


# **Computer Science** **Project Report**

## COMPUTER SALES AND SERVICING MANAGEMENT SYSTEM



Project Prepared by:  
Devashish Dhaulakhandi  
XII A  
Session: 2020-2021  
Roll Number: 8

# Table Of Contents

S.no	Topics
1.	Certificate
2.	Acknowledgement
3.	Introduction and Feasibility
4.	A brief history of Python
5.	Problem Definition and Project Objective
6.	Flowchart
7.	System requirements
8.	Libraries/Modules used
9.	Working description / Output
10.	Files generated
11.	Source Code
12.	Bibliography
13.	Shortcomings

# Certificate

This is to certify that DEVASHISH DHAULAKHANDI of class 12 A has completed this project titled  
“COMPUTER SALES AND SERVICING MANAGEMENT SYSTEM”  
under my guidance & this project may be considered as the part of the practical exam of AISSCE conducted by CBSE.

Ms. Annu Saini  
Computer Science

## Acknowledgement

I would like to express a deep sense of thanks and gratitude to my project guide Ms. Annu Saini, for guiding me immensely through the course of the project. She always evinced keen interest in my work. Her constructive advice & constant motivation have been responsible for the successful completion of this project.

I also thank my parents for their motivation and support. I must thank my classmates for their timely help & support for compilation of this project.

Last but not the least, I would like to thank all those who helped directly or indirectly towards the completion of this project.




## Introduction and Feasibility

This Python project is a unique software that can be used in a Computer system sales store to keep a record of bills and services.

It is a full fledged software with graphics and interactive functions that can be used by anyone with ease.  
It is feasible and cost effective.

No online access required and it is a standalone software.  
Only python and a few libraries are required to start.



The bill and services entries are stored as text files in a secure directory that can be set according to the user during installation of this system. An inbuilt calculator is also provided for the manager to calculate the final bill.

The bills are stored with time and dates of creation and changes. It is handy with a show all bills feature.

It is a useful , helpful project which was fun to make.

# A brief history of Python

## Introduction

Python is a high-level programming language that focuses on code readability.

Clear programming is enabled by the constructs provided in the language on both large and small scale.

Python has gained a lot of popularity.

## Early History

Guido van Rossum worked for a project “ABC Language” in Dutch CWI research institute. Guido van Rossum created a successor of ABC Language that was capable of exception handling and compatible with Amoeba operating system interface. This language was named as Python.



## Python 3

On December 3, 2008, Python 3.0 was released which was also known as Python 3000 and Py3K.

The main idea behind the design of Python 3.0 was to remove the fundamental design flaws of the language.

Backward compatibility could not be retained in order to make the changes and therefore, it needed an important version update.

A lot of efforts were put to remove duplicate constructs and modules to make sure that there is only a single way of doing things in Python 3.



# File Handling

Python supports file handling and allows users to handle files i.e., to read and write files, along with many other file handling options, to operate on files.

Python treats file differently as text or binary.

Each line of code includes a sequence of characters and they form text file.

Each line of a file is terminated with a special character, called the EOL or End of Line characters like comma {,} or newline character.

It ends the current line and tells the interpreter a new one has begun.

## Problem Definition:

The project is developed for all users, which aims to automate it's various tasks.

All the work of the tasks was earlier carried out manually. Automation provides a quick and easy step by step approach and Handle the tasks like Billing, Modifying bills, Updating, Keeping records , etc.

Thus, automation is no longer an option, but a necessity.

Writing and logging all the financial transactions of a Computer Sales store is tedious and is prone to errors and mistakes. It is feasible and better to use digital systems for such tasks.

## Advantages of the project



The project aims to develop a Management system that is clean, user-friendly and multifunctional. The development of this application includes a number of fields such that the user feels comfortable and system appears dynamic.

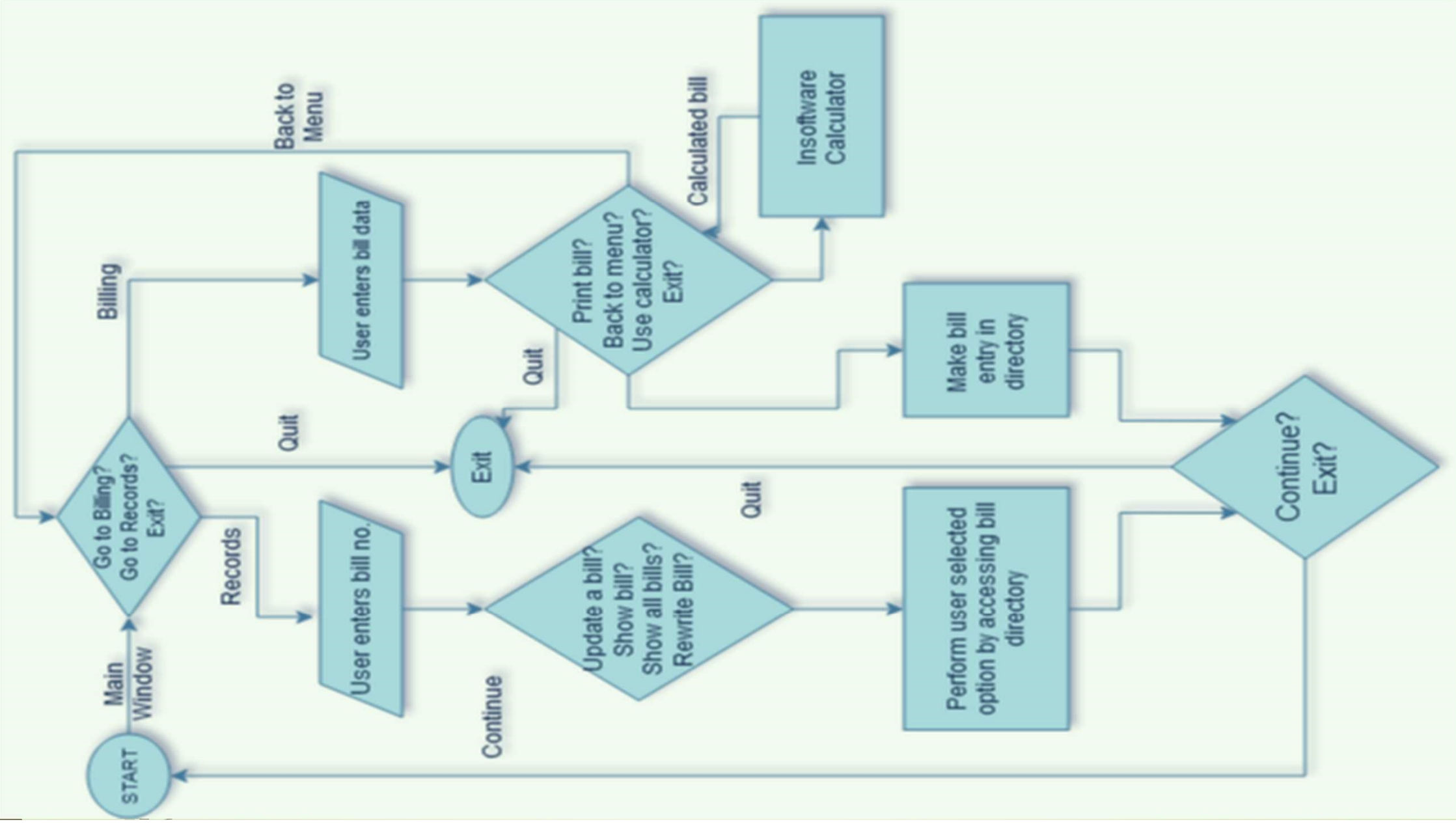
Through the new Sales Management system, all the work has been automated and replacing the process manual workload.

