COURSE DETAILS: Programming Fundamentals-CS111

NAMES: Muhammad Fahad Shabbir & Muhammad Ahmad

REG NO’s: 233017 & 231648

INSTRUCTOR” S NAME: Dr. Ashfaq Hussain Farooqi  
SUBMISSION DATE: 02 Jan 2024

LinkedIn Profile: <https://www.linkedin.com/in/fahad-shabbir-2710522a7/>

<https://www.linkedin.com/in/ahmad-afzal-4b2926280/>

GitHub Profile: <https://github.com/Fshabbir-X>

<https://github.com/AhmadAfzal16>

Description:

* There are 3 projects in this file the first 2 are the file handling and 2D-array’s but we have updated them with structures and the data of structures is stored in 2 separate files such as hardware.txt and workstation.txt.
* The 3rd project is the last one in which we had link between the new entity (Department) and the first 2 project entities. We acquired an entity which is using attributes of the last two entities such ID from Workstation Record Entity and Hardware Name from Purchase Record Entity.
* The concept of modular programming is implemented in the code:

1. Main file. (.cpp which includes menus and function calls)
2. Header file (.h which has all the functions written in detailed body)

The first entity is Purchase Record:

**First Attributes:**

* Hardware Name
* Brand
* Purchase Date
* Price

The second entity is Workstation Record:

**Second Attributes:**

1. ID

2. CPU cores

3. GPU VRAM

4. Power Supply

The third entity is Department Record:

**Third Attributes:**

* ID (previous attribute)
* Hardware Name (previous attribute)
* Department Name
* Manager Name

**FIRST PROJECT CODE**

**The first code consists of following functions**:

1. Add Hardware

2. View Hardware

3. Search Hardware

4. Delete Hardware

5. Update Hardware

6. Exit

**MAIN**

#include <iostream> //Header file to provide various functions related to input/output stream.

#include <fstream> //Header file to include file handling operations in a program.

#include<string> //Header file to include one variable type,one macro and many functions which can be used to mainupulate the string of arrays.

#include<iomanip> //Header file to include part of input/output library of C++ standard library such as setw() etc.

#include"Purchase\_record.h"

#include"Workstation.h"

#include"Department.h"

using namespace std; //Used for using names for objects and variables from standard library.

void add\_department();

void addHardware(); //Declaring function to 'add Hardware' in program as per requirement.

void viewHardware(); //Declaring function to 'view Hardware' in program as per requirement.

void updateHardware(); //Declaring function to 'update Hardware' in program as per requirement.

void deleteHardware(); //Declaring function to 'delete Hardware' in program as per requirement.

void searchHardware(); //Declaring function to 'search Hardware' in program as per requirement.

void menu(); //Declaring function to 'add menu' in program as per requirement.

void addWorkstation(); //Declaring function to 'add WorkStation' in program as per requirement.

void viewWorkstation(); //Declaring function to 'view Workstation' in program as per requirement.

void searchWorkstation(); //Declaring function to 'search Workstation' in program as per requirement.

void updateWorkstation(); //Declaring function to 'update Workstation' in program as per requirement.

void deleteWorkstation(); //Declaring function to 'delete Workstation' in program as per requirement.

void add\_department();

void view\_department();

int main() //Declaration of main function that should return an integer value.

{

while (true) //Application of 'while' loop and test condition checks whether it's (true) or not.

{

menu(); //Only displays the menu if the test condition is true.

}

}

//Calling Menu function, As declared at the top.

void menu() //Calling Menu function, As declared at the top.

{

cout << "COMPUTER HARDWARE MANAGEMENT SYSTEM \n WELCOME TO THE MAIN MENU\n"; //Ouputs the Main Title of the program.

cout << " \*\n"; //Spacing.

cout << " \*\n"; //Spacing.

int choice;

cout << "Please Select Your Desired Category\n \*\n1:Component Purchase Record\n2:WorkStation Record\n3:Department Record\n4.EXIT\n" << endl;

cin >> choice;

switch (choice)

{

case 1:

cout << " COMPONENETS MENU\n"; //Displays title MENU.

cout << " 1.VIEW Hardware\n 2.ADD Hardware\n 3.UPDATE Hardware\n 4.DELETE Hardware\n 5.SEARCH Hardware\n 6.EXIT\n"; //Displays options to be furthur determined.

int option; //Declaring integer (option) to select out of given structure.

cin >> option; //Taking option input by user.

switch (option) //Applying Decision Structure to check for the selected option by the user.

{

case 1: //if option no 1 is selected.

cout << "calling VIEW hardware function!\n"; //displays the following line to user.

viewHardware(); //calling the required function.

break; //ends if it's selected, dosen't display other statments directed by options.

case 2: //if option no 2 is selected.

cout << "calling ADD hardware function!\n"; //displays the following line to user.

addHardware(); //calling the required function.

break; //ends if it's selected, dosen't display other statments directed by options.

case 3: //if option no 3 is selected.

cout << "calling UPDATE hardware function!\n"; //displays the following line to user.

updateHardware(); //calling the required function.

break; //ends if it's selected, dosen't display other statments directed by options.

case 4: //if option no 4 is selected.

cout << "calling DELETE hardware function!\n"; //displays the following line to user.

deleteHardware(); //calling the required function.

break; //ends if it's selected, dosen't display other statments directed by options.

case 5: //if option no 5 is selected.

cout << "calling SEARCH hardware function!\n"; //displays the following line to user.

searchHardware(); //calling the required function.

break; //ends if it's selected, dosen't display other statments directed by options.

case 6: //if option no 6 is selected.

cout << "COMPONENTS\nPROGRAM EXITS HERE! AS PER YOUR REQUEST.\nThank You!\n"; //displays the following line to user.

exit(0); //exits the program if user selects this option.

default: //if anything other than given option is pressed so.

cout << "Invalid Option Selected!\n "; //following displays on screen.

break; //loop terminates infinitely.

}

break; //ends if it's selected, dosen't display other statments directed by Option.

case 2:

cout << " WORKSTATION MENU\n"; //Displays title MENU.

cout << " 1.VIEW WorkStation\n 2.ADD WorkStation\n 3.UPDATE WorkStation\n 4.SEARCH WorkStation\n 5.DELETE WorkStation\n 6.EXIT\n";

int WorkStation; //Declaring integer (WorkStation) to select out of given structure.

cin >> WorkStation; //Taking option input by user.

switch (WorkStation) //Applying Decision Structure to check for the selected option by the user.

