FEU Institute of Technology
COLLEGE OF ENGINEERING • COLLEGE OF COMPUTER STUDIES

College of Computer Studies

# COMPUTER PROGRAMMING 1

# (CCS0006)

## <Final Project >

| |
|---|
| |
| **GRADE** |


*Submitted by:*
Almario, Gian Lorenzo
Española, Walter Andrew
Fernandez, Elijah Cark
Ligutan, Aldrin Lorenz
Maminta, John Angelo
Yanela, Rye Angelo


*Submitted to:*

**<Julius Claour>**

Professor

November 30, 2023

1.1 Main.cpp

```cpp
#include <iostream>

#include "../include/snippets.h"  //ALWAYS ADD THIS
using namespace std;
#include "../include/systems.h"
int main() {
    bool exited = false;
    while (!exited) {
        system("cls");
        string choice = getStr("Welcome to the main menu. Please select an option:\n[B] Bank Management\n[C] Contact Management\n[R] Student Records\n[Q] Exit\n");
        switch (tolower(choice[0])) {
            case 'b': {
                system("cls");
                startBankManagement();
                break;
            }
            case 'c': {
                system("cls");
                startContactManagement();
                break;
            }
            case 'r': {
                system("cls");
                startStudentRecords();
                break;
            }
            case 'q':
                print("Thank you for using our system. Goodbye!");
                exited = true;
                break;
            default:
                print("Invalid choice. Please try again.");
        }
    }
    return 0;
}
```

1.2 snippets.h

```
#ifndef SNIPPETS_H
#define SNIPPETS_H

// Existing code in snippets.h

#include <conio.h>

#include <fstream>
#include <iostream>
#include <string>
#include <unordered_map>
#include <vector>
#define SEPERATOR "|"  // the seperator used in the csv file to seperate the data

// the hashmap to store the data
std::unordered_map<std::string, std::vector<std::string>> csvData;

// prints in a new line
template <typename T>
void print(T Value) {
    std::cout << Value << std::endl;
}
// Prints in the same line
template <typename U>
void printLn(U s) {
    std::cout << s;
};
// gets a number from the user
double getNum(std::string prompt = "") {
    std::string num;
    char* p;
    do {
        std::cout << prompt;
        std::cin >> num;
        double convertedNum = strtod(num.c_str(), &p);
        if (*p) {
            std::cout << "Invalid input" << std::endl;
        } else {
            std::cin.ignore();
            return convertedNum;
        }
    } while (true);
```

```
    return 0;
}

// gets a string from the user
std::string getStr(std::string prompt = "") {
    std::string s;
    std::cout << prompt;
    getline(std::cin, s);
    return s;
}

// reads the file and returns in string content memory address. returns true if
successful
bool readFile(std::string fileName, std::string& content) {
    std::ifstream file;
    file.open(fileName);
    if (file.is_open()) {
        std::string line;
        while (getline(file, line)) {
            content += line + "\n";
        }
        file.close();
        return true;
    } else {
        print("File not found! creating file...");
        std::ofstream file;
        file.open(fileName);
        if (file.is_open()) {
            print("File created successfully!");
            file << "sep=" + std::string(SEPERATOR) + "\n";
            file.close();
            return true;
        } else {
            return false;
        }
        return false;
    }
}

// adds data to the file. returns true if successful
bool appendFile(std::string fileName, std::string content) {
    std::ofstream file;
    // open the file in append mode
    file.open(fileName, std::ios::app);
    if (file.is_open()) {
```

```
        // write the content to the file
        file << content;
        file.close();
        return true;
    } else {
        return false;
    }
}
// splits the string into a vector
void splitData(std::string str, std::string delimiter, std::vector<std::string>&
vec) {
    // split the string into a vector (just like an array but can change size)
    size_t pos = 0;
    std::string token;
    while ((pos = str.find(delimiter)) != std::string::npos) {
        token = str.substr(0, pos);
        vec.push_back(token);
        str.erase(0, pos + delimiter.length());
    }
    vec.push_back(str);
}
// gets the row from the csvData. Args : search value
std::vector<std::string> getRow(const std::string& value) {
    std::vector<std::string> contacts;
    // read csvData and check if the name is in the csvData
    if (csvData.find(value) != csvData.end()) {
        return csvData[value];
    }
    // if not found return empty vector
    return {};
}

// deletes the row. Args : filename, the name of the first column
bool deleteRow(std::string fileName, std::string rowName) {
    std::string contents;
    readFile(fileName, contents);
    std::vector<std::string> data;
    splitData(contents, "\n", data);
    // loop through the data and add the indexes element to the hashmap
    for (int i = 0; i < data.size() - 1; i++) {
        std::vector<std::string> row;
        splitData(data[i], SEPERATOR, row);
        if (row[0] == rowName) {
            data.erase(data.begin() + i);
            std::string newContent;
```

```
            for (int j = 0; j < data.size() - 1; j++) {
                newContent += data[j] + "\n";
            }
            std::ofstream file;
            file.open(fileName);
            if (file.is_open()) {
                file << newContent;
                file.close();
                return true;
            } else {
                return false;
            }
        }
    }
    return false;
}
// updates the row. Args : filename, the name of the first column, the new value,
the index to update
bool updateRow(std::string fileName, std::string colName, std::string newValue,
int indexToUpdate) {
    std::string contents;
    readFile(fileName, contents);
    std::vector<std::string> data;
    splitData(contents, "\n", data);
    // loop through the data and add the indexes element to row vector
    for (int i = 0; i < data.size() - 1; i++) {
        std::vector<std::string> row;
        splitData(data[i], SEPERATOR, row);
        if (row[0] != colName) continue;
        row[indexToUpdate] = newValue;
        data[i] = "";
        for (int j = 0; j < row.size() - 1; j++) data[i] += row[j] + SEPERATOR;
        data[i] += row[row.size() - 1];
        std::string newContent;
        for (int j = 0; j < data.size() - 1; j++) newContent += data[j] + "\n";
        // write the new content to the file
        std::ofstream file;
        file.open(fileName);
        if (!file.is_open()) return false;
        i == 0 && file << "sep=" + std::string(SEPERATOR) + "\n";
        file << newContent;
        file.close();
        return true;
    }
    return true;
```

```
}
// pauses the program and waits for the user to press a key before continuing
void pauseProgram() {
    print("press any key to continue...");
    getch();
}

// initializes the csvData. Args : filename, the hashmap to store the data, the
indexes used for searching
void init(std::string content, std::unordered_map<std::string,
std::vector<std::string>>& csvData, const std::vector<int>& indexes) {
    std::string contents;
    print("Initializing...");
    readFile(content, contents)
        ? print(content + " read successfully.")
        : print(content + " read failed.");
    std::vector<std::string> data;
    std::vector<std::string> fields;
    splitData(contents, "\n", data);
    // checks if the first line is sep=SEPERATOR
    if (data[0].substr(0, 4) == "sep=") {
        data.erase(data.begin());
    }
    // loop through the data and add the indexes element to the hashmap
    for (int i = 0; i < data.size(); i++) {
        std::vector<std::string> row;
        if (data[i].empty()) continue;
        splitData(data[i], SEPERATOR, row);
        for (int j = 0; j < indexes.size(); ++j) {
            csvData[row[indexes[j]]] = row;
        }
    }
    print("Initialization complete.");
}
#endif
```

