الجامعة الإسلامية العالمية ماليزيا
INTERNATIONAL ISLAMIC UNIVERSITY MALAYSIA
يونيبرسيتي اسلام انتارابڠسا مليسيا

PROGRAMMING PROJECT EECE 1313
(REPORT)

STUDENT FINANCE DATABASE

NO.	NAME	MATRIC NUMBER
1.	AHMAD FUAD MAHDI BIN AHMAD FAIZAL	1916597
2.	MUHAMMAD FAREEZ IQMAL BIN MOHD SHARIPUDDIN	1914577
3.	FARAH BINTI KAMAL	1911974
4.	NURAISYAH BINTI SAMSUL	1915566

LECTURER’S NAME: DR. AFIDALINA BINTI TUMIAN

SECTION: 7

SEMESTER: 1 (2019/2020)

KULIYYAH: ENGINEERING

SUBMISSION DATE: 13 DECEMBER 2019

Introduction

For this project, we choose to create a program for high schools' use. The purpose of this program is to provide users with the ability to add new student, view financial statement, make a payment as well as delete the data of student that leaving the school. By using C++ Program, we are able to provide the users with a unique method to engage with the payment problems.

In brief about our program, AddNew function is called when the users want to add the information of new student. The function will ask for the students' names, matric number and course they are in. The fee that each student needs to pay is fixed based on their course. Next, the second choice in our main page is viewing financial statement, this program will display the list of students' names, their courses and balance of their fees. It will call function ViewStatus which includes two inputs to process this program.

Move on to the next element which is allowing the student to make a payment, this program will call the function named payFee. And it will ask the users for their matric number and the amount of money they want to pay. Then, the fees are deducted after the payment and update it in the system. Next, the program can delete students' data when they are leaving the school. deleteRecord function is called to satisfy this feature. However, before deleting, the program will ask the users to complete their fees. Last but not least, the program will exit if the users choose to not continue this program.

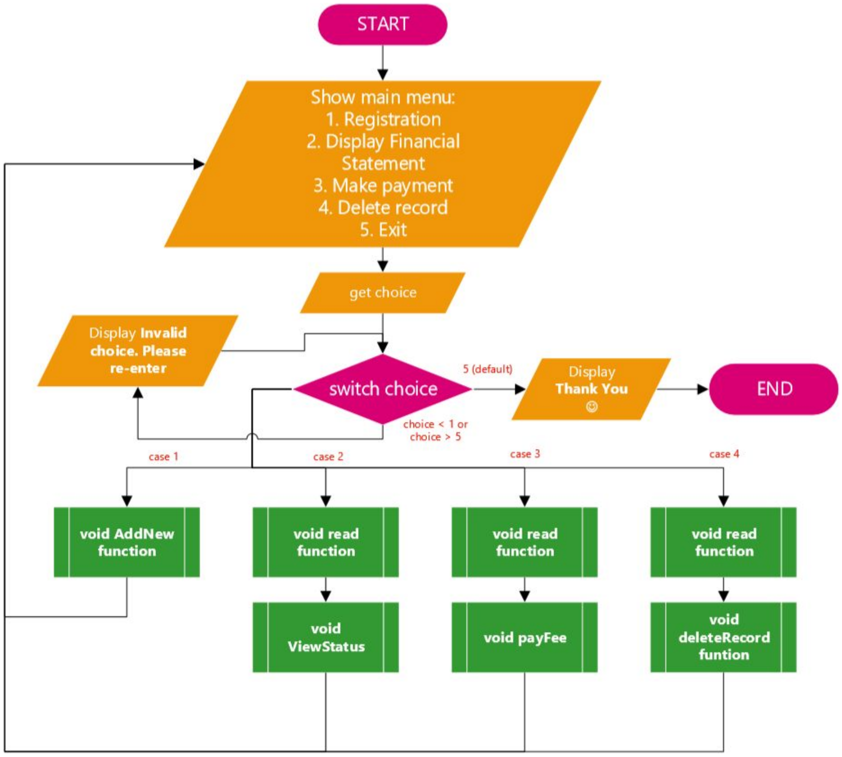
The main goal of this program is to create a database system that would allow the users to store students' information about their finances.