{

case 1: //if option no 1 is selected.

cout << "calling VIEW WorkStation function!\n"; //displays the following line to user.

viewWorkstation(); //calling the required function.

break; //ends if it's selected, dosen't display other statments directed by WorkStation.

case 2: //if option no 2 is selected.

cout << "calling ADD WorkStation function!\n"; //displays the following line to user.

addWorkstation(); //calling the required function.

break; //ends if it's selected, dosen't display other statments directed by WorkStation.

case 3: //if option no 3 is selected.

cout << "calling UPDATE WorkStation function!\n"; //displays the following line to user.

updateWorkstation();//calling the required function.

break; //ends if it's selected, dosen't display other statments directed by WorkStation.

case 4: //if option no 4 is selected.

cout << "calling Serach WorkStation function!\n"; //displays the following line to user.

searchWorkstation();//calling the required function.

break; //ends if it's selected, dosen't display other statments directed by WorkStation.

case 5: //if option no 5 is selected.

cout << "calling Delete WorkStation function!\n"; //displays the following line to user.

deleteWorkstation();//calling the required function.

break;

case 6: //if option no 6 is selected.

cout << "WORKSTATION\nPROGRAM EXITS HERE! AS PER YOUR REQUEST.\nThank You!\n"; //displays the following line to user.

exit(0); //exits the program if user selects this option.

break;

default: //if anything other than given option is pressed so.

cout << "Invalid Option Selected!\n "; //following displays on screen.

break; //loop terminates infinitely.

}

break; //ends if it's selected, dosen't display other statments directed by WorkStation.

case 3:

cout << "DEPARTMENT MENU\n";

cout << " 1.ADD DEPARTMENT\n 2.VIEW DEPARTMENT\n 3.EXIT\n";

int Department\_Name;

cin >> Department\_Name;

switch (Department\_Name)

{

case 1:

cout << "Calling ADD Department function!\n";

add\_department();

break;

case 2:

cout << "calling VIEW WorkStation function!\n";

view\_department();

break;

case 3:

cout << "DEPARTMENT\nPROGRAM EXITS HERE! AS PER YOUR REQUEST.\nThank You!\n";

exit(0);

break;

default:

cout << "Invalid Option Selected!\n ";

break;

}

break;

case 4:

cout << "COMPUTER HARWARE MANAGEMENT SYSTEM\nPROGRAM EXITS HERE! AS PER YOUR REQUEST.\nThank You!\n"; //displays the following line to user.

exit(0); //exits the program if user selects this WorkStation.

break;

default: //if anything other than given option is pressed so.

cout << "Invalid Option Selected!\n "; //following displays on screen.

break; //loop terminates infinitely.

}

}

**HEADER FILE**

#pragma once

#include <iostream> //Header file to provide various functions related to input/output stream.

#include <fstream> //Header file to include file handling operations in a program.

#include<string> //Header file to include one variable type,one macro and many functions which can be used to mainupulate the string of arrays.

#include<iomanip> //Header file to include part of input/output library of C++ standard library such as setw() etc.

using namespace std; //Used for using names for objects and variables from standard library.

struct Hardware {

string HARDWARE\_NAME;

string BRAND;

string PURCHASE\_DATE;

int PRICE;

string SEARCH\_HARDWARE;

};

void addHardware() //Calling Add function, As declared at the top.

{

Hardware addHardware;

ofstream fout; //using file handling operations to output the data into a file.

fout.open("hardwarefile.txt", ios::app); //creating a file with the following file name and opening it...appending data to it.

cout << "Enter the Hardware name\n(Enter the first letter in Capital letters):" << endl; //The statement displayed to user on screen.Asked to enter data.

cin >> addHardware.HARDWARE\_NAME; //The data enntered by the user taking input.

cout << "Enter the Brand name:" << endl;

cin >> addHardware.BRAND;

cout << "Enter the purchase date of hardware:" << endl;

cout << "Enter the date in this Format (Days/Months/Years):" << endl;

cin >> addHardware.PURCHASE\_DATE;

cout << "Enter the Price Rs: " << endl;

cin >> addHardware.PRICE;

fout << left << setw(25) << addHardware.HARDWARE\_NAME << setw(15) << addHardware.BRAND << setw(20) << addHardware.PURCHASE\_DATE << setw(10) << addHardware.PRICE << endl; //displaying following headings in output file with spaces using setw().

cout << "The hardware record Has Been Saved To The file Successfully!" << endl; //once the record has been saved, follwoing message is displayed over the screen.

fout.close(); //closing the output file.

}

void searchHardware()

{

Hardware searchHardware;

cout << "Enter the Name of Hardware you want to search\n(Enter the first letter in capital letters):" << endl;

cin >> searchHardware.SEARCH\_HARDWARE;

ifstream fin;

fin.open("hardwarefile.txt");

try

{

if (fin.fail())

{

throw runtime\_error("File is not available");

}

}

catch (runtime\_error& e)

{

cout << "Error occurred\n" << e.what() << endl;

exit(1);

}

bool found = false; // Variable to track if any matching hardware is found

while (fin >> searchHardware.HARDWARE\_NAME >> searchHardware.BRAND >> searchHardware.PURCHASE\_DATE >> searchHardware.PRICE)

{

if (searchHardware.SEARCH\_HARDWARE == searchHardware.HARDWARE\_NAME)

{

found = true;

cout << "This Hardware device is available" << endl;

char choice;

cout << "Enter 'Y' to show the record of this hardware device and 'N' to exit to the main menu:\n";

cin >> choice;

if (choice == 'y' || choice == 'Y')

{

cout << left << setw(25) << "HARDWARE\_NAME" << setw(15) << "BRAND" << setw(20) << "PURCHASE\_DATE" << setw(10) << "PRICE" << endl;

cout << left << setw(25) << searchHardware.HARDWARE\_NAME << setw(15) << searchHardware.BRAND << setw(20) << searchHardware.PURCHASE\_DATE << setw(10) << searchHardware.PRICE << endl;

}

else if (choice == 'n' || choice == 'N')

{

cout << "You have selected 'N' to exit to the main menu!" << endl;

}

}

}

fin.close();

if (!found)

{

cout << "Can't find such Hardware device!" << endl;

}

}

void viewHardware() //Calling View function, As declared at the top.

{

ifstream fin; //using file handling to take input from file, the data already present in the file.

fin.open("hardwarefile.txt"); //opening the file, under the following title.

try //using exceptional handling try block to check:

{

if (fin.fail()) //if the file under the follwing title fails to open/not found...so the error appears.

{

throw runtime\_error("File is not available"); //throws error on screen to show user that the file is not avaliable.

}

}

catch (runtime\_error& e) //catch block retaining runtime errors: such as file not found.

{

cout << "Error occurred\n" << e.what() << endl; //shows error has occured...and used to identify the exception.returns a null terminated character.

exit(1); //interrupted or abnormal termination.

}

Hardware viewHardware;

// declaring integer datatype with variable to store integer value entered by the user.

cout << left << setw(25) << "HARDWARE\_NAME" << setw(15) << "BRAND" << setw(20) << "PURCHASE\_DATE" << setw(10) << "PRICE" << endl; //headings displayed to the user.

while (fin >> viewHardware.HARDWARE\_NAME >> viewHardware.BRAND >> viewHardware.PURCHASE\_DATE >> viewHardware.PRICE) //using while loop,getting input from the file under following variable names,until unless the data is present;checks for it.

