Practical - I

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

step 1: create a file object using open method and use the write access mode followed by writing for some contents on the file and than closing the file.

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

step 3: now use the fileobject for finding the name of the file the file mode in which its opened whether the file is still open or close and finally display the output of the soft attribute.

step 4 : Now open the file write mode write obj content close subsequently again open the file mode that is the file obj and write content updated

step 5 : open file obj in display the updated content and close open in 'r+' mode with based and passed and for the output subsequently

step 6 : Now open file obj in mode open write method contents close the file obj again open the file obj read mode and display the appending output.

26

```
file obj = open ("abc.txt "w") # file open write mode
file obj write (" computer science subjects "\n")
file obj write (" DBMS in PYTHON in DS in ")
file obj. close () # file close
file obj = open (" abc.txt", "r") # read mode
file obj = read ()
str1 = fileobj. read ()
print (" The output of read method ", str1)
file obj. close ()
>>> (' The outputs of read method : ', computer science subjects in DBMS in python in
# readline ()
file obj = open
file obj = file obj . readline ()
str2 = file obj
print (" The output of readline meth
file obj. close ()
# readlines ()
fileobj = open ("abc.txt", "r")
str3 = fileobj. readline ()
print (" The output of readline
fileobj. close ()
# file attributes
a = fileobj. name
print (" name of file (nar
b = fileobj. closed
print (("(close) attribute :"
(close) attribute : ,
```

```
c = file obj. mode
print (" file mode", c)
>>> ( file mode : 'r')
d = fileobj softspace
print (" softspace", d)
>>> (' softspace : ', 0)
# w + mode
file obj = open ("abc.txt", "w+")
fileobj. write (" Arvind sir")
file obj . close ()
# r + mode
file obj= open ("abc.txt", "r+")
s1 = file obj. read (6)
print (" output of r+ ", s1)
file obj. close ()
>>> (' output of r+ ' , 'Arvind')
# append mode
fileobj = open ("abc.txt", "a")
file obj . write (" data structure
fileobj. close ()
fileobj
s3 = open ("abc.txt", "r")
s3 print (" output of append mode :"
file obj. close ()
>> (' output of append mode :
sir data structure')
```

Write
file obj.
file obj
file obj

step 7 : open the file obj in read mode, declare a variable and file object and tell method, store the output consequently variable

step 8 : use the seek method with argument with opening file obj in read mode and subsequently

step 9 : open file obj with r also use the readlines method the output consequently in a name you counting the length for conditional statement the length.

```
#
fileobj = open ("abc.txt", "r")
pos. fileobj.tree()
Round
fileobj.close
>>> ('tree ():', OL)
# seek ()
fileobj = open ("abc.txt", ("r"))
st = fileobj.seek(0,0)
print ("seek (0,0) is :", st)
fileobj.close
>>> ('seek(0,0)is:", st)
fileobj.close
>>> (seek(0,0)is:', name)
ileobj = open ("abc.txt", ("r"))
1 = fileobj.seek(0,1)
int ("seek(0,1)is:", st1)
leobj = open("abc.txt", ("r"))
= fileobj.seek(0,2)
nt ("seek(0,2),is:", st2)
obj.close
'seek(0,2) is:', none)
bj = open("abc.txt", "r")
fileobj.readline()
Output: " st)
in st:
(len(line))
```

Practical - 2 :

Objective : Iterators

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

Step 2: The similar output can be obtained by using for conditional statement. An iterative variable to be declared in for loop which will iterate.

step 7 : Define a function using parameter then using condition statement do not check whether the number is even and odd return completely.

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

Step 9 : Now use the next () to define the logic for displaying value

```
for x in range ...
    value = list ( map (lambda x : x(...
    print (value )
>>> [0, 0]
    [1, 1]
    [4, 8]
    [9, 27]
    [16, 64]

# map ()
listnum = (0, 4, 5, 7, 9, 11, 13, 15
listnum = list (map( lam...
listnum ))
print (listnum )
def even (x ) :
    if (x % 2 == 0):
        return " EVEN
else :
    return " EVEN"
else :
    return " ODD"
list (map (even, listnu...

# odd number
class odd :
    def __iter__ (self
        self .num = 1
        return self
    def __next__ (sel...
```

check to
any the
and it's
nd HCF
ng f
mat
en
h.

```
myobj = odd ()
= iter (myobj)
x = int (input ("enter a number"))
for i in myiter:
    if (i < x):
        print (i)
>>> enter a number : 15
1
3
5
7
9
11
13
```

factorial using map function:

```
def f(x):
    if (x == 1):
        return 1
    else:
        return (x f(x-1))
x = int (input ("enter a number : "))
list []
list.append(x)
n = map (f, list)
print ("The factorial is :", n)
> enter a number : 4
The factorial is : ', [24]).
```

31

step 10 : Define an object of a class

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

step 12 : Define a function with conditional if exactly equal to return else one parameter

step 13 : Enter a input from the an empty list we append meth appending input value call the map using two fo print the factorial value.

late
7/11

15

Practical - 3 :

Topic : exceptions

step 1 : In the try block open an existing file in the open mode with write some contents in file.

step 2 : In except (IO ERROR) block use the error in IO error and display the appropriate message to display

step 3 : else display the operation is successfull.

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

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

step 6 : Else display the operation is successfull.

IOError :

try :

32

IOERROR

try :

fo = open ("abc.txt", "w")

fo.write(" Python is an interdent language ".)

except IOError :

print (" enter appropriate mode for file operation !")

else :

print (" operation is successfull !")

