САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ ИНФОРМАТИКИ

Отчет по лабораторной работе №1

по курсу «объектно-ориентированное программирование»

Тема: Разработать консольное приложение

Выполнила:
Буй Тхук Хуен - К3239
Проверил:
Слюсаренко С. В.

Санкт-Петербург
2025 г.

Содержание отчета

I. Задача

Нужно разработать консольное приложение, которое будет эмулировать вендинговый автомат, позволяющее пользователю:

- Посмотреть список доступных товаров с их ценами и количеством.
- Вставить монеты разных номиналов.
- Выбрать товар и получить его, если внесённой суммы достаточно.
- Получить сдачу (если нужно) и вернуть неиспользованные монеты при отмене операции.
- Администраторский режим для пополнения ассортимента и сбора собранных средств.

1. Листинг кода

```
using System;
using System.Collections.Generic;

class Product
{
    public string Name { get; set; } = "";
    public int Price { get; set; }
    public int Quantity { get; set; }
}

public class VendingMachine
{
    private List<Product> products = new List<Product>();
    private int balance = 0;
    private int collectedMoney = 0;
    private const string adminPassword = "1234";

    public VendingMachine()
    {
          products.Add(new Product { Name = "Сырок", Price = 50, Quantity = 5
});
          products.Add(new Product { Name = "Чипсы", Price = 100, Quantity = 5
});
          products.Add(new Product { Name = "Вода", Price = 75, Quantity = 10
});
          products.Add(new Product { Name = "Сэндвич", Price = 180, Quantity =
5 });
```

```
        products.Add(new Product { Name = "Вафли", Price = 80, Quantity = 5
});
        products.Add(new Product { Name = "Сухарики", Price = 80, Quantity =
5 });
    }

    public void ShowProducts()
    {
        Console.WriteLine("\nСписок товаров:");
        for (int i = 0; i < products.Count; i++)
        {
                            Console.WriteLine($"{i+1}.  {products[i].Name}  -
{products[i].Price} руб. (осталось {products[i].Quantity})");
        }
    }

    public void InsertCoin(int coin)
    {
        balance += coin;
          Console.WriteLine($"Внесено: {coin} руб. Текущий баланс: {balance}
руб.");
    }

    public void SelectProduct(int index)
    {
        var product = products[index - 1];
        if (product.Quantity <= 0)
        {
            Console.WriteLine("Товар закончился!");
            return;
        }

        if (balance >= product.Price)
        {
            balance -= product.Price;
            collectedMoney += product.Price;
            product.Quantity--;
            Console.WriteLine($"Вы купили: {product.Name}");
        }
        else
        {
          Console.WriteLine($"Недостаточно средств. Требуется {product.Price}
рублей.");
        }
```

```
    }

    public void CancelOperation()
    {
        if (balance > 0)
        {
            Console.WriteLine($"Операция отменена. Возврат {balance} руб.");
            balance = 0;
        }
        else
        {
            Console.WriteLine("Баланс пуст.");
        }
    }

    public void AdminMode()
    {
        Console.Write("Введите пароль администратора: ");
        string pass = Console.ReadLine();

        if (pass != adminPassword)
        {
            Console.WriteLine("Неверный пароль!");
            return;
        }

        Console.WriteLine("=== Админ-меню ===");
        Console.WriteLine("1. Пополнить товары");
        Console.WriteLine("2. Собрать деньги");
        string choice = Console.ReadLine();

        if (choice == "1")
        {
            ShowProducts();
            Console.Write("Введите номер товара: ");
            string? inputId = Console.ReadLine();
            if (inputId != null && int.TryParse(inputId, out int id))
            {
                Console.Write("Введите количество для добавления: ");
                string? inputQty = Console.ReadLine();
                if (inputQty != null && int.TryParse(inputQty, out int qty))
                {
                    products[id - 1].Quantity += qty;
                    Console.WriteLine("Товар пополнен!");
```

```
                }
                else
                {
                    Console.WriteLine("Ошибка ввода количества!");
                }
            }
            else
            {
                Console.WriteLine("Ошибка ввода номера товара!");
            }

        }
        else if (choice == "2")
        {
            Console.WriteLine($"Собрано {collectedMoney} руб.");
            collectedMoney = 0;
        }
    }
        // Возвращает текущую сумму
    public int GetBalance() => balance;

        //Возвращает общую сумму, собранную автоматом
    public int GetCollectedMoney() => collectedMoney;

        // Возвращает количество товаров по индексу
    public int GetProductQuantity(int index) => products[index - 1].Quantity;

        // Возвращает цену товаров по индексу
    public int GetProductPrice(int index) => products[index - 1].Price;

        // Помощник администратора: добавить товары
    public void AdminAddProduct(int index, int qty)
    {
        products[index - 1].Quantity += qty;
    }

        // Помощник администратора: собрать деньги
    public int AdminCollectMoney()
    {
        int money = collectedMoney;
        collectedMoney = 0;
        return money;
    }
}
```

```
class Program
{
    static void Main()
    {
        VendingMachine vm = new VendingMachine();
        while (true)
        {
            Console.WriteLine("\n=== Вендинговый автомат ===");
            Console.WriteLine("1. Показать товары");
            Console.WriteLine("2. Внести монету");
            Console.WriteLine("3. Выбрать товар");
            Console.WriteLine("4. Отменить операцию");
            Console.WriteLine("5. Админ-режим");
            Console.WriteLine("0. Выход");
            Console.Write("Выберите действие: ");

            string choice = Console.ReadLine();

            switch (choice)
            {
                case "1":
                    vm.ShowProducts();
                    break;
                case "2":
                    Console.Write("Введите номинал монеты: ");
                    int coin = int.Parse(Console.ReadLine());
                    vm.InsertCoin(coin);
                    break;
                case "3":
                    Console.Write("Введите номер товара: ");
                    int index = int.Parse(Console.ReadLine());
                    vm.SelectProduct(index);
                    break;
                case "4":
                    vm.CancelOperation();
                    break;
                case "5":
                    vm.AdminMode();
                    break;
                case "0":
                    return;
                default:
                    Console.WriteLine("Неверный ввод!");
```

```
                    break;
                }
            }
        }
    }
```

