Addis Ababa Science and Technology University

Fundamentals of Programming II (C++)

Project Work Solutions

---

1. Project Overview

The Student Record Management System (SRMS) is a menu-driven C++ application designed to automate the management of student information, course registration, academic performance evaluation, and reporting.

The system enables academic departments to efficiently manage student records, maintain course information, record grades, compute GPA/CGPA, determine academic status, and generate various reports.

2. Problem Statement

Traditional manual record management suffers from:

- Data redundancy
- Data inconsistency
- Slow retrieval of records
- Difficulty in generating reports
- Errors in GPA/CGPA computation
- Difficulty in ranking students

The SRMS addresses these problems by providing a computerized solution.

3. Functional Requirements

1	Student Management	• Register students. • Update Student Information. • Delete student records. • Search students. • Display Students
2	Course Management	• Register courses. • Modify course information. • Delete Course • Search courses • Check prerequisite requirements • Display courses

3	Grade Management	• Record grades. • Update grades. • Delete Grade • Compute semester GPA. • Compute cumulative CGPA
4	Academic Evaluation	• Determine student academic status • Generate student ranking • Identify top-performing students
5	Reporting	• Number of students. • Generate Transcript • Generate Department Report • Generate Pass Rate Report • Generate Top Students Report
6	File Management	• Save Data • Load Data

4. Detailed IPO Tables

IPO Table 1: Student Registration

Input	Process	Output
Student ID	Validate uniqueness	Student registered
Name	Store information	Confirmation message
Gender	Save record	Updated database
Department		
Admission Year		

IPO Table 2: Course Registration

Input	Process	Output
Course Code	Validate code	Course added
Course Name	Store course	Confirmation
Credit Hour	Save record	Updated course list

IPO Table 3: Grade Recording

Input	Process	Output
Student ID	Verify student	Grade saved
Course Code	Verify course	Confirmation
Grade	Convert to grade point	Updated record

IPO Table 4: GPA and CGPA Calculation

Input	Process	Output
Course Grades	Convert to grade points	GPA
Credit Hours	Calculate weighted average	GPA Report
Semester GPA Records	Compute cumulative average	CGPA

