**STUDENT GRADE**

**MANAGEMENT**

**SYSTEM**

**BMIS105 OOP with C++**

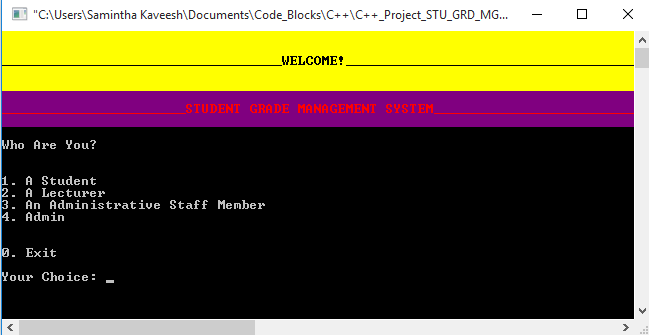
**Group Assignment**

**Batch: CS/MIS/15.1**

Hand out date : 27 Aug 2015

Hand in date : 9 Nov 2015

|  |  |
| --- | --- |
| **INDEX NUMBER** | **NAME** |
| BSC-UCD-CSC-15.1-063 | Chulan Kotudurage |
| BSC-UCD-CSC-15.1-088 | Samintha Kaveesh |
| BSC-UCD-CSC-15.1-091 | Kalana Wijesekara |
| BSC-UGC-MIS-15.1-012 | Sandun Nimsara |



Content

Description and Justification of the Design of the Implementation Codes 3

Description and Justification of the Implementation Codes In Terms of 7

Object Oriented Programming Concepts

Description and Justification of the Validation Codes Applied Into The 9

Implementation Codes

Description and Discussion of the Testing Plan and Result 11

Technical Details 14

Conclusion 15

References 16

**Description and Justification of the Design of the Implementation Codes**

**DESCRIPTION:**

In this Student Management System, Administrator has the highest access priority that means he can perform each and every functionality specified in the system. The functionalities he can perform includes add student, register student, register lecturers, register administrative staff so that users can access the system, view student marks, view user registration details, view module details. And the lowest access priority is given to the students who can only view his own profile and module details. There are two more other types of users who are lecturers and administrative staff members. Lecturers can view student details, marks and edit student marks whereas administrative staff members can only view student details and marks.

**JUSTIFICATION:**

Administrator : Has the highest priority. Admin can view/ enter/ edit/ delete,

Students, Lecturers, Administrative Staff Members details. Also can view/ enter/ edit/ delete student marks. Admin can do all the functions in the application.

Lecturer : Lecturer can only view student details and view/ enter/ edit/ delete

Student marks.

Administrative : Only view student details student marks.

Staff Members

Students : Can only view their details and marks.

*MAIN WINDOW CODE:*

cout << "Who Are You?\n\n" << endl;

cout << "1. A Student" << endl;

cout << "2. A Lecturer" << endl;

cout << "3. An Administrative Staff Member" << endl;

cout << "4. Admin" << endl;

cout << "\n\n0. Exit" << endl;

cout << "\nYour Choice: "; cin >> whoareyou;

if( whoareyou == '1' )

{

RegisterUser regstudent;

string username;

cout << "Enter Username: "; cin >> username;

This part of the code executes the main window of the student grade management system. There are 5 choices which the user can choose from to interact with the application.

**DESCRIPTION:**

To get a more user friendly view, we used colors for final output. We also used timing function to view certain message for certain time.

**JUSTIFICATION:**

Main thing is this is a console program. So it has a very boring, black background and off-white foreground color. When working with console programs it’s hard to find output of the program. So we changed the colors of some outputs.

Especially we used timing function to display certain small messages for certain time. For example when we needed to display “User logged in successfully” we used timing function to display that message for 2 seconds. Also used “CLS” command in “Command Prompt” to clear the screen.

To apply these colour codes we had to use additional header libraries.

“#include <windows.h>” header file used to change colors and “#include <dos.h>” header file used to apply timing function.

*COLOR CHANGE CODE:*

SetConsoleTextAttribute( hstdout, 0xA ); //COLOR (“A” for green)

cout << “Successfully Logged In” << endl; //cout will turn green.

FlushConsoleInputBuffer( hstdin ); // Back to normal color

SetConsoleTextAttribute( hstdout, csbi.wAttributes );

*TIMING FUNCTION CODE:*

Sleep(2000); //This will sleep the program for 2 seconds.

system(“CLS”): //This will clear the console window.

**DESCRIPTION:**

To make the code easy to use, we used functions. Record display, record write, record edit, record search functionalities based on function calls.

**JUSTIFICATION:**

When it comes to the smaller programs we can code it without functions. But when it comes to the programs which has thousands of lines of codes, we can’t simply code it like a list. Sometimes we have to use the same code several times. So in our code we used many functions. To view/ edit/ search like functionalities we used functions. In some parts we passed values to functions to execute some codes with ease.

*FUNCTIONS CODE:*

void write\_record()

{

ofstream outFile;

outFile.open("users/students.dat", ios::binary | ios::app);

Student obj;

obj.setData();

outFile.write((char\*)&obj, sizeof(obj));

outFile.close();

}

**DESCRIPTION:**

When it comes to the control structures (selections), we used mainly “If conditions” and “switch case”.

**JUSTIFICATION:**

First thing is this is a console program. So that we cannot assign codes to “buttons”. To get inputs from users we had to use “If conditions” and “switch case”. If a condition is “true” program will execute the code or else it will execute the next code. There is also “Default” case in “switch case” selections, which we can use to catch error inputs and output something else.

*PART OF “IF CONDITION” CODE:*

char\* to\_srch\_keyword;

//This will search for the keyword

if (srch\_keyword == 2)

to\_srch\_keyword = obj\_fn\_srch.retFname();

else if (srch\_keyword == 3)

to\_srch\_keyword = obj\_fn\_srch.retLname();

else if (srch\_keyword == 4)

to\_srch\_keyword = obj\_fn\_srch.retNIC();

else if (srch\_keyword == 5)

to\_srch\_keyword = obj\_fn\_srch.retM1();

*PART OF “SWITCH CASE” CODE:*

int swi;

switch (swi)

{

case 0:

system("CLS");

break;

case 1:

int j;

cout << "\nHow Many Records You Want to Enter: "; cin >> j;

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

write\_record();

cout << "Records Entered Successfully" <<endl;

Sleep(2000);

system("CLS");

break;

}

**Description and Justification of the Implementation Codes In Terms of Object Oriented Programming Concepts**

**DESCRIPTION:**

Object-oriented programming (OOP) is a programming paradigm based on the concept of "objects", which are data structures that contain data, in the form of fields, often known as attributes; and code, in the form of procedures, often known as methods. A distinguishing feature of objects is that an object's procedures can access and often modify the data fields of the object with which they are associated (objects have a notion of "this" or "self"). In OOP programming, computer programs are designed by making them out of objects that interact with one another. There is significant diversity in object-oriented programming, but most popular languages are class-based, meaning that objects are instances of classes, which typically also determines their type.

(Wikipedia)

**JUSTIFICATION:**

To make use of OOP concepts, we used 3 main classes. Those are “class RegisterUser”, ”class Student” and “class Staff”. “class RegisterUser” will check on functions to register a user and whenever registering a user this class will check if the user we are going to register is already exists or not. Staff and Student classes will execute when codes related students, lecturers and administrative staff are executed.

*PART OF CLASS STUDENT CODE:*

class Student

{

private:

int stuno;

char index[20];

char fname[20];

char lname[20];

char nic[20];

char gender[20];

char dob[20];

char batch[20];

char joined[20];

*PART OF CLASS REGISTER USER CODE:*

class RegisterUser{

private:

string password;

string uname;

string pwd;

public:

bool IsLoggedIN(string usertype, string username)

{

cout << "Enter Password: "; cin >> password;

transform( username.begin(), username.end(), username.begin(), ptr\_fun <int, int> ( tolower ) );

string reader = usertype+"\\";

reader += username;

reader += ".dat";

ifstream read(reader.c\_str());

getline(read, uname);

getline(read, pwd);

read.close();

if(uname == username && pwd == password)

{

return true;

}

else

{

return false;

}

}

**Description and Justification of the Validation Codes Applied Into the Implementation Codes**

**DESCRIPTION:**

Validation is applied into the system for check some inputs from the user. This is specially used when they enter a new password or modify the password. In some cases it uses to check whether user want to continue or not.

**JUSTIFICATION:**

*CONFIRM PASSWORD CODE:*

if (old\_pw == pwd)

{

cout << "Enter New Password : "; cin >> new\_pw;

cout << "Confirm New Password : "; cin >> cnf\_pw;

if (cnf\_pw == new\_pw)

{

ofstream writeCUP(readerCUP.c\_str(), ios::out);

writeCUP << username << endl << new\_pw << endl << adminNo;

writeCUP.close();

return "Password Changed Successfully";

}

*CONFIRM DELETION CODE:*

cout << "Enter Username of the Student : "; cin >> student\_uname;

cout << "Are You Sure You Want to Delete \"" << student\_uname << "\" [Y/N] ? " ; cin >> yes\_no;

**DESCRIPTION:**

Username repetition preventing and validating.

**JUSTIFICATION:**

Administrator has the privilege to register any user. But if he register’s the same user twice, it could replace the user file and it will also replace the password. This is bad for the whole system because it has no security at all. So we figured out this and validate the username to be insert only in simple letters. If the administrator input username in capital or mix case it will automatically convert the username to simple case.

