Manish Mahato

The British College

Student ID: 23085136

work sheet 2

**Worksheet 2**

**Task 1: Basic student grading system prototype using classes and objects. [20 Marks]**

Write a program that manages a simple student grade calculator with the following requirements. Create a Student class that has:

1. Student name (string)
2. Three subject marks (integers)
3. A basic member function to calculate average

The program should:

1. Accept student details (name and marks) from user input
2. Calculate and display:
   1. Total marks
   2. Average marks
   3. Grade (A for ≥90%, B for ≥80%, C for ≥70%, D for ≥60%, F for <60%)
3. Display a message if any mark is below 0 or above 100

**Source code for this:**

#include <iostream>  
using namespace std;  
  
class Student {  
public:  
 string name;  
 int m1, m2, m3;  
  
 void input() {  
 cout << "Enter student's name: ";  
 cin >> name;  
  
 cout << "Enter marks for 3 subjects: ";  
 cin >> m1 >> m2 >> m3;  
 }  
  
 int total() {  
 return m1 + m2 + m3;  
 }  
  
 float average() {  
 return total() / 3.0;  
 }  
  
 char grade() {  
 float avg = average();  
 if (avg >= 90) return 'A';  
 else if (avg >= 80) return 'B';  
 else if (avg >= 70) return 'C';  
 else if (avg >= 60) return 'D';  
 else return 'F';  
 }  
  
 void display() {  
 cout << "\nName: " << name << endl;  
 cout << "Total: " << total() << endl;  
 cout << "Average: " << average() << endl;  
 cout << "Grade: " << grade() << endl;  
 }  
};  
  
int main() {  
 Student s;  
 s.input();  
 s.display();  
 return 0;  
}

**Output:**

**A screenshot of a computer program

AI-generated content may be incorrect.**

**Task 2: Programming assignments: All questions are mandatory**

1. Write a program with a class Circle having:
   1. Private member: radius (float)
   2. A constructor to initialize radius
   3. A friend function compareTwoCircles that takes two Circle objects and prints which circle has the larger area

**Source code for this:**

#include <iostream>  
using namespace std;  
  
class Circle {  
public:  
 float radius;  
  
 Circle(float r) {  
 radius = r;  
 }  
  
 float area() {  
 return 3.14 \* radius \* radius;  
 }  
};  
  
void compare(Circle c1, Circle c2) {  
 float a1 = c1.area();  
 float a2 = c2.area();  
  
 cout << "Area of Circle 1: " << a1 << endl;  
 cout << "Area of Circle 2: " << a2 << endl;  
  
 if (a1 > a2)  
 cout << "Circle 1 is larger.\n";  
 else if (a2 > a1)  
 cout << "Circle 2 is larger.\n";  
 else  
 cout << "Both circles have the same area.\n";  
}  
  
int main() {  
 float r1, r2;  
  
 cout << "Enter radius of Circle 1: ";  
 cin >> r1;  
  
 cout << "Enter radius of Circle 2: ";  
 cin >> r2;  
  
 Circle c1(r1), c2(r2);  
 compare(c1, c2);  
  
 return 0;  
}

**Output:**

**A screenshot of a computer

AI-generated content may be incorrect.**

1. Create a program with these overloaded functions named findMax:
   1. One that finds maximum between two integers
   2. One that finds maximum between two floating-point numbers
   3. One that finds maximum among three integers
   4. One that finds maximum between an integer and a float

**Source code for this:**

#include <iostream>  
using namespace std;  
  
// Max of two integers  
int findMax(int a, int b) {  
 return (a > b) ? a : b;  
}  
  
// Max of two floats  
float findMax(float a, float b) {  
 return (a > b) ? a : b;  
}  
  
// Max of three integers  
int findMax(int a, int b, int c) {  
 return max(a, max(b, c)); // Using built-in max  
}  
  