{

{

cout << left << setw(25) << viewHardware.HARDWARE\_NAME << setw(15) << viewHardware.BRAND << setw(20) << viewHardware.PURCHASE\_DATE << setw(10) << viewHardware.PRICE << endl; //data from file with spaces using setw() displayed under the assigned headings.

}

}

fin.close(); //closing the input file.

}

void deleteHardware()

{

ifstream fin("hardwarefile.txt");

try

{

if (fin.fail())

{

throw runtime\_error("File is not available");

}

}

catch (runtime\_error& e)

{

cout << "Error occurred\n" << e.what() << endl;

exit(1);

}

ofstream fout;

fout.open("tempfile.txt");

Hardware deleteHardware;

cout << "Enter the name of the hardware you want to delete\n(Enter the first letter in capital letters):" << endl;

cin >> deleteHardware.SEARCH\_HARDWARE;

bool found = false;

while (fin >> deleteHardware.HARDWARE\_NAME >> deleteHardware.BRAND >> deleteHardware.PURCHASE\_DATE >> deleteHardware.PRICE)

{

if (deleteHardware.SEARCH\_HARDWARE == deleteHardware.HARDWARE\_NAME)

{

found = true;

cout << "Hardware record found and deleted Successfully!\n";

}

else

{

fout << left << setw(25) << deleteHardware.HARDWARE\_NAME << setw(15) << deleteHardware.BRAND << setw(20) << deleteHardware.PURCHASE\_DATE << setw(10) << deleteHardware.PRICE << endl;

}

}

if (!found)

{

cout << "Cannot find such hardware device to delete record!\n";

}

fin.close();

fout.close();

remove("hardwarefile.txt");

rename("tempfile.txt", "hardwarefile.txt");

}

void updateHardware() //Calling Update function, As declared at the top.

{

Hardware updateHardware;

ifstream fin("hardwarefile.txt"); //takes input from the file named, using file handling.

try //using exceptional handling try block to check:

{

if (fin.fail()) //if the file under the follwing title fails to open/not found...so the error appears.

{

throw runtime\_error("File is not available"); //throws error on screen to show user that the file is not avaliable.

}

}

catch (runtime\_error& e) //catch block retaining runtime errors: such as file not found.

{

cout << "Error occurred\n" << e.what() << endl; //shows error has occured...and used to identify the exception.returns a null terminated character.

exit(1); //interrupted or abnormal termination.

}

ofstream fout; //using file handling, writing the output to a temporaray file.

fout.open("tempfile.txt"); //output temporaray file created,with the folllowing name and opening it.

cout << "Enter the name of the hardware you want to update\n(Enter the first letter in capital letters):" << endl; //shows on the user screen to enter the name of the hardware to update.

cin >> updateHardware.SEARCH\_HARDWARE; //takes input whatever the user has entered.

while (fin >> updateHardware.HARDWARE\_NAME >> updateHardware.BRAND >> updateHardware.PURCHASE\_DATE >> updateHardware.PRICE) //using while loop,getting input from the file under following variable names,until unless the data is present;checks for it.

{

if (updateHardware.SEARCH\_HARDWARE == updateHardware.HARDWARE\_NAME) //applying control structure to check whether the written data is avaliable in the file or not? if it's present in file then,

{

cout << "Enter new hardware name: "; //asks to fill all the fields once again to update each field in the file.

cin >> updateHardware.HARDWARE\_NAME; //taking all the inputs from the user.

cout << "Enter new Brand: ";

cin >> updateHardware.BRAND;

cout << "Enter new Purchase Date: ";

cin >> updateHardware.PURCHASE\_DATE;

cout << "Enter new Price: ";

cin >> updateHardware.PRICE;

fout << updateHardware.HARDWARE\_NAME << "\t\t" << updateHardware.BRAND << "\t\t" << updateHardware.PURCHASE\_DATE << "\t\t" << updateHardware.PRICE << endl; //outputs(writes) all the data to output file with declared variables. New data that user updates is written to file.

cout << "Hardware record has been updated Successfully!\n"; //after writing successfully displays the following message to user on screen.

}

else //if data is not found inside the file then the else statment executes the block of statements.

{

cout << "Cannot find such hardware device to update record!\n"; //displays that there is no such entry matching in the file to update.

fout << updateHardware.HARDWARE\_NAME << "\t\t" << updateHardware.BRAND << "\t\t" << updateHardware.PURCHASE\_DATE << "\t\t" << updateHardware.PRICE << endl; //hence the old data(not updated) is again written to the file in same manner.

}

}

fin.close(); //closes file from which data was input to significantly update specific data record.

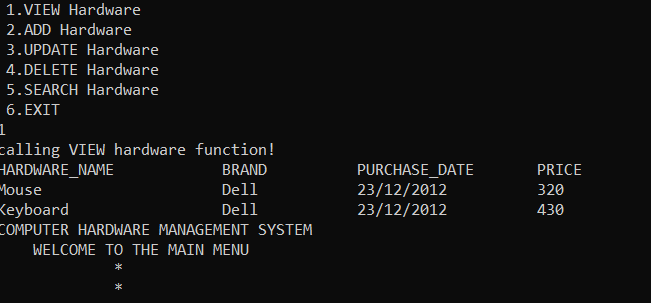
fout.close(); //closes the temporaray file, which has the written(updated) data to it.

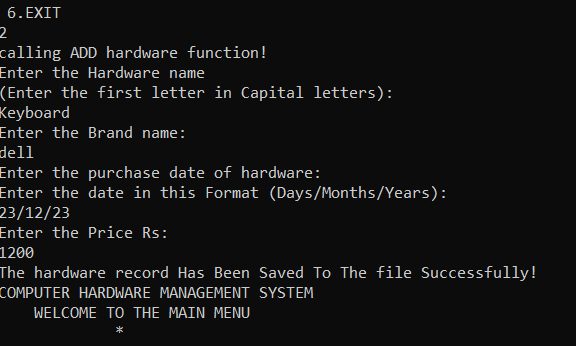
remove("hardwarefile.txt"); //removing the original file having all the data entries(un-updated).

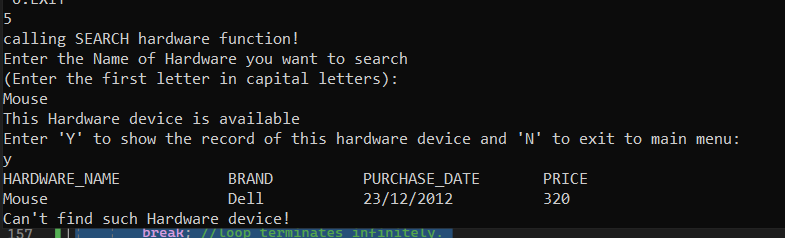
rename("tempfile.txt", "hardwarefile.txt"); //renaming the temporary file to the same named file as before it was originally created so that there is no difference in the file containing data; even after updating records.

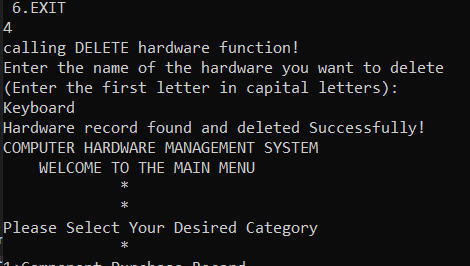
}

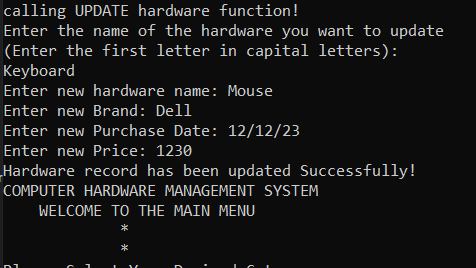
**OUTPUT’s**

****

****

****

****

****

**SECOND PROJECT CODE**

**The Second code consists of following functions**:

1. Add Workstation

2. View Workstation

3. Search Workstation

4. Delete Workstation

5. Update Workstation

6. Exit

**MAIN**

#include <iostream> //Header file to provide various functions related to input/output stream.

#include <fstream> //Header file to include file handling operations in a program.

#include<string> //Header file to include one variable type,one macro and many functions which can be used to mainupulate the string of arrays.

#include<iomanip> //Header file to include part of input/output library of C++ standard library such as setw() etc.

#include"Purchase\_record.h"

#include"Workstation.h"

#include"Department.h"

using namespace std; //Used for using names for objects and variables from standard library.

void add\_department();

void addHardware(); //Declaring function to 'add Hardware' in program as per requirement.

void viewHardware(); //Declaring function to 'view Hardware' in program as per requirement.

void updateHardware(); //Declaring function to 'update Hardware' in program as per requirement.

void deleteHardware(); //Declaring function to 'delete Hardware' in program as per requirement.

void searchHardware(); //Declaring function to 'search Hardware' in program as per requirement.

void menu(); //Declaring function to 'add menu' in program as per requirement.

void addWorkstation(); //Declaring function to 'add WorkStation' in program as per requirement.

void viewWorkstation(); //Declaring function to 'view Workstation' in program as per requirement.

void searchWorkstation(); //Declaring function to 'search Workstation' in program as per requirement.

void updateWorkstation(); //Declaring function to 'update Workstation' in program as per requirement.

void deleteWorkstation(); //Declaring function to 'delete Workstation' in program as per requirement.

void add\_department();

void view\_department();

int main() //Declaration of main function that should return an integer value.

{

while (true) //Application of 'while' loop and test condition checks whether it's (true) or not.

{

menu(); //Only displays the menu if the test condition is true.

}

}

//Calling Menu function, As declared at the top.

void menu() //Calling Menu function, As declared at the top.

{

cout << "COMPUTER HARDWARE MANAGEMENT SYSTEM \n WELCOME TO THE MAIN MENU\n"; //Ouputs the Main Title of the program.

cout << " \*\n"; //Spacing.

cout << " \*\n"; //Spacing.

int choice;

cout << "Please Select Your Desired Category\n \*\n1:Component Purchase Record\n2:WorkStation Record\n3:Department Record\n4.EXIT\n" << endl;

cin >> choice;

switch (choice)

{

case 1:

cout << " COMPONENETS MENU\n"; //Displays title MENU.

cout << " 1.VIEW Hardware\n 2.ADD Hardware\n 3.UPDATE Hardware\n 4.DELETE Hardware\n 5.SEARCH Hardware\n 6.EXIT\n"; //Displays options to be furthur determined.

int option; //Declaring integer (option) to select out of given structure.

cin >> option; //Taking option input by user.

switch (option) //Applying Decision Structure to check for the selected option by the user.

{

case 1: //if option no 1 is selected.

cout << "calling VIEW hardware function!\n"; //displays the following line to user.

viewHardware(); //calling the required function.

break; //ends if it's selected, dosen't display other statments directed by options.

case 2: //if option no 2 is selected.

cout << "calling ADD hardware function!\n"; //displays the following line to user.

addHardware(); //calling the required function.

break; //ends if it's selected, dosen't display other statments directed by options.

case 3: //if option no 3 is selected.

cout << "calling UPDATE hardware function!\n"; //displays the following line to user.

updateHardware(); //calling the required function.

break; //ends if it's selected, dosen't display other statments directed by options.

case 4: //if option no 4 is selected.

cout << "calling DELETE hardware function!\n"; //displays the following line to user.

deleteHardware(); //calling the required function.

break; //ends if it's selected, dosen't display other statments directed by options.

case 5: //if option no 5 is selected.

cout << "calling SEARCH hardware function!\n"; //displays the following line to user.

searchHardware(); //calling the required function.

break; //ends if it's selected, dosen't display other statments directed by options.

case 6: //if option no 6 is selected.

cout << "COMPONENTS\nPROGRAM EXITS HERE! AS PER YOUR REQUEST.\nThank You!\n"; //displays the following line to user.

exit(0); //exits the program if user selects this option.

default: //if anything other than given option is pressed so.

cout << "Invalid Option Selected!\n "; //following displays on screen.

break; //loop terminates infinitely.

}

break; //ends if it's selected, dosen't display other statments directed by Option.

case 2:

cout << " WORKSTATION MENU\n"; //Displays title MENU.

cout << " 1.VIEW WorkStation\n 2.ADD WorkStation\n 3.UPDATE WorkStation\n 4.SEARCH WorkStation\n 5.DELETE WorkStation\n 6.EXIT\n";

int WorkStation; //Declaring integer (WorkStation) to select out of given structure.

cin >> WorkStation; //Taking option input by user.

switch (WorkStation) //Applying Decision Structure to check for the selected option by the user.

{

case 1: //if option no 1 is selected.

cout << "calling VIEW WorkStation function!\n"; //displays the following line to user.

viewWorkstation(); //calling the required function.

break; //ends if it's selected, dosen't display other statments directed by WorkStation.

case 2: //if option no 2 is selected.

cout << "calling ADD WorkStation function!\n"; //displays the following line to user.

addWorkstation(); //calling the required function.

break; //ends if it's selected, dosen't display other statments directed by WorkStation.

case 3: //if option no 3 is selected.

cout << "calling UPDATE WorkStation function!\n"; //displays the following line to user.

updateWorkstation();//calling the required function.

break; //ends if it's selected, dosen't display other statments directed by WorkStation.

case 4: //if option no 4 is selected.

cout << "calling Serach WorkStation function!\n"; //displays the following line to user.

searchWorkstation();//calling the required function.

break; //ends if it's selected, dosen't display other statments directed by WorkStation.

case 5: //if option no 5 is selected.

cout << "calling Delete WorkStation function!\n"; //displays the following line to user.

deleteWorkstation();//calling the required function.

break;

case 6: //if option no 6 is selected.

cout << "WORKSTATION\nPROGRAM EXITS HERE! AS PER YOUR REQUEST.\nThank You!\n"; //displays the following line to user.

exit(0); //exits the program if user selects this option.

break;

default: //if anything other than given option is pressed so.

cout << "Invalid Option Selected!\n "; //following displays on screen.

break; //loop terminates infinitely.