>>> operation is successfull !

R OUTPUT :

>>> enter appropriate mode for file operation !

value error :

try :

x = int (input (" enter a state

except value error :

print (" ARITHMETIC ERR

else :

print (" operation is success

>>> Enter a statement : abc

ARITHMETIC ERROR !

>>> Enter a statement : 123

```
# type error, zero division error
a = 1
b = input ("enter : ")
try :
print (a/b)
except type error :
print ("incompatible value")
except zero division error :
print ("denominator is zero so
operation cannot be performed")
```

```
>>> Enter : t
incompatible value
>>> Enter : 2
0.5
>>> Enter : 0
denominator is zero so operation
cannot be performed
```

```
# multiline exception
a, b = 1, 0
try :
print (1/0)
print ("10"+10)
except (typeerror, zero division error)
print ("invalid input !").
```

```
1.0
010
nvalid input.
```

step 14: define a function [illegible] empty list and calculate the length of the list.

step 15: define a function [illegible] initialise or declare [illegible] elements in list and [illegible] calculate the length of the [illegible] and display the same.

step 16: in try block accept input from the user and if the user enters character values raise error that is saying to enter integer values

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

output: successful
0
None
4
None

raise keyword:

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

18

match ()
Import re
pattern = r"FYCS"
sequence = "FYCS report computer science"
if re.match(pattern, sequence)
Print ("Attached, Scattered")

Else
Print ("Not found")
>>> match patterned found

numerical value (regulation
Import re
pattern = r'\d+'
sequence = 'Hello 123, howdy by 2[illegible]
output = re.findall (pattern, sequence)
Print (output)
>>> (123', '(789', 45')

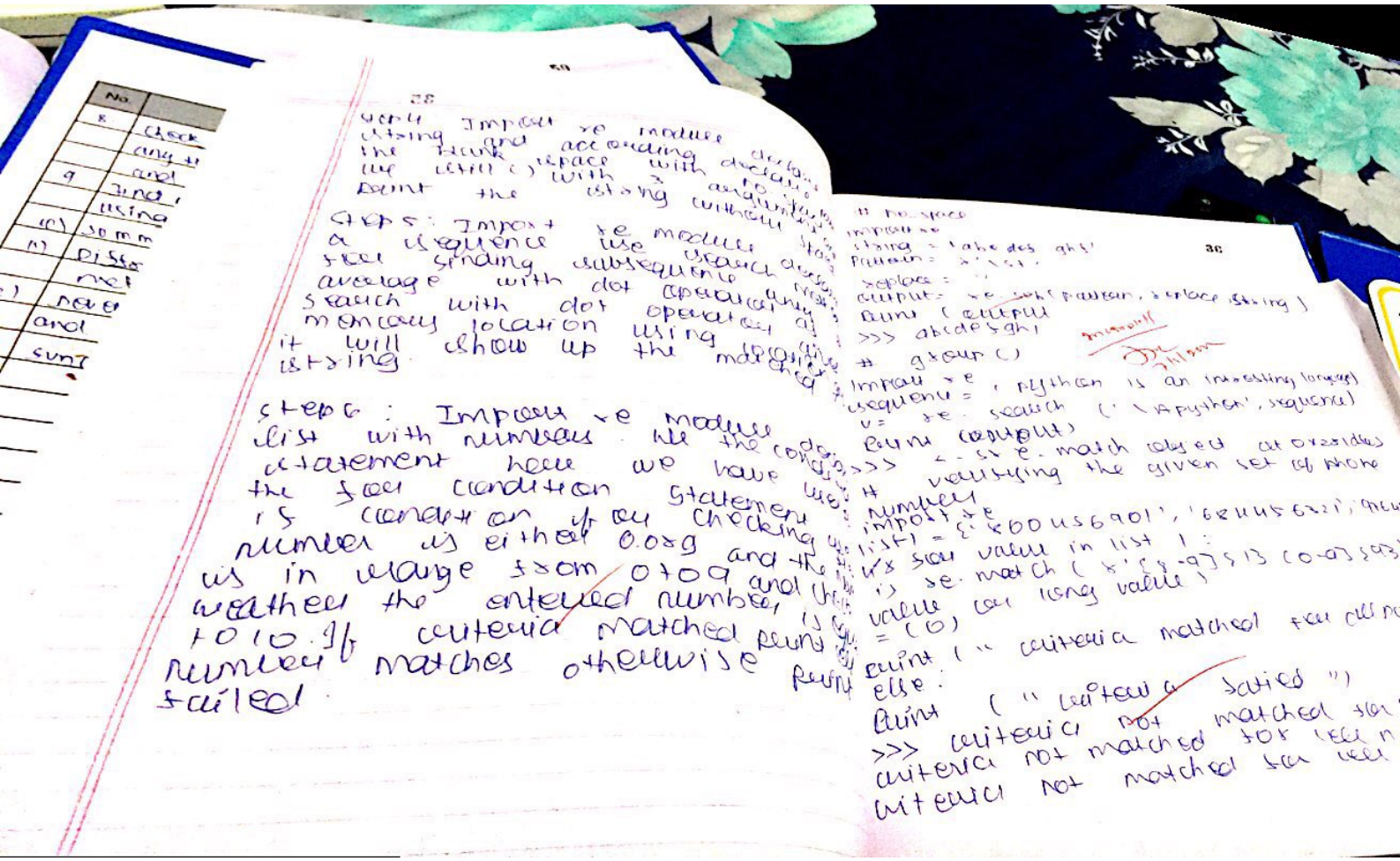
split ()
Import re
pattern = r'/d+'
string = 'Hello, [illegible] howdy 784 - 45 Hours'
output = re.split (pattern, string)
Print (output)
('Hello', 'Howdy', 'Hours')

Practical 4
Topic: Regular expression

Step 1: Import re module declare pattern and declare sequence use match method with declare argument matched than print the same otherwise print pattern not found.