*USERNAME CASE CONVERT CODE:*

transform( student\_uname.begin(), student\_uname.end(), student\_uname.begin(), ptr\_fun <int, int> ( tolower ) ); //converting student\_uname to lowercase

To execute this codes we used extra header files. Those are “#include <algorithm>”, “#include <cctype>” and “#include <functional>”.

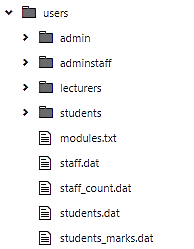
**Description and Discussion of the Testing Plan and Result**

**DESCRIPTION:**

To make use of file structures we used file handling in this program. In that case we have a folder called “users” and it has subfolder for each user type to categorize each functions for each users.

**JUSTIFICATION:**

*FILE STRUCTURE:*



When it comes to the “Administrator” the first subfolder-“admin”, inside “users” folder is for admin. In admin folder it saves login functions for admin. Other users like Lecturers, Administrative staff members and Students logging files save under “lecturers”, “adminstaff” and “students” folders.

In “modules.txt” file includes all the descriptions about modules. Admin can manually modify it or in program also can modify. “staff.dat” and “students.dat” files saves all the details about staff and students. “students\_marks.dat” file saves Student marks for each module. In program there is an auto generating “Staff Number”. To automatically regenerate the “Staff Number” when program runs next time, it saves the counting automatically in “staff\_count.dat”.

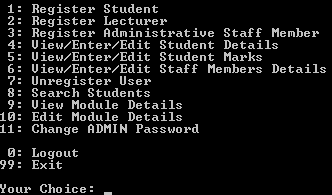
**DESCRIPTION:**

User friendly environment with much more functionalities for each user. Like we mentioned above Administrator has the highest access priority.

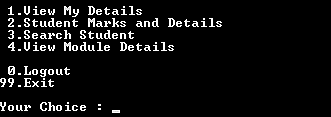
**JUSTIFICATION:**

Only Admin has the access to register all members to the system. Admin has to register a student with details and the primary key for this details are “Student Number”. When admin is going to give access to that student he has to again register that student providing a “Username” and “Password”. So that, the student can login to the system. System will capture the “Student Number” from “Username” and it will automatically show student details corresponding to that “Student Number”. Same method will apply for “Lecturers” and “Administrative Staff Members”.

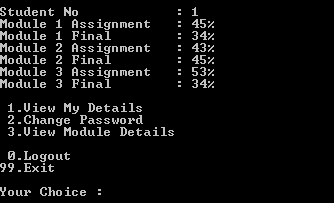
*ADMINISTRATOR FUNCTIONS:*



*LECTURER’S FUNCTIONS:*



*STUDENT’S FUNCTIONS:*



**DESCRIPTION:**

Admin and staff members can search students but admin has the privilege to edit and delete records.

**JUSTIFICATION:**

Like we mentioned earlier only admin can edit and delete student details. For search purpose we provided 9 options.

1. Search by Student Number
2. Search by First Name
3. Search by Last Name
4. Search by NIC Number
5. Search by Module 1
6. Search by Module 2
7. Search by Module 3
8. Search by Batch
9. Search by Date Joined

Search by Batch and Search by Date Joined are listed under “Other Categories”. After searching purpose is done Admin can edit or delete records and these changes will automatically save.

**Technical Details**

* We compiled this Win32 Console Application using “Code Blocks” and

“Visual C++” (Both includes in the CD).

* Default Admin password : 5454
* User Accounts:  
    
  Students

1. Username : samintha  
   Password : 123456

* Lecturers  
  1. Username : darshana  
     Password : 123456
* Administrative Staff members  
  1. Username : chulan  
     Password : 123456
* Lines of codes : Over 2800

**Conclusion**

It was a great experience to design and implement the Student Management System by using an Object Oriented Programming Language C++ and to work on its documentation. While working on this project, we have learned many things especially how to apply the concepts of OOP paradigm in modelling of real world systems. This assignment helped us to get the better understanding to develop new class structures and organize them such that they will model real world systems within computers. This Assignment that we embarked on helped us to explore ideas related to the project. It also helped us in getting in the better understanding of basic programming concepts of C++ language such as control structures, file handling etc. In this assignment, we have used many concepts of C++ language, we had learned.

We have also used OOPs concept in an excellent manner. We also were inspired by many projects we looked at from the internet that helped us apply OOP concepts to the console application.

After doing this assignment, the experience we received from the great team work we are in a position to explain object oriented programming concepts and apply them to the modelling of real world systems by utilizing its offered facilities.

**References**

*1. Object-oriented programming, [Online] Available from: https://en.wikipedia.org/wiki/Object-oriented\_programming*

*2.* *Colors in console for beginners, [Online] Available from: http://www.cplusplus.com/forum/beginner/5830/*

*3.* *Sleep() function windows 8 C++, [Online] Available from: http://stackoverflow.com/questions/21904745/sleep-function-windows-8-c*