// Max of int and float  
float findMax(int a, float b) {  
 return (a > b) ? a : b;  
}  
  
int main() {  
 int a, b, c;  
 float x, y;  
  
 cout << "Enter two integers: ";  
 cin >> a >> b;  
 cout << "Max: " << findMax(a, b) << endl;  
  
 cout << "Enter two number in floats: ";  
 cin >> x >> y;  
 cout << "Max: " << findMax(x, y) << endl;  
  
 cout << "Enter three integers: ";  
 cin >> a >> b >> c;  
 cout << "Max: " << findMax(a, b, c) << endl;  
  
 cout << "Enter one integer and one float: ";  
 cin >> a >> x;  
 cout << "Max: " << findMax(a, x) << endl;  
  
 return 0;  
}

**Output:**

**A screenshot of a computer program

AI-generated content may be incorrect.**

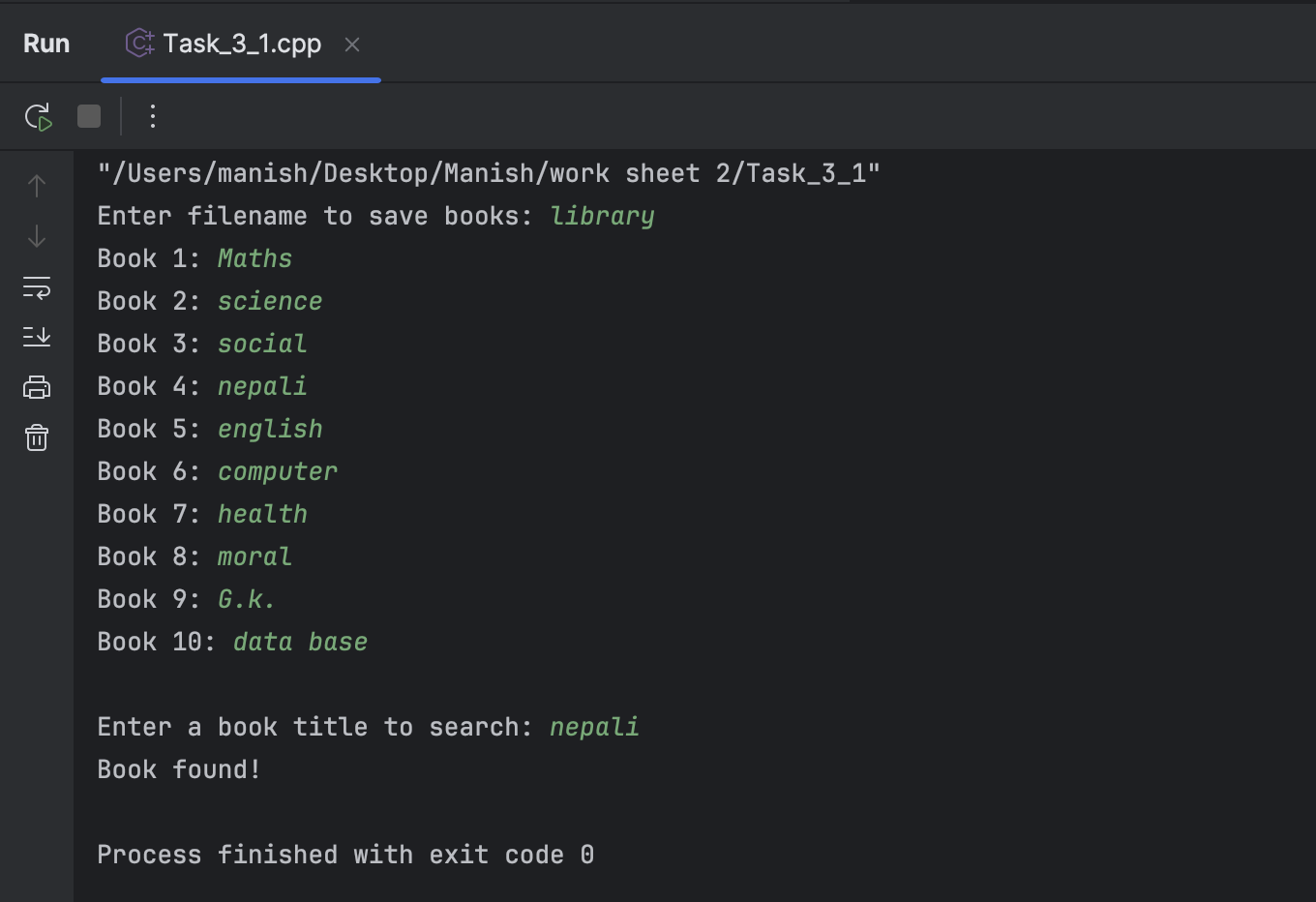
**Task 3: Basics of File Handling**

Write a program that reads the titles of 10 books (use an array of 150 characters) and writes them in a binary file selected by the user. The program should read a title and display a message to indicate if it is contained in the file or not.