Step 2: Import re module declare pattern with literal and meta character declare string value use the with argument and print

Step 3: The Import re module declare pattern with meta character use the split and print the output.



No.	
8	check any + and
9	find using
10	som... Disse... me' ...

36

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

# host and domain
import re
user = 'abc+csc@edu.com, xyz@...'
pattern = r'[\w\.-]+@[\w+\.-]+'
output = re.findall(pattern, user)
print(output)
>>> ['abc+csc.edu.com', 'xyz'
.com)

# counting of 2 letters
import re
s = 'mr.a, ms.b, ms.c, mr.d'
p = r'[ms|mr]+'
o = re.findall(p, s)
print(o)
m = 0
f = 0
for y in o:
  if (y == 'ms')
    f = f+1
  else:
    m = m+1
print("no. of males is:", m)
print("no. of females is:", f)
```

('no. of males
('no. of females

37

step 7. import re module declare string use the findall() for finding the vowels in the string and declare the same.

step 8. import re module declare host and domain name separating the host and domain name. Declare pattern use the findall() and print the respectively.

step 9. Import re module enter a string use pattern to display only two elements of the particular string use findall() declare two variable with initial value as zero use for if condition and subsequent use the if condition check weather condition meat value and display the value subsequently.

14/01

Step 1. use the tkinter library by importing the features of the text widget

Step 2: create an object using the

Step 3: create a variable using the widget label and use the text

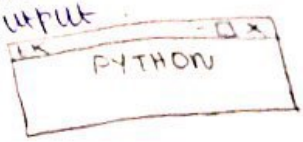
Step 4: use the mainloop() to see the corresponding above events.

#2:
Step 1: use the tkinter library by importing the features of the message. widget

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

```
#. creation of Tkinter window
from Tkinter import *
root = Tk()
l = label (root, text = "Python")
l.pack()
root mainloop()
```

output

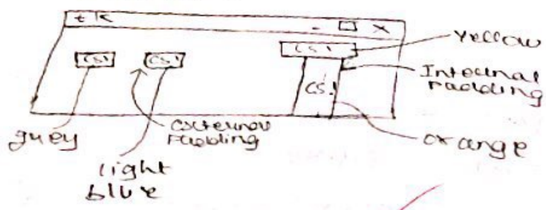


#2:

```
from Tkinter import *
root Tk()
l= label (root, text = "Python")
l.pack()
l1 = label (root, text = "(1", bg = "grey",
            fg = "black", font = "10")
l1.pack (side = LEFT, padx = 20)
l2 = label (root, text = "(1", bg = "
            fg = "black", font = "20"
l2.pack (side = LEFT, pady = "30"
l3 = label (root, text = "(1"), bg =
            fg = "Black", font = "
l3.pack (side = TOP, ipadx = 40
```

```
l4 = Label (root, text= "CS 1", fg = "Black", font= "10", bg= ...
l4. pack (side = TOP, pady = 50)
root. mainloop ()
```

output



step 3: use the pack () along with the object created from the text () and use the parameter,
1) side = LEFT
2) side = LEFT , padx = 20
3) side = TOP , pady = 30
4) side = TOP , ipadx = 40
ipad y = 50

step 4: use the main loop () 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 (bg)
4) The foreground fg and then use the pack () with a relevant padding attributes.

Practical. 5 (2)

Aim : Radio button and scrollbar using tkinter

1) Radiobutton

Algorithm :

Step 1 : use the tkinter module to import the relevant method

Step 2 : Define a function which tells the user about the selection made from given option available menu

Step 3 : use the config () along with label object and call the variable as an argument within the method

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

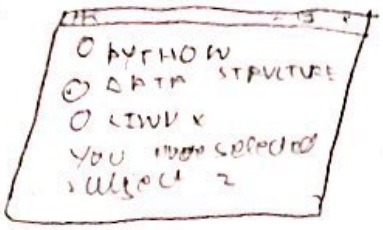
Step 5 : create object from the radiobutton () which will take the following argument.

source code :

```
from tkinter import *
def sel ():
    sel = "\n You have selected subject" + str (var.get ())
    label.config (text = sel)
root = Tk ()
var = IntVar ()
R = radiobutton (root, text = "python", font = '25', variable = var, value = 1, command = sel)
R. pack (anchor = w)
R1 = Radiobutton (root, text = "Data structure", font = '25', variable = var, value 2, command = sel)
R1. pack (anchor = w)
R2 = Radiobutton (root, text = "Linux", variable = var, font = '25', value = 3, command = sel)
R2. pack (anchor = w)
label = Label (root)
Label .pack ()
root. main loop ==p()
```

Scanned with CamScanner

OUTPUT FOR



ii) Defining the text variable which will take the value option 1,2,3,4

iii) Define the variable argument the corresponding value and trigger the function

step 6: For corresponding Radiobutton object, call the pack method and specify the argument as anchor attribute.

2) scrollbar

step 1: Import relevant methods from tkinter library

step 2: create an object corresponding to the parent window and create an object for scroll bar

step 3: create an object from the text window with height and width specified.

step 4: use the pack method along with the object of the scrollbar method and use the argument as side and fill

step 5: now use the text ... along with pack and ... the side

step 6: now use the config ... method along the object ... the scroll bare and use ... attribute

step 7: similarly use the config ... method along with text object ... and use yscroll command