1.3 systems.h

```
#include "../src/bankManagement.cpp"
#include "../src/contactManagement.cpp"
#include "../src/studentRecords.cpp"
#ifndef FIRST_H
#define FIRST_H
```

```
int startBankManagement();
int startContactManagement();
int startStudentRecords();

#endif
```

1.4 bankManagement.cpp

```cpp
#include <map>
#include <string>

#include "../include/snippets.h"  //ALWAYS ADD THIS
using namespace std;
const string bankFileName = "accounts.csv";
struct Account {
    string name, address, accountType;
    double amount;
};

// Function to show account details
void showAccountDetails(vector<string> accounts) {
    print("Name: " + accounts[0]);
    print("Address: " + accounts[1]);
    print("Account Type: " + accounts[2]);
    print("Balance: $" + accounts[3]);
}

// Function to deposit funds
void deposit(vector<string> account, double amount) {
    if (amount > 0) {
        double balance = stoi(account[3]);
        balance += amount;
        print("Deposit successful. New balance: $" + to_string(balance));
        updateRow(bankFileName, account[0], to_string(balance), 3);
        csvData[account[0]][3] = to_string(balance);
    } else
        print("Invalid deposit amount. Please enter a positive amount.");
}

// Function to withdraw funds
void withdraw(vector<string> acc, double amount) {
    double balance = stoi(acc[3]);
    if (amount > 0 && amount <= balance) {
        balance -= amount;
        print("Withdrawal successful. New balance: $" + to_string(balance));
        updateRow(bankFileName, acc[0], to_string(balance), 3);
        csvData[acc[0]][3] = to_string(balance);
    } else {
        print("Invalid withdrawal amount or insufficient funds.");
    }
}
```

```
void searchAccount(string name) {
    vector<string> account = getRow(name);
    if (account.size() == 0)
        print("Account not found.");
    else {
        print("Account found!");
        print("Name: " + account[0]);
        print("Address: " + account[1]);
        print("Account Type: " + account[2]);
        print("Balance: $" + account[3]);
    }
}
string createAccount() {
    Account newAccount;
    bool isSuccess;
    print("Great! Let's get started with your account creation process.");
    newAccount.name = getStr("Please enter your name: \n");
    newAccount.address = getStr("Enter your address: \n");
    newAccount.accountType = getStr("Enter your account type: \n");
    newAccount.amount = getNum("Enter amount: \n");
    string content = newAccount.name + SEPERATOR + newAccount.address + SEPERATOR
+ newAccount.accountType + SEPERATOR + to_string(newAccount.amount);
    isSuccess = appendFile(bankFileName, content + "\n");
    if (isSuccess) {
        csvData[newAccount.name] = {newAccount.name, newAccount.address,
newAccount.accountType, to_string(newAccount.amount)};
        print("Account created successfully!");
        return "";
    }
    return newAccount.name;
}
int startBankManagement() {
    vector<int> indexes = {0};
    init(bankFileName, csvData, indexes);
    string answer;
    bool exited = false;
    vector<string> account;
    print("------------------------------------------------------");
    print("||Welcome to Gian's & Elijah's Bank Management System!||");
    print("------------------------------------------------------");
    answer = getStr("Would you like to create an account? (yes/no)");
    if (answer == "no") {
        print("No problem. Let us know if you changed your mind.");
        while (!exited) {
            print("Choose an option:");
```

```
            print("1. Create an account");
            print("4. Search for an account");
            print("5. Exit");
            int choice = getNum("Enter your choice: ");
            switch (choice) {
                case 1:
                    account = csvData[createAccount()];
                    exited = account.size() > 0;
                    break;
                case 4: {
                    string searchName = getStr("Enter the name to search for: ");
                    searchAccount(searchName);
                    break;
                }
                case 5: {
                    print("Exiting the program. Thank you!");
                    exited = true;
                    break;
                }
                default: {
                    print("Invalid choice. Please try again.");
                }
            }
        }
    } else {
        createAccount();
        exited = false;
    }
    while (!exited) {
        print("Choose an option:");
        print("1. Show account details");
        print("2. Deposit");
        print("3. Withdraw");
        print("4. Search for an account");
        print("5. Exit");
        int choice = getNum("Enter your choice: ");
        switch (choice) {
            case 1:
                showAccountDetails(account);
                break;
            case 2: {
                double depositAmount = getNum("Enter the deposit amount: $");
                deposit(account, depositAmount);
                break;
            }
```

```
            case 3: {
                double withdrawAmount = getNum("Enter the withdrawal amount: $");
                withdraw(account, withdrawAmount);
                break;
            }
            case 4: {
                string searchName = getStr("Enter the name to search for: ");
                vector<string> account = getRow(searchName);
                if (account.size() == 0)
                    print("Account not found.");
                else
                    showAccountDetails(account);
                break;
            }
            case 5: {
                print("Exiting the program. Thank you!");
                exited = true;
                break;
            }
            default:
                printLn("Invalid choice. Please try again.");
        }
    }

    return 0;
}
```

