ПРИЛОЖЕНИЕ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М.А. БОНЧ-БРУЕВИЧА»

(СПбГУТ)

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Кафедра Систем обработки данных

**Дисциплина «Технологии программирования»**

**ОТЧЕТ**

**за практическое занятие №4**

**Тема: Взаимодействие функций в программах**

Выполнил

Студент 2 курса, гр. ИБ-32вп

\_\_\_\_\_\_\_\_\_\_\_\_\_ Братишкин Д.Е.

Принял

Доцент кафедры БИС

\_\_\_\_\_\_\_\_\_\_\_\_\_ Медведев В.А.

Санкт-Петербург

2024

Тема: **Проектирование классов и создание объектов**

Цель: Получение практических навыков разработки классов, создания объектов и использования дружественных функций для обработки данных.  
  
Задание  
  
**Вариант А.**

**Задание 1**. Спроектировать класс **point,** обозначающий координаты точки на плоскости x, y (закрытые переменные), и функции доступа к закрытым переменным, а также конструктор и деструктор.

Динамически выделить память под массив из 4-х объектов так, чтобы они соответствовали точкам вершин прямоугольника. Разработать функцию, вычисляющую длины сторон и гипотенузы.

**Задание 2**. В проект добавить дружественную функцию, вычисляющую площадь прямоугольника, заданного точками его вершин.

**Задание 3.** Спроектировать класс в соответствии с *индивидуальным заданием* (см. таблица 6). Создать массив из 6-ти объектов спроектированного класса. Рассчитать *Вычисляемый показатель*.

Интерфейс программы оформить в виде меню.

Код программы

#include <iostream>

#include <cmath>

using namespace std;

// Задание 1: Класс для представления точки на плоскости

class Point {

private:

    float x, y; // Закрытые переменные для координат

public:

    // Конструктор с параметрами для инициализации координат

    Point(float xCoord, float yCoord) : x(xCoord), y(yCoord) {}

    // Функции доступа к закрытым переменным

    float getX() const { return x; }

    float getY() const { return y; }

    // Деструктор, сообщающий о разрушении объекта

    ~Point() {

        cout << "Object at (" << x << ", " << y << ") is destroyed." << endl;

    }

    // Статический метод для вычисления расстояния между двумя точками

    static float distance(const Point &p1, const Point &p2) {

        return sqrt(pow(p2.x - p1.x, 2) + pow(p2.y - p1.y, 2));

    }

};

// Дружественная функция для вычисления площади прямоугольника

float calculateRectangleArea(Point\* points) {

    float length = Point::distance(points[0], points[1]);

    float width = Point::distance(points[1], points[2]);

    return length \* width; // Площадь = длина \* ширина

}

// Задание 2: Класс для представления локальной сети

class LocalNetwork {

private:

    int numberOfDevices; // Количество устройств в сети

    float installationCostPerDevice; // Стоимость установки на одно устройство

public:

    // Конструктор для инициализации количества устройств и стоимости установки

    LocalNetwork(int devices, float cost) : numberOfDevices(devices), installationCostPerDevice(cost) {}

    // Метод для расчета минимальной стоимости установки

    float calculateMinimumInstallationCost() const {

        return numberOfDevices \* installationCostPerDevice;

    }

    // Деструктор, сообщающий о разрушении объекта

    ~LocalNetwork() {

        cout << "LocalNetwork object destroyed." << endl;

    }

};

int main() {

    // Задание 1: Создание массива из 4 объектов класса Point

    Point\* rectangleVertices = new Point[4]{

        Point(0.0, 0.0), // Нижний левый угол

        Point(0.0, 4.0), // Верхний левый угол

        Point(3.0, 4.0), // Верхний правый угол

        Point(3.0, 0.0)  // Нижний правый угол

    };

    // Вычисление площади прямоугольника

    float area = calculateRectangleArea(rectangleVertices);

    cout << "Area of the rectangle: " << area << endl;

    // Освобождение памяти

    delete[] rectangleVertices;

    // Задание 2: Создание локальной сети

    LocalNetwork network(10, 150.0);

    // Расчет минимальной стоимости установки

    float installationCost = network.calculateMinimumInstallationCost();

    cout << "Minimum installation cost: " << installationCost << endl;

    return 0; // Завершение программы

}

Результаты работы программы

Задание 1

- Ввод: Создание массива объектов `rectangleVertices`, представляющих вершины прямоугольника.

- Вывод: Площадь прямоугольника: 12.0

Задание 2

- Ввод: Создание объекта `network` с 10 устройствами и стоимостью монтажа 150.0 за устройство.

- Вывод: Минимальная стоимость монтажа: 1500.0

Выводы по разработанным элементам программы

Программа успешно выполняет все поставленные задачи. В первом задании она корректно создает объекты класса `Point`, инициализирует их координаты, вычисляет площадь прямоугольника с использованием дружественной функции и выводит результат. Во втором задании программа создает объекты класса `LocalNetwork`, вычисляет минимальную стоимость монтажа и выводит результат.

Код структурирован в виде классов с конструкторами, функциями доступа и деструкторами, что делает его более понятным и легким для отладки. Обработка ошибок и ввод данных пользователем выполнены корректно, что повышает надежность и удобство использования программы.