Formula: $GPA = \sum(GP \times CH) / \sum CH$

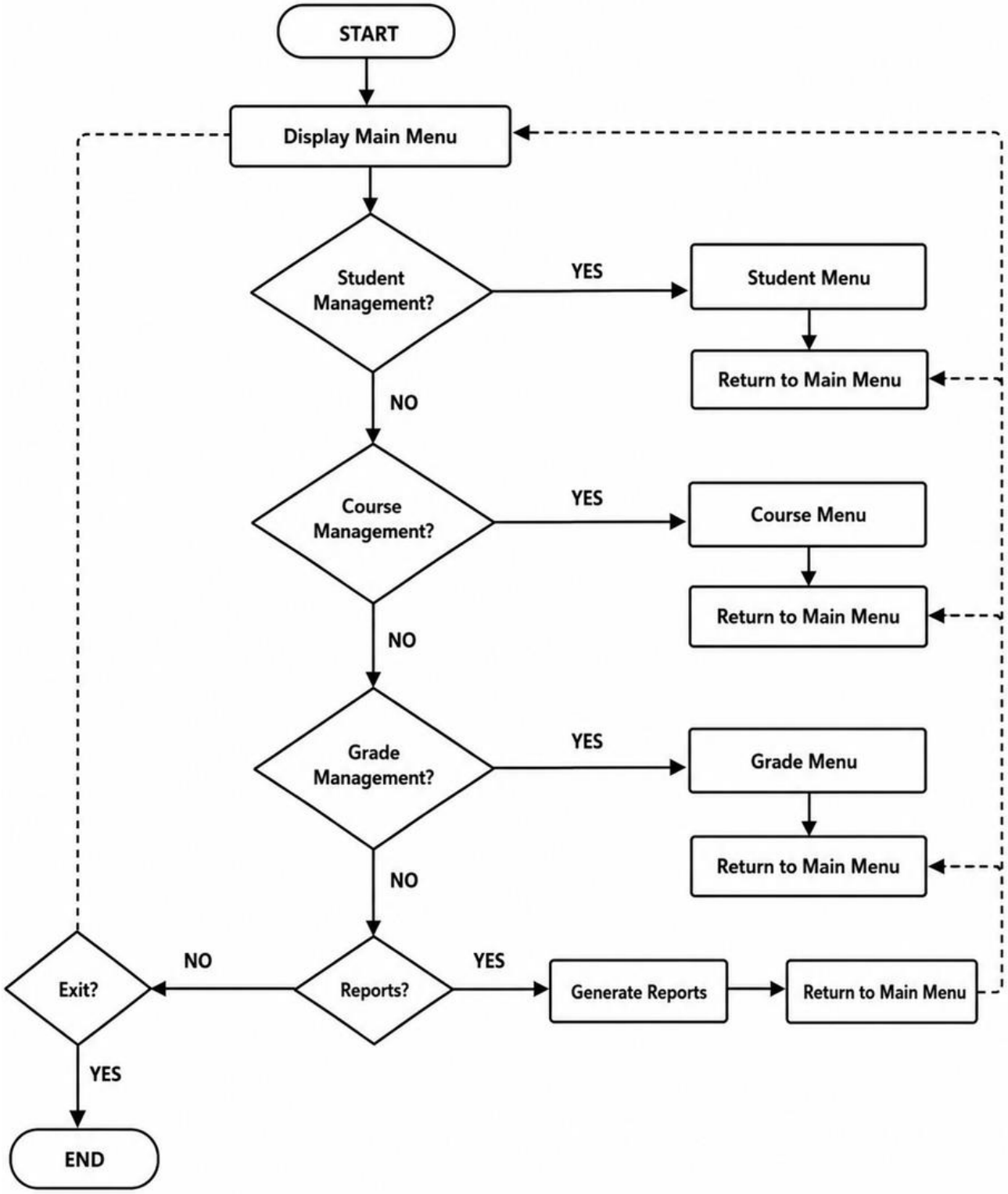
IPO Table 5: Student Ranking

Input	Process	Output
Student CGPAs	Sort descending	Ranked list

IPO Table 6: Academic Status

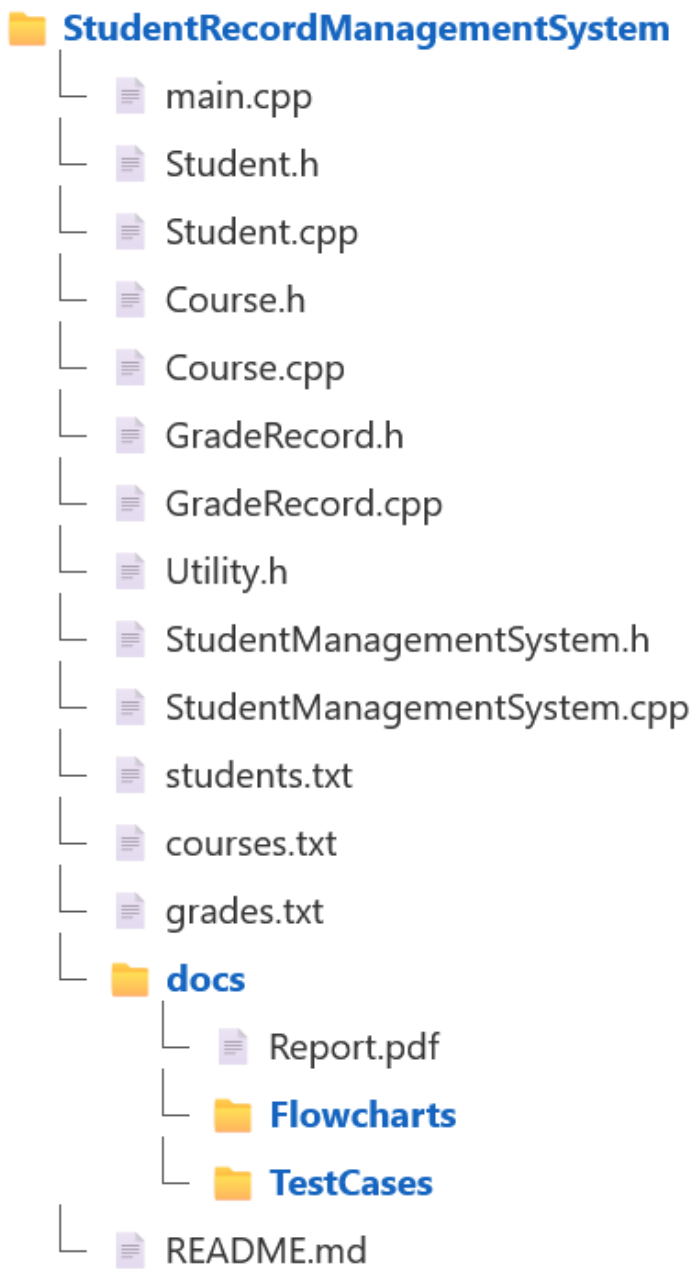
Input	Process	Output
CGPA	Compare against rules	PASS WARNING DISMISSAL