```
source code
from tkinter import*
root = Tk()
scrollbar = scrollbar(root)
s.scrollbar.pack(side = right, fill = y)
mylist = Listbox(root, yscroll command
                 = scrollbar)
for line in range (100):
    my list.insert (END, "this is line
my list.pack( command = scrollbar)
                side = LEFT, fill = BOTH)
scrollbar.config( command = mylist.yview
root.mainloop( )
```

Scanned with CamScanner

Practical 5 (3)

Aim: Messagebox, Relief, Traversing Window

1) message box method

Algorithm:

step 1: Input the relevant method from Tkinter library

step 2: Define a function and use the messagebox along with the different methods avaible which contains one or more argumen

step 3: Thus different option avaible are showinfo(), showwarning(), showerror(), askyesno(), askquestion(), askcancel()

u: create an object from the button ethod and place on the parent window h the one of button specified and esponding event called for triggering

50

* Relief style

Step 1 : Use the button object with following attributes :

1) Parent window
2) Text attribute
3) Relief

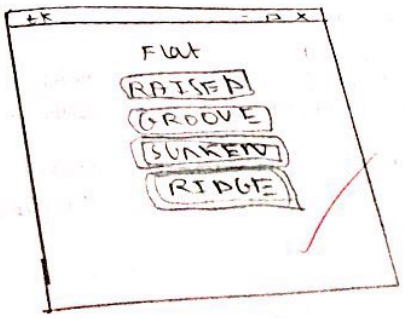
step 2 : use the corresponding pack method is the respective button objects and the corresponding event

step 3 : finally use the mainloop method

source code :

```
from tkinter import *
root = Tk ()
b1 = Button ( root, Text = "FLAT", relief=FLAT
b1 . pack ()
b2 = Button ( root, text = "RAISED", relief=
RAISED)
b2 pack ()
b3 = Button ( root, text = "GROOVE", relief=
GROOVE)
b3 . pack ()
b4 = Button ( root, text = "SUNKEN", relief=
b4 . pack ()
b5 = Button ( root, text = "RIDGE", relief
b5 . pack
mainloop ()
```

output :



source code :

```
from tkinter import
# call motors
def call():
    root = TK()
    root.config()
    root.title("Welcome To car ...
    root.minsize(height=700, width...
w1 = TK()
l = label(w1, text="In car models")
l.pack()
b = BUTTON(w1, text="...", command=...
    width= height=2. command=...
pack()
bike motors.
bike()
root = TK()
root.config()
root.title("Welcome to bike ...
root.minsize(height= 700, width 700)
on(w1, text="Bike motors", wid...
    40, height 2, command=...
()
```


Travelling from one window to another and making use of geometry layout managers

Algorithm:

step 1: Define a function and create a object of the given window by using the three methods 1) config 2) title 3) minsize

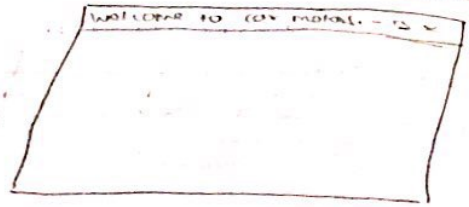
step 2: create a button object and use the text and the command attribute for the triggering of the given event and use the grid method. similarly create another button object which will allow the application to terminate.

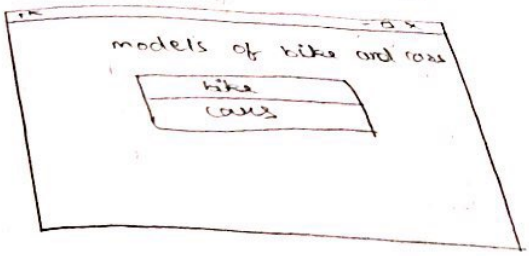
step 3: Define the second function corresponding to the second window with the attribute for the window object and define one button goes which will shift to the third window.

step 4: create 3rd window object and two button for making onto quit from the destroying program.

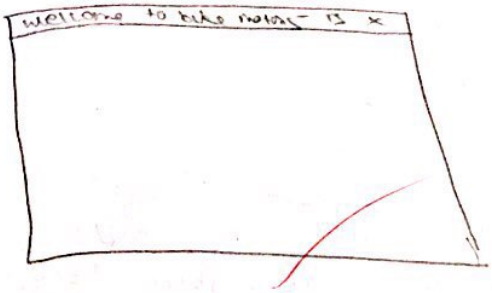
step 5: Define a function for terminate and call the quit() method and

on clicking car motors





on clicking Bike motors