Through this new system, the manager can log all the financial transactions and Billing. Further any bill entry can be Manually updated/modified. A calculator is also an additional feature for the user to calculate the bill.



## This Program consists of the following options:

1. To Write /Add a new text file
2. To show one or all files
3. To search and edit files
4. To delete files
5. To exit
6. An inbuilt calculator



# Libraries/Modules used in the program

1.Tkinter ~ Tkinter is the standard GUI (Graphical User Interface) library for python.

It is a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

2.Datetime ~ The datetime module supplies classes for manipulating dates and times.

3.Playsound ~ The playsound module contains only one thing, the function playsound.

4.PIL/pillow ~ Pillow is a Python Imaging Library (PIL), which adds support for opening, manipulating, and saving images.

5.Os ~The os module in python provides functions for interacting with the operating system. Os comes under Python's standard utility modules.

# Working description / Output

The interface is tkinter based windows with buttons and entry widgets.

The software uses the os library to access, open, delete, remove files from the system.

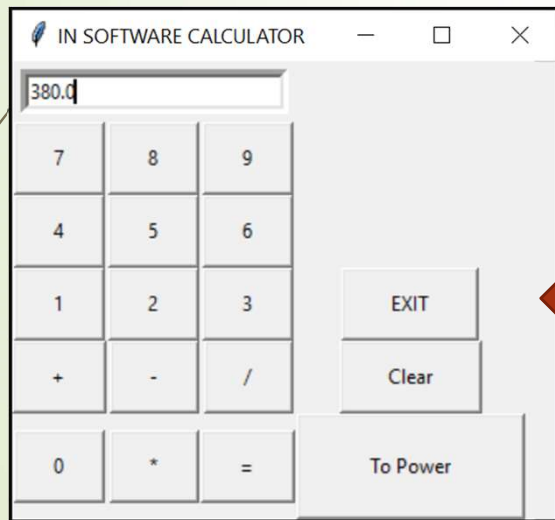
## The main window:

The window consists of different buttons and other windows have entry fields to input information to save :



## The Billing Window:

Inbuilt calculator:



A screenshot of a window titled "BILLING SOFTWARE". The interface has a blue background with white text and buttons. At the top, the word "BILLING" is displayed in a large, stylized font. Below it, there are three input fields labeled "NAME", "Phone no.", and "Bill No.". The "NAME" field contains "Dev", "Phone no." contains "9953951873", and "Bill No." contains "01". Below these fields is a button labeled "Parts~Services~Prices". To the right of this button is a table with two columns: "Part" and "Price". The table contains the following data:

Part	Price
keyboard rs250	xxxxxxxxxxxx
mouse rs.50	xxxxxxxxxxxx
monitor rs.80	xxxxxxxxxxxx
xxxxxxxxxxxx	xxxxxxxxxxxx
xxxxxxxxxxxx	xxxxxxxxxxxx

Below the table, there are two buttons: "<---Mainscreen" and "Print Bill Entry". To the right of the "Print Bill Entry" button is a button labeled "CALC.". A red arrow points from the "CALC." button to the "Print Bill Entry" button, and another red arrow points from the "Print Bill Entry" button to the "IN SOFTWARE CALCULATOR" window.

Saves/Prints the User entered information  
In a bill entry



## The Records window:

The screenshot shows a window titled "RECORDKEEPING" with a sub-header "RECORDS". The window has a blue background and contains several interactive elements:

- A label "Bill No." followed by a text input field and a "Show Bill" button.
- A "Rewrite bill" button.
- A label "New Parts/Services To Add" followed by a text input field and an "Add to bill" button.
- A "Show All" button.
- An "Exit" button.
- A "Delete Bill" button.

This window has various Buttons and Entry fields.

The user enters the bill no. and then decides which operation to carry out.

The user clicks the buttons they choose, this window is also used to add new data to the selected bill.

It can also be completely rewritten and edited.

Main Window Has Two Button Options:  
Billing ~ Records



Along with Entry fields , Billing Window Has Options:  
Main window      Print/Save Bill      Calculator      Exit

BILLING SOFTWARE

***BILLING***

*NAME*

Dev

*Phone no.*

9953951873

*Bill No.*

01

*Parts~Services~Prices*

keyboard rs250	xxxxxxxxxxx
mouse rs.50	xxxxxxxxxxx
monitor rs.80	xxxxxxxxxxx
xxxxxxxxxxx	xxxxxxxxxxx
xxxxxxxxxxx	xxxxxxxxxxx