5. Flowchart for Main System



6. Complete implementation

6.1 Project Structure



6.2 Implementation Code

File 1: Student.h

```
#ifndef STUDENT_H
#define STUDENT_H

#include <string>
using namespace std;

class Student {
    private:
        string studentID;
        string fullName;
        string gender;
        string dept;
        int admissionYear;

    public:
        Student ();

        Student (string id, string name, string gender, string dept, int year);

        void setStudentID (string id);
        string getStudentID () const;

        void setFullName (string name);
        string getFullName () const;

        void setGender (string gender);
        string getGender () const;

        void setDepartment (string dept);
        string getDepartment () const;

        void setAdmissionYear (int year);
        int getAdmissionYear () const;

        void input ();
        void display () const;

        string toFileString () const;
};

#endif
```

File 2: Student.cpp

```cpp
#include "Student.h"
#include <sstream>

Student::Student () {
    studentID = "";
    fullName = "";
    gender = "";
    department = "";
    admissionYear = 0;
}

Student::Student (string id, string name, string gen, string dept, int year) {
    studentID = id;
    fullName = name;
    gender = gen;
    department = dept;
    admissionYear = year;
}

void Student::setStudentID (string id) {  studentID = id;  }

string Student::getStudentID () const { return studentID; }

void Student::setFullName (string name) {  fullName = name; }

string Student::getFullName () const {  return fullName; }

void Student::setGender (string gen) {  gender = gen;  }

string Student::getGender () const {  return gender; }

void Student::setDepartment (string dept) {  department = dept; }

string Student::getDepartment () const {  return department; }

void Student::setAdmissionYear (int year) {  admissionYear = year; }

int Student::getAdmissionYear () const {  return admissionYear; }

void Student::input () {
    cout << "Student ID: ";
    cin >> studentID;
```

```
    cin.ignore ();

    cout << "Full Name: ";
    getline (cin, fullName);

    cout << "Gender: ";
    getline (cin, gender);

    cout << "Department: ";
    getline (cin, department);

    cout << "Admission Year: ";
    cin >> admissionYear;
}

void Student::display () const {
    cout<< studentID << "\t"<< fullName << "\t"<< gender << "\t";
    cout<< department << "\t" << admissionYear << endl;
}

string Student::toFileString () const {
    stringstream ss;

    ss << studentID << ","<< fullName << ","<< gender << ",";
    ss << department << "," << admissionYear;

    return ss.str ();
}
```

File 3: Course.h

```
#ifndef COURSE_H
#define COURSE_H

#include <iostream>
#include <string>
using namespace std;

class Course {
    private:
        string courseCode;
        string courseName;
        int creditHour;
```

```
    public:
        Course();

        Course(
            string code,
            string name,
            int credit);

        void setCourseCode (string code);
        string getCourseCode () const;

        void setCourseName (string name);
        string getCourseName () const;

        void setCreditHour (int credit);
        int getCreditHour () const;

        void input ();
        void display () const;

        string toFileString () const;
};

#endif
```

File 4: Course.cpp

```
#include "Course.h"
#include <sstream>

Course::Course () {
    courseCode = "";
    courseName = "";
    creditHour = 0;
}
Course::Course (string code, string name, int credit) {
    courseCode = code;
    courseName = name;
    creditHour = credit;
}

void Course::setCourseCode (string code) {
    courseCode = code;
}
```

```cpp
string Course::getCourseCode () const {
    return courseCode;
}

void Course::setCourseName (string name) {
    courseName = name;
}

string Course::getCourseName () const {
    return courseName;
}

void Course::setCreditHour (int credit) {
    creditHour = credit;
}

int Course::getCreditHour () const {
    return creditHour;
}

void Course::input () {
    cout << "Course Code: ";
    cin >> courseCode;

    cin.ignore();

    cout << "Course Name: ";
    getline(cin, courseName);

    cout << "Credit Hour: ";
    cin >> creditHour;
}

void Course::display () const {
    cout<< courseCode << "\t"<< courseName << "\t"<< creditHour<< endl;
}

string Course::toFileString () const {
    stringstream ss;
    ss << courseCode << ","<< courseName << ","<< creditHour;

    return ss.str();
}
```