}

break; //ends if it's selected, dosen't display other statments directed by WorkStation.

case 3:

cout << "DEPARTMENT MENU\n";

cout << " 1.ADD DEPARTMENT\n 2.VIEW DEPARTMENT\n 3.EXIT\n";

int Department\_Name;

cin >> Department\_Name;

switch (Department\_Name)

{

case 1:

cout << "Calling ADD Department function!\n";

add\_department();

break;

case 2:

cout << "calling VIEW WorkStation function!\n";

view\_department();

break;

case 3:

cout << "DEPARTMENT\nPROGRAM EXITS HERE! AS PER YOUR REQUEST.\nThank You!\n";

exit(0);

break;

default:

cout << "Invalid Option Selected!\n ";

break;

}

break;

case 4:

cout << "COMPUTER HARWARE MANAGEMENT SYSTEM\nPROGRAM EXITS HERE! AS PER YOUR REQUEST.\nThank You!\n"; //displays the following line to user.

exit(0); //exits the program if user selects this WorkStation.

break;

default: //if anything other than given option is pressed so.

cout << "Invalid Option Selected!\n "; //following displays on screen.

break; //loop terminates infinitely.

}

}

**HEADER FILE**

#pragma once

#include <iostream> //Header file to provide various functions related to input/output stream.

#include <fstream> //Header file to include file handling operations in a program.

#include<string> //Header file to include one variable type,one macro and many functions which can be used to mainupulate the string of arrays.

#include<iomanip> //Header file to include part of input/output library of C++ standard library such as setw() etc.

using namespace std; //Used for using names for objects and variables from standard library.

const int maxworkstation = 5;

struct Workstation {

int ID;

int CPUcores;

int GPUvRAM;

int Powersupply;

}Myworkstation[5];

void addWorkstation() //Calling add function, As declared at the top.

{

Workstation\* MyWorkstation = new Workstation[5];

ofstream fout; //Gives output in the following file.

fout.open("workstation.txt", ios::app); //opening the file and writing data to it. append used to store at the end of last entry.

char choice; //character variable is declared.

for (int j = 0; j <= 5; j++)

{

cout << "Enter WorkStation ID: "; //asks user to enter the new record to store the updated version of all the record attributes present in one row.

cin >> MyWorkstation[j].ID;

cout << "Enter CPU Cores: ";

cin >> MyWorkstation[j].CPUcores;

cout << "Enter GPU VRAM: ";

cin >> MyWorkstation[j].GPUvRAM;

cout << "Enter Power Supply: ";

cin >> MyWorkstation[j].Powersupply;

fout << left << setw(25) << MyWorkstation[j].ID << setw(15) << MyWorkstation[j].CPUcores << setw(20) << MyWorkstation[j].GPUvRAM << setw(10) << MyWorkstation[j].Powersupply << endl; //outputs the new data to the file and saves in it.

cout << "Enter Y if you want to add data again and N if not" << endl;

cin >> choice;

if (choice == 'N' || choice == 'n') {

j = 5;

}

}

fout.close(); //closing the output file.

cout << "The records have been saved to the file" << endl; //displays the follwoing message to the user. after successfully exection.

delete[] MyWorkstation;

}

void viewWorkstation() //Calling view function, As declared at the top.

{

ifstream fin; //takes input from the file.

fin.open("workstation.txt"); //opening and inputing the data from the file.

try //using exceptional handling to increase the efficiency of the code.

{

if (fin.fail())

{

throw runtime\_error("File is not available"); //throws following error on screen in case of failure of not finding a file.

}

}

catch (runtime\_error& e) //catching errors.

{

cout << "File not found!\n"

<< e.what() << endl;

exit(1);

}

int recordcount = 0; //number of records the user wants to save. increasing rows, shifting to the next rows. one by one.

cout << left << setw(25) << "ID" << setw(15) << "CPU\_CORES" << setw(20) << "GPUV\_RAM(IN GBs)" << setw(10) << "POWER\_SUPPLY(IN WATTS)" << endl; // displays following headings with the spacing.

while (recordcount<maxworkstation && fin >> Myworkstation[recordcount].ID >> Myworkstation[recordcount].CPUcores >> Myworkstation[recordcount].GPUvRAM >> Myworkstation[recordcount].Powersupply) //using loop to take input from the file releated to the specific fields with 4 columns.

{

cout << left << setw(25) << Myworkstation[recordcount].ID << setw(15) << Myworkstation[recordcount].CPUcores << setw(20) << Myworkstation[recordcount].GPUvRAM << setw(10) << Myworkstation[recordcount].Powersupply << endl; //displays the record on the screen in the follwoing format.

recordcount++; //adding rows one after another everytime loop executes.

}

fin.close(); //closing the file which was opened before.

}

void searchWorkstation() //Calling search function, As declared at the top.

{

ifstream fin; //takes input from the file.

fin.open("workstation.txt"); //opening and inputing the data from the file.

try //using exceptional handling to increase the efficiency of the code.

{

if (fin.fail())

{

throw runtime\_error("File is not available"); //throws following error on screen in case of failure of not finding a file.

}

}

catch (runtime\_error& e) //catching errors.

{

cout << "File not found!\n"

<< e.what() << endl;

exit(1);

}

int searchID; //declaring variable of int type search id.

bool found = false; //applying bool by making false.

cout << "Enter the WorkStation ID to search for: "; //asks user to enter id to search.

cin >> searchID; //takes input.

int recordcount = 0; //recordcount variable declared to shift to next no of rows. as the user enter the records successfully.

while (recordcount<maxworkstation && fin >> Myworkstation[recordcount].ID >> Myworkstation[recordcount].CPUcores >> Myworkstation[recordcount].GPUvRAM >> Myworkstation[recordcount].Powersupply) //using loop to take input from the file releated to the specific fields with 4 columns.

{

if (Myworkstation[recordcount].ID == searchID) //applying condition to check if the entered id matches the records placed inside the file beforehand.

{

cout << "Workstation found!\n"; //if found displays following message.

cout << left << setw(25) << "ID" << setw(15) << "CPU\_Cores" << setw(20) << "GPUV\_RaM" << setw(10) << "Power\_Supply" << endl; //with details under the headings.

cout << left << setw(25) << Myworkstation[recordcount].ID << setw(15) << Myworkstation[recordcount].CPUcores << setw(20) << Myworkstation[recordcount].GPUvRAM << setw(10) << Myworkstation[recordcount].Powersupply << endl;

found = true; //bool remains true as the record is found.

break;

}

recordcount++; //this add the row and the control shifts to the next row.

