

Name: Adil Khan

Department: BS AI

Roll no: 2330-0030

Subject: Object Oriented Programming

Teacher: Mam Saba Awan

**Lab # 04**

Task no 1:

#include <iostream>

using namespace std;

class Polygon {

private:

float length;

float\* width;

public:

Polygon(float l, float w) {

length = l;

width = new float(w);

}

Polygon(const Polygon& obj) {

length = obj.length;

width = obj.width;

}

void display() {

cout << "Length: " << length << ", Width: " << \*width << endl;

}

void setWidth(float w) {

\*width = w;

}

~Polygon() {

delete width;

}

};

int main() {

cout << "Shallow Copy Demonstration" << endl;

Polygon one(10.5, 20.5);

Polygon two = one;

one.display();

two.display();

one.setWidth(30.5);

cout << "After modifying 'one' width:" << endl;

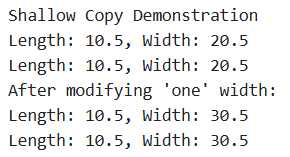
one.display();

two.display();

return 0;

}

Output:



Task no 2:

#include <iostream>

using namespace std;

class Student {

private:

int ID;

float\* grades;

int numGrades;

public:

Student(int id, int num) {

ID = id;

numGrades = num;

grades = new float[numGrades];

}

Student(const Student& obj) {

ID = obj.ID;

numGrades = obj.numGrades;

grades = new float[numGrades];

for (int i = 0; i < numGrades; i++) {

grades[i] = obj.grades[i];

}

}

void setGrades(float g[]) {

for (int i = 0; i < numGrades; i++) {

grades[i] = g[i];

}

}

void modifyGrade(int index, float newGrade) {

if (index >= 0 && index < numGrades) {

grades[index] = newGrade;

} else {

cout << "Invalid index!\n";

}

}

void display() {

cout << "Student ID: " << ID << ", Grades: ";

for (int i = 0; i < numGrades; i++) {

cout << grades[i] << " ";

}

cout << endl;

}

~Student() {

delete[] grades;

}

};

int main() {

int numGrades = 3;

float gradesArray[3] = {85.5, 90.0, 78.2};

Student student1(101, numGrades);

student1.setGrades(gradesArray);

Student student2 = student1;

cout << "\nBefore modifying student1:\n";

student1.display();

student2.display();

student1.modifyGrade(0, 99.9);

cout << "\nAfter modifying student1's first grade:\n";

student1.display();

student2.display();

return 0;

}

Output:

