Федеральное государственное автономное образовательное учреждение высшего образования «Санкт-Петербургский государственный электротехнический университет «ЛЭТИ»

им. В.И. Ульянова (Ленина)»

кафедра Апу

**ОТЧЕТ**

**по лабораторной работе №1**

**«Улучшенные методы сортировки»**

Выполнил: Панфёров М. А.

Группа №3371

Преподаватель: Ряскова Е. Б.

Санкт-Петербург

2024

Постановка задачи

Требуется в программу из лабораторной работы №1 добавить 2 пункта меню с соответствием индивидуального задания, в данном случае добавить сортировки быстрая и Шелла.

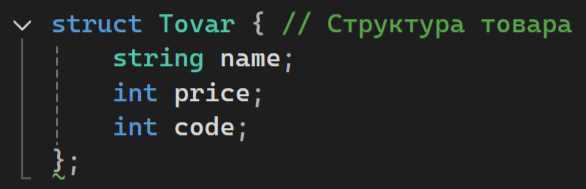
Листинг программы

Рис. 1 – структура данных товара.

На рисунке 1 показана структура данных товара. Он в себе содержит данные от его названии, сцены и кода.

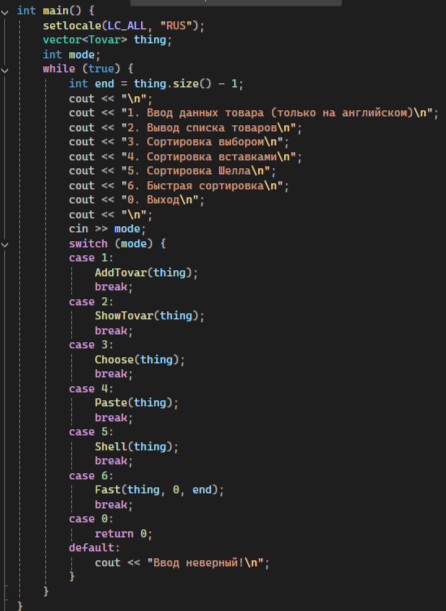


Рис. 2 – главная функция.

На рисунке 2 показана глафная функция. В ней создаётся массив структур и находится меню. После идёт выбор пункта меню, если указать число, которое не указано в пункте меню, то программа сообщит, что ввод неверен и будет ожидать ввода верного числа. Для работы быстрой сортировки была создана функция для вызова последнего элемента.

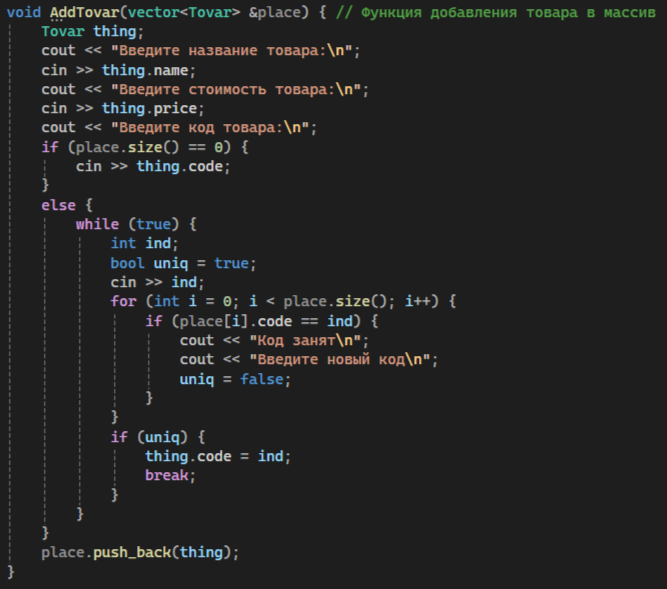


Рис. 3 – функция добавления товара в массив

На рисунке 3 показана функция добавления товара в массив. На вход подаётся массив. После создаётся “thing” – товар, которому после присваивается название и стоимость. Код присваивается иным способом. Когда массив пустой, код просто присваивается, но когда в массиве есть элементы, то идёт проверка уникальности кода. Если код повторяется – то пограмма сообщит, что код занят и нужно ввести новый, иначе код присвоится и товар будет добавлен в массив.

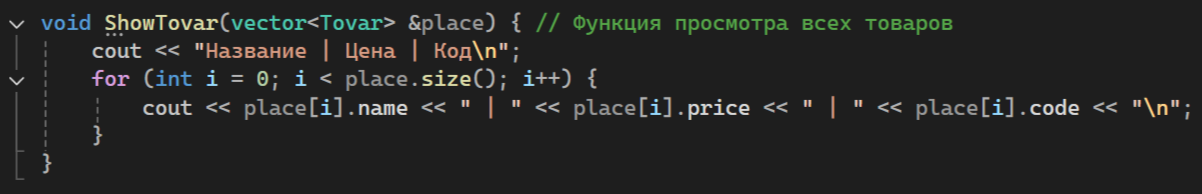


Рис. 4 – функция вывода содежимого массива

На рисунке 4 показана функция вывода данных массива. На вход подаётся массив. Вывод происходит в виде некоторой таблицы: «Название | Цена | Код». Вывод реализован при помощи цикла, в котором каждый i-тый элемент выводится.

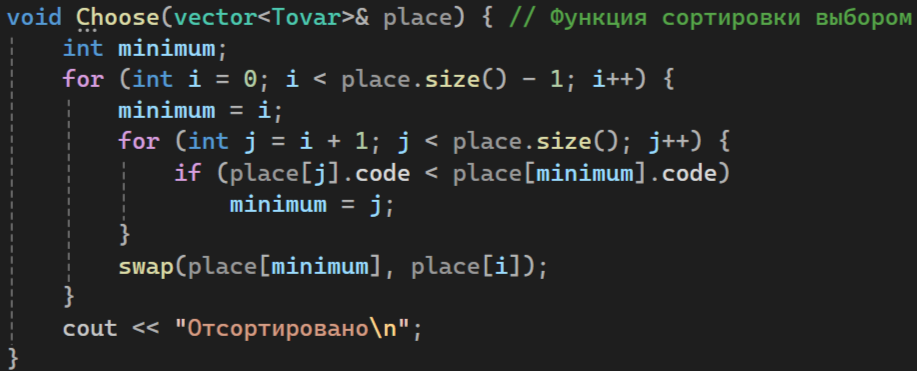


Рис. 5 – функция сортировки выбором

На рисунке показана функция, где реализована сортировка выбором. На вход подаётся массив. Создаётся переменная «minimum» - она определяет положение элемента, нужна для сравнения последующих элементов с текущим. Далее идёт цикл, который сравивает текущее значение, с «minimum», если текущее значение меньше «minimum», то текущее значение становится им и они меняются местами. И так до того, как массив не будет отсортирован.

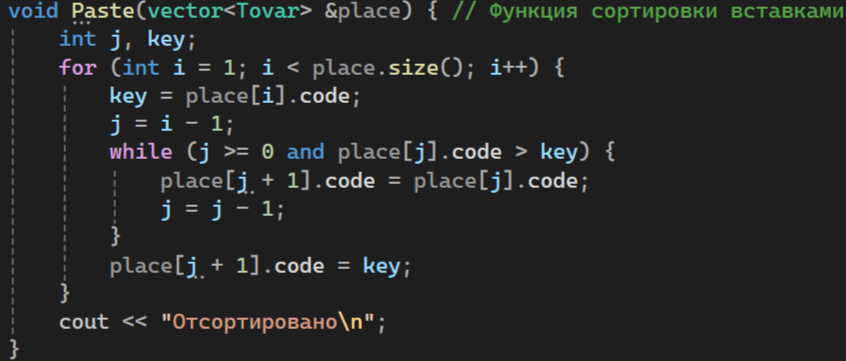
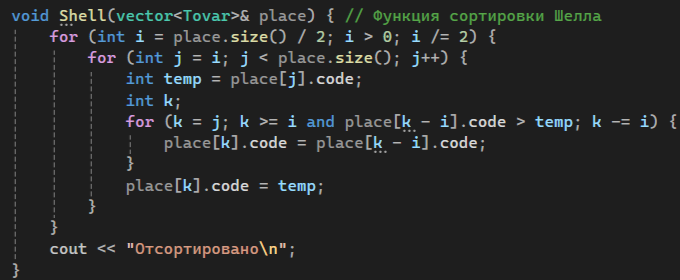


Рис. 6 – функция сортировки вставками

На рисунке 6 показана функция сортировка вствками. На вход подаётся массив. Создаются переменные «j» и «key». В начале цикла «key» принимает значение текущего элемента, а «j» - предыдущего. После идёт цикл, где следующий элемент заменяется текущим, после «key» становится следующим элементом. И так до того как массив не будет отсортирован.

Рис. 7 – функция сортировки Шелла

На рисунке 7 показана сортировка Шелла. Сама сортировка предствавляет собой улучшенную сортировку вставками, только элементы сравниваются с определённым интервалом.