*<---Mainscreen*

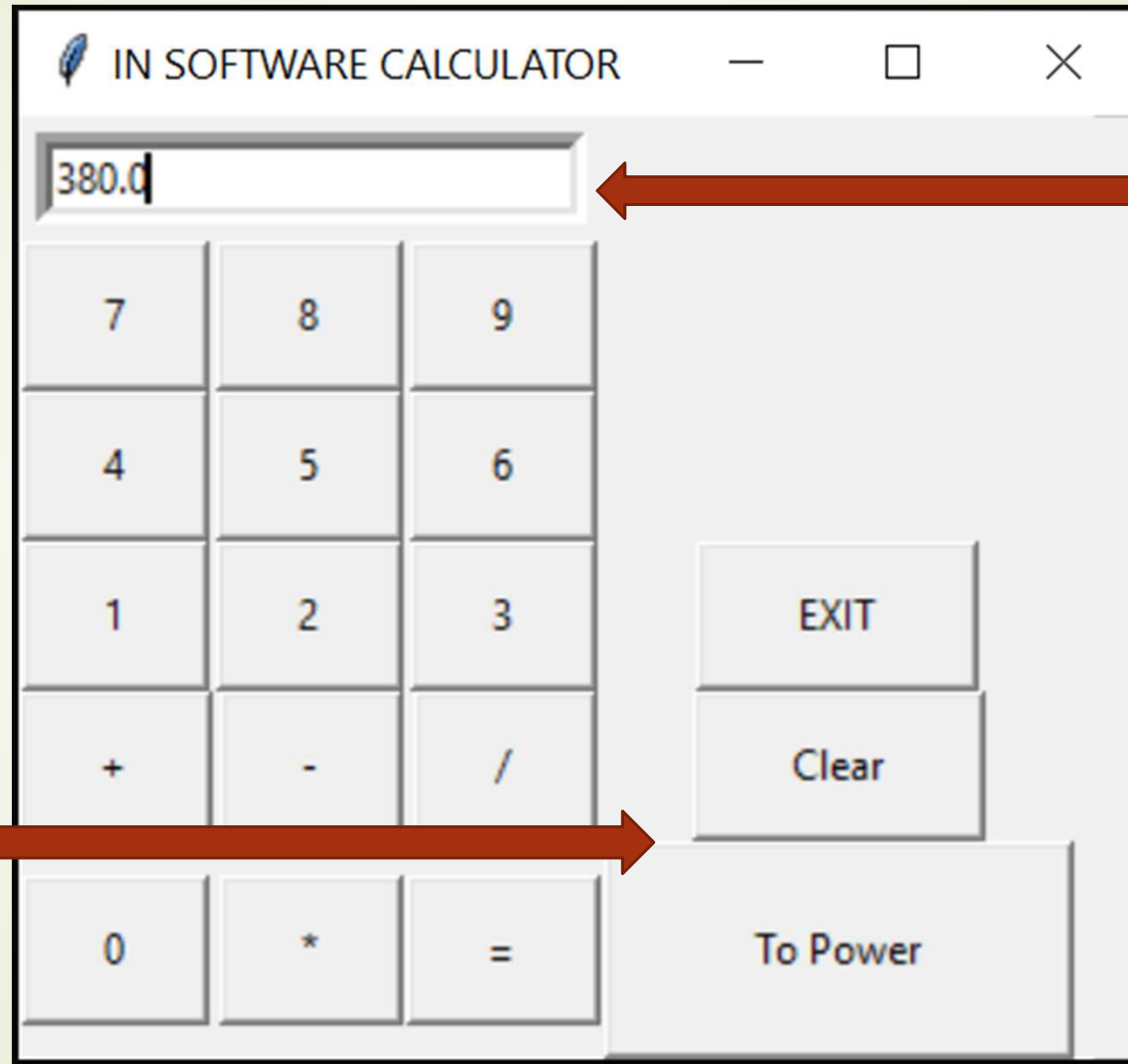
*Print Bill Entry*

*CALC.*

## In Software Calculator:

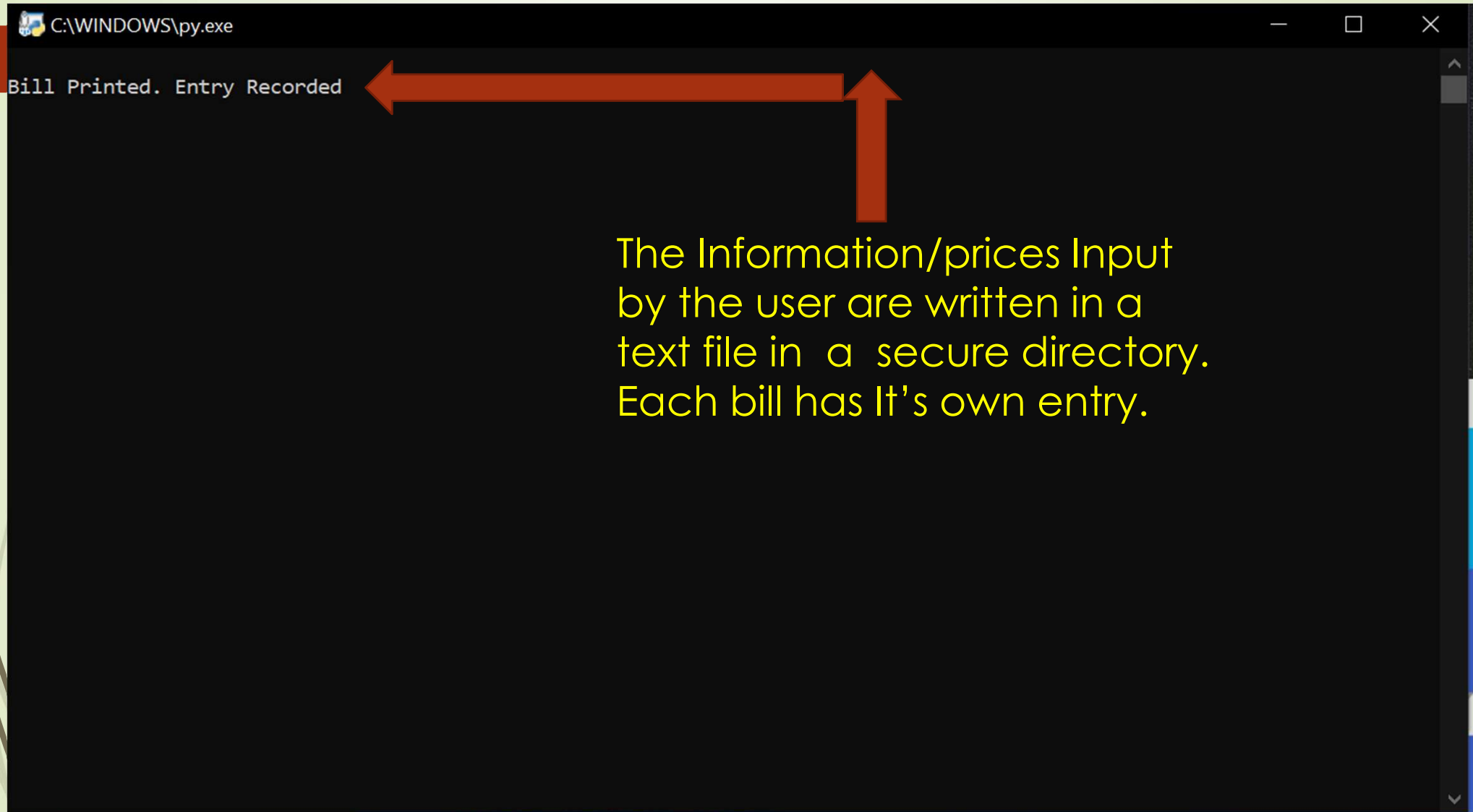
Buttons used as  
in a usual calculator.

The given buttons  
are used according  
To the symbols  
Shown.



Direct output

## Output Of “Print Bill” button:




A screenshot of a Windows command prompt window titled "C:\WINDOWS\py.exe". The window has a black background and white text. The text "Bill Printed. Entry Recorded" is displayed on the first line. A red arrow points from the text "The Information/prices Input by the user are written in a text file in a secure directory. Each bill has It's own entry." to the text "Bill Printed. Entry Recorded".

```
C:\WINDOWS\py.exe
Bill Printed. Entry Recorded
```

The Information/prices Input  
by the user are written in a  
text file in a secure directory.  
Each bill has It's own entry.

Records Window has options :



An Entry field for the Bill no. of the chosen Bill.  
Rewrite bill : it allows user to rewrite any and all details of the bill entry.