1.5 contactManagement.cpp

```
#include "../include/snippets.h"
using namespace std;
#include <conio.h>

#include <iostream>
#include <vector>
void printIntro() {
    print("-----------------------------------------------");
    print("|| Welcome to the Contact Management System ||");
    print("-----------------------------------------------");
    pauseProgram();
}
void addContact() {
    system("cls");
    print("Add a contact");
    print("------------");
    string name = getStr("Enter name: ");
    string phone = getStr("Enter phone number: ");
    while (phone.length() != 11) {
        print("Invalid phone number!");
        phone = getStr("Enter phone number: ");
    }
    string email = getStr("Enter email: ");
    while (email.find("@") == string::npos) {
        print("Invalid email!");
        email = getStr("Enter email: ");
    }
    string address = getStr("Enter current address: ");
    string contact = name + SEPERATOR + phone + SEPERATOR + email + SEPERATOR +
address + "\n";
    bool isSuccess = appendFile("contacts.csv", contact);
    if (isSuccess) {
        print("Contact added successfully!");
        vector<string> row = {name, phone, email, address};
        csvData[name] = row;
        csvData[email] = row;
    } else
        print("Failed to add contact!");
    pauseProgram();
}
void searchContact() {
    system("cls");
    string name = getStr("Enter name or email: ");
```

```
    vector<string> result = getRow(name);
    if (result.empty())
        print("Contact not found.");
    else
        print("Name: " + result[0] + "\nPhone: " + result[1] + "\nEmail: " +
result[2] + "\nAddress: " + result[3]);
    pauseProgram();
}
void deleteContact() {
    system("cls");
    print("Delete a contact");
    print("------------");
    string name = getStr("Enter name or email: ");
    vector<string> result = getRow(name);
    deleteRow("contacts.csv", result[0]);
    print("Contact deleted successfully!");
    pauseProgram();
}
bool updateContactProcess(string noun, vector<string> result, int index) {
    string newValue = getStr("Enter new" + noun + ":");
    bool isSuccess = updateRow("contacts.csv", result[0], newValue, index);
    result[index] = isSuccess ? newValue : result[index];
    csvData[result[0]] = result;
    return isSuccess;
}
void updateContact() {
    system("cls");
    bool isSuccess = false;
    string newValue;
    print("Update a contact");
    print("------------");
    string name = getStr("Enter the name you wish to update: ");
    vector<string> result = getRow(name);
    if (result.empty()) {
        print("Contact not found.");
        pauseProgram();
        return;
    }
    while (!isSuccess) {
    char choice = getStr("Select to update: \n [N] Update name\n [P] Update phone
number\n [E] Update email\n [A] Update address\n [B] Go back \nYour choice:")[0];
        switch (tolower(choice)) {
            case 'n':
                isSuccess = updateContactProcess("name", result, 0);
                break;
```

```
            case 'p':
                isSuccess = updateContactProcess("phone number", result, 1);
                break;
            case 'e':
                isSuccess = updateContactProcess("email", result, 2);
                break;
            case 'a':
                isSuccess = updateContactProcess("address", result, 3);
                break;
            case 'b':
                isSuccess = true;
                break;
            default:
                print("Invalid choice!");
                break;
        }
    }
    if (isSuccess)
        print("Contact updated successfully!");
    else
        print("Failed to update contact!");
    pauseProgram();
}

int startContactManagement() {
    vector<int> indexes = {0, 2};
    init("contacts.csv", csvData, indexes);
    printIntro();
    bool exit = false;
    while (!exit) {
        system("cls");
        string choice = getStr("Select an option: \n[C] Add a contact\n[R] Search
a contact\n[U] Update a contact\n[D] Delete a contact\n[Q] Exit\nEnter your
choice: ");
        switch (tolower(choice[0])) {
            case 'c':
                addContact();
                break;
            case 'r':
                searchContact();
                break;
            case 'u':
                updateContact();
                break;
            case 'd':
```

```
                deleteContact();
                break;
            case 'q':
                exit = true;
                break;
            default:
                print("Invalid choice!");
                break;
        }
    }
    print("Good bye");
    return 0;
}
```