**Source code for this:**

#include <iostream>  
#include <fstream>  
#include <cstring>  
using namespace std;  
  
const int MAX\_BOOKS = 10;  
const int TITLE\_SIZE = 100;  
  
  
void writeBooks(const char\* fileName, char books[MAX\_BOOKS][TITLE\_SIZE]) {  
 ofstream out(fileName, ios::*binary*);  
 if (!out) {  
 cout << "Error writing to file!" << endl;  
 return;  
 }  
 out.write((char\*)books, sizeof(char) \* MAX\_BOOKS \* TITLE\_SIZE);  
 out.close();  
}  
  
  
void readBooks(const char\* fileName, char books[MAX\_BOOKS][TITLE\_SIZE]) {  
 ifstream in(fileName, ios::*binary*);  
 if (!in) {  
 cout << "Error reading from file!" << endl;  
 return;  
 }  
 in.read((char\*)books, sizeof(char) \* MAX\_BOOKS \* TITLE\_SIZE);  
 in.close();  
}  
  
  
bool searchBook(char books[MAX\_BOOKS][TITLE\_SIZE], const char\* title) {  
 for (int i = 0; i < MAX\_BOOKS; i++) {  
 if (strcmp(books[i], title) == 0)  
 return true;  
 }  
 return false;  
}  
  
int main() {  
 char books[MAX\_BOOKS][TITLE\_SIZE];  
 char fileName[50];  
 char searchTitle[TITLE\_SIZE];  
  
 cout << "Enter filename to save books: ";  
 cin >> fileName;  
 cin.ignore();  
  
  
 for (int i = 0; i < MAX\_BOOKS; i++) {  
 cout << "Book " << i + 1 << ": ";  
 cin.getline(books[i], TITLE\_SIZE);  
 }  
  
  
 writeBooks(fileName, books);  
  
  
 cout << "\nEnter a book title to search: ";  
 cin.getline(searchTitle, TITLE\_SIZE);  
  
  
 readBooks(fileName, books);  
  
  
 if (searchBook(books, searchTitle))  
 cout << "Book found!" << endl;  
 else  
 cout << "Book not found." << endl;  
  
 return 0;  
}

**Output:**



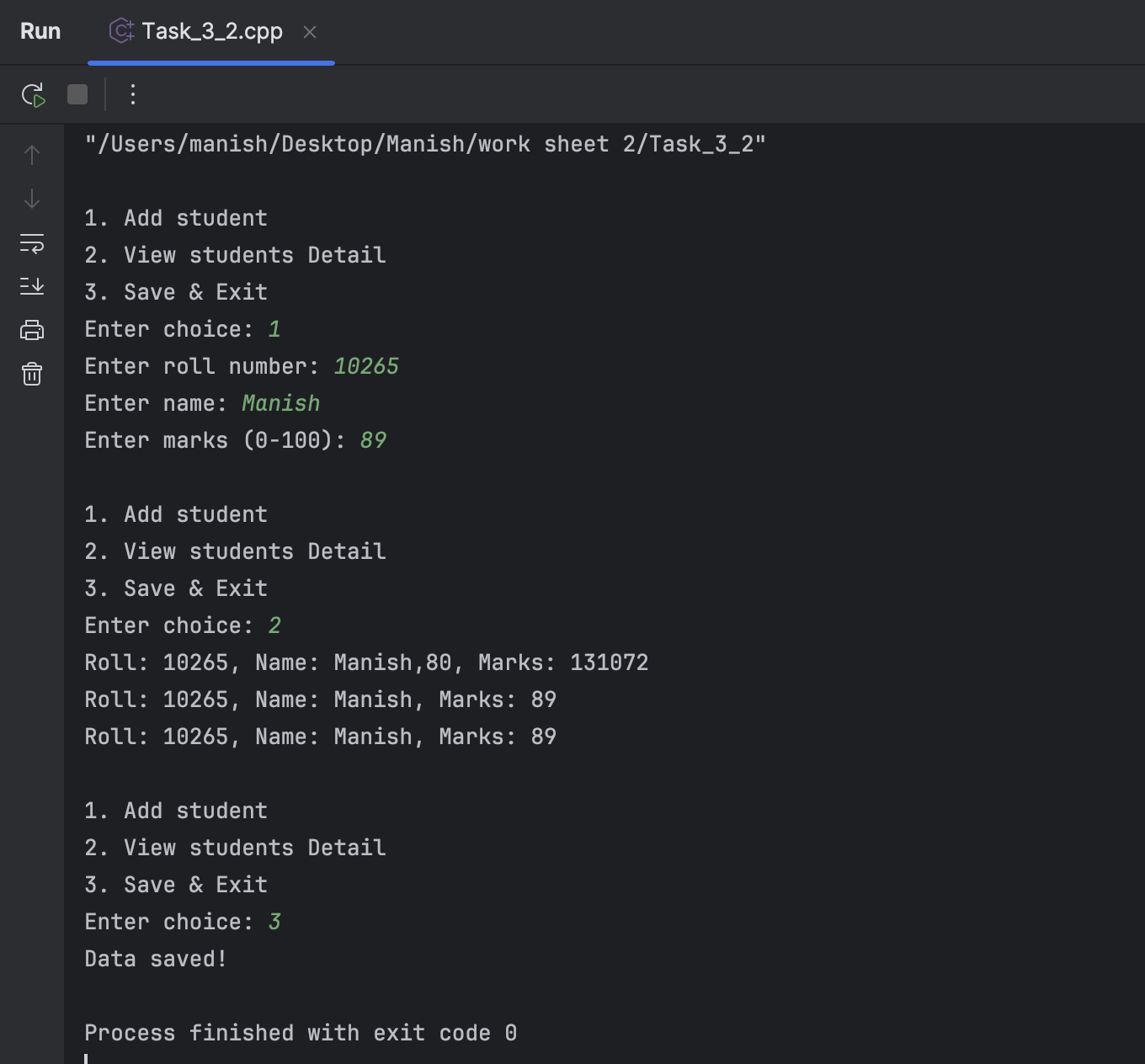
Create a program that:

1. Reads student records (roll, name, marks) from a text file
2. Throws an exception if marks are not between 0 and 100
3. Allows adding new records with proper validation
4. Saves modified records back to file

**Source code for this:**

#include <iostream>  
#include <fstream>  
#include <vector>  
using namespace std;  
  
class Student {  
public:  
 int roll;  
 string name;  
 int marks;  
  
 void input() {  
 cout << "Enter roll number: ";  
 cin >> roll;  
 cout << "Enter name: ";  
 cin.ignore(); // clear buffer  
 getline(cin, name);  
 cout << "Enter marks (0-100): ";  
 cin >> marks;  
  
 if (marks < 0 || marks > 100) {  
 cout << "Invalid marks!\n";  
 marks = 0;  
 }  
 }  
  
 void display() const {  
 cout << "Roll: " << roll << ", Name: " << name << ", Marks: " << marks << endl;  
 }  
};  
  
class StudentManager {  
 vector<Student> students;  
 string filename = "students.txt";  
  
public:  
 void load() {  
 ifstream file(filename);  
 if (!file) {  
 cout << "No existing data.\n";  
 return;  
 }  
  
 Student s;  
 while (file >> s.roll) {  
 file.ignore();  
 getline(file, s.name);  
 file >> s.marks;  
 students.push\_back(s);  
 }  
  
 file.close();  
 }  
  
 void add() {  
 Student s;  
 s.input();  
 students.push\_back(s);  
 }  
  
 void view() const {  
 if (students.empty()) {  
 cout << "No records found.\n";  
 return;  
 }  
 for (const auto& s : students)  
 s.display();  
 }  
  
 void save() {  
 ofstream file(filename);  
 for (const auto& s : students)  
 file << s.roll << "\n" << s.name << "\n" << s.marks << "\n";  
 file.close();  
 cout << "Data saved!\n";  
 }  
};  
  
int main() {  
 StudentManager manager;  
 manager.load();  
  
 int choice;  
 do {  
 cout << "\n1. Add student\n2. View students Detail\n3. Save & Exit\nEnter choice: ";  
 cin >> choice;  
  
 if (choice == 1)  
 manager.add();  
 else if (choice == 2)  
 manager.view();  
 else if (choice == 3)  
 manager.save();  
 else  
 cout << "Invalid choice!\n";  
  
 } while (choice != 3);  
  
 return 0;  
}

**Output:**

****