- Текстовое объяснение решения
  + Программа моделирует **работу вендингового автомата** (автомата по продаже товаров).

    Пользователь может:
    - посмотреть список товаров,
    - внести монеты,
    - купить товар,
    - отменить операцию,
    - войти в **админ-режим**, чтобы пополнить запасы или забрать деньги.
  + Класс Product: Это простой объект, который хранит информацию о каждом продукте в автомате.

    Name — название товара (строка).

    Price — цена товара в рублях (целое число).

    Quantity — количество оставшихся единиц товара.
  + Класс VendingMachine: Этот класс отвечает за всю логику автомата.

    products — список всех товаров.

    balance — текущие деньги, внесённые пользователем.

    collectedMoney — общая сумма, собранная автоматом.

    adminPassword — пароль для входа в админ-режим.

  + Метод ShowProducts(): Выводит список всех товаров и их состояние:
  + Метод InsertCoin(int coin): Добавляет введённые деньги в баланс пользователя
  + Метод SelectProduct(int index): Логика покупки товара:
    1. Проверяет, есть ли товар в наличии.
    2. Проверяет, достаточно ли денег.
    3. Если да — уменьшает количество товара, списывает деньги и сообщает о покупке.

+ Метод CancelOperation(): Позволяет пользователю отменить операцию и вернуть внесённые деньги.
+ Метод AdminMode(): Режим администратора:
    - Запрашивает пароль.
    - Если пароль правильный — открывает меню администратора:
        1. Пополнить товары
        2. Собрать деньги

    Админ может выбрать действие, пополнить запасы или обнулить кассу, собрав выручку.

+ Вспомогательные методы: Эти методы возвращают или изменяют внутренние данные автомата — в основном нужны для админа или тестирования.

```
public int GetBalance() => balance;
public int GetCollectedMoney() => collectedMoney;
public int GetProductQuantity(int index) => products[index - 1].Quantity;
public int GetProductPrice(int index) => products[index - 1].Price;
public void AdminAddProduct(int index, int qty) { ... }
public int AdminCollectMoney() { ... }
```

+ Класс Program: Пользователь вводит номер команды, и программа выполняет соответствующий метод автомата через switch.