1.6 studentrecords.cpp

```
#include "../include/snippets.h"
using namespace std;
#include <conio.h>

#include <iomanip>
#include <iostream>
#include <vector>
const string studentRecordFile = "studentRecords.csv";
void addrecord() {
    / system("cls");
    string last = getStr("Enter last name: ");
    string first = getStr("Enter first name: ");
    string middle = getStr("Enter middle name: ");
    string rollNumber = getStr("Enter roll number: ");
    while (rollNumber.length() != 9) {
        print("Invalid roll number, try again.");
        rollNumber = getStr("Enter roll number: ");
    }
    string contact = getStr("Enter contact number: ");
    while (contact.length() != 11 && contact.length() != 7) {
        print("Invalid contact number, try again.");
        contact = getStr("Enter contact number: ");
    }
    int year = getNum("Enter year level (number): ");
    string yearLevel;
    switch (year) {
        case 1:
            yearLevel = "1st";
            break;
        case 2:
            yearLevel = "2nd";
            break;
        case 3:
            yearLevel = "3rd";
            break;
        default:
            yearLevel = to_string(year) + "th";
    }
    string course = getStr("Enter course: ");
    string email = getStr("Enter email: ");
    string fullName = last + SEPERATOR + first + SEPERATOR + middle;
    string record = rollNumber + SEPERATOR + fullName + SEPERATOR + contact +
SEPERATOR + yearLevel + SEPERATOR + course + SEPERATOR + email + "\n";
```

```
    bool isSuccess = appendFile(studentRecordFile, record);
    if (isSuccess) {
        print("Contact added successfully!");
        vector<string> row = {rollNumber, last, first, middle, contact,
yearLevel, course, email};
        csvData[rollNumber] = row;
        csvData[last] = row;
    } else
        print("Failed to add contact!");
    pauseProgram();
}

void deleterecord() {
    // system("cls");
    print("--------------------------------");
    print("DELETE A RECORD");
    print("--------------------------------");
    print("[1] - Roll number");
    print("[2] - Last name");
    print("--------------------------------");
    string choice = getStr("Enter your choice: ");
    switch (choice[0]) {
        case '1': {
            string RN = getStr("Enter roll number: ");
            vector<string> result = getRow(RN);
            deleteRow(studentRecordFile, result[0]);
            print("Record deleted successfully!");
            break;
        }
        case '2': {
            string last = getStr("Enter last name: ");
            vector<string> result = getRow(last);
            deleteRow(studentRecordFile, result[1]);
            print("Contact deleted successfully!");
            break;
        }
        default: {
            print("Invalid Choice");
        }
    }
    pauseProgram();
}

bool updateProcess(string noun, vector<string> result, int index) {
    string newValue = getStr("Enter new" + noun + ":");
```

```
    bool isSuccess = updateRow(studentRecordFile, result[0], newValue, index);
    result[index] = isSuccess ? newValue : result[index];
    csvData[result[0]] = result;
    csvData[result[1]] = result;

    return isSuccess;
}

void modifyrecord() {
    // system("cls");
    bool isSuccess = false;
    string newValue;
    print("--------------------------------");
    print("MODIFY A RECORD");
    print("--------------------------------");
    print("[1] - Roll number");
    print("[2] - Last name");
    print("--------------------------------");
    string choice = getStr("Enter your choice: ");
    switch (choice[0]) {
        case '2':
        case '1': {
            string rollNumber = getStr("Enter record: ");
            vector<string> result = getRow(rollNumber);
            // system("cls");
            if (result.empty()) {
                print("--------------------------------");
                print("Record not found.");
                print("--------------------------------");
                pauseProgram();
                break;
            } else {
                // system("cls");
                print("--------------------------------");
                print("PICK THE SUBJECT YOU WANT TO MODIFY");
                print("--------------------------------");
                print("[1] - Roll Number: " + result[0] + "\n[2] - Last Name: " +
result[1] + "\n[3] - First Name: " + result[2] + "\n[4] - Middle Name: " +
result[3] + "\n[5] - Contact Number: " + result[4] + "\n[6] - Year Level: " +
result[5] + "\n[7] - Course: " + result[6] + "\n[8] - Email: " + result[7]);
                print("--------------------------------");
            }
            string choice = getStr("Enter your choice: ");
