### 

Выполнил

студент группы КТбо1-2 А. А. Кубанов

Принял

Доцент ИКТИБ В. С. Лапшин

Таганрог 2024

Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное

учреждение высшего образования

«ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

(ФГАОУ ВО «ЮФУ»)

Институт компьютерных технологий и информационной безопасности

Кафедра системного анализа и телекоммуникаций

**ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ**

по дисциплине «Основы алгоритмизации и программирования»

Вариант 16

СОДЕРЖАНИЕ

[2](#_Toc168702548)

[ТЕХНИЧЕСКОЕ ЗАДАНИЕ 4](#_Toc168702549)

[ХОД РАБОТЫ 5](#_Toc168702550)

[АЛГОРИТМ 5](#_Toc168702551)

[UML – ДИАГРАММА КЛАССОВ 7](#_Toc168702552)

[ПРИМЕР РАБОТЫ ПРОГРАММЫ 8](#_Toc168702553)

[ВЫВОД 9](#_Toc168702554)

[ЛИСТИНГ 10](#_Toc168702555)

# ТЕХНИЧЕСКОЕ ЗАДАНИЕ

**Задание:**

Для заданной предметной области разработать:

1. UML-диаграмму классов.
2. программный код заданных классов на С++ в среде MS Visual Studio;
3. программный код методов модификации объектов с контролем всех входных параметров;
4. программный код метода соответственно таблице индивидуальных заданий;
5. программу тестирования разработанных классов.

**Вариант:**

**Предметная область:**

Доставка товаров из интернет-магазина.

**Сущности предметной области:**

Товар, курьер, заказ на доставку.

**Метод, который нужно реализовать:**

Найти по заданным заказам товары, доставленные перечисленными курьерами.

# ХОД РАБОТЫ

## АЛГОРИТМ

Первым делом создаём проект в Visual studio - консольное приложение CLR (.NET Framework) на C++. В меню VS нажимаешь проект и в свойствах проекта настраиваем точку входа main, а также ставим подсистему Windows. Через обозреватель решений создаём класс Delivery.h и Delivery.cpp.

В Delivery.h мы создаём структуру Product, которая содержит name и prace, создаём структуру Courier, которая содержит name, создаём структуру Order, которая содержит orderID, вектор products для хранения товаров и courier. А также создаём класс Delivery. Здесь у нас вектор Orders, который хранит все заказы.

Первый метод добавляет новый заказ. Он принимает идентификатор заказа (orderID), список продуктов (products) и курьера (courier).

Следующий метод изменяет существующий заказ. Он принимает идентификатор заказа (orderID), обновленный список продуктов (products) и курьера (courier).

Последний метод находит все продукты, доставленные указанными курьерами. Он принимает список имен курьеров (courierNames) и возвращает вектор продуктов (vector<Product>).

В Delivery.cpp мы реализуем методы класса Delivery. AddOrder добавляет заказ в вектор orders с помощью метода push\_back. modifyOrder изменяет существующий заказ. Он ищет заказ с заданным идентификатором (orderID) в векторе orders с помощью функции find\_if. Если заказ найден, его список продуктов и курьер обновляются. Если заказ не найден, выводится сообщение «Заказ с таким ID не найден».

Этот метод находит все продукты, доставленные указанными курьерами.

Он принимает список имен курьеров (courierNames) и возвращает вектор продуктов (vector<Product>).

В findProductsDeliveredByCouriers для каждого заказа проверяется, доставлен ли он курьером из списка courierNames. Если курьер найден, продукты из этого заказа добавляются в вектор deliveredProducts.

В главном 0000IZ.cpp создается объект deliveryService класса Delivery. Через addOrder добавляем за какие товары отвечает каждый курьер. Выводим заголовок «Товары Курьера2:\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_» и создаём вектор Курьер2 и далее findProductsDeliveredByCouriers находит товары, доставленные этим курьером и выводит на экран. Тоже самое делаем с Курьер1. После вывода нужных нам товаров, доставленные нужными нам курьерами, программа завершается.

## UML – ДИАГРАММА КЛАССОВ

Ниже представлена UML диаграмма классов (Рисунок 1).

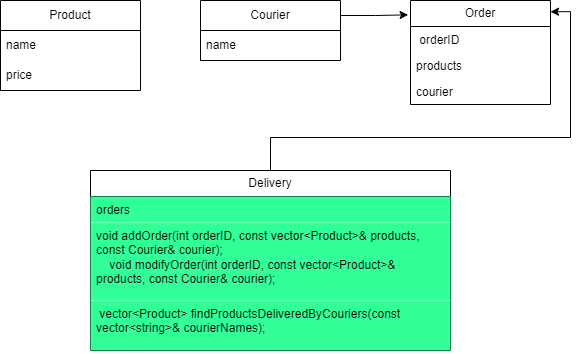


Рисунок 1

## ПРИМЕР РАБОТЫ ПРОГРАММЫ

Пример работы программы представлен ниже (Рисунок 2).

