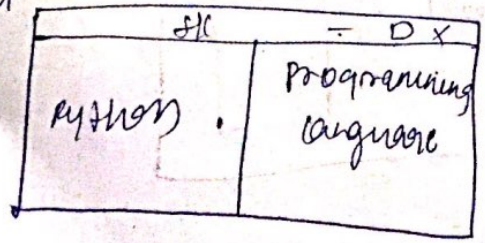
```
from Tkinter import *
master = Tk()
s = spinbox (master, from_ = 0, to = 10)
s.pack()
master.mainloop()
```

Output



```
from Tkinter import *
root = Tk()
p = Panedwindow()
p.pack(fill = BOTH, expand = 10)
l = Label(p, text = "PYTHON")
p.add(l)
p1 = Panedwindow(p, orient = VERTICAL
                 bg = "black")
p.add(p1)
l1 = Label(p1, text = "Programming language")
p1.add()
root.mainloop()
```

output



Practical 10.

Aim : GUI components

#Spinbox

Step 1 : Import relevant (s) from the tkinter library.

Step 2 : Creat p.w object along with Tk()

Step 3 : Creat an object from spinbox method and use attributes p.w object, from_ and to attributes)

Step 4 : Subsequently call the pack() method along with spin box object and called the mainloop()

#2 PANED WINDOW

Step 1 : Import relevant methods from the tkinter library also creat a parent window object

Step 2 : creat an object along with panel window and subsequently use the

the pack() along with paried window style along with attributes like fill and expand :

step 3 : creat an label object along with label() use paned window object, orient & background color.

step 4 : similarly create another label method and use the add()

step 5 : finally called the mainloop () for event migration

CANVAS :

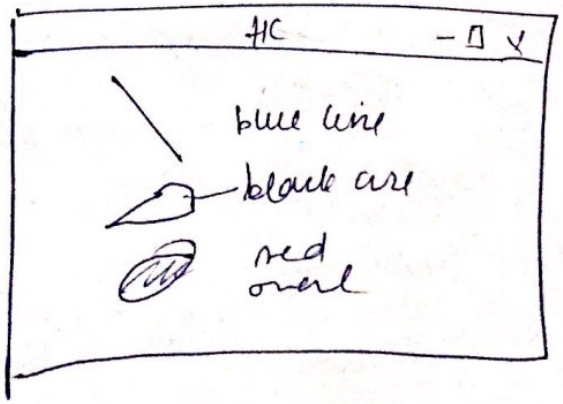
step 1 : import relevant ()'s from tkinter library declare a parent window obj.

step 2 : creat a canvas object along with canvas() with attributes parent window objects height width, bg

#3 from Tkinter import *

```
root = Tk()
c = canvas(root, height = 100, width = 200,
           bg = "red")
carl = c.create_arc(10,20,30,40, start=10,
                    extend = 50, fill = "Blue")
coval = C.create_oval(20,31,45,50, fill "red")
Line = C.create_line(10,15,20,30, fill = "yellow",
c.pack()
root.mainloop()
```

output



Ex 4: Similarly for anal and use use the pack() and call() manloop() for event driven programming

Practical :06

Aim : Demonstrate the use of GUI creating a human face and converting celsius unto fahrenheit.

Q.1] 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 and width.

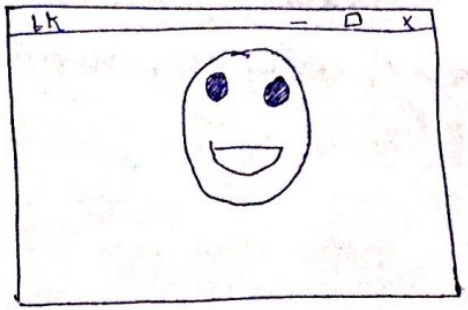
Step 4 : Now use pack () for positioning of widget onto parent window.

Step 5 : Now create an object face and use object create_oval () with coordinates 50, 50, 300 300 & outline : black, fill : yello

#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, 75, 175, fill = "black")
eye2 = c.create_oval (225, 125, 275, 175, fill = "black")
mouth = c.create_arc (125, 225, 295, 275, start = 0, extent
                      = -180, width = 5, fill = "red")
root.mainloop()
```

output



GPCU