- Результат работы код

```
=== Вендинговый автомат ===
1. Показать товары
2. Внести монету
3. Выбрать товар
4. Отменить операцию
5. Админ-режим
0. Выход
Выберите действие: 1

Список товаров:
1. Сырок - 50 руб. (осталось 5)
2. Чипсы - 100 руб. (осталось 5)
3. Вода - 75 руб. (осталось 10)
4. Сэндвич - 180 руб. (осталось 5)
5. Вафли - 80 руб. (осталось 5)
6. Сухарики - 80 руб. (осталось 5)

=== Вендинговый автомат ===
1. Показать товары
2. Внести монету
3. Выбрать товар
4. Отменить операцию
5. Админ-режим
0. Выход
Выберите действие: 2
Введите номинал монеты: 200
Внесено: 200 руб. Текущий баланс: 200 руб.
```

```
=== Вендинговый автомат ===
1. Показать товары
2. Внести монету
3. Выбрать товар
4. Отменить операцию
5. Админ-режим
0. Выход
Выберите действие: 3
Введите номер товара: 4
Вы купили: Сэндвич

=== Вендинговый автомат ===
1. Показать товары
2. Внести монету
3. Выбрать товар
4. Отменить операцию
5. Админ-режим
0. Выход
Выберите действие: 3
Введите номер товара: 2
Недостаточно средств. Требуется 100 рублей.
```

```
=== Вендинговый автомат ===
1. Показать товары
2. Внести монету
3. Выбрать товар
4. Отменить операцию
5. Админ-режим
0. Выход
Выберите действие: 2
Введите номинал монеты: 100
Внесено: 100 руб. Текущий баланс: 120 руб.

=== Вендинговый автомат ===
1. Показать товары
2. Внести монету
3. Выбрать товар
4. Отменить операцию
5. Админ-режим
0. Выход
Выберите действие: 4
Операция отменена. Возврат 120 руб.
```

```
=== Вендинговый автомат ===
1. Показать товары
2. Внести монету
3. Выбрать товар
4. Отменить операцию
5. Админ-режим
0. Выход
Выберите действие: 5
Введите пароль администратора: 1234
=== Админ-меню ===
1. Пополнить товары
2. Собрать деньги
1

Список товаров:
1. Сырок - 50 руб. (осталось 5)
2. Чипсы - 100 руб. (осталось 5)
3. Вода - 75 руб. (осталось 10)
4. Сэндвич - 180 руб. (осталось 4)
5. Вафли - 80 руб. (осталось 5)
6. Сухарики - 80 руб. (осталось 5)
Введите номер товара: 5
Введите количество для добавления: 10
Товар пополнен!
```

```
=== Вендинговый автомат ===
1. Показать товары
2. Внести монету
3. Выбрать товар
4. Отменить операцию
5. Админ-режим
0. Выход
Выберите действие: 5
Введите пароль администратора: 1234
=== Админ-меню ===
1. Пополнить товары
2. Собрать деньги
2
Собрано 180 руб.

=== Вендинговый автомат ===
1. Показать товары
2. Внести монету
3. Выбрать товар
4. Отменить операцию
5. Админ-режим
0. Выход
Выберите действие: 0
PS D:\OOP\OOP2025\Lab1\src>
```

- Test

```
[xUnit.net 00:00:01.47]   Finished:    VendingMachineTest
  VendingMachineTest test succeeded (5.1s)

Test summary: total: 10, failed: 0, succeeded: 10, skipped: 0, duration: 5.0s
Build succeeded in 7.1s
```

II. Вывод

В ходе выполнения лабораторной работы была разработана и протестирована программа — модель вендингового автомата на языке C# с использованием объектно-ориентированного подхода.

Были реализованы следующие возможности:

- создание класса **Product** для хранения информации о товарах (название, цена, количество);
- разработка класса **VendingMachine**, реализующего основные функции автомата: отображение списка товаров, приём монет, покупку товаров, отмену операции и работу в режиме администратора;
- добавлен администраторский режим с возможностью пополнения запасов и сбора выручки;
- проведено тестирование с помощью библиотеки **xUnit**, подтвердившее корректность работы всех основных методов.

В результате работы получена консольная программа, демонстрирующая принципы инкапсуляции, работы с коллекциями, а также основы тестирования программных модулей.