File 5: GradeRecord.h

```
#ifndef GRADERECORD_H
#define GRADERECORD_H

#include <iostream>
#include <string>
using namespace std;

class GradeRecord {
    private:
        string studentID;
        string courseCode;
        string letterGrade;

    public:
        GradeRecord();

        GradeRecord(
            string sid,
            string ccode,
            string grade);

        void setStudentID (string id);
        string getStudentID () const;

        void setCourseCode (string code);
        string getCourseCode () const;

        void setLetterGrade (string grade);
        string getLetterGrade () const;

        double getGradePoint() const;

        void input ();
        void display () const;

        string toFileString () const;
};

#endif
```

File 6: GradeRecord.cpp

```cpp
#include "GradeRecord.h"
#include <sstream>

GradeRecord::GradeRecord() {
    studentID = "";
    courseCode = "";
    letterGrade = "";
}

GradeRecord::GradeRecord(string sid, string ccode, string grade) {
    studentID = sid;
    courseCode = ccode;
    letterGrade = grade;
}

void GradeRecord::setStudentID (string id) { studentID = id;}

string GradeRecord::getStudentID () const { return studentID;}

void GradeRecord::setCourseCode (string code) { courseCode = code;}

string GradeRecord::getCourseCode () const { return courseCode;}

void GradeRecord::setLetterGrade (string grade) { letterGrade = grade;}

string GradeRecord::getLetterGrade () const { return letterGrade;}

double GradeRecord::getGradePoint () const {
    if(letterGrade=="A")
         return 4.0;
    if(letterGrade=="B+")
         return 3.5;
    if(letterGrade=="B")
         return 3.0;
    if(letterGrade=="C+")
         return 2.5;
    if(letterGrade=="C")
         return 2.0;
    if(letterGrade=="D")
         return 1.0;

    return 0.0;
}
```

```
void GradeRecord::input () {
    cout << "Student ID: ";
    cin >> studentID;

    cout << "Course Code: ";
    cin >> courseCode;

    cout << "Grade: ";
    cin >> letterGrade;
}

void GradeRecord::display () const {
    cout<< studentID << "\t"<< courseCode << "\t"<< letterGrade<< endl;
}

string GradeRecord::toFileString () const {
    stringstream ss;
    ss << studentID << ","<< courseCode << ","<< letterGrade;

    return ss.str ();
}
```

Sample Output Screenshots:

1. **Main Menu:**

```
==================================
 STUDENT RECORD MANAGEMENT SYSTEM
==================================

1. Student Management
2. Course Management
3. Grade Management
4. Reports
5. Save Data
0. Exit

Select Option:
```

2. Student Registration

```
-----------------------------------
 STUDENT REGISTRATION
-----------------------------------

Student ID      : ST001
Full Name       : Abebe Kebede
Gender          : Male
Department      : Software Engineering
Admission Year  : 2024

Student Registered Successfully!
```

3. Student List

```
-------------------------------------------------------------
ID      Name              Gender   Department         Year
-------------------------------------------------------------
ST001   Abebe Kebede      Male     Software Eng.      2024
ST002   Hana Tesfaye      Female   Computer Science   2024
-------------------------------------------------------------
```

4. Transcript Report

```
==================================================
               STUDENT TRANSCRIPT
==================================================

Student ID : ST001
Name       : Abebe Kebede

Course      Grade     Credit
--------------------------------
CS101       A         4
CS102       B+        3
CS103       B         3

GPA         : 3.45
CGPA        : 3.41
Status      : PASS
```

5. Student Ranking Report

```
==========================================
              STUDENT RANKING
==========================================

Rank    ID       Name             CGPA
------------------------------------------
1       ST005    Samuel T.        3.89
2       ST001    Abebe K.         3.45
3       ST002    Hana T.          3.21
4       ST007    Meron B.         2.98

==========================================
```