```
from tkinter import *
window = Tk()
fahrenheit = DoubleVar()
fahrenheit.set(32.0)
def convert(celsius):
    fahrenheit.set(1.8*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 = StringVar()
l2 = Label(window, textvariable = fahrenheit
l2.grid(row = 2, column = 0, columnspan = 2)
B = Button(window, text = "calculate",
           command = lambda: convert
           (Celsius.get()))
B.grid(row = 1, column = 0, columnspan = 2)
mainloop()
window
```


8) write a program to convert celsius into farenheit using GUI

Algorithm

Step 1 : Import all the relevant methods in tkinter library.

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

Step 3 : Now initialize farenheit as DoubleVar() and set it to 32.0.

Step 4 : Now define a funct^n convert with arguement celsius & to convert celsius into farenheit using set()

Step 5 : Now create an object l2 using label() and place it onto parent window and 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()

Step 8: Create another object and use entry widget to enter the input and, place it onto the parent window

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

Step 10: Finally use mainloop

#1

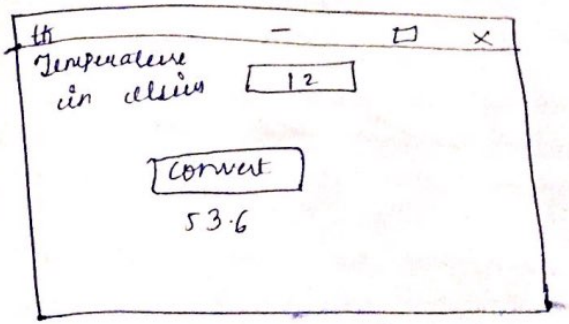
Steps continue: as attribute to create face.

Step 6: Now create any object & again use oval, create_oval() with appropriate co-ordinates along with fill as attribute along create left eye.

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

Step 8: Create an object mouth and use object create_arc() with start = 0, extend = -180 and bd = "raised", width = 5 as attribute

output



Algorithm

Step 1: Import re module and 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.

3. Now display the result.

[OR] Write a regular expression for extracting the date in format dd-mm-yyyy by using the findall() format Input: 2024-12-24

Step 1: Import re module and apply string

Step 2: We findall method and use \|d{2}-\|d{2}-\|d{4} as an parameter

Step 3: Now display the output

to create mouth.

(Step 7: finally use the main loop!)

Practical : 4

AIM : Write a program to find factorial of a number and use arithmetic operations on 2 nos using GUI.

Q.1] Write a program to find factorial of a no using GUI.

Algorithm :-

Step 1 : Import relevant methods from tkinter using GUI.

Step 2 : Define a Func^n factorial using recursive function.

Step 3 : Define another Func^n calculate to call factorial Func^n

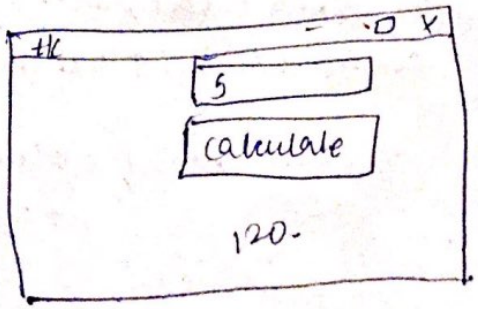
Step 4 : Now create an object with entry() and use pack() for positioning on parent-window.

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

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

output



```
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 nos")
l2 = grid(row = 1, column = 0)
e2 = Entry(root)
e2.grid(row = 1, column = 1)
v = IntVar()
r1 = RadioButton(root, text = "add", variable = v
                 value = 1)
r1.grid(row = 2, column = 0)
~~r2 = Radiobutton~~ r2 = RadioButton(root, text = "sub"
                 variable = v, value = 2)

r3 = RadioButton(root, text = "mul", 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)
```


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

step 7: Finally use the mainloop()

(12) WAP to perform arithmetic operation on 2 nos 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 carry out arithmetic operations on 2 nos.

step 4: Now create object with label() as num1 & num2 and use grid() to place it onto parent window.

step 5: create objects with entry() to take input from user()

Step 6 : Now initialize v as integer using IntVar().

Step 7 : Now create 4 object with Radiobutton() to choose any one of a.o 2 use grid() for positioning on to parent window

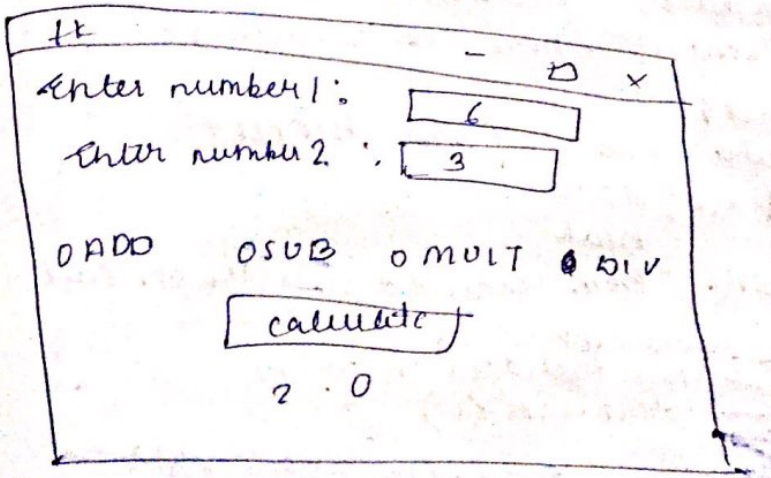
Step 8 : Now create a object with button() along with command attribute to carry out the a.o of users choice.

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

Step 10 : Finally use the mainloop()