}

if (!found) //if record is not found.

{

cout << "Workstation with ID " << searchID << " not found.\n"; //displays the follwoing message.

}

fin.close(); ///closes the file.

}

void updateWorkstation() //Calling update function, As declared at the top.

{

ifstream fin("workstation.txt"); //input the data from the file after opening it.

try //using exceptional handling to increase the efficiency of the code.

{

if (fin.fail())

{

throw runtime\_error("File is not available"); //throws following error on screen in case of failure of not finding a file.

}

}

catch (runtime\_error& e) //catching errors.

{

cout << "File not found!\n"

<< e.what() << endl;

exit(1);

}

ofstream fout("tempfile1.txt"); //gives output of data to a temporary file created.

int recordcount = 0; //declartions of variables as before accordingly.

int updateID;

bool found = false; //applying bool variable which is false.

cout << "Enter the WorkStation ID to update: "; //asks user to enter the ID to update.

cin >> updateID; //takes input from the user.

while (recordcount<maxworkstation && fin >> Myworkstation[recordcount].ID >> Myworkstation[recordcount].CPUcores >> Myworkstation[recordcount].GPUvRAM >> Myworkstation[recordcount].Powersupply) //reads the data accordingly using the variables from the file.

{

if (Myworkstation[recordcount].ID == updateID) //if records in arrays matches the ID's input by the user.

{

found = true; //bool returns true.

cout << "Workstation Record found!\n"; //displays following on the screen.

cout << left << setw(25) << "ID" << setw(15) << "CPU\_Cores" << setw(20) << "GPUV\_RaM" << setw(10) << "Power\_Supply" << endl; //shows the ouput of the attributes headings() in the following order with the spaces.

cout << left << setw(25) << Myworkstation[recordcount].ID << setw(15) << Myworkstation[recordcount].CPUcores << setw(20) << Myworkstation[recordcount].GPUvRAM << setw(10) << Myworkstation[recordcount].Powersupply << endl;

cout << "Enter new WorkStation ID: "; //asks user to enter the new record to store the updated version of all the record attributes present in one row.

cin >> Myworkstation[recordcount].ID;

cout << "Enter new CPU Cores: ";

cin >> Myworkstation[recordcount].CPUcores;

cout << "Enter new GPU VRAM: ";

cin >> Myworkstation[recordcount].GPUvRAM;

cout << "Enter new Power Supply: ";

cin >> Myworkstation[recordcount].Powersupply;

fout << left << setw(25) << Myworkstation[recordcount].ID << setw(15) << Myworkstation[recordcount].CPUcores << setw(20) << Myworkstation[recordcount].GPUvRAM << setw(10) << Myworkstation[recordcount].Powersupply << endl; //outputs the new data to the file and saves in it.

cout << "Updated Record Saved Successfully to the file\n"; //displays the following message to the user on the screen.

}

else //otherwise if it dosen't match then,

{

fout << left << setw(25) << Myworkstation[recordcount].ID << setw(15) << Myworkstation[recordcount].CPUcores << setw(20) << Myworkstation[recordcount].GPUvRAM << setw(10) << Myworkstation[recordcount].Powersupply << endl;

}

recordcount++; //shifts to the next row everytime adding the rows as the user enters the records.

}

if (!found) //if the records is not found.

{

cout << "Cannot find such Workstation ID to update record!\n"; //displays the following message on screen.

}

fin.close(); //closes the input data file.

fout.close(); //closes the output data file.

remove("workstation.txt"); //removes the original file with the data stored un-updated.

rename("tempfile1.txt", "workstation.txt"); //renames the tempfile having all the records unupdated and updated all together....back to the same title as before. there's no purpose for tempfile rather than saving data accumulated after updating.

}

void deleteWorkstation() //Calling delete function, As declared at the top.

{

ifstream fin("workstation.txt"); //input the data from the file after opening it.

try //using exceptional handling to increase the efficiency of the code.

{

if (fin.fail())

{

throw runtime\_error("File is not available"); //throws following error on screen in case of failure of not finding a file.

}

}

catch (runtime\_error& e) //catching errors.

{

cout << "File not found!\n"

<< e.what() << endl;

exit(1);

}

ofstream fout("tempfile.txt"); //gives output of data to a temporary file created.

bool found = false; //applying bool variable which is false.

int recordcount = 0; //declartions of variables as before accordingly.

int deleteID;

cout << "Enter the WorkStation ID to delete: "; //asks user to enter the ID to delete.

cin >> deleteID; //user inputs the ID and record related to that to delete.

while (recordcount<maxworkstation && fin >> Myworkstation[recordcount].ID >> Myworkstation[recordcount].CPUcores >> Myworkstation[recordcount].GPUvRAM >> Myworkstation[recordcount].Powersupply) //reads the data accordingly using the variables from the file.

{

if (Myworkstation[recordcount].ID == deleteID) //if records in arrays matches the ID's input by the user.

{

found = true; //bool returns true.

cout << "Workstation ID Record found and deleted!\n"; //deletes the record and displays the following messsage.

}

else

{

fout << left << setw(25) << Myworkstation[recordcount].ID << setw(15) << Myworkstation[recordcount].CPUcores << setw(20) << Myworkstation[recordcount].GPUvRAM << setw(10) << Myworkstation[recordcount].Powersupply << endl; // otherwise is undeleted and gives the records with undeleted iD record.

}

recordcount++; //this add the row and the control shifts to the next row.

}

if (!found) //if the records is not found.

{

cout << "Cannot find such Workstation ID to update record!\n"; //displays the following message on screen.

}

fin.close(); //closes the input data file.

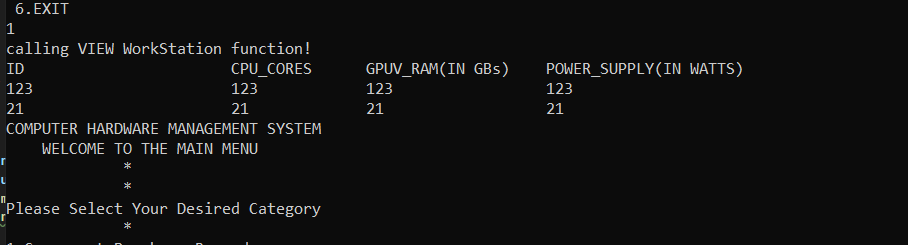
fout.close(); //closes the output data file.

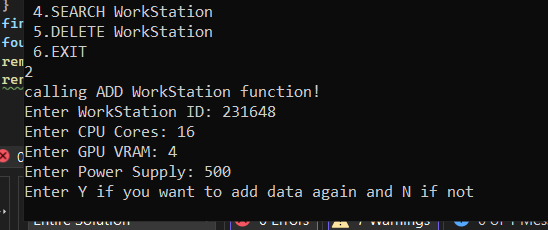
remove("workstation.txt"); //removes the original file with the data stored un-updated.

