**Name : Abdul Rehman Hafeez**

**REG NO: BCS233127**

**ASSIGNMENT NO 3**

**Question No.1**

**Task 1: File Handling**

* **Create a class Student with data members: name, roll\_number, and marks.**
* **Implement member functions to input and display student data.**
* **Create a program to:**
* **Create a file named students.txt.**
* **Write data of 5 students to the file using ofstream.**
* **Read the data from the file and display it using ifstream.**
* **Modify the marks of a specific student using fstream.**

#include <iostream>

#include <fstream>

#include <string>

#include <vector>

class Student {

public:

std::string name;

int roll\_number;

float marks;

void input() {

std::cout << "Enter name: ";

std::cin >> name;

std::cout << "Enter roll number: ";

std::cin >> roll\_number;

std::cout << "Enter marks: ";

std::cin >> marks;

}

void display() const {

std::cout << "Name: " << name << ", Roll Number: " << roll\_number << ", Marks: " << marks << std::endl;

}

};

void writeToFile(const std::vector<Student>& students) {

std::ofstream outFile("students.txt");

for (const auto& student : students) {

outFile << student.name << " " << student.roll\_number << " " << student.marks << std::endl;

}

outFile.close();

}

void readFromFile() {

std::ifstream inFile("students.txt");

std::string name;

int roll\_number;

float marks;

while (inFile >> name >> roll\_number >> marks) {

std::cout << "Name: " << name << ", Roll Number: " << roll\_number << ", Marks: " << marks << std::endl;

}

inFile.close();

}

void modifyMarks(int roll\_number, float new\_marks) {

std::fstream file("students.txt", std::ios::in | std::ios::out);

std::string name;

int r\_num;

float marks;

std::streampos pos;

while (file >> name >> r\_num >> marks) {

if (r\_num == roll\_number) {

pos = file.tellg();

file.seekp(pos - std::streamoff(sizeof(float)));

file << new\_marks << std::endl;

break;

}

}

file.close();

}

int main() {

std::vector<Student> students(5);

for (auto& student : students) {

student.input();

}

writeToFile(students);

std::cout << "\nData read from file:\n";

readFromFile();

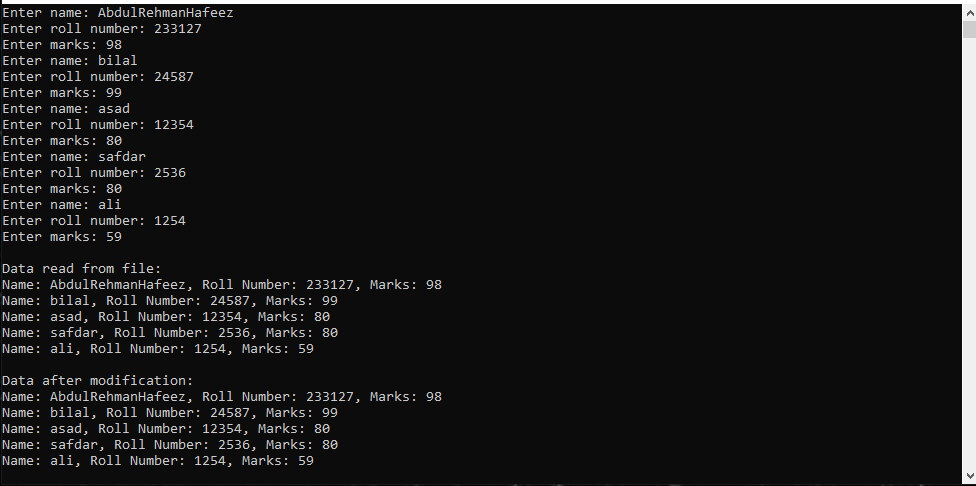
modifyMarks(1, 95.0);

std::cout << "\nData after modification:\n";

readFromFile();

return 0;

}

****

**Task 2: Inheritance**

**Part A: Single Inheritance**

* Create a base class Shape with data member color and member functions to set and get
* color.
* Create a derived class Rectangle inheriting Shape with data members length and
* breadth.
* Implement member functions to calculate area and perimeter of the rectangle.
* Demonstrate constructor chaining in both base and derived classes.
* #include <iostream>
* #include <string>

class Shape {

protected:

std::string color;

public:

Shape(const std::string& c) : color(c) {}

void setColor(const std::string& c) {

color = c;

}

std::string getColor() const {

return color;

}

};

class Rectangle : public Shape {

private:

double length;

double breadth;

public:

Rectangle(const std::string& c, double l, double b) : Shape(c), length(l), breadth(b) {}

double area() const {

return length \* breadth;

}

double perimeter() const {

return 2 \* (length + breadth);

}

};

int main() {

Rectangle rect("Red", 5.0, 3.0);