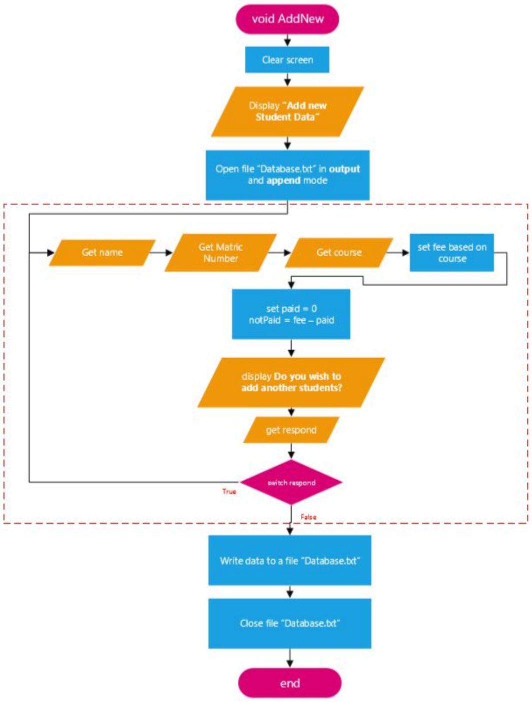
Method

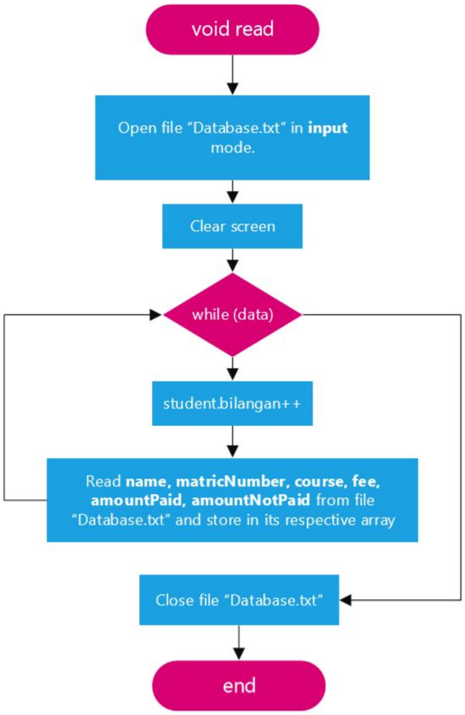
Student finance database is one of the exciting projects that does not require much header which it runs with just four header files which are iostream, string, fstream and iomanip. Each of the header files has its own function. Iostream stands for standard input-output stream that contains definitions to objects like cin and cout while iomanip stands for input-output manipulators that contains definitions of setw, setprecision and many more and the methods we stated in this files are used for manipulating streams. Next, fstream is used to organize the data being written as an output and data being read as an input while string header must be included to be able to stores a sequence of letters or characters.

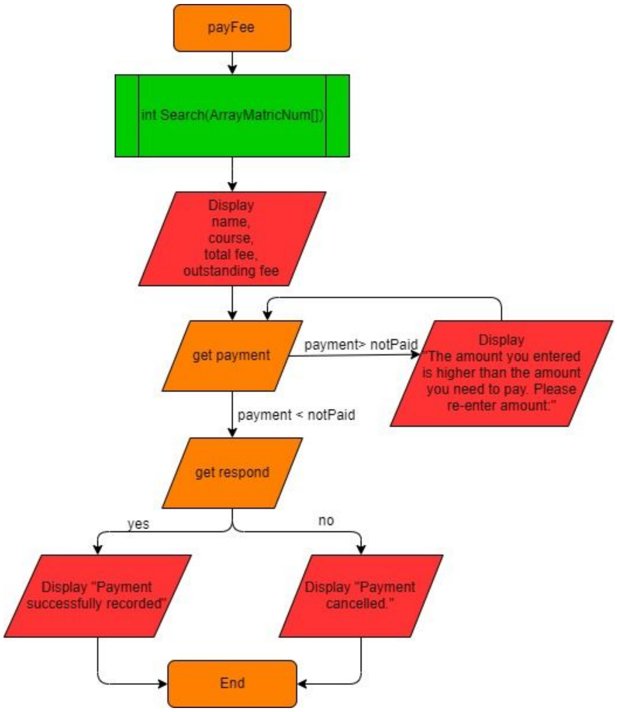
For this project, the switch statement is being used in the main selection structure for the user to make a choice. This program also used several do-while loop to make sure at least one iteration performs even the expression is false to begin with like example in the main function. Next, a global function is used as well as we defined outside all function and call back the function in main function inside the switch statement. System (“cls”) is used to clear screen so that only the remaining output will be displayed. To store data that is written, a file is used to store all the information that has been input by the user. Besides, if/else if statement also being used in this program to make certain types of certain nested decision logic simpler to write.

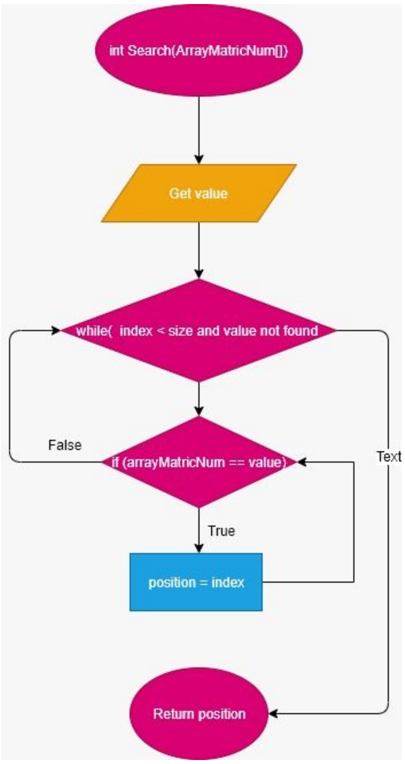
FLOWCHART

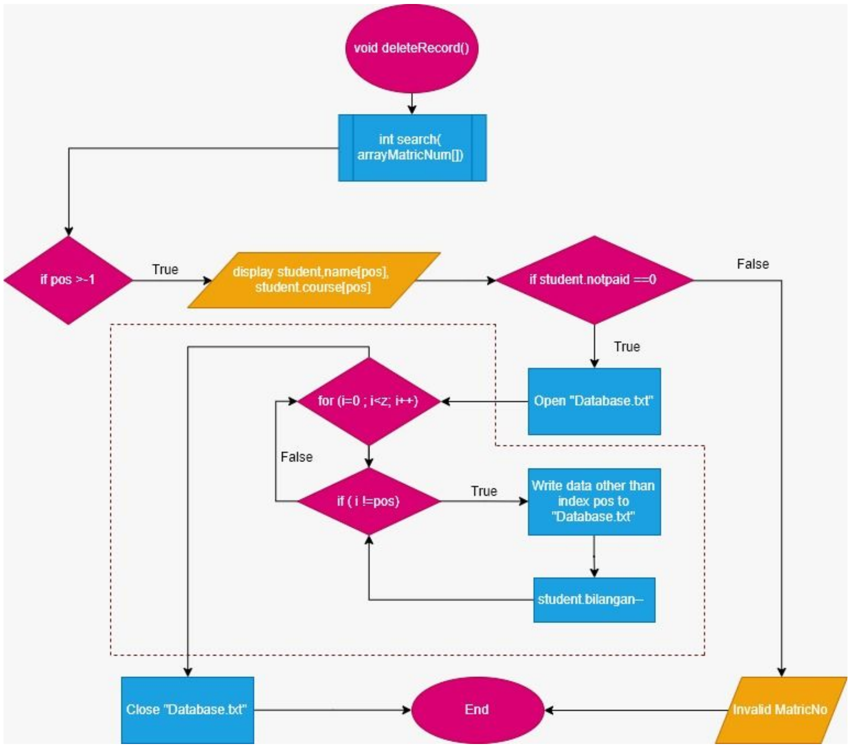

START
Show main menu:
1. Registration
2. Display Financial Statement
3. Make payment
4. Delete record
5. Exit
get choice
Display Invalid choice. Please re-enter
switch choice
5 (default)
Display Thank You ☺
END
choice < 1 or choice > 5
case 1
case 2
case 3
case 4
void AddNew function
void read function
void read function
void read function
void ViewStatus
void payFee
void deleteRecord funtion