            while (!isSuccess) {
                switch (choice[0]) {
```

```
                case '1': {
                    isSuccess = updateProcess(" Roll number", result, 0);
                    break;
                }
                case '2': {
                    isSuccess = updateProcess(" Last name", result, 1);
                    break;
                }
                case '3': {
                    isSuccess = updateProcess(" First name", result, 2);
                    break;
                }
                case '4': {
                    isSuccess = updateProcess(" Middle name", result, 3);
                    break;
                }
                case '5': {
                    isSuccess = updateProcess(" Contact number", result, 4);
                    break;
                }
                case '6': {
                    isSuccess = updateProcess(" Year level", result, 5);
                    break;
                }
                case '7': {
                    isSuccess = updateProcess(" Course", result, 6);
                    break;
                }
                case '8': {
                    isSuccess = updateProcess(" Email", result, 7);
                    break;
                }
                case 'q': {
                    isSuccess = true;
                    break;
                }
                default: {
                    print("Invalid Choice");
                    choice = getStr("Enter your choice: ");
                }
            }
        }
        break;
    }
    default:
```

```
            print("Invalid Choice");
    }
    if (isSuccess)
        print("updated successfully!");
    else
        print("Failed to update contact!");
    pauseProgram();
}

void viewrecord() {
    // system("cls");
    print("--------------------------------");
    print("VIEWING RECORD");
    print("--------------------------------");
    print("[1] - Roll number");
    print("[2] - Last name");
    print("--------------------------------");
    string choice = getStr("Enter your choice: ");
    switch (choice[0]) {
        case '1': {
            string RN = getStr("Enter roll number: ");
            vector<string> result = getRow(RN);
            pauseProgram();
            // system("cls");
            if (result.empty()) {
                print("--------------------------------");
                print("Record not found.");
                print("--------------------------------");
                pauseProgram();
                break;
            } else {
                print("--------------------------------");
                print("Roll Number: " + result[0] + "\nLast Name: " + result[1] +
"\nFirst Name: " + result[2] + "\nMiddle Name: " + result[3] + "\nContact Number:
" + result[4] + "\nYear Level: " + result[5] + "\nCourse: " + result[6] +
"\nEmail: " + result[7]);
                print("--------------------------------");
            }
            break;
        }
        case '2': {
            string last = getStr("Enter last name: ");
            vector<string> result = getRow(last);
            // system("cls");
            pauseProgram();
```

```
            if (result.empty()) {
                print("-------------------------------");
                print("Record not found.");
                print("-------------------------------");
                pauseProgram();
            } else {
                print("-------------------------------");
                print("Roll Number: " + result[0] + "\nLast Name" + result[1] +
"\nFirst Name" + result[2] + "\nMiddle Name" + result[3] + "\nContact Number" +
result[4] + "\nYear Level" + result[5] + "\nCourse" + result[6] + "\nEmail" +
result[7]);
                print("-------------------------------");
            }
            break;
        }
        default: {
            print("Invalid Choice");
        }
    }
    pauseProgram();
}

int startStudentRecords() {
    std::vector<int> indexes = {0, 1};
    init(studentRecordFile, csvData, indexes);
    bool exit = false;
    while (!exit) {
        // system("cls");
        print("-------------------------------");
        print("Student Record Management System");
        print("-------------------------------");
        print("[A] - Add record");
        print("[D] - Delete record");
        print("[M] - Modify record");
        print("[V] - View record");
        print("[Q] - QUIT");
        print("-------------------------------");
        string choice = getStr("Enter your choice: ");
        switch (tolower(choice[0])) {
            case 'a':
                addrecord();
                break;
            case 'd':
                deleterecord();
                break;
```

