



**Project Tittle:** Computer Lab Management System

**Date of Submission:** 28\02\23

**Course Title:** Programming in Python

**Section:** [A] **Semester:** Spring 2022-2023

**Course Teacher:** Dr. Akinul Islam Jony

**Submitted By:**

**NAME:** Tanjim, Samiul Arif

**ID:** 20-42694-1

**Project Overview:** It is a Python-based console-based computer lab management application. By entering information such as the PC number, operating system, and status of the PC, we can add a new computer to our system. We can also update the data on the outgoing Computer. Any PC that is currently in the system can be deleted or removed. Just writing the inserted data to a text file in the directory of the running application serves as data storage. Also, we have access to all computer data and may search or query it.

This program offers certain fundamental computer functionalities like adding, deleting, searching, updating, presenting, and storing data.

**Project Solution Design:**

START

Create Lab Computer Object with given number, operating system and status

LOOP:

Print menu options

Get user input

If user input is 1, **GET** information from user and **CALL** Add\_New\_Pc() method

If user input is 2, **GET** pc\_number from user and **CALL** Search\_Pc () method

If user input is 3, **GET** pc\_number from user and **CALL** Search\_PC () method

and Update\_Pc () method

If user input is 4, **GET** pc\_number from user and **CALL** Remove\_Pc () method

If user input is 5, **CALL** Show\_All\_Pc\_Info () method

If user input is 6, **CALL** Store\_Info () method

If user input is 7, **CALL** quit () method and terminates the program

If user input is invalid, CONTINUE LOOP

END

**IMPLEMENTATION:**

**Lab\_Module.py**

# Lab\_Module which contains Lab\_Computer Class

**class Lab\_Computer**:  
 pc\_list = [] # To store all the object of this class  
  
 **def \_\_init\_\_(**self, pc\_num, pc\_os, pc\_stat): # Constructor method  
 self.number = pc\_num  
 self.operating\_system = pc\_os  
 self.status = pc\_stat  
  
 Lab\_Computer.pc\_list.append(self)  
 print(f"\n NEW {self.operating\_system} COMPUTER HAS BEEN REGISTERED\n") # Appends instances to the list  
  
 **def Add\_New\_Pc**(self): # For Adding New Computer  
 print(" NEW PC REGISTRATION\n")  
 pc\_num = input("ENTER PC NUMBER: ")  
 pc\_os = input("ENTER PC OPERATING SYSTEM: ")  
 pc\_stat = input("ENTER PC STATUS: ")  
  
 computer\_found = Lab\_Computer.Pc\_Check(pc\_num) # To check whether pc with same exits or not  
  
 if computer\_found == 0: # If no pc has the same pc number, computer found sets to 0  
 lab\_pc = Lab\_Computer(pc\_num, pc\_os, pc\_stat) # Adding to pc to the list object  
 else:  
 print("\nWARNING: THERE IS A EXITING PC WITH SAME NUMBER\n")  
 print("TO UPDATE THE NUMBER OF EXITING COMPUTER - ENTER '1'")  
 print("TO REMOVE EXITING COMPUTER - ENTER '2'")  
 print("TO CANCEL REGISTRATION - ENTER '3'")  
  
 i = input("\n ENTER CHOICE: ")  
  
 if i == '1':  
 new\_pc\_num = input("\nENTER NEW PC NUMBER: ")  
 Lab\_Computer.Update\_Pc(computer\_found, new\_pc\_num, computer\_found.operating\_system,  
 computer\_found.status)  
 lab\_pc = Lab\_Computer(pc\_num, pc\_os, pc\_stat)  
 elif i == '2':  
 Lab\_Computer.Remove\_Pc(computer\_found)  
 lab\_pc = Lab\_Computer(pc\_num, pc\_os, pc\_stat)  
 elif i == '3':  
 Lab\_Computer.Main()  
 else:  
 Lab\_Computer.Main()  
  
 **def Remove\_Pc**(computer): # To Delete a exiting pc  
 print(f"\n {computer.number} IS SUCCESSFULLY DELETED")  
 Lab\_Computer.pc\_list.remove(computer)  
  
 **def Update\_Pc(computer, new\_pc\_num, new\_pc\_os, new\_pc\_stat):** # TO Update Pc's Information  
 if Lab\_Computer.Pc\_Check(new\_pc\_num) == 0:  
 old\_pc\_number = computer.number  
 computer.number = new\_pc\_num  
 computer.operating\_system = new\_pc\_os  
 computer.status = new\_pc\_stat  
 print(f"PC {old\_pc\_number} HAS BEEN UPDATED\n")  
 else:  
 print(f"CAN NOT UPDATE !!! THE NUMBER {computer.number} HAS ALREADY ASSIGNED\n")  
  