void AddNew
Clear screen
Display "Add new Student Data"
Open file "Database.txt" in output and append mode
Get name
Get Matric Number
Get course
set fee based on course
set paid = 0
notPaid = fee – paid
display Do you wish to add another students?
get respond
switch respond
True
False
Write data to a file "Database.txt"
Close file "Database.txt"
end


void read
Open file "Database.txt" in input mode.
Clear screen
while (data)
student.bilangan++
Read name, matricNumber, course, fee, amountPaid, amountNotPaid from file "Database.txt" and store in its respective array
Close file "Database.txt"
end


payFee
int Search(ArrayMatricNum[])
Display name, course, total fee, outstanding fee
get payment
payment> notPaid
Display "The amount you entered is higher than the amount you need to pay. Please re-enter amount:"
payment < notPaid
get respond
yes
no
Display "Payment successfully recorded"
Display "Payment cancelled."
End

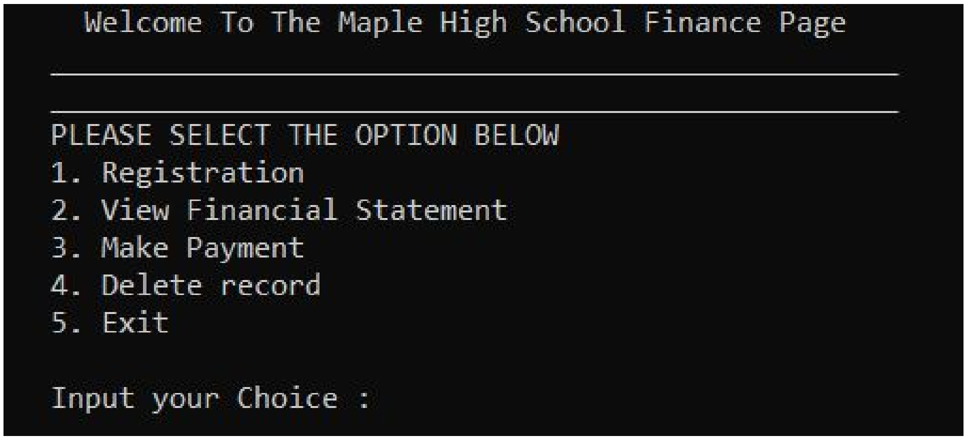

int Search(ArrayMatricNum[])
Get value
while(index < size and value not found
False
if (arrayMatricNum == value)
Text
True
position = index
Return position


void deleteRecord()
int search(arrayMatricNum[])
if pos >-1
True
display student.name[pos], student.course[pos]
if student.notpaid ==0
False
True
Open "Database.txt"
for (i=0 ; i<z; i++)
False
if (i !=pos)
True
Write data other than index pos to "Database.txt"
student.bilangan--
Close "Database.txt"
End
Invalid MatricNo

Result

1. Main Page

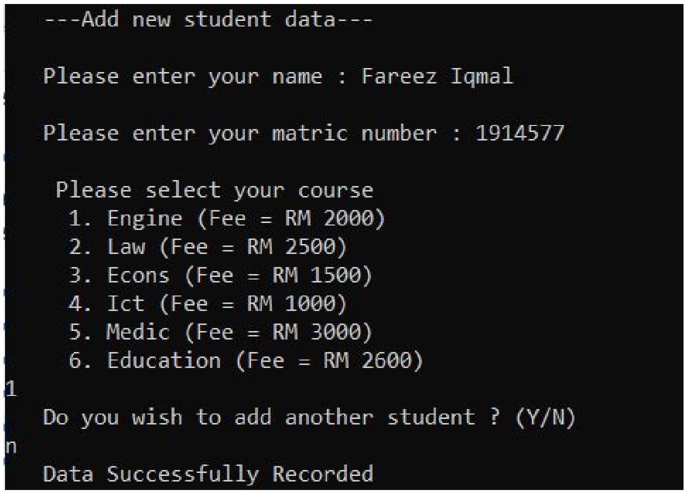
   This feature is the main page of “Maple High School Finance Page”. It displays multiple options such as registration, view financial statement, make payment, delete record and exit. The program will ask the user to input digit from one to five. If the user entered below than one or higher than five, the program will ask to re-enter the number.



```
  Welcome To The Maple High School Finance Page
__________________________________________________
__________________________________________________
PLEASE SELECT THE OPTION BELOW
1. Registration
2. View Financial Statement
3. Make Payment
4. Delete record
5. Exit

Input your Choice :
```

2. Student’s Registration

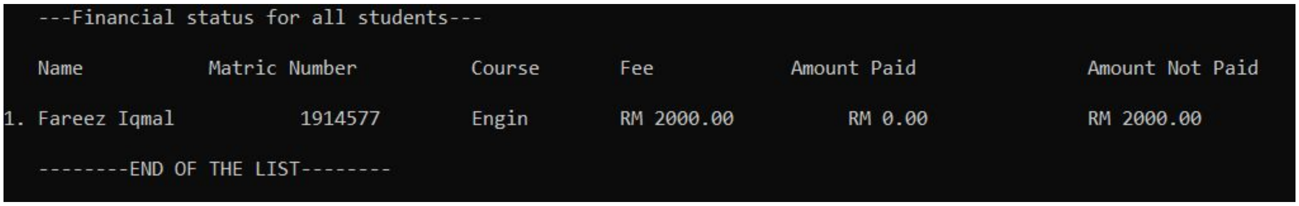
   This feature creates a new students information including their matric number. It also provided with multiple courses such as engineering, law, economics, ICT, medical and education that has to be chosen by the student. The program also asks if the user wants to add another student. Upon successful registration, the program will display “Data Successfully Recorded” and this data will be written in a “Database.txt” file in the background.



```
   ---Add new student data---

   Please enter your name : Fareez Iqmal

   Please enter your matric number : 1914577

    Please select your course
     1. Engine (Fee = RM 2000)
     2. Law (Fee = RM 2500)
     3. Econs (Fee = RM 1500)
     4. Ict (Fee = RM 1000)
     5. Medic (Fee = RM 3000)
     6. Education (Fee = RM 2600)
1
   Do you wish to add another student ? (Y/N)
n
   Data Successfully Recorded
```