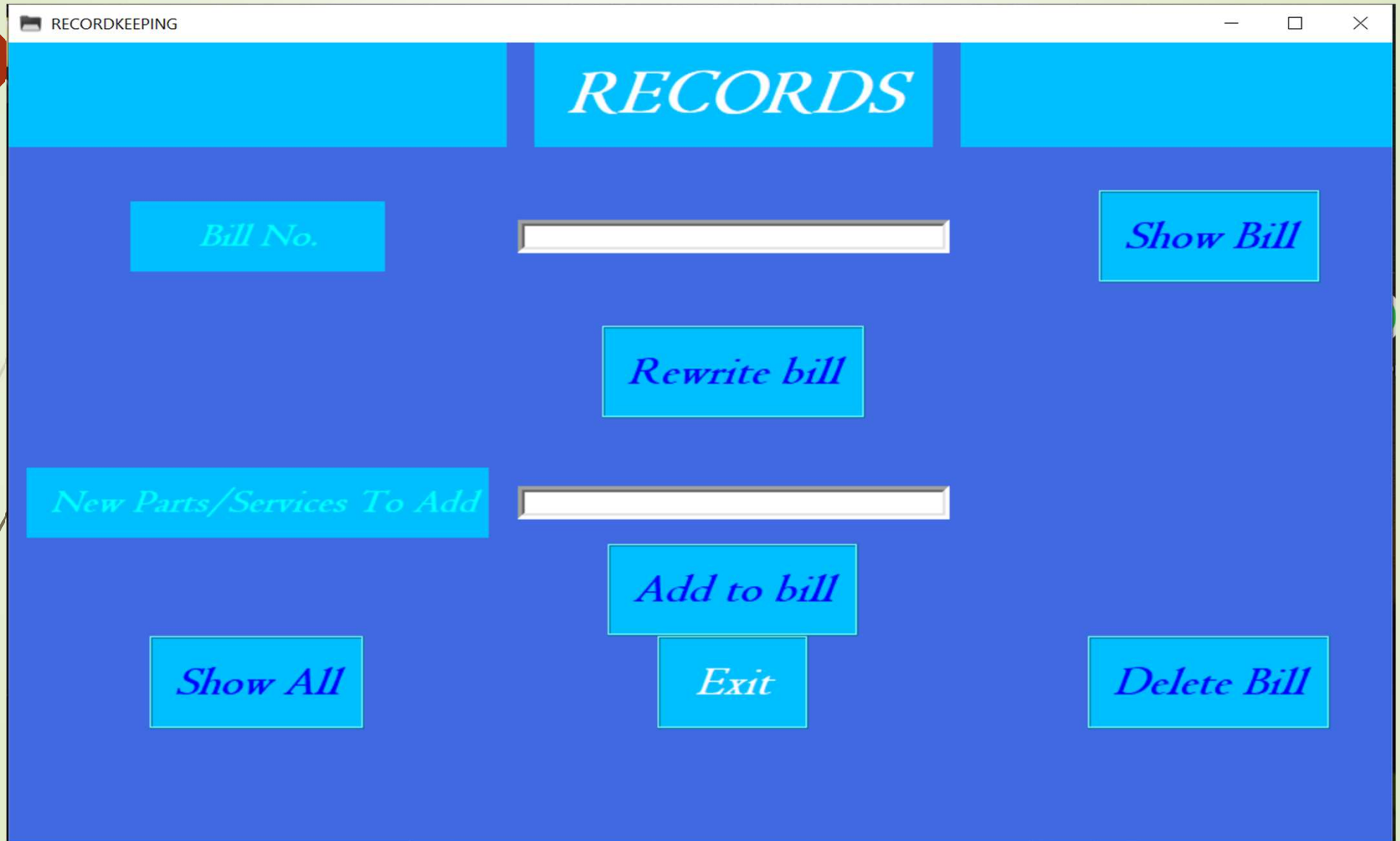
Show bill : prints the bill entry whose bill no. is entered.

Add to bill : It adds the entered information to the bill entry.

Delete bill : It deletes the bill entry.

Show all : It prints all the bills in the directory.

## Records Window:



The image shows a software window titled "RECORDKEEPING" with a blue header bar containing the word "RECORDS" in white. The main area has a blue background and contains several interactive elements: two text input fields, one labeled "Bill No." and the other "New Parts/Services To Add", both with light blue labels; and seven buttons labeled "Show Bill", "Rewrite bill", "Add to bill", "Show All", "Exit", and "Delete Bill". The buttons are arranged in a grid-like fashion around the input fields. The window has standard Windows-style window controls (minimize, maximize, close) in the top right corner.

RECORDKEEPING

*RECORDS*

*Bill No.*

*Show Bill*

*Rewrite bill*

*New Parts/Services To Add*

*Add to bill*

*Show All*

*Exit*

*Delete Bill*

## Output of Show bill:

```
Dev 9953951873 01 2020-11-26 23:10:00.585381
```

```
keyboard rs150
```

```
mouse rs50
```

```
monitor rs80
```

```
xxxxxxxxxxxxxx
```

```
xxxxxxxxxxxxxx
```

```
xxxxxxxxxxxxxx
```

```
xxxxxxxxxxxxxx
```

```
xxxxxxxxxxxxxx
```

```
xxxxxxxxxxxxxx
```

```
xxxxxxxxxxxxxx
```

The Bill Is Saved with  
time and date of creation.


The Entries made in the  
Billing window are saved  
In the entry as shown~



## Output of add to bill :

```
Dev 9953951873 01 2020-11-26 23:10:00.585381 #####  
keyboard rs150  
  
mouse rs50  
  
monitor rs80  
  
XXXXXXXXXXXXX  
XXXXXXXXXXXXX  
XXXXXXXXXXXXX  
XXXXXXXXXXXXX  
XXXXXXXXXXXXX  
XXXXXXXXXXXXX  
XXXXXXXXXXXXX  
XXXXXXXXXXXXX  
  
latest entry: THIS NEW PRODUCT/SERVICE IS NOW ADDED TO BILL 01 2020-11-26 23:12:59.541520 *
```

The Add to bill option  
Updates the bill with the  
New Information entered:



This application is small, portable and user friendly.  
Minimum Requirements:

#### System And Hardware:

4gb or greater RAM  
Microprocessor: INTEL i3 or Higher  
Hard disk/SD storage: 256gb or greater  
Printer  
Speakers/audio system

Files Generated:  
~Text files

#### Software Specification:

Platform: Windows7/8/10 OS  
Front end: Python 3.9  
Any text editor  
Various python Modules and Libraries

## Source code :

#Inbuilt modules used and Other modules installed using python's pip installer

```
from tkinter import *          #tkinter module helps make separate windows
import datetime                #datetime module gets local time and day
from playsound import playsound #playsound module plays mp3 files
from PIL import ImageTk,Image  #PIL or pillow library imports .jpg and .png files in tkinter for logo
import os                      #os module used to manage and write files in record directory
```

#main/original window which has logo, Billing and Record window buttons

```
def mainwin():
```

```
    #initialising the main window
```

```
    rootwin = Tk()
```

```
    rootwin.title("Computer Sales and Repairs Database")
```

```
    rootwin.geometry("1000x800")
```

```
    rootwin.iconbitmap(r'C:\Users\DTG\Desktop\Cs class 12th Project\Icons\2towers.ico')
```

```
    rootwin.configure(bg = 'royalblue')
```