 **def Pc\_Check(pc\_num):** # For Checking whether Pc number is unique  
 flag = 1  
 for computer in Lab\_Computer.pc\_list:  
 if computer.number == pc\_num:  
 flag = 0  
 break  
 if flag == 0:  
 return computer  
 else:  
 return 0  
  
 **def Show\_All\_Pc\_Info()**: # For showing all computer information  
 if len(Lab\_Computer.pc\_list) != 0:  
 print(" ALL PC INFORMATION\n")  
 for computer in Lab\_Computer.pc\_list:  
 print(f"PC NUMBER: {computer.number}")  
 print(f"PC OS: {computer.operating\_system}")  
 print(f"PC STATUS: {computer.status}\n")  
 else:  
 print("THERE IS NO PC REGISTERED TO THE SYSTEM!!!")  
  
 **def Search\_Pc(pc\_num):** # Search whether PC exit or not using pC number  
 flag = 0  
 for computer in Lab\_Computer.pc\_list:  
 if computer.number == pc\_num:  
 flag = 1  
 print(" SEARCHED RESULT\n")  
 print(f"PC NUMBER: {computer.number}")  
 print(f"PC OS: {computer.operating\_system}")  
 print(f"PC STATUS: {computer.status}")  
 computer\_found = computer  
 break  
 if flag == 1:  
 return computer\_found  
 else:  
 print(f"NO PC FOUND AS PC NUMBER {pc\_num}")  
 return 0  
  
 **def Store\_Info()**: # Saves the Data in the Directory  
 try:  
 file = "Computer\_List.txt"  
 with open(file, 'w') as file\_ob:  
 for computer in Lab\_Computer.pc\_list:  
 file\_ob.write(f"{computer.number},{computer.operating\_system},{computer.status}\n")  
 print("\nALL PC INFORMATION HAS BEEN SAVED TO THE DIRECTORY !!!")  
  
 except FileNotFoundError:  
 print("\n COULD NOT SAVE INFORMATION !!! PLEASE TRY AGAIN!")  
  
 **def Main()**: # Main Function  
 print(" WELCOME TO COMPUTER LAB MANAGEMENT SYSTEM\n")  
 print("1: ADD A NEW PC")  
 print("2: SEARCH PC INFORMATION")  
 print("3: UPDATE PC INFORMATION")  
 print("4: REMOVE A PC")  
 print("5: SHOW ALL PC INFORMATION")  
 print("6: STORE ALL PC INFORMATION")  
 print("7: EXIT!\n")  
  
 counter = input("ENTER CHOICE: \n")  
  
 if counter == '1':  
 Lab\_Computer.Add\_New\_Pc(Lab\_Computer)  
  
 elif counter == '2':  
 comp\_num = input("ENTER PC NUMBER TO SEARCH: ")  
 Lab\_Computer.Search\_Pc(comp\_num)  
  
 elif counter == '3':  
 comp\_num = input("INSERT PC NUMBER TO BE UPDATED: ")  
 computer\_found = Lab\_Computer.Search\_Pc(comp\_num)  
 if computer\_found != 0:  
 print("FOR {comp\_num} SELECTED PC")  
 new\_pc\_num = input("ENTER NEW PC NUMBER: ")  
 new\_pc\_os = input("ENTER NEW PC'S OPERATING SYSTEM: ")  
 new\_pc\_stat = input("ENTER NEW PC'S STATUS: ")  
 Lab\_Computer.Update\_Pc(computer\_found, new\_pc\_num, new\_pc\_os, new\_pc\_stat)  
  
 elif counter == '4':  
 pc\_num = input("ENTER PC NUMBER TO BE REMOVED: ")  
 computer\_found = Lab\_Computer.Search\_Pc(pc\_num)  
 if computer\_found != 0:  
 Lab\_Computer.Remove\_Pc(computer\_found)  
  
 elif counter == '5':  
 Lab\_Computer.Show\_All\_Pc\_Info()  
  
 elif counter == '6':  
 Lab\_Computer.Store\_Info()  
  
 elif counter == '7':  
 quit()

**Main.py**

import Lab\_Module # Import Module  
while 1>0: # Calling the main function separating the main part of the code  
 Lab\_Module.Lab\_Computer.Main()

BY importing the Lab\_Module we invoked the Lab\_computer.Main() function from Lab\_Computer class.