```
root.title("...")
leftframe = Frame(root, height=100, width=...
rightframe = Frame(root, height=100, width=...
rightframe.grid(row=0, column=1, ...
pady=20)
label(leftframe, text="original Image...", ...
RAISED).
...el.grid(row=0, column=0)
...TO = PhotoImage(file="sample.gif")
...el(leftframe, image=photo)
...el.grid(row=2, column=3, padx=10, pady=10)
...(rightframe, image=photo)
...grid(row=1, column=2)
...= Frame(leftframe, height=50, width=...
...grid(row=2, column=2, padx=...
(0)
...el(name()
...text1 = "RAJ"
...label.config(text=text1, Justify...
...onfig(text=text3, justify=LEFT)
...e():
...t = "24"
...config(text=text2, justify=LEFT)
...ht():
...text3 = "75"
...config(text=text, justify=LEFT
```


Practical 5(a)

Aim: Write a program for displaying the image by using the concept of frame toolbar, grid method and button method

Algorithm:

step 1: Create an object corresponding to the parent window and use the following three method
1) title 2) maxsize (3) config()

step 2: create a leftframe object from frame method and place it onto the parent window with height and width specified subsequently use grid method. pad, pady.

step 3: now create a rightframe object from frame method with height and width specified and the row value should be specified.

step 4: create a label object from label method and place it onto the frame with text attribute as original image grid method with column value with some options

step 5: now use the photo image with the file attribute

step 6: use the subexample with the object so the image give x & y co-ordinate values

step 7: use the label method position it onto the leftframe place the image after the and use the grid method positioning in the first values

step 8: create another label object positioning onto rightframe and the image and background row and column specific

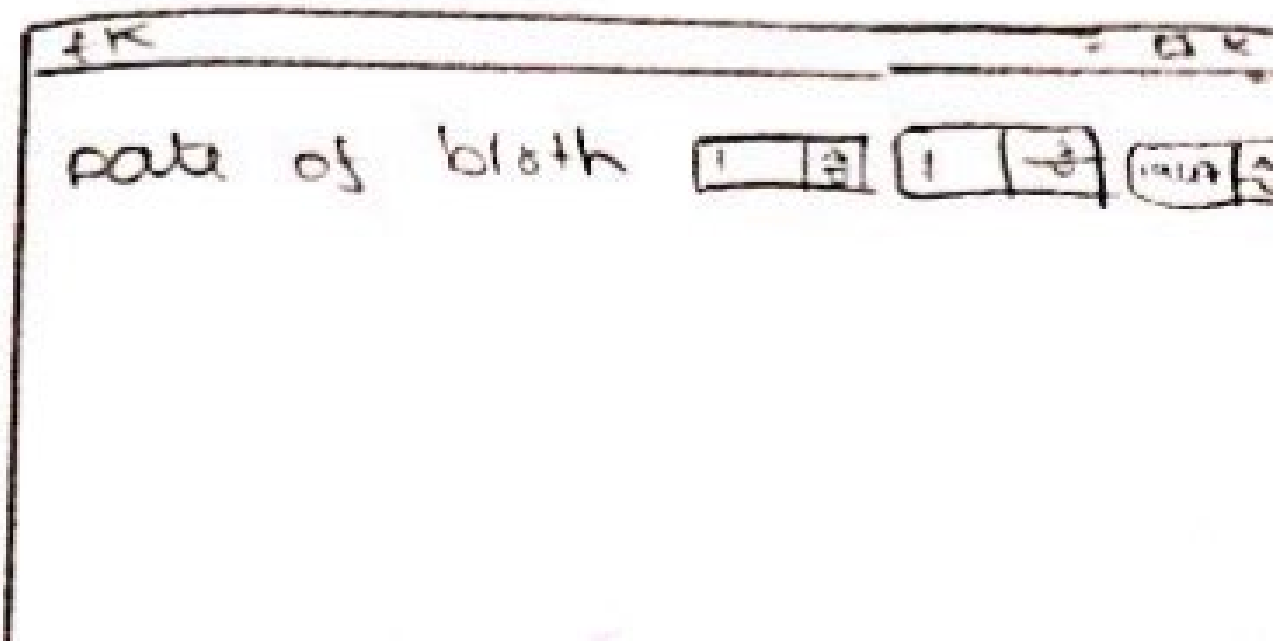
step 9: now create a toolbar object from the form method and position it onto the left frame with height to width specified and position on the corel row

p 10: now define the various the different toolbar option left frame

: From the label method position

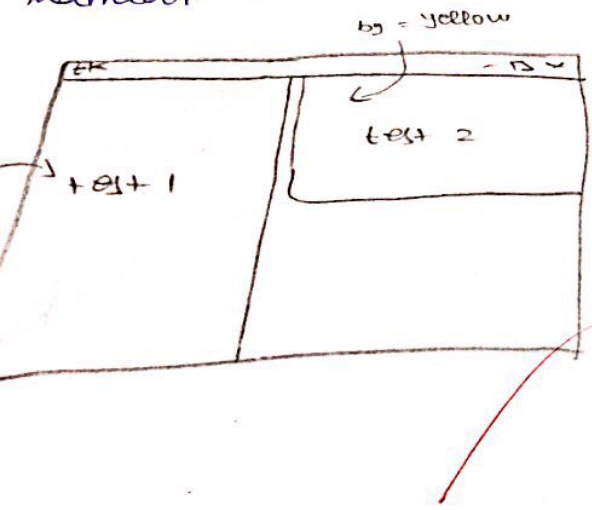
```
def right():
    text1= "6 ft in"
    label = Label(rightFrame, text= text1).grid(sticky=LEFT)
label = Label(leftFrame, text= "Info")
label.grid(row=0, column=0)
b1 = Button(toolbar, text = "name", command=
b1.grid(row=1, column=0)
b2 = Button(toolbar, text = "AGE", command=
b2.grid(row=1, column=1)
b3 = Button(toolbar, text="weight", command
b3.grid(row=2, column=0)
bt = Button
bu = Button(toolbar, text = "Height",
           command = height)
bu.grid(row=2, column=1)
label = Label(rightFrame)
label.grid
mainloop()
```