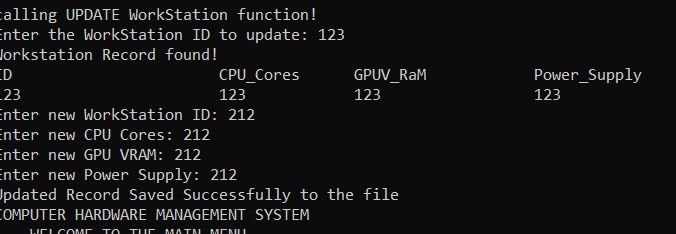
rename("tempfile.txt", "workstation.txt"); //renames the tempfile having all the records unupdated and updated all together....back to the same title as before. there's no purpose for tempfile rather than saving data accumulated after updating.

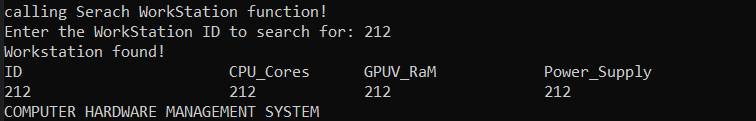
}

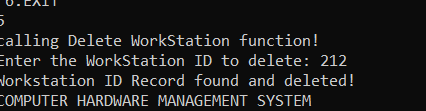
**OUTPUT’s**

****

****

****

****

****

**THIRD PROJECT CODE**

**The Third code consists of following functions**:

1. Add Department

2. View Department

3. Exit

**MAIN**

#include <iostream> //Header file to provide various functions related to input/output stream.

#include <fstream> //Header file to include file handling operations in a program.

#include<string> //Header file to include one variable type,one macro and many functions which can be used to mainupulate the string of arrays.

#include<iomanip> //Header file to include part of input/output library of C++ standard library such as setw() etc.

#include"Purchase\_record.h"

#include"Workstation.h"

#include"Department.h"

using namespace std; //Used for using names for objects and variables from standard library.

void add\_department();

void addHardware(); //Declaring function to 'add Hardware' in program as per requirement.

void viewHardware(); //Declaring function to 'view Hardware' in program as per requirement.

void updateHardware(); //Declaring function to 'update Hardware' in program as per requirement.

void deleteHardware(); //Declaring function to 'delete Hardware' in program as per requirement.

void searchHardware(); //Declaring function to 'search Hardware' in program as per requirement.

void menu(); //Declaring function to 'add menu' in program as per requirement.

void addWorkstation(); //Declaring function to 'add WorkStation' in program as per requirement.

void viewWorkstation(); //Declaring function to 'view Workstation' in program as per requirement.

void searchWorkstation(); //Declaring function to 'search Workstation' in program as per requirement.

void updateWorkstation(); //Declaring function to 'update Workstation' in program as per requirement.

void deleteWorkstation(); //Declaring function to 'delete Workstation' in program as per requirement.

void add\_department();

void view\_department();

int main() //Declaration of main function that should return an integer value.

{

while (true) //Application of 'while' loop and test condition checks whether it's (true) or not.

{

menu(); //Only displays the menu if the test condition is true.

}

}

//Calling Menu function, As declared at the top.

void menu() //Calling Menu function, As declared at the top.

{

cout << "COMPUTER HARDWARE MANAGEMENT SYSTEM \n WELCOME TO THE MAIN MENU\n"; //Ouputs the Main Title of the program.

cout << " \*\n"; //Spacing.

cout << " \*\n"; //Spacing.

int choice;

cout << "Please Select Your Desired Category\n \*\n1:Component Purchase Record\n2:WorkStation Record\n3:Department Record\n4.EXIT\n" << endl;

cin >> choice;

switch (choice)

{

case 1:

cout << " COMPONENETS MENU\n"; //Displays title MENU.

cout << " 1.VIEW Hardware\n 2.ADD Hardware\n 3.UPDATE Hardware\n 4.DELETE Hardware\n 5.SEARCH Hardware\n 6.EXIT\n"; //Displays options to be furthur determined.

int option; //Declaring integer (option) to select out of given structure.

cin >> option; //Taking option input by user.

switch (option) //Applying Decision Structure to check for the selected option by the user.

{

case 1: //if option no 1 is selected.

cout << "calling VIEW hardware function!\n"; //displays the following line to user.

viewHardware(); //calling the required function.

break; //ends if it's selected, dosen't display other statments directed by options.

case 2: //if option no 2 is selected.

cout << "calling ADD hardware function!\n"; //displays the following line to user.

addHardware(); //calling the required function.

break; //ends if it's selected, dosen't display other statments directed by options.

case 3: //if option no 3 is selected.

cout << "calling UPDATE hardware function!\n"; //displays the following line to user.

updateHardware(); //calling the required function.

break; //ends if it's selected, dosen't display other statments directed by options.

case 4: //if option no 4 is selected.

cout << "calling DELETE hardware function!\n"; //displays the following line to user.

deleteHardware(); //calling the required function.

break; //ends if it's selected, dosen't display other statments directed by options.

case 5: //if option no 5 is selected.

cout << "calling SEARCH hardware function!\n"; //displays the following line to user.

searchHardware(); //calling the required function.

break; //ends if it's selected, dosen't display other statments directed by options.

case 6: //if option no 6 is selected.

cout << "COMPONENTS\nPROGRAM EXITS HERE! AS PER YOUR REQUEST.\nThank You!\n"; //displays the following line to user.

exit(0); //exits the program if user selects this option.

default: //if anything other than given option is pressed so.

cout << "Invalid Option Selected!\n "; //following displays on screen.

break; //loop terminates infinitely.

}

break; //ends if it's selected, dosen't display other statments directed by Option.

case 2:

cout << " WORKSTATION MENU\n"; //Displays title MENU.

cout << " 1.VIEW WorkStation\n 2.ADD WorkStation\n 3.UPDATE WorkStation\n 4.SEARCH WorkStation\n 5.DELETE WorkStation\n 6.EXIT\n";

int WorkStation; //Declaring integer (WorkStation) to select out of given structure.

cin >> WorkStation; //Taking option input by user.

switch (WorkStation) //Applying Decision Structure to check for the selected option by the user.