**APPLICATION OVERVIEW:**

1) **Main Menu:** As the software runs, the user sees the screen below and all the features available for selection there.

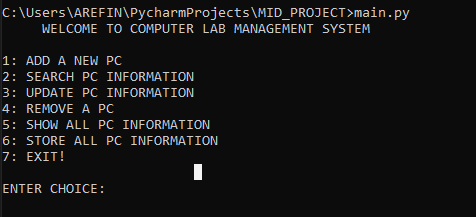


Fig 1: Main Program

2) **Adding a new Computer:** The user is able to add new computers to the program. The user must provide information such the PC number, PC operating system, and PC status when adding a new PC to the system.

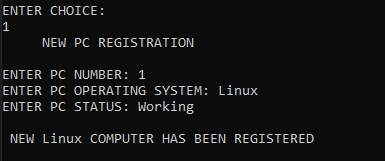


Fig 2: Adding New Computer

3) **Searching PC using Pc number:** Information from PCs is kept as an object in a list. A computer is searched using linear brute force search.

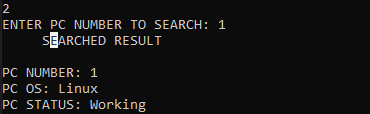


Fig 3: Searching Using PC Number

4) **Updating exiting Pc:** If user try to register using exiting pc number, then it would show three options to the user.

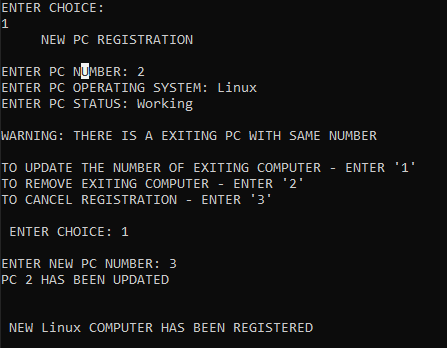


Fig 4: Trying to Add PC using exiting PC number

Same goes for when updating information of an existing pc of the system. It shows cannot update already PC number is assigned.

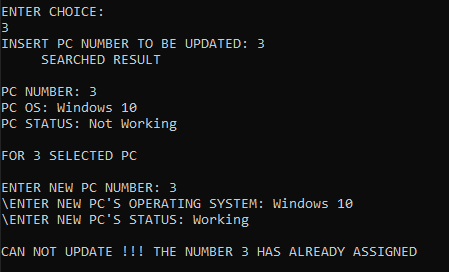


Fig 5: Trying to Update using same PC number

5) **Removing Exiting Pc:** If PC is exits in the system, then we can delete the PC’s information using this functionality.

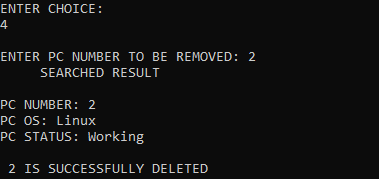


Fig 6: Removing PC from the system

6) **Showing All PC’s Information:** it shows all the information that has been registered on the runtime.

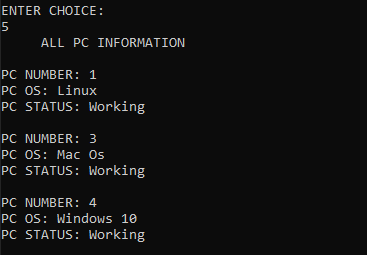


Fig 7: ALL PC’s Information

7) **Storing Information to a file:** This function will save all the information it saved to the lab\_computer object list a .text file.

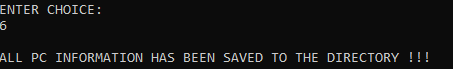


Fig 8: Gives message about storing the data

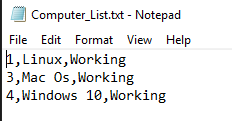


Fig 9: Data are saved in .txt file

8) **Exiting application:** This function terminates the program.

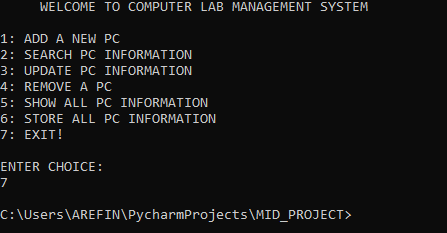


Fig 10: Exiting the application