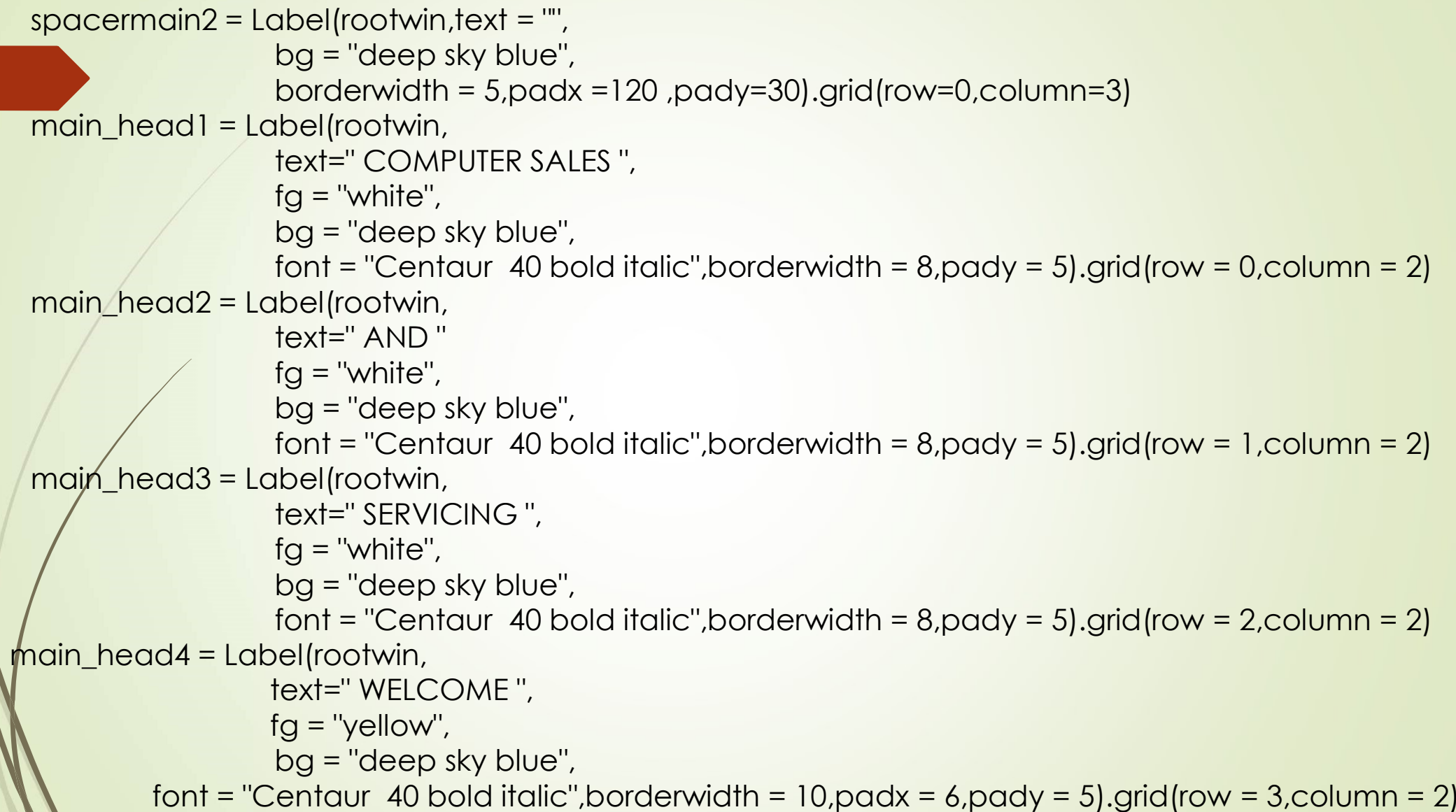
```
    playsound (r'C:\Users\DTG\Desktop\Cs class 12th Project\Sounds\powup3.mp3')
```

```
    #welcome window format and customization
```

```
    spacemain1 = Label(rootwin,text = "",
```

```
                        bg = "deep sky blue",
```

```
                        borderwidth = 5,padx =120 ,pady=30).grid(row=0,column=1)
```




```
spacermain2 = Label(rootwin,text = "",
                    bg = "deep sky blue",
                    borderwidth = 5,padx = 120 ,pady=30).grid(row=0,column=3)
main_head1 = Label(rootwin,
                  text=" COMPUTER SALES ",
                  fg = "white",
                  bg = "deep sky blue",
                  font = "Centaur 40 bold italic",borderwidth = 8,pady = 5).grid(row = 0,column = 2)
main_head2 = Label(rootwin,
                  text=" AND "
                  fg = "white",
                  bg = "deep sky blue",
                  font = "Centaur 40 bold italic",borderwidth = 8,pady = 5).grid(row = 1,column = 2)
main_head3 = Label(rootwin,
                  text=" SERVICING ",
                  fg = "white",
                  bg = "deep sky blue",
                  font = "Centaur 40 bold italic",borderwidth = 8,pady = 5).grid(row = 2,column = 2)
main_head4 = Label(rootwin,
                  text=" WELCOME ",
                  fg = "yellow",
                  bg = "deep sky blue",
                  font = "Centaur 40 bold italic",borderwidth = 10,padx = 6,pady = 5).grid(row = 3,column = 2)
```

#function defined to close the main window and exit the program

```
def closing1():  
    rootwin.destroy()  
    quit()
```

#setting up buttons on main window fonts,size,type etc.

```
button_to_billing = Button(text = "BILLING",padx=6,pady=5,relief = RIDGE,fg = "blue",font = "Centaur  
25 bold italic",  
                           bg = "deep sky blue",command = bill).grid(row = 4,column = 1,)  
button_to_records = Button(text = " RECORDS ",padx=6,pady=5,relief = RIDGE,fg = "blue",  
                           bg = "deep sky blue",font = "Centaur 25 bold italic",command = records ).grid(row  
= 5,column = 1,)  
exit_button = Button(text = "EXIT",padx=6,pady=5,relief = RIDGE,fg = "white",  
                    bg = "deep sky blue",font = "Centaur 30 bold italic",command = closing1 ).grid(row  
= 4,column = 3,)  
#pillow library used to import logo and display it on the window  
myimg1 = Image.open("C:/Users/DTG/Desktop/Cs class 12th Project/Icons/final matching logo.png")  
myimg = myimg1.resize((250,200),Image.ANTIALIAS)  
myimg = ImageTk.PhotoImage(myimg)  
labe = Label(rootwin, image = myimg).grid(row=5,column=2)  
rootwin.mainloop()  
#sets up tkinter window on a loop that waits for some other funtion to occur
```



#secondary window of billing

def bill():

#initialising billing window

rootbill = Tk()

rootbill.title("BILLING SOFTWARE")

rootbill.geometry("1120x550")

rootbill.iconbitmap(r'C:\Users\DTG\Desktop\Cs class 12th Project\Icons\fldr.ico')

rootbill.configure(bg = 'royalblue')

playsound (r'C:\Users\DTG\Desktop\Cs class 12th Project\Sounds\bt3.mp3')