3. View Financial Statement

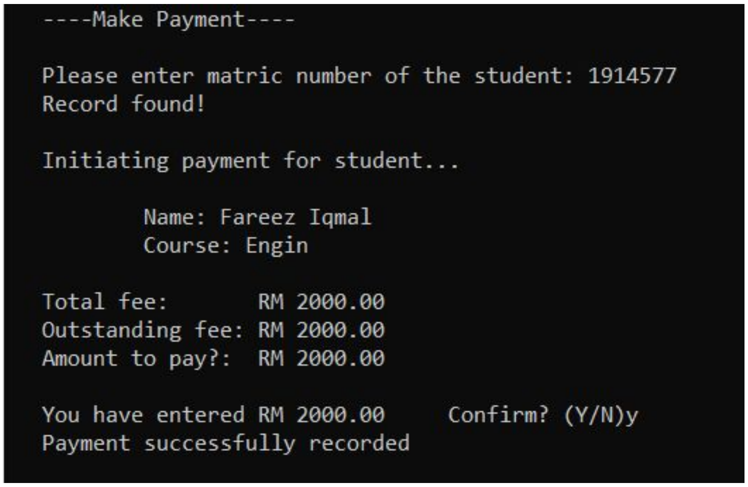
This feature basically shows the progress of the financial statement of all the students added in the system. It displays the name of all students with their matric number and course, the fee that students have to pay and current update of their financial statement whether the students already paid or the remaining fee that the students have to pay.



```
   ---Financial status for all students---

   Name           Matric Number          Course       Fee            Amount Paid               Amount Not Paid

1. Fareez Iqmal           1914577        Engin        RM 2000.00         RM 0.00               RM 2000.00

   --------END OF THE LIST--------
```

4. Make payment

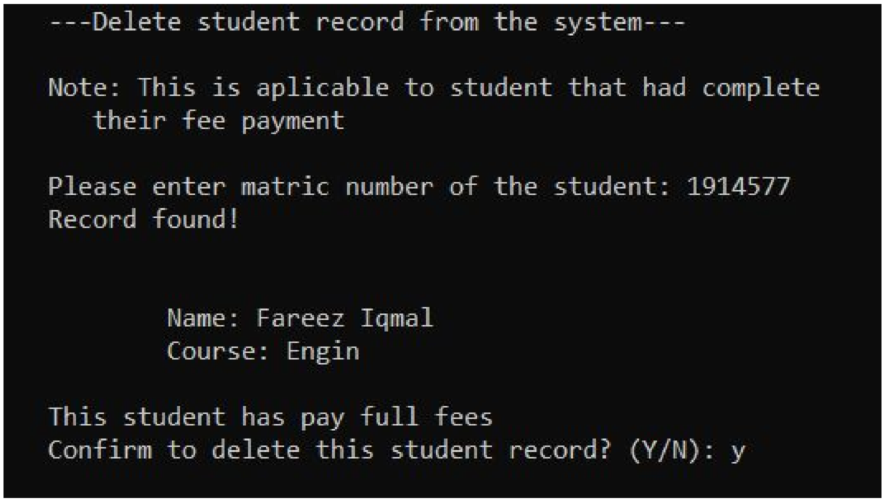
This feature is used to able a particular student to complete their payment. First, the program will ask for matric number and the name and course of a particular student is sought. After that, it will display the name and course of the student and shows the total fee, outstanding fees and amount that students have to pay. Lastly, the program asks the user to enter the amount they want to pay. Upon the successful transaction, the program displays “Payment successfully recorded.”.



```
----Make Payment----

Please enter matric number of the student: 1914577
Record found!

Initiating payment for student...

        Name: Fareez Iqmal
        Course: Engin

Total fee:       RM 2000.00
Outstanding fee: RM 2000.00
Amount to pay?:  RM 2000.00

You have entered RM 2000.00     Confirm? (Y/N)y
Payment successfully recorded
```

5. Delete Record

   This feature deletes all the information on a particular student. The program will ask for matric number that the user wants to delete. If the matric number is not found, the program will ask to re-enter matric number of the student. After record found, name and course of the student will be displayed and if the student already completed their payment, the student can be deleted, if not, the program displays the message “ This student still has outstanding fees.”.



```
---Delete student record from the system---

Note: This is aplicable to student that had complete
   their fee payment

Please enter matric number of the student: 1914577
Record found!


        Name: Fareez Iqmal
        Course: Engin

This student has pay full fees
Confirm to delete this student record? (Y/N): y
```

Conclusion

The high school management has to handle the finance issue for large numbers of students and maintenance was difficult.

The difficulties that we face in this program is in the beginning, we try to use array to store the data entered by the users but the data cannot be saved. Then, we use file instead of array to save it.

Furthermore, there is a conflict caused by the fstream flags. The program didn't work as intended when we use open the file both in output and input mode `(ios::out and ios::in)` at the same time. As a result, the output text file is messed up. Then we fixed by separate read and write file into different functions.