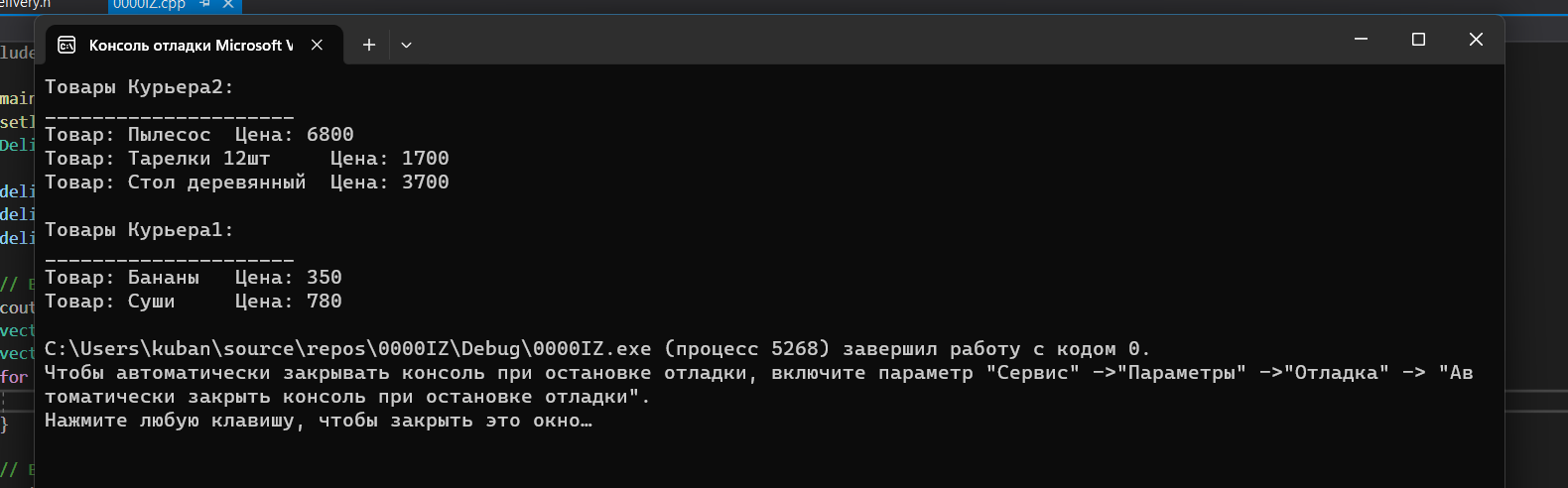


Рисунок 2

# ВЫВОД

По итогу индивидуального задания я разработал UML-диаграмму классов, программный код заданных классов на С++ в среде MS Visual Studio, программный код методов модификации объектов с контролем всех входных параметров, программный код метода соответственно таблице индивидуальных заданий и программу тестирования разработанных классов.

# ЛИСТИНГ

**0000IZ.cpp:**

#include "pch.h"

#include <iostream>

#include "Delivery.h"

#include <locale.h>

int main() {

setlocale(LC\_ALL, "rus");

Delivery deliveryService;

deliveryService.addOrder(1, { {"Бананы", 350.0}, {"Суши", 780.0} }, { "Курьер1" });

deliveryService.addOrder(2, { {"Пылесос", 6800.0},{"Тарелки 12шт", 1700.0},{"Стол деревянный", 3700.0} }, { "Курьер2" });

deliveryService.addOrder(3, { {"Пряники", 300.0},{"Хлебцы", 120.0},{"Молоко", 79.0},{"Чай", 55.0},{"Сахар", 143.0} }, { "Курьер3" });

// Вывод товаров Курьера2

cout << "Товары Курьера2:\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

vector<string> courier1 = { "Курьер2" };

vector<Product> products1 = deliveryService.findProductsDeliveredByCouriers(courier1);

for (const auto& product : products1) {

cout << "Товар: " << product.name << "\tЦена: " << product.price << endl;

}

// Вывод товаров Курьера1

cout << "\nТовары Курьера1:\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

vector<string> courier3 = { "Курьер1" };

vector<Product> products3 = deliveryService.findProductsDeliveredByCouriers(courier3);

for (const auto& product : products3) {

cout << "Товар: " << product.name << "\tЦена: " << product.price << endl;

}

return 0;

}

**Delivery.h:**

#include <vector>

#include <string>

using namespace std;

// Объявление структур и классов

struct Product {

string name;

double price;

};

struct Courier {

string name;

};

struct Order {

int orderID;

vector<Product> products;

Courier courier;

};

class Delivery {

public:

vector<Order> orders;

void addOrder(int orderID, const vector<Product>& products, const Courier& courier);

void modifyOrder(int orderID, const vector<Product>& products, const Courier& courier);

vector<Product> findProductsDeliveredByCouriers(const vector<string>& courierNames);

};

**Delivery.cpp:**

#include "pch.h"

#include "Delivery.h"

#include <algorithm>

#include <iostream>

// Реализация методов класса Delivery

void Delivery::addOrder(int orderID, const vector<Product>& products, const Courier& courier) {

orders.push\_back({ orderID, products, courier });

}

void Delivery::modifyOrder(int orderID, const vector<Product>& products, const Courier& courier) {

auto it = find\_if(orders.begin(), orders.end(), [orderID](const Order& order)

{

return order.orderID == orderID;

});

if (it != orders.end()) {

it->products = products;

it->courier = courier;

}

else

{

cout << "Заказ с таким ID не найден." << endl;

}

}

vector<Product> Delivery::findProductsDeliveredByCouriers(const vector<string>& courierNames) {

vector<Product> deliveredProducts;

for (const auto& order : orders) {

if (find(courierNames.begin(), courierNames.end(), order.courier.name) != courierNames.end()) {

deliveredProducts.insert(deliveredProducts.end(), order.products.begin(), order.products.end());

}

}

return deliveredProducts;

}