#calculator window initialised

def calculator():

root=Tk()

root.title("IN SOFTWARE CALCULATOR ")

```
e = Entry(root, width = 25, borderwidth=5,) #defining shape and size of window and buttons
e.grid(row=0, column=0, columnspan=3, padx=5, pady =5)
```

```
def clear():
```

```
    e.delete(0, END)
```

```
    #clears screen i.e. clears "entry " so that other value may be entered
```

```
    " basically, the entry screen acts as an onscreen container.
```

```
    In the bclick function, it takes a value from user.
```

```
    Then when an operator is used,
```

```
    that initial value is stored in fnum as float value,
```

```
    op is given a math operation and the screen is cleared.
```

```
    Then another value is entered on the screen by user,
```

```
    and finally the equal function takes these and according to op goes to
    the if else statement for add sub mul div etc.
```

```
    """
```

```
def power():
```

```
    global fnum
```

```
    global op
```


```
    fn = e.get()
```

```
    fnum = float(fn)
```

```
    e.delete(0, END)
```

```
    op = "exp"
```






```
def add():
    global fnum
    global op
    fn = e.get()
    fnum = float(fn)
    e.delete(0,END)
    op = "add"

def sub():
    global fnum
    global op
    fn = e.get()
    fnum = float (fn)
    e.delete(0,END)
    op = "sub"

def mul():
    global fnum
    global op
    fn = e.get()
    fnum = float(fn)
    e.delete(0,END)
    op = "mul"
```





```
def div():
    global fnum
    global op
    fn = e.get()
    fnum = float(fn)
    e.delete(0,END)
    op = "div"

def bequal():
    sn = float(e.get())
    e.delete(0,END)
    if op == "add":
        e.insert(0,fnum+sn)
    if op == "sub":
        e.insert(0,fnum-sn)
    if op == "mul":
        e.insert(0,fnum*sn)
    if op == "div":
        e.insert(0,fnum/sn)
    if op == "exp":
        e.insert(0,fnum**sn)
```



#enters value to screen

def bclick(n):

current = e.get()

e.delete(0,END)

y= str(current)+str(n)

e.insert(0,y)

#no. buttons defined

b1=Button(root,text="1",padx=20,pady=10,command = lambda : bclick(1)).grid(row=3,column=0)

b2=Button(root,text="2",padx=20,pady=10,command = lambda : bclick(2)).grid(row=3,column=1)

#put them on the screen

b3=Button(root,text="3",padx=20,pady=10,command = lambda : bclick(3)).grid(row=3,column=2)

b4=Button(root,text="4",padx=20,pady=10,command = lambda : bclick(4)).grid(row=2,column=0)

b5=Button(root,text="5",padx=20,pady=10,command = lambda : bclick(5)).grid(row=2,column=1)


b6=Button(root,text="6",padx=20,pady=10,command = lambda : bclick(6)).grid(row=2,column=2)

b7=Button(root,text="7",padx=20,pady=10,command = lambda : bclick(5)).grid(row=1,column=0)


b8=Button(root,text="8",padx=20,pady=10,command = lambda : bclick(8)).grid(row=1,column=1)

b9=Button(root,text="9",padx=20,pady=10,command = lambda : bclick(9)).grid(row=1,column=2)

b0=Button(root,text="0",padx=20,pady=10,command = lambda : bclick(0)).grid(row=5,column=0)




```
bclickadd= Button(root,text="+",padx=20,pady=10,command = add).grid(row=4,column=0)
bclicksub= Button(root,text="-",padx=20,pady=10,command = sub).grid(row=4,column=1)
bclickdiv= Button(root,text="/",padx=20,pady=10,command = div).grid(row=4,column=2)
bclickmul= Button(root,text="*",padx=20,pady=10,command = mul).grid(row=5,column=1)
bclickpower= Button(root,text="To Power",padx=40,pady=20,command = power).grid(row=5,column=3)
bequal=Button(root,text="=",padx=20,pady=10,command = bequal).grid(row=5,column=2)
bec1=Button(root,text="Clear",padx=25,pady=10,command = clear).grid(row=4,column=3)
#defining the funtion to close the calculator window,button is present on the billing window
def closecalc():
    root.destroy()
bexit=Button(root,text = "EXIT",padx = 26, pady = 10,command = closecalc).grid(row = 3,column = 3)
root.mainloop()
```



#function to add a new data entry to the records directory from  
#the billing window information provided by user  
def newbill():


```
x = ename.get()
ename.delete(0,END)
y = ephone.get()
ephone.delete(0,END)
z = ebillno.get()
ebillno.delete(0,END)
a = econtent2.get()
econtent2.delete(0,END)
b = econtent3.get()
econtent3.delete(0,END)
c = econtent4.get()
econtent4.delete(0,END)
d = econtent5.get()
econtent5.delete(0,END)
e = econtent6.get()
econtent6.delete(0,END)
```



```
f = econtent02.get()
econtent02.delete(0,END)
g = econtent03.get()
econtent03.delete(0,END)
h = econtent04.get()
econtent04.delete(0,END)
i = econtent05.get()
econtent05.delete(0,END)
j = econtent06.get()
econtent06.delete(0,END)
p = "\n"
qwerty = '#####'
k = [a,b,c,d,e,f,g,h,i,j]
now = datetime.datetime.now()
v = str(now)
r = x + " " + y + " " + z + " " + v + " "
```

#a line of name, phone no., bill no., date and time saved in the text file

```
mf = open('C:/Users/DTG/Desktop/trial project cs/'+ z +'.txt', 'w')
mf.write(r)
mf.close()
mf = open('C:/Users/DTG/Desktop/trial project cs/'+ z +'.txt', 'a')
mf.write(qwerty)
```




```
for content in k:
    if content != '\n':
        mf.write(p)
        mf.write(content)
        mf.write(p)
    else:
        continue

mf.close()
print("")
print("Bill Printed. Entry Recorded ")
print("")

def closing2():
    rootbill.destroy()
```

```
main_head1 = Label(rootbill, text="    BILLING    ",
                    fg = "white",
                    bg = "deep sky blue",
                    font = "Centaur 40 bold italic",borderwidth = 10,pady = 5).grid(row = 0,column = 2)
Name = Label(rootbill,
              text="    NAME    ",
              fg = "cyan",
              bg = "deep sky blue",
              font = "Centaur 20 bold italic",borderwidth = 8,pady = 5).grid(row = 1,column = 1)
Phone = Label(rootbill,
               text="    Phone no.    ",
               fg = "cyan",
               bg = "deep sky blue",
               font = "Centaur 20 bold italic",borderwidth = 8,pady = 5).grid(row = 1,column = 2)
Bill = Label(rootbill,
              text="    Bill No.    ",
              fg = "cyan",
              bg = "deep sky blue",
              font = "Centaur 20 bold italic",borderwidth = 8,pady = 5).grid(row = 1,column = 3)
Parts = Label(rootbill,
               text="Parts~Services~Prices",
               fg = "blue",
               bg = "deep sky blue",
               font = "Centaur 15 bold italic",borderwidth = 8,pady = 5).grid(row = 3,column = 1)
```