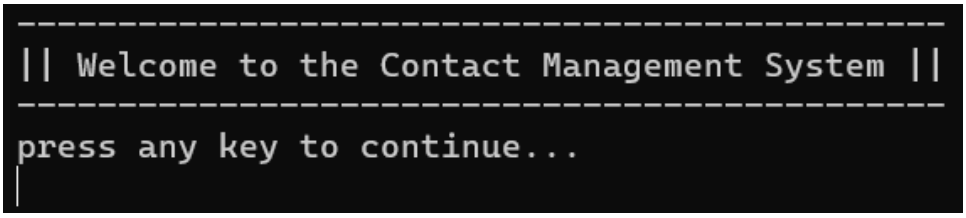
```
            case 'm':
                modifyrecord();
                break;
            case 'v':
                viewrecord();
                break;
            case 'q':
                exit = true;
                print("Exit");
                break;
            default:
                print("Invalid choice");
        }
    }
    return 0;
}
```

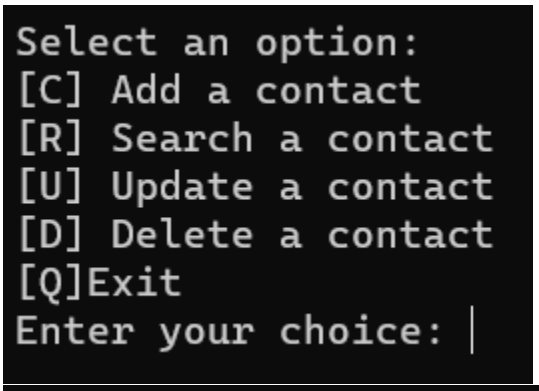
2. Screenshots

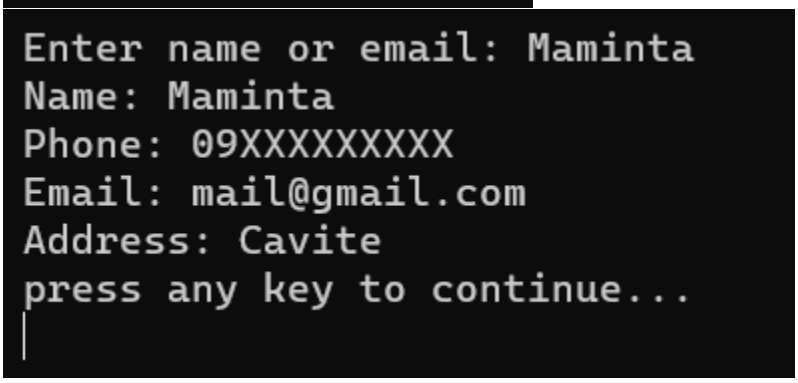
2.1 Bank Management System :