```
source code
from tkinter import *
root = Tk()
root.minsize(height = 500, w
root.title("date of birth")
label(root, text = "date of birth").place(x = 50, y = 100)
s1 = Spinbox(root, from_ = 1, to = 31, ).place(x = 140, y = 100)
s2 = Spinbox(root, from_ = 1, to = 12, ).place(x = 100, y = 100)
s3 = Spinbox(root, from_ = 1900, to = 2020, ).place(x = 240, y = 100)
mainloop()
```



source code

```
from tkinter import *
root = Tk()
P1 = Panedwindow()
P1.pack(fill = both, expand = 1)
L1 = Label(P1, text = "Test1", bg = "yellow")
P1.add(L1)
P2 = Panedwindow(P1, orient = VERTICAL)
P1.add(P2)
L2 = Label(P2, text = "Test 2", bg = "yellow")
P2.add(L2)
mainloop
```



53

Paned window

Algorithm:

step 1: Import the relevant method from tkinter library

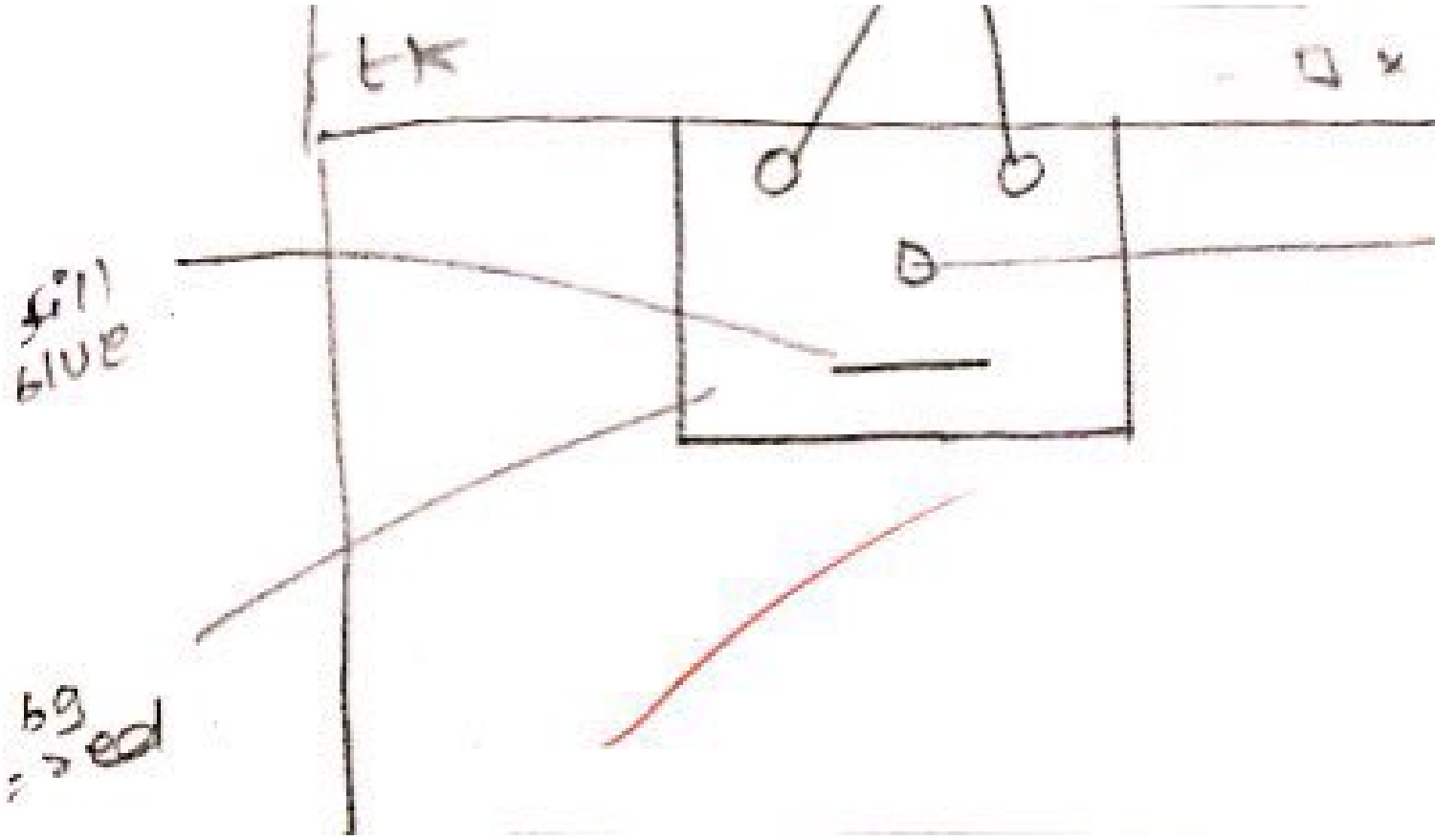
step 2: create an object of parent window and also an object of paned window method

step 3: Now use attribute fill and expand in pack method

step 4: create an object of label method Now use add() along with and object of paned it in window specified.

step 5: Now use add() along with object at paned window and label

step 7: finally, trigger the main loop method.



:8

source code :

