**Отчет**

**Лабораторная работа №7**

Задание:

Вариант 10

1. Разработать класс «Товар», содержащий информацию:

▪ название товара,

▪ название магазина, в котором продается товар,

▪ стоимость товара в рублях.

***2. Для разрабатываемого класса необходимо выполнить:*** 1) Описание полей класса.

2) Конструктор без параметров, конструктор с параметрами, конструкторы копирования, деструктор.

3) Перегрузку одного из бинарных операторов как метод класса.

4) Перегрузку одного из бинарных операторов через дружественную функцию.

5) Перегрузку одного из операторов отношения как метод класса.

6) Перегрузку одного из операторов отношения через дружественную функцию.

7) Перегрузку оператора присваивания.

8) Перегрузку префиксного инкремента как метод класса.

9) Перегрузку постфиксного инкремента через дружественную функцию.

10) Перегрузку операторов << и >> через дружественные функции.

3. Создать массив из объектов разработанного класса. Ввести в массив данные из текстового файла. Имя файла со входными данными должны вводиться из потока cin.

4. Определить самый дорогой товар. Для сравнения использовать перегруженный оператор отношения, поиск элемента выполнить в отдельной процедуре.

5. Выполнить сортировку массива методом простого выбора по стоимости товара. Для сортировки написать и использовать шаблон функции. Результат вывести в файл в виде таблицы, сделать заголовки колонок таблицы.

6. Создать список магазинов. Результат вывести в файл в виде таблицы, сделать заголовки колонок таблицы

7. Предусмотреть обработку и инициализацию исключительных ситуаций, связанных, например, с проверкой значения полей перед инициализацией и присваиванием

8. Программа должна быть написана в соответствии со стандартом программирования: C++ Programming Style Guidelines (<http://geosoft.no/development/cppstyle.html>).

**Требования**

* Пробелы между данными
* Не допускается ввод пустых строчек

**Спецификация**

**Класс Product**

Поля:

float productCost\_; // цена товара

MyString productName\_; // имя товара

MyString marketName\_; // название магазина с товаром

Конструкторы:

Product();//Конструктор без параметров

Product(const MyString& productName, const MyString& marketName, const float& productCost);//конструктор с параметрами

Product(const Product& other);//конструкторы копирования

Деструктор:

~Product();//деструктор

Перегруженные операторы:

bool operator==(const Product& firstVal) const;// Перегрузка одного из бинарных операторов как метод класса

friend bool operator!=(const Product& secondVal, const Product& firstVal);//Перегрузка одного из бинарных операторов через дружественную функцию

bool operator>(const Product& firstVal) const;//Перегрузка одного из операторов отношения как метод класса

bool operator<=(const Product& firstVal) const;

Product& operator++();//Перегрузка префиксного инкремента как метод класса

friend Product operator++(Product& payment, int); //Перегрузка постфиксного инкремента через дружественную функцию

friend std::istream& operator>>(std::istream& is, Product& payment); // Перегрузка операторов >> через дружественные функции

friend std::ostream& operator<<(std::ostream& os, const Product& payment); // Перегрузка операторов << через дружественные функции

bool operator<(Product& value);

Функции:

MyString getProductName() const;

MyString getMarketName() const;

float getProductCost() const;

size\_t getProductNameLength();

size\_t getMarketNameLength();

**Класс MyString**

Поля:

char\* str\_;

char\* begin\_ = nullptr;

char\* end\_ = nullptr;

size\_t length\_;

static const size\_t npos = -1;

static const size\_t CIN\_LIM = 100000;

Конструкторы:

MyString();

MyString(const char\* str);

MyString(const MyString& other);

MyString(const MyString& other, size\_t pos, size\_t len = npos);

MyString(MyString&& other) noexcept;

Деструктор:

~MyString();

Перегруженные операторы:

MyString& operator=(const char\* other);

MyString& operator=(const MyString& other);

MyString& operator+=(char ch);

MyString& operator+=(const char\* other);

MyString& operator+=(const MyString& other);

bool operator==(const MyString& other) const;

bool operator!=(const MyString& other) const;

char& operator[](size\_t index);

friend std::ostream& operator<<(std::ostream& os, const MyString& str);

friend std::istream& operator>>(std::istream& is, MyString& str);

friend MyString operator+(const MyString& lhs, const MyString& rhs);

friend MyString operator+(const MyString& lhs, const char\* rhs);

friend MyString operator+(const char\* lhs, const MyString& rhs);

**Класс Exception**

Поля:

char\* text\_;

Конструкторы:

explicit MyException(const char\* text);

Деструктор:

~MyException() noexcept override;

**Шаблонный класс MyArray**

Поля:

size\_t size\_;

C\* data\_;

Конструкруторы:

Array()

Array(size\_t size)

Array(const Array<C>& object)

Деструктор:

~Array()

Перегруженные операторы:

C& operator[]

Array& operator= (Array& other)

Функции:

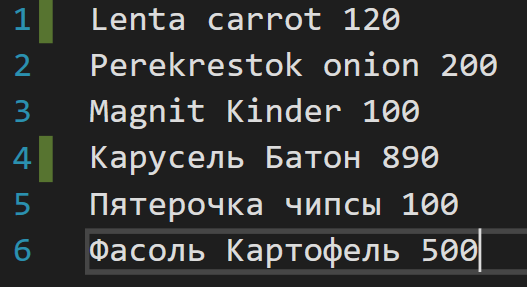
void push\_back(C& element)

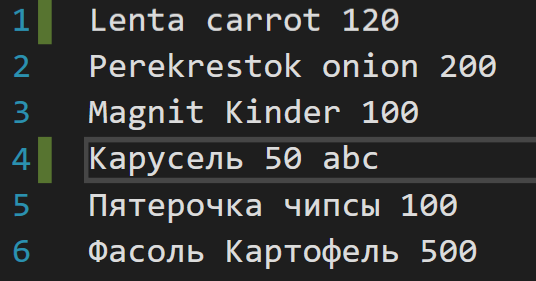
size\_t getSize() const // геттер

**Тестовый набор с выходными данными**

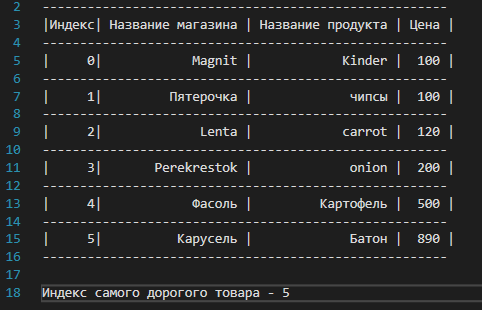
Input.txt:

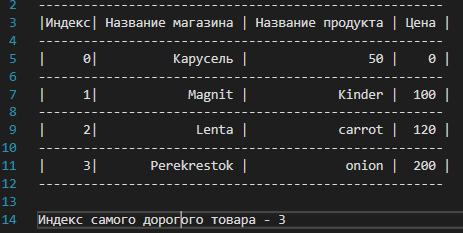
Ввод:

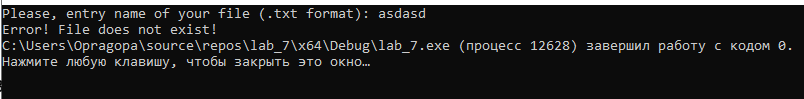




Вывод:







Программа:

https://github.com/Opragopa/lab\_7/tree/lab\_7