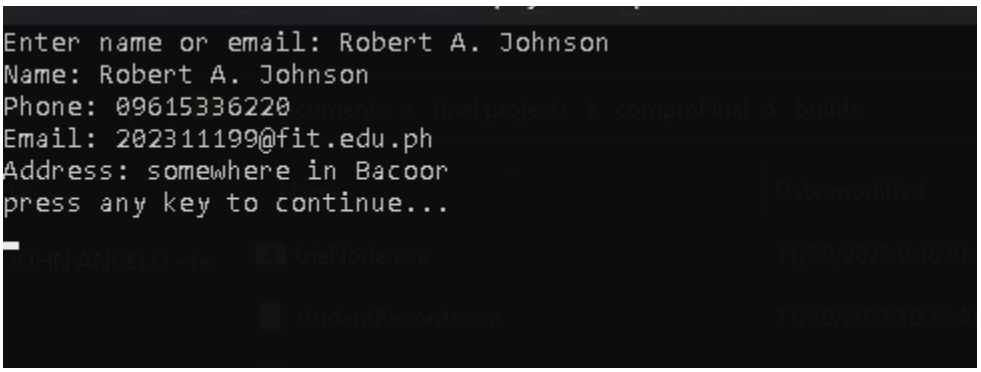
```
------------------------------------------------------
||Welcome to Gian's & Elijah's Bank Management System!||
------------------------------------------------------
Would you like to create an account? (yes/no)no
No problem. Let us know if you changed your mind.
Choose an option:
1. Create an account
4. Search for an account
5. Exit
Enter your choice: 4
Enter the name to search for: Walter
Account found!
Name: Walter
Address: Valenzuela City
Account Type: savings
Balance: $200.000000
Choose an option:
1. Create an account
4. Search for an account
5. Exit
Enter your choice:
```