{

case 1: //if option no 1 is selected.

cout << "calling VIEW WorkStation function!\n"; //displays the following line to user.

viewWorkstation(); //calling the required function.

break; //ends if it's selected, dosen't display other statments directed by WorkStation.

case 2: //if option no 2 is selected.

cout << "calling ADD WorkStation function!\n"; //displays the following line to user.

addWorkstation(); //calling the required function.

break; //ends if it's selected, dosen't display other statments directed by WorkStation.

case 3: //if option no 3 is selected.

cout << "calling UPDATE WorkStation function!\n"; //displays the following line to user.

updateWorkstation();//calling the required function.

break; //ends if it's selected, dosen't display other statments directed by WorkStation.

case 4: //if option no 4 is selected.

cout << "calling Serach WorkStation function!\n"; //displays the following line to user.

searchWorkstation();//calling the required function.

break; //ends if it's selected, dosen't display other statments directed by WorkStation.

case 5: //if option no 5 is selected.

cout << "calling Delete WorkStation function!\n"; //displays the following line to user.

deleteWorkstation();//calling the required function.

break;

case 6: //if option no 6 is selected.

cout << "WORKSTATION\nPROGRAM EXITS HERE! AS PER YOUR REQUEST.\nThank You!\n"; //displays the following line to user.

exit(0); //exits the program if user selects this option.

break;

default: //if anything other than given option is pressed so.

cout << "Invalid Option Selected!\n "; //following displays on screen.

break; //loop terminates infinitely.

}

break; //ends if it's selected, dosen't display other statments directed by WorkStation.

case 3:

cout << "DEPARTMENT MENU\n";

cout << " 1.ADD DEPARTMENT\n 2.VIEW DEPARTMENT\n 3.EXIT\n";

int Department\_Name;

cin >> Department\_Name;

switch (Department\_Name)

{

case 1:

cout << "Calling ADD Department function!\n";

add\_department();

break;

case 2:

cout << "calling VIEW WorkStation function!\n";

view\_department();

break;

case 3:

cout << "DEPARTMENT\nPROGRAM EXITS HERE! AS PER YOUR REQUEST.\nThank You!\n";

exit(0);

break;

default:

cout << "Invalid Option Selected!\n ";

break;

}

break;

case 4:

cout << "COMPUTER HARWARE MANAGEMENT SYSTEM\nPROGRAM EXITS HERE! AS PER YOUR REQUEST.\nThank You!\n"; //displays the following line to user.

exit(0); //exits the program if user selects this WorkStation.

break;

default: //if anything other than given option is pressed so.

cout << "Invalid Option Selected!\n "; //following displays on screen.

break; //loop terminates infinitely.

}

}

**HEADER FILE**

#pragma once

#include <iostream> //Header file to provide various functions related to input/output stream.

#include <fstream> //Header file to include file handling operations in a program.

#include<string> //Header file to include one variable type,one macro and many functions which can be used to mainupulate the string of arrays.

#include<iomanip> //Header file to include part of input/output library of C++ standard library such as setw() etc.

using namespace std; //Used for using names for objects and variables from standard library.

struct department {

int Workstation\_ID;

string Hardware\_Name;

string Department\_Name;

string Manager;

}MyDepartment[20];

void add\_department(){

ifstream fin;

fin.open("workstation.txt");

try //using exceptional handling try block to check:

{

if (fin.fail()) //if the file under the follwing title fails to open/not found...so the error appears.

{

throw runtime\_error("File is not available"); //throws error on screen to show user that the file is not avaliable.

}

}

catch (runtime\_error& e) //catch block retaining runtime errors: such as file not found.

{

cout << "Error occurred\n" << e.what() << endl; //shows error has occured...and used to identify the exception.returns a null terminated character.

exit(1); //interrupted or abnormal termination.

}

int recordcount = 0;

int count = 0;

int Search\_ID;

cout << "Enter the Workstation ID you want Added to your desired Department"<<endl;

cin >> Search\_ID;

while (fin >> MyDepartment[recordcount].Workstation\_ID) {

if (Search\_ID == MyDepartment[recordcount].Workstation\_ID) {

fin.close();

ifstream fin; //using file handling to take input from file, the data already present in the file.

fin.open("hardwarefile.txt"); //opening the file, under the following title.

try //using exceptional handling try block to check:

{

if (fin.fail()) //if the file under the follwing title fails to open/not found...so the error appears.

{

throw runtime\_error("File is not available"); //throws error on screen to show user that the file is not avaliable.

}

}

catch (runtime\_error& e) //catch block retaining runtime errors: such as file not found.

{

cout << "Error occurred\n" << e.what() << endl; //shows error has occured...and used to identify the exception.returns a null terminated character.

exit(1); //interrupted or abnormal termination.

}

string Search\_Hardware;

cout << "Enter the name of hardware" << endl;

cin>> Search\_Hardware;

while (fin >> MyDepartment[count].Hardware\_Name) {

if (Search\_Hardware == MyDepartment[count].Hardware\_Name) {

fin.close();

cout << "The workstation with the required hardware is found!" << endl;

cout << "Enter the name of the Department you want your workstation to be added:" << endl;

cin >> MyDepartment[count].Department\_Name;

cout << "Enter the name of the Manager of your Department:" << endl;

cin>>MyDepartment[count].Manager;

ofstream fout;

fout.open("department.txt",ios::app);

fout << MyDepartment[count].Workstation\_ID <<" " << MyDepartment[count].Hardware\_Name <<" " << MyDepartment[count].Department\_Name <<" " << MyDepartment[count].Manager << endl;

}

else {

count++;

}

}

}

else {

recordcount++;

}

}

}

void view\_department() {

ifstream fin;

fin.open("department.txt");

int count = 15;

cout << left << setw(25) <<"Workstation\_ID" << setw(15) << "Hardware\_Name" << setw(20) <<"Department\_Name" << setw(10) << "Manager" << endl;

while (fin >> MyDepartment[count].Workstation\_ID >> MyDepartment[count].Hardware\_Name >>MyDepartment[count].Department\_Name >> MyDepartment[count].Manager) {

cout << left << setw(25) << MyDepartment[count].Workstation\_ID << setw(15) << MyDepartment[count].Hardware\_Name << setw(20) << MyDepartment[count].Department\_Name << setw(10) << MyDepartment[count].Manager << endl;

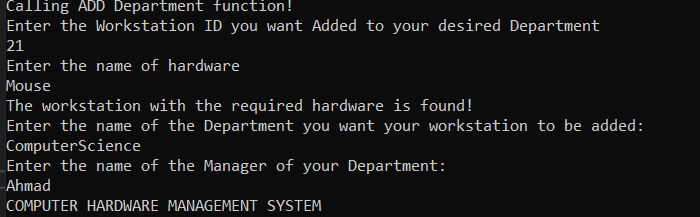
count++;

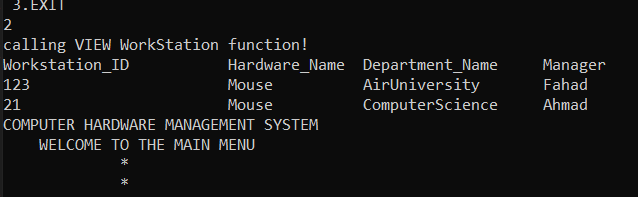
}

fin.close();

}

**OUTPUT’s**

****

****