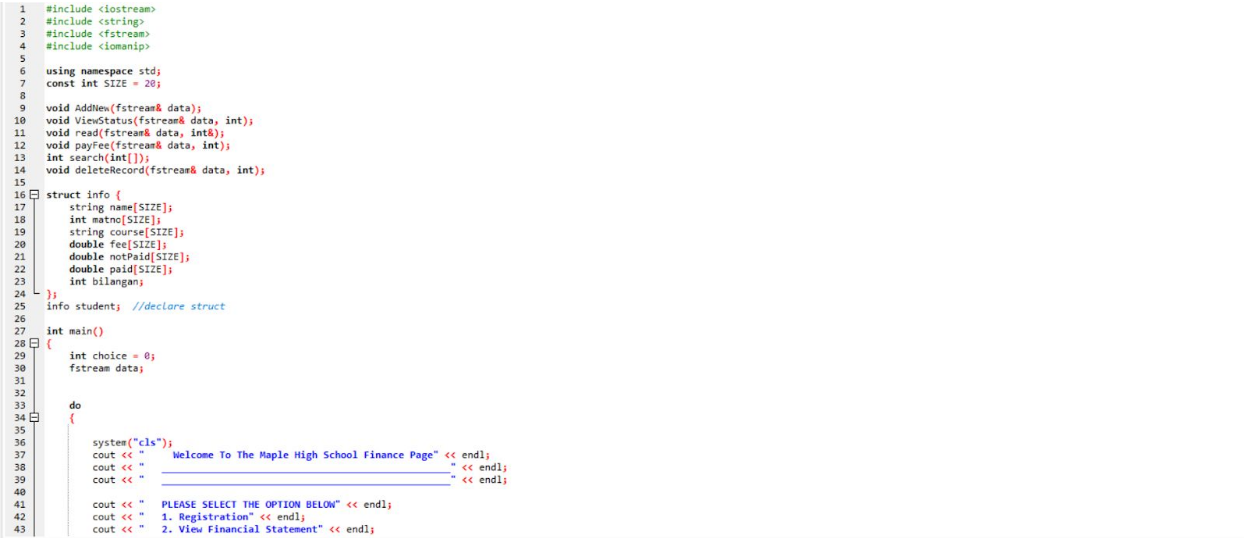
There is still room for improvement that can be done with the program. Many features can be implemented to increase the productivity of the user. One of them is an automatic sync feature. Student database file created by the program can be linked to cloud storage so that the database can be accessed or modified anywhere. Furthermore, this program can accept online payment. Student can pay their fees by using JomPay, FPX or others, then the receiving account send the payment data to the program and the program will process and update the database seamlessly.

Next, the other improvement that we can do for this program is making sure there are no repeated matric numbers so that the other users do not share the same matric number. Due date for the students to pay their fees can also be done in this program. This feature will help to ease the management of their finances.

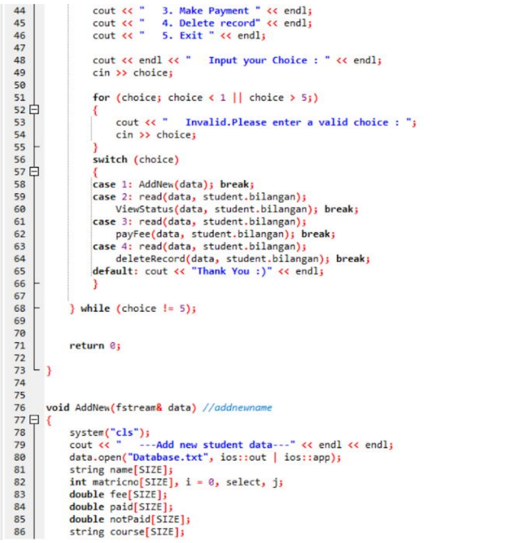
Reference

1. Gaddis, T. (2015). *Starting out with C++* (8th ed.). United State, America: Addison-Wesley.

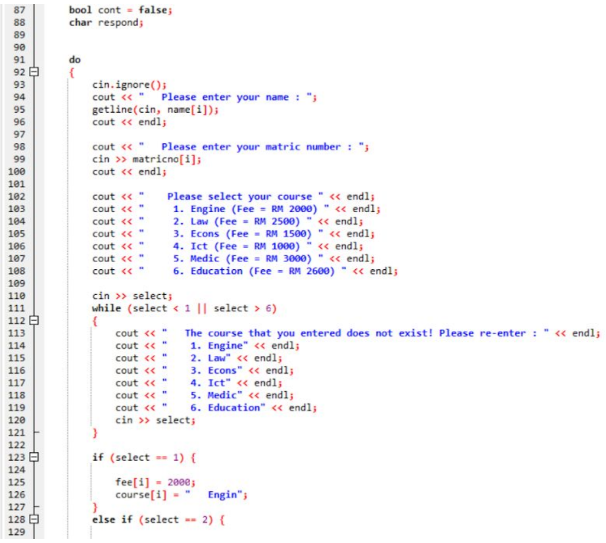
Appendices



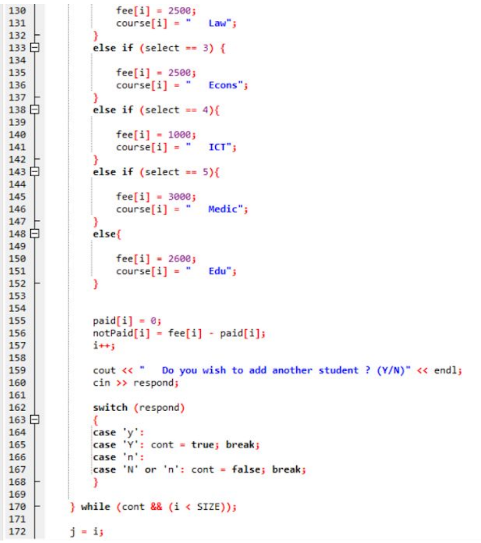
```cpp
#include <iostream>
#include <string>
#include <fstream>
#include <iomanip>

using namespace std;
const int SIZE = 20;

void AddNew(fstream& data);
void ViewStatus(fstream& data, int);
void read(fstream& data, int&);
void payFee(fstream& data, int);
int search(int[]);
void deleteRecord(fstream& data, int);

struct info {
    string name[SIZE];
    int matno[SIZE];
    string course[SIZE];
    double fee[SIZE];
    double notPaid[SIZE];
    double paid[SIZE];
    int bilangan;
};
info student;  //declare struct

int main()
{
    int choice = 0;
    fstream data;


    do
    {

        system("cls");
        cout << "     Welcome To The Maple High School Finance Page" << endl;
        cout << "   ______________________________________________" << endl;
        cout << "   ______________________________________________" << endl;

        cout << "   PLEASE SELECT THE OPTION BELOW" << endl;
        cout << "   1. Registration" << endl;
        cout << "   2. View Financial Statement" << endl;
```