2.2 Contact Management System



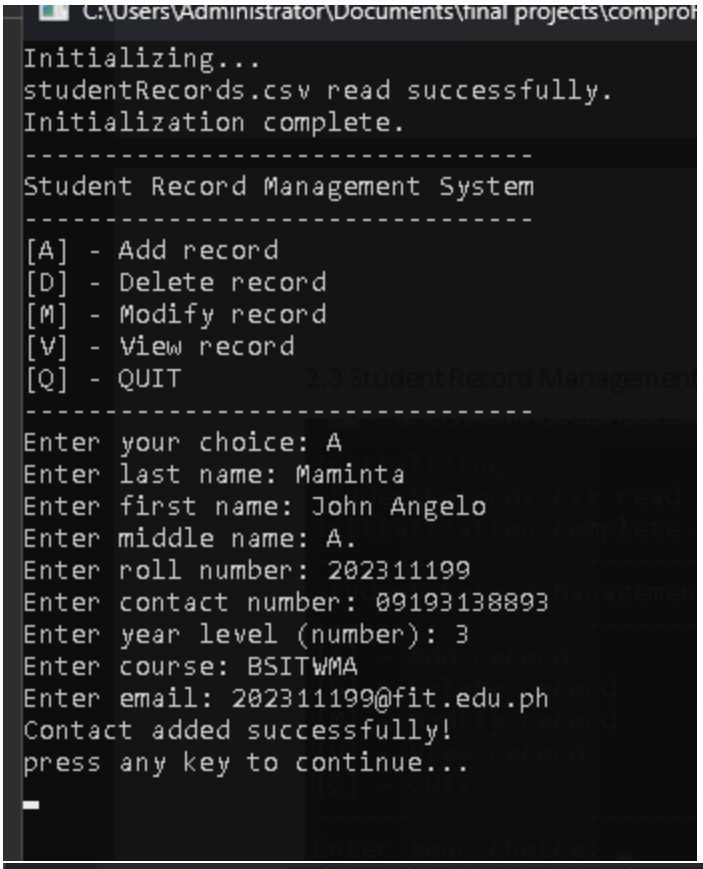



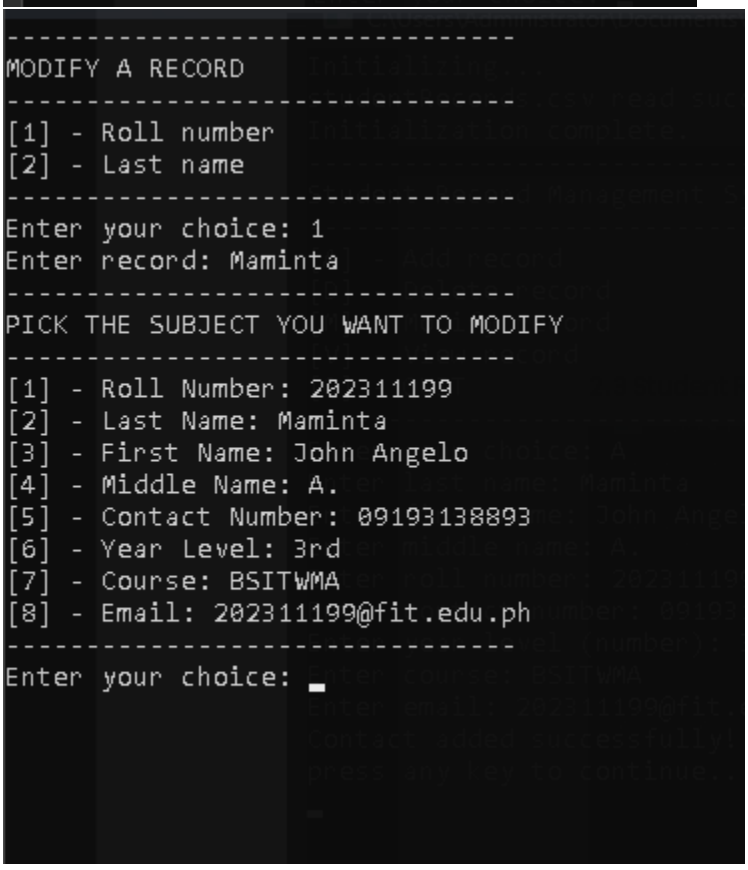
```
Update a contact
------------
Enter the name you wish to update: Robert A. Johnson
Select to update:
 [N] Update name
 [P] Update phone number
 [E] Update email
 [A] Update address
 [B] Go back
Your choice:
```

```
Update a contact
------------
Enter the name you wish to update: Robert A. Johnson
Select to update:
 [N] Update name
 [P] Update phone number
 [E] Update email
 [A] Update address
 [B] Go back
Your choice:A
Enter newaddress:Somewhere in Manila
Contact updated successfully!
press any key to continue...
```

```
C:\Users\Administrator\Documents\final projects\comproFinal\builds\main.exe
Delete a contact
------------
Enter name or email: Robert A. Johnson
Contact deleted successfully!
press any key to continue...
```

2.3 Student Record Management System


C:\Users\Administrator\Documents\final projects\compro
Initializing...
studentRecords.csv read successfully.
Initialization complete.
Student Record Management System
[A] - Add record
[D] - Delete record
[M] - Modify record
[V] - View record
[Q] - QUIT
Enter your choice: A
Enter last name: Maminta
Enter first name: John Angelo
Enter middle name: A.
Enter roll number: 202311199
Enter contact number: 09193138893
Enter year level (number): 3
Enter course: BSITWMA
Enter email: 202311199@fit.edu.ph
Contact added successfully!
press any key to continue...


MODIFY A RECORD
[1] - Roll number
[2] - Last name
Enter your choice: 1
Enter record: Maminta
PICK THE SUBJECT YOU WANT TO MODIFY
[1] - Roll Number: 202311199
[2] - Last Name: Maminta
[3] - First Name: John Angelo
[4] - Middle Name: A.
[5] - Contact Number: 09193138893
[6] - Year Level: 3rd
[7] - Course: BSITWMA
[8] - Email: 202311199@fit.edu.ph
Enter your choice:

```
Enter record: Maminta
--------------------------------
PICK THE SUBJECT YOU WANT TO MODIFY
--------------------------------
[1] - Roll Number: 202311199
[2] - Last Name: Maminta
[3] - First Name: John Angelo
[4] - Middle Name: A.
[5] - Contact Number: 09193138893
[6] - Year Level: 3rd
[7] - Course: BSITWMA
[8] - Email: 202311199@fit.edu.ph
--------------------------------
Enter your choice: 9
Invalid Choice
Enter your choice: 1
Enter your choice: 1
Enter new Roll number:202311192
updated successfully!
press any key to continue...
```