```
import dbm
db = dbm.open("demo", 'flag')
db["demo"] = "Database"
if db["demo"] != None:
    print("Found!!!")
else:
    print("Not Found")
db.close()
```

output :

Found!

Algorithm:

step 1: Import corresponding library making database connection

step 2: Now create object cursor cursor from the connection using object

step 3: now use the execute() for creating the table with the column name to respective data type

step 4: use the commit() to commit the transaction using the connection object

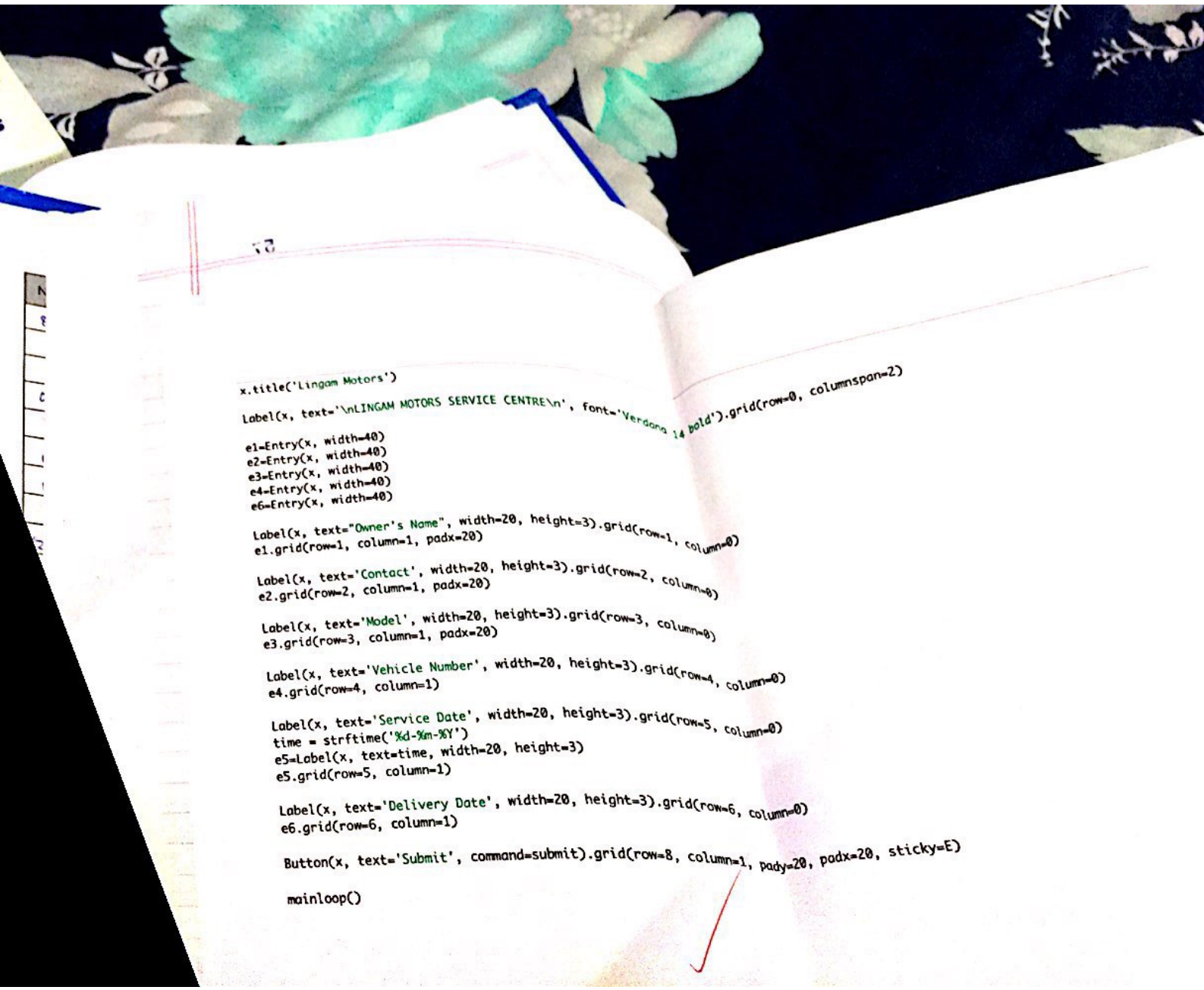
step 5: use the execute statement along with cursor object for the values from database using the select from where clause.

step 6: Finally use fetchall() for displaying values

step 7: use the execute() and

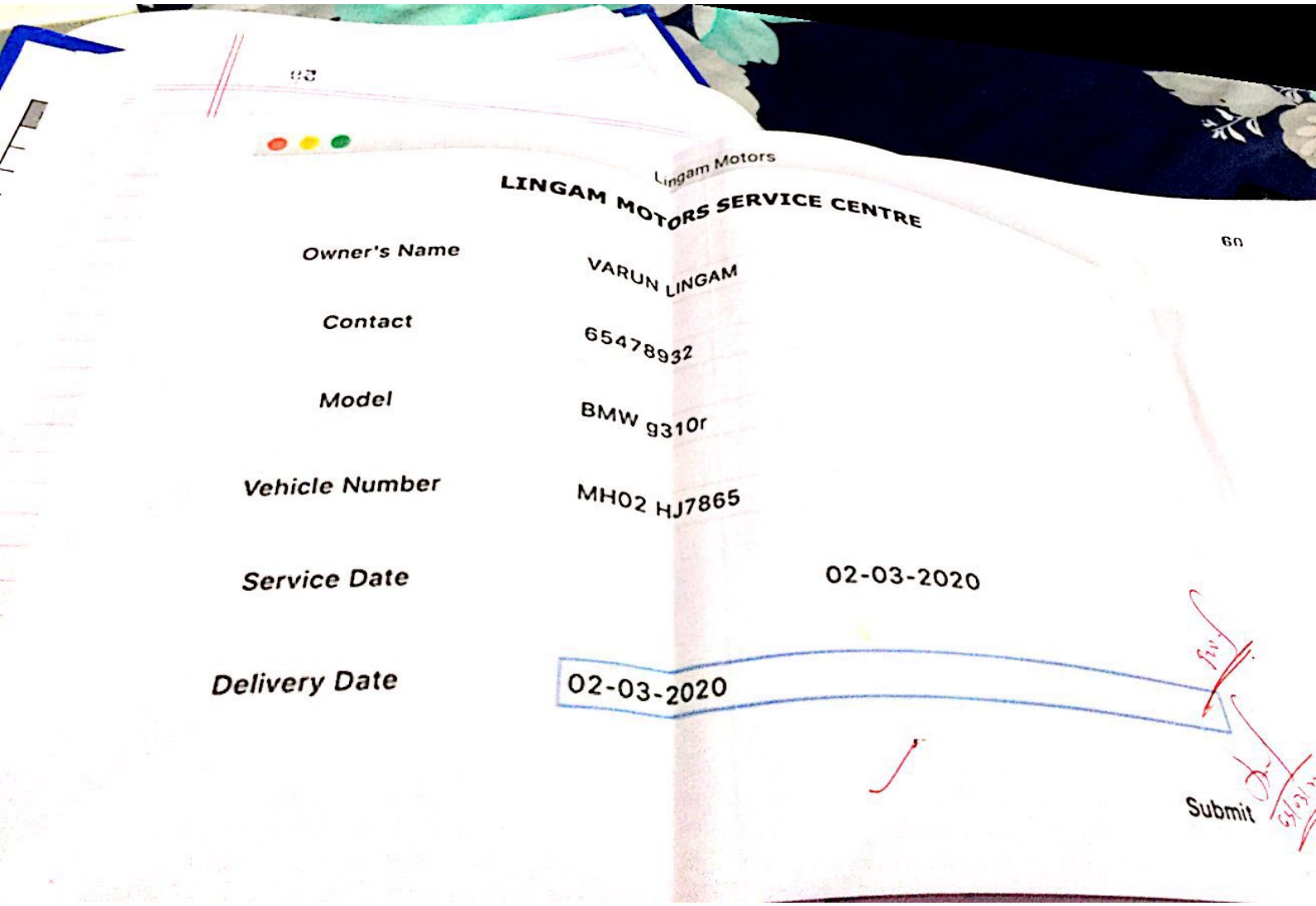
Project :- (I) service centre