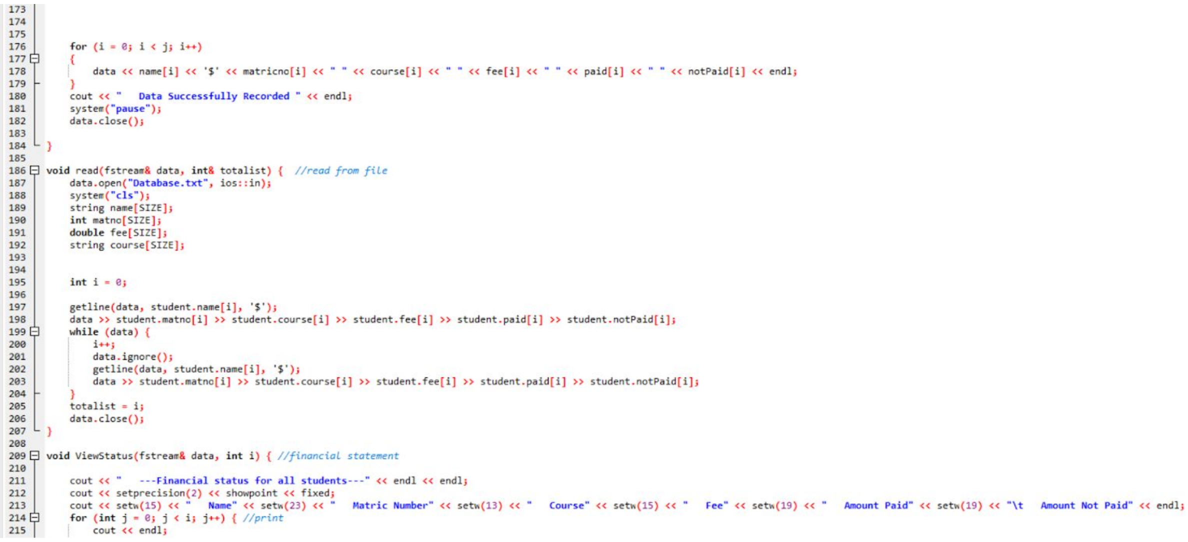
```cpp
        cout << "   3. Make Payment " << endl;
        cout << "   4. Delete record" << endl;
        cout << "   5. Exit " << endl;

        cout << endl << "   Input your Choice : " << endl;
        cin >> choice;

        for (choice; choice < 1 || choice > 5;)
        {
            cout << "   Invalid.Please enter a valid choice : ";
            cin >> choice;
        }
        switch (choice)
        {
        case 1: AddNew(data); break;
        case 2: read(data, student.bilangan);
            ViewStatus(data, student.bilangan); break;
        case 3: read(data, student.bilangan);
            payFee(data, student.bilangan); break;
        case 4: read(data, student.bilangan);
            deleteRecord(data, student.bilangan); break;
        default: cout << "Thank You :)" << endl;
        }

    } while (choice != 5);


    return 0;

}


void AddNew(fstream& data) //addnewname
{
    system("cls");
    cout << "   ---Add new student data---" << endl << endl;
    data.open("Database.txt", ios::out | ios::app);
    string name[SIZE];
    int matricno[SIZE], i = 0, select, j;
    double fee[SIZE];
    double paid[SIZE];
    double notPaid[SIZE];
    string course[SIZE];
```



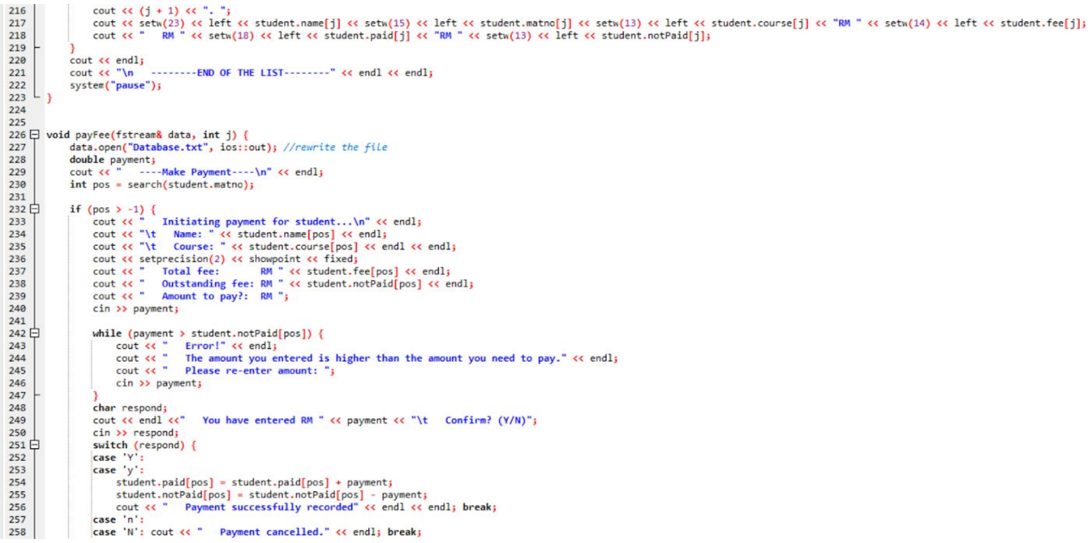
```
    bool cont = false;
    char respond;


    do
    {
        cin.ignore();
        cout << "   Please enter your name : ";
        getline(cin, name[i]);
        cout << endl;

        cout << "   Please enter your matric number : ";
        cin >> matricno[i];
        cout << endl;

        cout << "    Please select your course " << endl;
        cout << "     1. Engine (Fee = RM 2000) " << endl;
        cout << "     2. Law (Fee = RM 2500) " << endl;
        cout << "     3. Econs (Fee = RM 1500) " << endl;
        cout << "     4. Ict (Fee = RM 1000) " << endl;
        cout << "     5. Medic (Fee = RM 3000) " << endl;
        cout << "     6. Education (Fee = RM 2600) " << endl;

        cin >> select;
        while (select < 1 || select > 6)
        {
            cout << "   The course that you entered does not exist! Please re-enter : " << endl;
            cout << "    1. Engine" << endl;
            cout << "    2. Law" << endl;
            cout << "    3. Econs" << endl;
            cout << "    4. Ict" << endl;
            cout << "    5. Medic" << endl;
            cout << "    6. Education" << endl;
            cin >> select;
        }

        if (select == 1) {

            fee[i] = 2000;
            course[i] = "   Engin";
        }
        else if (select == 2) {