#entry fields for information of customer

```
ename = Entry(rootbill, width = 50,borderwidth=5,)
ename.grid(row = 2,column = 1,columnspan = 1,padx = 8 ,pady = 20)
ephone = Entry(rootbill, width = 50,borderwidth=5,
ephone.grid(row = 2,column = 2,columnspan = 1,padx= 8 ,pady = 20)
ebillno = Entry(rootbill, width = 50,borderwidth=5,)
ebillno.grid(row = 2,column = 3,columnspan = 1,padx = 8,pady = 20)
```



```
'''econtent2
econtent3
econtent4
econtent5
econtent6
econtent02
econtent03
econtent04
econtent05
econtent06'''
```





```
econtent2 = Entry(rootbill,width = 60 ,borderwidth= 5,)
econtent2.grid(row = 4,column = 2)
econtent3 = Entry(rootbill,width = 60 ,borderwidth= 5,)
econtent3.grid(row = 5,column = 2)
econtent4 = Entry(rootbill,width = 60 ,borderwidth= 5,)
econtent4.grid(row = 6,column = 2)
econtent5 = Entry(rootbill,width = 60 ,borderwidth= 5,)
econtent5.grid(row = 7,column = 2)
econtent6 = Entry(rootbill,width = 60 ,borderwidth= 5,)
econtent6.grid(row = 8,column = 2)
```

```
econtent02 = Entry(rootbill,width = 60 ,borderwidth= 5,)
econtent02.grid(row = 4,column = 3)
econtent03 = Entry(rootbill,width = 60 ,borderwidth= 5,)
econtent03.grid(row = 5,column = 3)
econtent04 = Entry(rootbill,width = 60 ,borderwidth= 5,)
econtent04.grid(row = 6,column = 3)
econtent05 = Entry(rootbill,width = 6,borderwidth= 5,)
econtent05.grid(row = 7,column = 3)
econtent06 = Entry(rootbill,width = 60 ,borderwidth= 5,)
econtent06.grid(row = 8,column = 3)
```

#button that invokes the previously defined function of adding a new bill

```
bill_button = Button(rootbill,text = " Print Bill Entry ",padx=6,pady=5,relief = RIDGE,fg = "blue",  
bg = "deep sky blue",font = "Centaur 25 bold italic",  
command = newbill ).grid(row = 10,column = 2)
```

```
main_win_button = Button(rootbill,text = "<---Mainscreen",padx=6,pady=5,relief = RIDGE,fg = "white",  
bg = "deep sky blue",font = "Centaur 25 bold italic",  
command = closing2 ).grid(row = 9,column = 1)
```

```
calc_button = Button(rootbill,text = "CALC.",padx=6,pady=5,relief = RIDGE,fg = "white",  
bg = "deep sky blue",font = "Centaur 25 bold italic",  
command = calculator ).grid(row = 9,column = 3)
```

```
rootbill.mainloop()
```

```
def records():
```

```
    rootSales = Tk()
```

```
    rootSales.title(" RECORDKEEPING ")
```

```
    rootSales.geometry("1000x650")
```

```
    rootSales.iconbitmap(r'C:\Users\DTG\Desktop\Cs class 12th Project\Icons\fld.ico')
```

```
    rootSales.configure(bg = 'royalblue')
```

```
    playsound (r'C:\Users\DTG\Desktop\Cs class 12th Project\selected sounds\bt3.mp3')
```


```
def closing2():  
    rootSales.destroy()
```

```
#welcome window format and customization
```

```
spacermain1 = Label(rootSales,text = "",  
                    bg = "deep sky blue",  
                    borderwidth = 5,padx = 175 ,pady=30).grid(row=0,column=1)
```

```
spacermain2 = Label(rootSales,text = "",  
                    bg = "deep sky blue",  
                    borderwidth = 5,padx = 175 ,pady=30).grid(row=0,column=3)
```

```
main_head1 = Label(rootSales,  
                  text=" RECORDS ",  
                  fg = "white",  
                  bg = "deep sky blue",  
                  font = "Centaur 40 bold italic",borderwidth = 8,pady = 5).grid(row = 0,column = 2)
```



```
def Replacebill():
```

```
    z = ebillno2.get()
```

```
    ebillno2.delete(0,END)
```

```
    c = 'C:/Users/DTG/Desktop/trial project cs/'+ z +'.txt'
```

```
    if os.path.exists(c):
```

```
        os.remove(c)
```

```
    else:
```

```
        print("")
```

```
        print(" The Bill does not exist. ")
```

```
        print("")
```


```
    mf = open('C:/Users/DTG/Desktop/trial project cs/'+ z +'.txt', 'w')
```

```
    print('~Bill Replaced~')
```

```
    mf.close()
```

```
    closing2()
```

```
    bill()
```



```
def Appendtobill():
```

```
    z = ebillno2.get()
```

```
    ebillno2.delete(0,END)
```

```
    now = datetime.datetime.now()
```

```
    v = str(now)
```

```
    c = ebilladding.get()
```

```
    ebilladding.delete(1,END)
```

```
    finalcontent = c + "" + v + ' * '
```

```
    o = 'C:/Users/DTG/Desktop/trial project cs/'+ z +'.txt'
```

```
    b = ' latest entry: '
```

```
    p = '\n'
```

```
    if os.path.exists(o):
```

```
        mf = open('C:/Users/DTG/Desktop/trial project cs/'+ z +'.txt', 'a')
```

```
        mf.write(p)
```

```
        mf.write(b)
```

```
        mf.write(finalcontent)
```

```
        mf.write(p)
```

```
        mf.close()
```

```
    else:
```

```
        print("")
```

```
        print(" No record of sale. Contact Customer care at Ph = 9953951873. ")
```

```
        print("")
```



```
def Show_bill():
```

```
    z = ebillno2.get()
```

```
    ebillno2.delete(0,END)
```

```
    c = 'C:/Users/DTG/Desktop/trial project cs/'+ z +'.txt'
```

```
    if os.path.exists(c):
```

```
        mf = open('C:/Users/DTG/Desktop/trial project cs/'+ z +'.txt', 'r')
```

```
        x = mf.read()
```

```
        print("")
```

```
        print(x)
```

```
        print("")
```

```
        mf.close()
```

```
    else:
```

```
        print(" The Bill does not exist. ")
```



```
def deletebill():
```

```
    z = ebillno2.get()
    ebillno2.delete(0,END)
    c = 'C:/Users/DTG/Desktop/trial project cs/'+ z +'.txt'
    if os.path.exists(c):
        os.remove(c)
    else:
        print("")
        print(" The Bill does not exist. ")
        print("")
```

```
def showingall():
```

```
    x = ' '
    for filename in os.listdir('C:/Users/DTG/Desktop/trial project cs'):
        content = open(os.path.join('C:/Users/DTG/Desktop/trial project cs/', filename), 'r')
        x = content.read()
        content.close()
        print(' ')
        print (x)
        print(' ')
```

```
def exitrecords():  
    rootSales.destroy()
```

```
    spacermain1 = Label(rootSales,text = "",  
        bg = "royalblue",  
        borderwidth = 5,padx =145 ,pady=5).grid(row=1,column=1)
```