```
l3 = Label (root)
l3.grid (row = 4, column = 1)
root.mainloop

# output
```



#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))

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()
```

(NOTE) run the program & now right-write client program.

~~#code~~

Practical : 08

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

WAP a program to demonstrate use of socket module and server client program

Algorithm :

Step 1 : Import the socket module to import relevant methods.

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

Step 3 : Now get value for port variable to initialize port no above 10.24

Step 4 : use -socket() to get instance.

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

Step 6: Now use accept() to accept new condition.

Step 7: Now print address

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

Step 9: Now close the program.

(FOR SOCKET CLIENT PROGRAM):

Algorithm

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

Step 2: Define a func client_program get the host name and 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 ("->")

#socket


```
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 socket_server.py
connection from: ('127.0.0.1', 47922)
from connected user: Hi
-> Hello.
from connected user: How are you?
-> good.
from connected user: Awesome!
-> ok then, bye.
```

Name :

Roll No.:

Subject :

Batch No

\# output for client-program

$python 3.6 socket-client.py :

→ Hi

Recieved from server : Hello

→ How are you

Recieved from server : Good

→ Awesome!

Recieved from server : ok then bye!

→ Bye.

output for client_program

$python 3.6 socket_client.py

→ Hi

Recieved from server : Hello

→ How are you

Recieved from server : Good

→ Awesome!

Recieved from server : ok then bye!

→ bye.

step 6: use while conditional loop to send a message.

step 7: Now use decode to recieve response

step 8: now show the data

step 9: again take input

step 10: close the program by using close()

Practical : 09

AIM : Demonstrate the use of database connectivity.

Algorithm

Step 1: Import sqlite3 module to import relevant methods.

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

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

Step 4: Now use cur.execute() to make a table, insert values into table and use DML, DDL statements to manipulate the data in this database

Steps: use fetchall() to show the output.

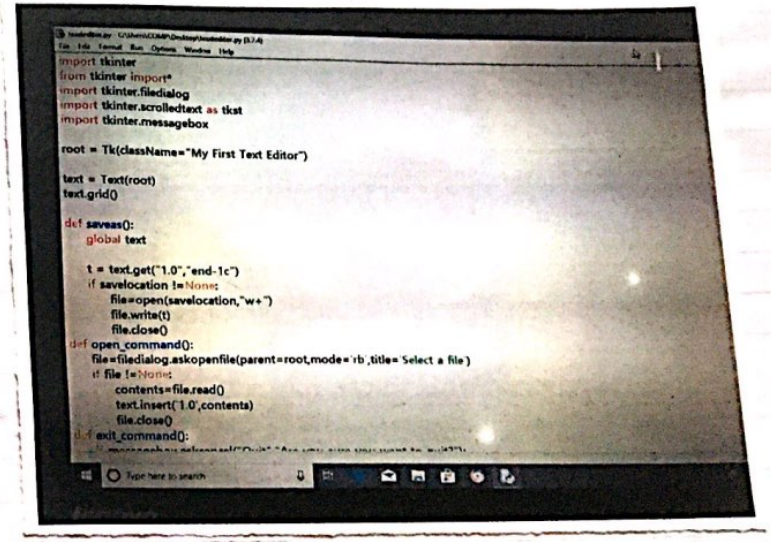
Step 6: use commit to save all changes

Step 7: use close() to terminate to terminate the process.