```



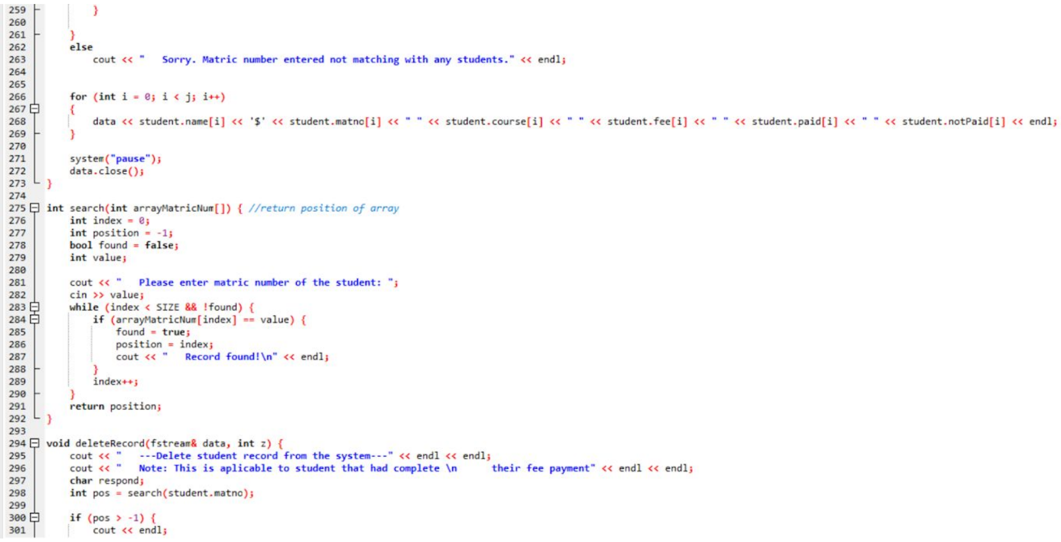
```
            fee[i] = 2500;
            course[i] = "   Law";
        }
        else if (select == 3) {

            fee[i] = 2500;
            course[i] = "   Econs";
        }
        else if (select == 4){

            fee[i] = 1000;
            course[i] = "   ICT";
        }
        else if (select == 5){

            fee[i] = 3000;
            course[i] = "   Medic";
        }
        else{

            fee[i] = 2600;
            course[i] = "   Edu";
        }


        paid[i] = 0;
        notPaid[i] = fee[i] - paid[i];
        i++;

        cout << "   Do you wish to add another student ? (Y/N)" << endl;
        cin >> respond;

        switch (respond)
        {
        case 'y':
        case 'Y': cont = true; break;
        case 'n':
        case 'N' or 'n': cont = false; break;
        }

    } while (cont && (i < SIZE));

    j = i;
```