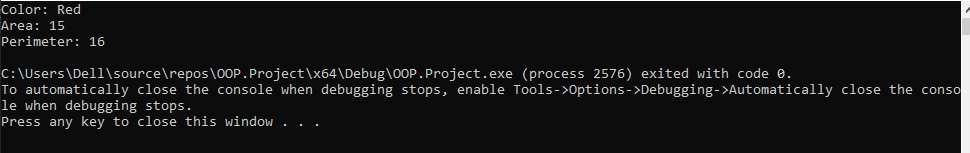
std::cout << "Color: " << rect.getColor() << std::endl;

std::cout << "Area: " << rect.area() << std::endl;

std::cout << "Perimeter: " << rect.perimeter() << std::endl;

return 0;

}

****

**Multiline inheritence**

* Create a base class Animal with data member name.
* Create a derived class Mammal inheriting Animal with data member numberOfLegs.
* Create a derived class Dog inheriting Mammal with member functions to bark.
* Demonstrate object creation and access to members of all classes.

#include <iostream>

#include <string>

class Animal {

protected:

std::string name;

public:

Animal(std::string n) : name(n) {}

void display() {

std::cout << "Animal Name: " << name << std::endl;

}

};

class Mammal : public Animal {

protected:

int numberOfLegs;

public:

Mammal(std::string n, int legs) : Animal(n), numberOfLegs(legs) {}

void display() {

Animal::display();

std::cout << "Number of Legs: " << numberOfLegs << std::endl;

}

};

class Dog : public Mammal {

public:

Dog(std::string n, int legs) : Mammal(n, legs) {}

void bark() {

std::cout << name << " says: Woof!" << std::endl;

}

};

int main() {

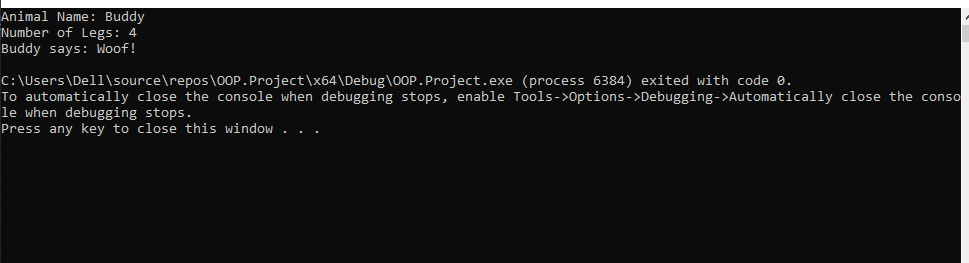
Dog myDog("Buddy", 4);

myDog.display();

myDog.bark();

return 0;

}



**Part C: Multiple Inheritance**

* Create a base class Person with data members name and address.
* Create another base class Employee with data members employeeID and salary.
* Create a derived class Teacher inheriting both Person and Employee with additional
* data member subject.
* Demonstrate object creation and access to members of all classes.
* Discuss the ambiguity problem in multiple inheritance and how to resolve it.

#include <iostream>

#include <string>

class Person {

public:

std::string name;

std::string address;

Person(std::string n, std::string a) : name(n), address(a) {}

};

class Employee {

public:

int employeeID;

double salary;

Employee(int id, double sal) : employeeID(id), salary(sal) {}

};

class Teacher : public Person, public Employee {

public:

std::string subject;

Teacher(std::string n, std::string a, int id, double sal, std::string sub)

: Person(n, a), Employee(id, sal), subject(sub) {}

};

int main() {

Teacher teacher("HASSAN", " St 1A", 101, 50000.0, "Mathematics");

std::cout << "Name: " << teacher.name << "\n"

<< "Address: " << teacher.address << "\n"

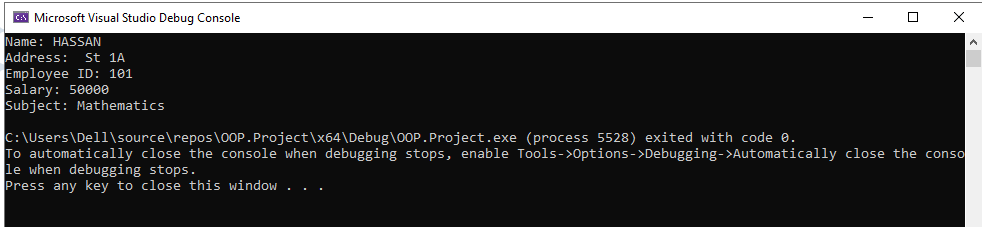
<< "Employee ID: " << teacher.employeeID << "\n"

<< "Salary: " << teacher.salary << "\n"

<< "Subject: " << teacher.subject << std::endl;

return 0;

}

****