DAILY ASSESSMENT

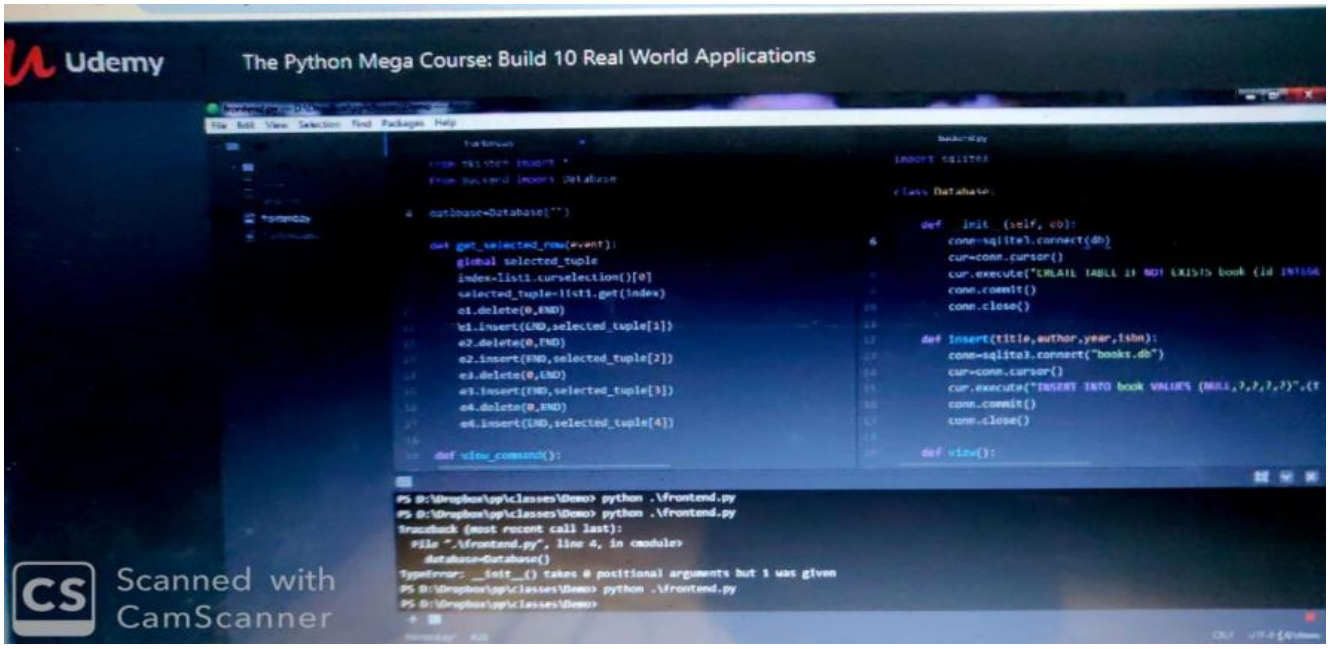
FORENOON SESSION DETAILS

DAILY ASSIGEMENTS DETAILS:

Date:	**29/05/2020**	**Name:**	**Shilpa.S**
Course:	Python	**USN:**	**4AL14EC078**
Topic:	Object oriented programming	**Semester & Section:**	**8(A) sec**
Github Repository:	**Shilpa_online**		

AFTERNOON SESSION

Session image:



REPORT:

- Python has been an object-oriented language since it existed. Because of this, creating and using classes and objects are downright easy.

Creating Classes

The *class* statement creates a new class definition. The name of the class immediately follows the keyword *class*

- The class has a documentation string, which can be accessed via *ClassName.__doc__*.
- The *class_suite* consists of all the component statements defining class members, data attributes and functions.

Creating Instance Objects

To create instances of a class, you call the class using class name and pass in whatever arguments its *__init__* method accepts.

Built-In Class Attributes

Every Python class keeps following built-in attributes and they can be accessed using dot operator like any other attribute −

- **__dict__** − Dictionary containing the class's namespace.
- **__doc__** − Class documentation string or none, if undefined.
- **__name__** − Class name.
- **__module__** − Module name in which the class is defined. This attribute is "__main__" in interactive mode.
- **__bases__** − A possibly empty tuple containing the base classes, in the order of their occurrence in the base class list

Garbage Collection

Python deletes unneeded objects (built-in types or class instances) automatically to free the memory space. The process by which Python periodically reclaims blocks of memory that no longer are in use is termed Garbage Collection

Program:

```
from tkinter import *
from backend import Database
```

```
database=Database("books.db")

class Window(object):

def __init__(self,window):

self.window = window

self.window.wm_title("BookStore")
l2=Label(window,text="Author")
    l2.grid(row=0,column=2)

    l3=Label(window,text="Year")
    l3.grid(row=1,column=0)

    l4=Label(window,text="ISBN")
    l4.grid(row=1,column=2)

self.title_text=StringVar()
    self.e1=Entry(window,textvariable=self.title_text)
    self.e1.grid(row=0,column=1)
self.author_text=StringVar()
    self.e2=Entry(window,textvariable=self.author_text)
    self.e2.grid(row=0,column=3)

self.year_text=StringVar()
    self.e3=Entry(window,textvariable=self.year_text)
    self.e3.grid(row=1,column=1)

self.isbn_text=StringVar()
    self.e4=Entry(window,textvariable=self.isbn_text)
    self.e4.grid(row=1,column=3)

    self.list1=Listbox(window, height=6,width=35)
    self.list1.grid(row=2,column=0,rowspan=6,columnspan=2)

    sb1=Scrollbar(window)
    sb1.grid(row=2,column=2,rowspan=6)

    self.list1.configure(yscrollcommand=sb1.set)
    sb1.configure(command=self.list1.yview)
self.list1.bind('<<ListboxSelect>>',self.get_selected_row)
```

```
        b1=Button(window,text="View all", width=12,command=self.view_command)
        b1.grid(row=2,column=3)

        b2=Button(window,text="Search entry", width=12,command=self.search_command)
        b2.grid(row=3,column=3)

        b3=Button(window,text="Add entry", width=12,command=self.add_command)
        b3.grid(row=4,column=3)
b4=Button(window,text="Update selected", width=12,command=self.update_command)
        b4.grid(row=5,column=3)

        b5=Button(window,text="Delete selected", width=12,command=self.delete_command)
        b5.grid(row=6,column=3)

        b6=Button(window,text="Close", width=12,command=window.destroy)
        b6.grid(row=7,column=3)
defget_selected_row(self,event):
        index=self.list1.curselection()[0]
self.selected_tuple=self.list1.get(index)
        self.e1.delete(0,END)
self.e1.insert(END,self.selected_tuple[1])
        self.e2.delete(0,END)
        self.e2.insert(END,self.selected_tuple[2])
        self.e3.delete(0,END)
        self.e3.insert(END,self.selected_tuple[3])
        self.e4.delete(0,END)
        self.e4.insert(END,self.selected_tuple[4])

defview_command(self):
        self.list1.delete(0,END)
        for row in database.view():
            self.list1.insert(END,row)

defsearch_command(self):
        self.list1.delete(0,END)
        for row is
 database.search(self.title_text.get(),self.author_text.get(),self.year_text.get(),self.isbn_text.get()):
            self.list1.insert(END,row)

defadd_command(self):
        database.insert(self.title_text.get(),self.author_text.get(),self.year_text.get(),self.isbn_text.get())
        self.list1.delete(0,END)
        self.list1.insert(END,(self.title_text.get(),self.author_text.get(),self.year_text.get(),self.isbn_text.get()))
```

```
window=Tk()
Window(window)
window.mainloop()
```