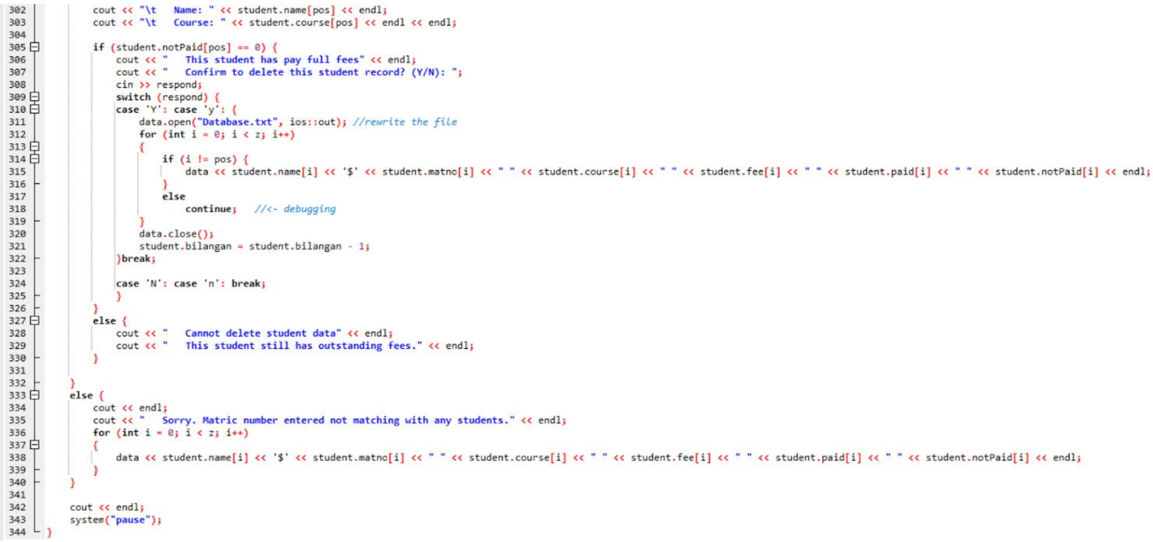
```



    for (i = 0; i < j; i++)
    {
        data << name[i] << '$' << matricno[i] << " " << course[i] << " " << fee[i] << " " << paid[i] << " " << notPaid[i] << endl;
    }
    cout << "   Data Successfully Recorded " << endl;
    system("pause");
    data.close();

}

void read(fstream& data, int& totalist) {  //read from file
    data.open("Database.txt", ios::in);
    system("cls");
    string name[SIZE];
    int matno[SIZE];
    double fee[SIZE];
    string course[SIZE];


    int i = 0;

    getline(data, student.name[i], '$');
    data >> student.matno[i] >> student.course[i] >> student.fee[i] >> student.paid[i] >> student.notPaid[i];
    while (data) {
        i++;
        data.ignore();
        getline(data, student.name[i], '$');
        data >> student.matno[i] >> student.course[i] >> student.fee[i] >> student.paid[i] >> student.notPaid[i];
    }
    totalist = i;
    data.close();
}

void ViewStatus(fstream& data, int i) { //financial statement

    cout << "   ---Financial status for all students---" << endl << endl;
    cout << setprecision(2) << showpoint << fixed;
    cout << setw(15) << "   Name" << setw(23) << "   Matric Number" << setw(13) << "   Course" << setw(15) << "   Fee" << setw(19) << "   Amount Paid" << setw(19) << "\t   Amount Not Paid" << endl;
    for (int j = 0; j < i; j++) { //print
        cout << endl;
```



```
        cout << (j + 1) << ". ";
        cout << setw(23) << left << student.name[j] << setw(15) << left << student.matno[j] << setw(13) << left << student.course[j] << "RM " << setw(14) << left << student.fee[j];
        cout << "   RM " << setw(18) << left << student.paid[j] << "RM " << setw(13) << left << student.notPaid[j];
    }
    cout << endl;
    cout << "\n   --------END OF THE LIST--------" << endl << endl;
    system("pause");
}


void payFee(fstream& data, int j) {
    data.open("Database.txt", ios::out); //rewrite the file
    double payment;
    cout << "   ----Make Payment----\n" << endl;
    int pos = search(student.matno);

    if (pos > -1) {
        cout << "   Initiating payment for student...\n" << endl;
        cout << "\t   Name: " << student.name[pos] << endl;
        cout << "\t   Course: " << student.course[pos] << endl << endl;
        cout << setprecision(2) << showpoint << fixed;
        cout << "   Total fee:        RM " << student.fee[pos] << endl;
        cout << "   Outstanding fee: RM " << student.notPaid[pos] << endl;
        cout << "   Amount to pay?:  RM ";
        cin >> payment;

        while (payment > student.notPaid[pos]) {
            cout << "   Error!" << endl;
            cout << "   The amount you entered is higher than the amount you need to pay." << endl;
            cout << "   Please re-enter amount: ";
            cin >> payment;
        }
        char respond;
        cout << endl <<"   You have entered RM " << payment << "\t   Confirm? (Y/N)";
        cin >> respond;
        switch (respond) {
        case 'Y':
        case 'y':
            student.paid[pos] = student.paid[pos] + payment;
            student.notPaid[pos] = student.notPaid[pos] - payment;
            cout << "   Payment successfully recorded" << endl << endl; break;
        case 'n':
        case 'N': cout << "   Payment cancelled." << endl; break;
```



```
        }

    }
    else
        cout << "   Sorry. Matric number entered not matching with any students." << endl;


    for (int i = 0; i < j; i++)
    {
        data << student.name[i] << '$' << student.matno[i] << " " << student.course[i] << " " << student.fee[i] << " " << student.paid[i] << " " << student.notPaid[i] << endl;
    }

    system("pause");
    data.close();
}

int search(int arrayMatricNum[]) { //return position of array
    int index = 0;
    int position = -1;
    bool found = false;
    int value;

    cout << "   Please enter matric number of the student: ";
    cin >> value;
    while (index < SIZE && !found) {
        if (arrayMatricNum[index] == value) {
            found = true;
            position = index;
            cout << "   Record found!\n" << endl;
        }
        index++;
    }
    return position;
}

void deleteRecord(fstream& data, int z) {
    cout << "   ---Delete student record from the system---" << endl << endl;
    cout << "   Note: This is aplicable to student that had complete \n      their fee payment" << endl << endl;
    char respond;
    int pos = search(student.matno);

    if (pos > -1) {
        cout << endl;
```



```
        cout << "\t   Name: " << student.name[pos] << endl;
        cout << "\t   Course: " << student.course[pos] << endl << endl;

        if (student.notPaid[pos] == 0) {
            cout << "   This student has pay full fees" << endl;
            cout << "   Confirm to delete this student record? (Y/N): ";
            cin >> respond;
            switch (respond) {
            case 'Y': case 'y': {
                data.open("Database.txt", ios::out); //rewrite the file
                for (int i = 0; i < z; i++)
                {
                    if (i != pos) {
                        data << student.name[i] << '$' << student.matno[i] << " " << student.course[i] << " " << student.fee[i] << " " << student.paid[i] << " " << student.notPaid[i] << endl;
                    }
                    else
                        continue;   //<- debugging
                }
                data.close();
                student.bilangan = student.bilangan - 1;
            }break;

            case 'N': case 'n': break;
            }
        }
        else {
            cout << "   Cannot delete student data" << endl;
            cout << "   This student still has outstanding fees." << endl;
        }

    }
    else {
        cout << endl;
        cout << "   Sorry. Matric number entered not matching with any students." << endl;
        for (int i = 0; i < z; i++)
        {
            data << student.name[i] << '$' << student.matno[i] << " " << student.course[i] << " " << student.fee[i] << " " << student.paid[i] << " " << student.notPaid[i] << endl;
        }
    }

    cout << endl;
    system("pause");
}
```