```
lo" INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE\
 NOT NULL, "Vehicle_no" TEXT NOT NULL\

e_no,Service_Date,Delivery_Date) VALUES (?,?,?,?,?,?)'\
```



```
x.title('Lingam Motors')

Label(x, text='\nLINGAM MOTORS SERVICE CENTRE\n', font='Verdana 14 bold').grid(row=0, columnspan=2)

e1=Entry(x, width=40)
e2=Entry(x, width=40)
e3=Entry(x, width=40)
e4=Entry(x, width=40)
e6=Entry(x, width=40)

Label(x, text="Owner's Name", width=20, height=3).grid(row=1, column=0)
e1.grid(row=1, column=1, padx=20)

Label(x, text='Contact', width=20, height=3).grid(row=2, column=0)
e2.grid(row=2, column=1, padx=20)

Label(x, text='Model', width=20, height=3).grid(row=3, column=0)
e3.grid(row=3, column=1, padx=20)

Label(x, text='Vehicle Number', width=20, height=3).grid(row=4, column=0)
e4.grid(row=4, column=1)

Label(x, text='Service Date', width=20, height=3).grid(row=5, column=0)
time = strftime('%d-%m-%Y')
e5=Label(x, text=time, width=20, height=3)
e5.grid(row=5, column=1)

Label(x, text='Delivery Date', width=20, height=3).grid(row=6, column=0)
e6.grid(row=6, column=1)

Button(x, text='Submit', command=submit).grid(row=8, column=1, pady=20, padx=20, sticky=E)

mainloop()
```

```
Python 3.7.4 (v3.7.4:e09359112e, Jul  8 2019, 14:54:52)
[Clang 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license()" for more information.
>>>
========== RESTART: /Users/varun/Documents/ddva.py ==========
(1, 'varun lingam', 1234567891, 'BMW X4', 'MH02 VL1234', '02-03-2020', '03-03-2020')
(2, 'dharan shah', 213454657, 'BMW g310r', 'MH02 YU0009', '02-03-2020', '04-03-2020')
(3, 'dinesh lohar', 45367853, 'DUCATI PANIGALE', 'MH02 DK3452', '02-03-2020', '06-03-2020')
(4, 'raj konkar', 435267879, 'TRIUMPH ROCKET', 'MH02 RK1243', '02-03-2020', '05-03-2020')
(5, 'VARUN LINGAM', 65478932, 'BMW g310r', 'MH02 HJ7865', '02-03-2020', '02-03-2020')
```



Lingam Motors

LINGAM MOTORS SERVICE CENTRE

60

Owner's Name	VARUN LINGAM
Contact	65478932
Model	BMW g310r
Vehicle Number	MH02 HJ7865
Service Date	02-03-2020
Delivery Date	02-03-2020

Submit