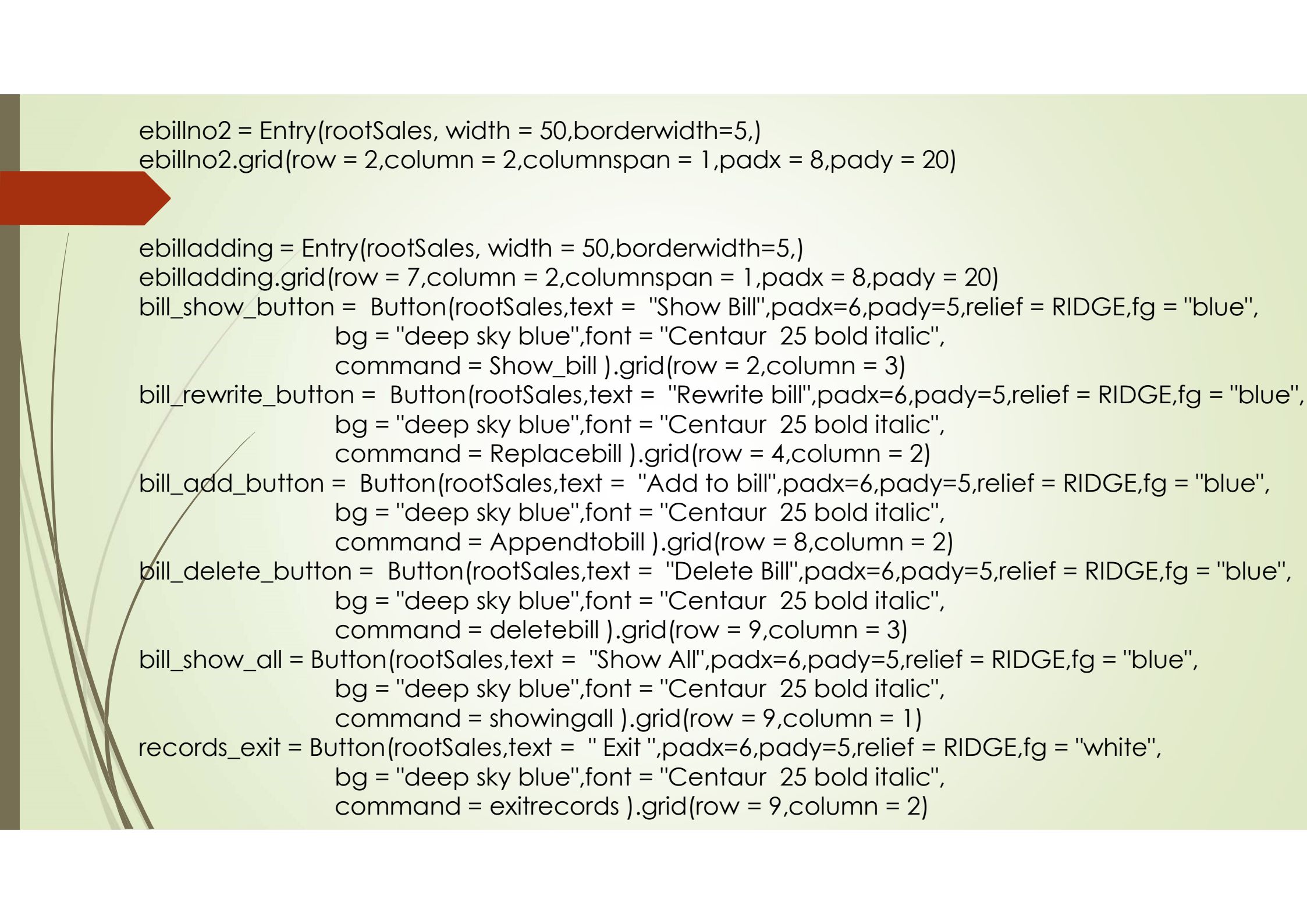
```
    spacermain2 = Label(rootSales,text = "",  
        bg = "royalblue",  
        borderwidth = 5,padx =145 ,pady=5).grid(row=3,column=1)
```

```
    spacermain4 = Label(rootSales,text = "",  
        bg = "royalblue",  
        borderwidth = 5,padx =145 ,pady=5).grid(row=6,column=1)
```

```
    Bill0 = Label(rootSales,  
        text="    Bill No.    ",  
        fg = "cyan",  
        bg = "deep sky blue",  
        font = "Centaur 20 bold italic",borderwidth = 8,pady = 5).grid(row = 2,column = 1)
```


```
    Bill3 = Label(rootSales,  
        text=" New Parts/Services To Add",  
        fg = "cyan",  
        bg = "deep sky blue",  
        font = "Centaur 20 bold italic",borderwidth = 8,pady = 5).grid(row = 7,column = 1)
```





```
ebillno2 = Entry(rootSales, width = 50,borderwidth=5,)
ebillno2.grid(row = 2,column = 2,columnspan = 1,padx = 8,pady = 20)
```

```
ebilladding = Entry(rootSales, width = 50,borderwidth=5,)
ebilladding.grid(row = 7,column = 2,columnspan = 1,padx = 8,pady = 20)
bill_show_button = Button(rootSales,text = "Show Bill",padx=6,pady=5,relief = RIDGE,fg = "blue",
                           bg = "deep sky blue",font = "Centaur 25 bold italic",
                           command = Show_bill ).grid(row = 2,column = 3)
bill_rewrite_button = Button(rootSales,text = "Rewrite bill",padx=6,pady=5,relief = RIDGE,fg = "blue",
                              bg = "deep sky blue",font = "Centaur 25 bold italic",
                              command = Replacebill ).grid(row = 4,column = 2)
bill_add_button = Button(rootSales,text = "Add to bill",padx=6,pady=5,relief = RIDGE,fg = "blue",
                          bg = "deep sky blue",font = "Centaur 25 bold italic",
                          command = Appendtobill ).grid(row = 8,column = 2)
bill_delete_button = Button(rootSales,text = "Delete Bill",padx=6,pady=5,relief = RIDGE,fg = "blue",
                             bg = "deep sky blue",font = "Centaur 25 bold italic",
                             command = deletebill ).grid(row = 9,column = 3)
bill_show_all = Button(rootSales,text = "Show All",padx=6,pady=5,relief = RIDGE,fg = "blue",
                       bg = "deep sky blue",font = "Centaur 25 bold italic",
                       command = showingall ).grid(row = 9,column = 1)
records_exit = Button(rootSales,text = " Exit ",padx=6,pady=5,relief = RIDGE,fg = "white",
                      bg = "deep sky blue",font = "Centaur 25 bold italic",
                      command = exitrecords ).grid(row = 9,column = 2)
```



```
while True :
```

```
    mainwin()
```



# Bibliography:

[www.python-graph-gallery](http://www.python-graph-gallery)

[www.python.org](http://www.python.org)

[www.freecodecamp.org](http://www.freecodecamp.org)

[www.shutterstock.com](http://www.shutterstock.com)

[www.codecademy](http://www.codecademy)


[www.freesound.org](http://www.freesound.org)

[www.pythonworld.in](http://www.pythonworld.in)

[www.flaticon.com](http://www.flaticon.com)

[www.programiz.com](http://www.programiz.com)

## Shortcomings:



The bill entries made are text files and no specific detail can be accessed individually.

The directory has to be secured externally by an antivirus/operating system.

The saved bills can be externally accessed unlike binary files.