CODE IN SHELL ENVIRONMENT:-

```
>>> import sqlite3
>>> conn = sqlite3.connect("student1.db")
>>> cur = conn.cursor()
>>> cur.execute('create table student
        (roll_no int(3) primary key, name
        varchar(50) not null, address
        Marchar(50) not null, class varher
        (50), dob date)')
<sqlite3.cursor object at 0x0322EBED>
>>> cur.execute('insert into student values
        (101, "Ankita", "borivali", "FY CS", "12/07/2001")
<sqlite3.cursor. object at 0x3226EBED>
>>> cur.execute('insert into student values
        (102, "Esha", "kandivali", 74 CS", "13-08-2001")
<sqlite3.cursor. object at 0x0322EBED>
>>> cur.execute('insert into student values
        (103, "Esha", "kandivali", 745.11, "14-08-2001")
<sqlite3.cursor. object at 0x0322EBED>
>>> cur.execute('select * from student")
<sqlite3.cursor object at 0x0322EBED>
>>> cur.fetchall()
[(101, 'Ankita', 'borivali', 74 CS', 12/07/2001)
 (102, 'Esha', 'kandivali', '74 CS', 13/08/2
>>> cur.execute("update student set
        dob = "13/08/2001" where roll_no = 1
<sqlite3.cursor object at 0x0322EBED>
```

```
>>> cur.execute("select * from
        student where address='karad
        naka'")
<sqlite3.Cursor object at 0x0322E1B0>
>>> cur.fetchall()
[(102,'Esha','Kandalgaon','74CS'
        13/08/2000')]
>>> cur.execute('commit')
<sqlite.cursor object at 0x0322E1B0>
>>> cur.close()
```

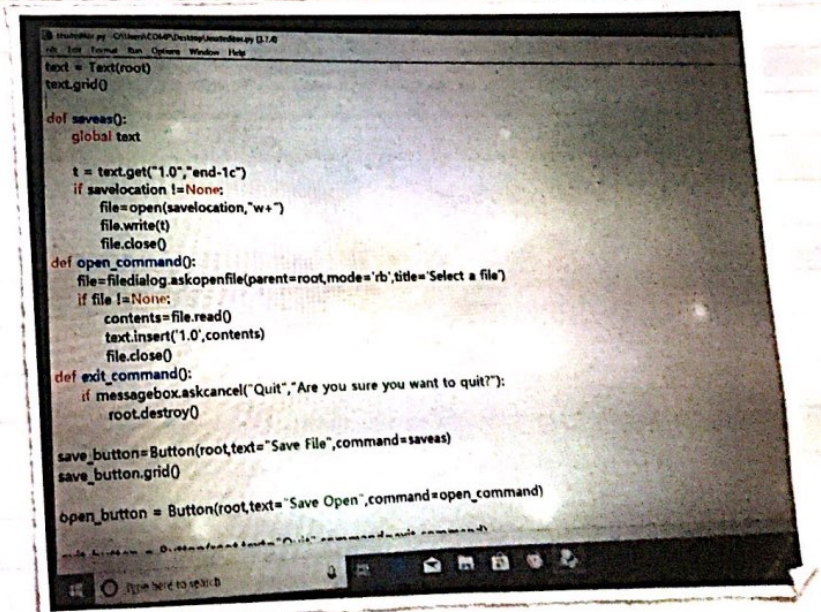
```
import tkinter
from tkinter import*
import tkinter.filedialog
import tkinter.scrolledtext as tkst
import tkinter.messagebox

root = Tk(className="My First Text Editor")

text = Text(root)
text.grid()

def saveas():
    global text

    t = text.get("1.0","end-1c")
    if savelocation !=None:
        file=open(savelocation,"w+")
        file.write(t)
        file.close()
def open_command():
    file=filedialog.askopenfile(parent=root,mode='rb',title='Select a file')
    if file !=None:
        contents=file.read()
        text.insert('1.0',contents)
        file.close()
def exit_command():
```



```
text = Text(root)
text.grid()

def saveas():
    global text

    t = text.get("1.0","end-1c")
    if savelocation !=None:
        file=open(savelocation,"w+")
        file.write(t)
        file.close()
def open_command():
    file=filedialog.askopenfile(parent=root,mode='rb',title='Select a file')
    if file !=None:
        contents=file.read()
        text.insert('1.0',contents)
        file.close()
def exit_command():
    if messagebox.askcancel("Quit","Are you sure you want to quit?"):
        root.destroy()

save_button=Button(root,text="Save File",command=saveas)
save_button.grid()

open_button = Button(root,text="Save Open",command=open_command)
```

```
        if file != None:
            contents = file.read()
            text.insert('1.0',contents)
            file.close()
    def exit_command():
        if messagebox.askcancel("Quit","Are you sure you want to quit?"):
            root.destroy()
    save_button=Button(root,text = "Save File",command=saveas)
    save_button.grid()

    open_button = Button(root,text = "Save Open",command=open_command)
    exit_button = Button(root,text = "Quit",command=exit_command)
    exit_button.grid()

    menu = Menu(root)
    root.config(menu=menu)
    filemenu=Menu(menu)
    menu.add_cascade(label="File",menu=filemenu)
    filemenu.add_command(label="Quit",command=exit_command)
    root.mainloop()
```