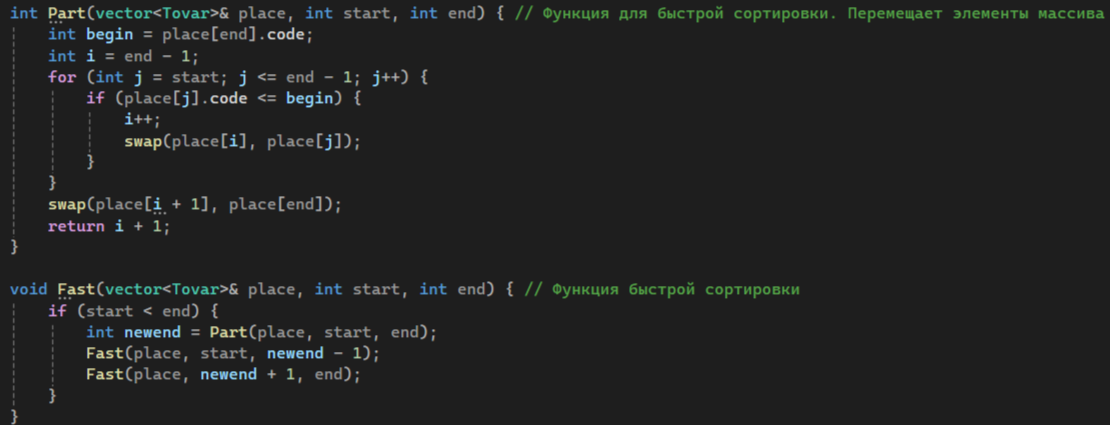


Рис 8. – реализация быстрой сортировки

На рисунке 8 показана реализация быстрой сортировки. Для неё было написано 2 функции – Part и Fast. Fast – это функция, при помощи которой эта функция вызывается. В ней реализованы рекурсивные функции, в которых элементы перемещаются в зависимости от размера выбранного числа. В Part происходит перемещение элементов.

Код и пример работы программы

AiSD\_Laba2.cpp:

#include "iostream"

#include "string"

#include "vector"

using namespace std;

// Вариант 15: дополнение кода программы двумя вариантами сортировки: Шелла и быстрая

struct Tovar { // Структура товара

string name;

int price;

int code;

};

void AddTovar(vector<Tovar>& place) { // Функция добавления товара в массив

Tovar thing;

cout << "Введите название товара:\n";

cin >> thing.name;

cout << "Введите стоимость товара:\n";

cin >> thing.price;

cout << "Введите код товара:\n";

if (place.size() == 0) {

cin >> thing.code;

}

else {

while (true) {

int ind;

bool uniq = true;

cin >> ind;

for (int i = 0; i < place.size(); i++) {

if (place[i].code == ind) {

cout << "Код занят\n";

cout << "Введите новый код\n";

uniq = false;

}

}

if (uniq) {

thing.code = ind;

break;

}

}

}

place.push\_back(thing);

}

void ShowTovar(vector<Tovar>& place) { // Функция просмотра всех товаров

cout << "Название | Цена | Код\n";

for (int i = 0; i < place.size(); i++) {

cout << place[i].name << " | " << place[i].price << " | " << place[i].code << "\n";

}

}

void Choose(vector<Tovar>& place) { // Функция сортировки выбором

int minimum;

for (int i = 0; i < place.size() - 1; i++) {

minimum = i;

for (int j = i + 1; j < place.size(); j++) {

if (place[j].code < place[minimum].code)

minimum = j;

}

swap(place[minimum], place[i]);

}

}

void Paste(vector<Tovar>& place) { // Функция сортировки вставками

int j, key;

for (int i = 1; i < place.size(); i++) {

key = place[i].code;

j = i - 1;

while (j >= 0 and place[j].code > key) {

place[j + 1].code = place[j].code;

j = j - 1;

}

place[j + 1].code = key;

}

cout << "Отсортировано\n";

}

void Shell(vector<Tovar>& place) { // Функция сортировки Шелла

for (int i = place.size() / 2; i > 0; i /= 2) {

for (int j = i; j < place.size(); j++) {

int temp = place[j].code;

int k;

for (k = j; k >= i and place[k - i].code > temp; k -= i) {

place[k].code = place[k - i].code;

}

place[k].code = temp;

}

}

cout << "Отсортировано\n";

}

int Part(vector<Tovar>& place, int start, int end) { // Функция для быстрой сортировки. Перемещает элементы массива

int begin = place[end].code;

int i = end - 1;

for (int j = start; j <= end - 1; j++) {

if (place[j].code <= begin) {

i++;

swap(place[i], place[j]);

}

}

swap(place[i + 1], place[end]);

return i + 1;

}

void Fast(vector<Tovar>& place, int start, int end) { // Функция быстрой сортировки

if (start < end) {

int newend = Part(place, start, end);

Fast(place, start, newend - 1);

Fast(place, newend + 1, end);

}

}

int main() {

setlocale(LC\_ALL, "RUS");

vector<Tovar> thing;

int mode;

while (true) {

int end = thing.size() - 1;

cout << "\n";

cout << "1. Ввод данных товара (только на английском)\n";

cout << "2. Вывод списка товаров\n";

cout << "3. Сортировка выбором\n";

cout << "4. Сортировка вставками\n";

cout << "5. Сортировка Шелла\n";

cout << "6. Быстрая сортировка\n";

cout << "0. Выход\n";

cout << "\n";

cin >> mode;

switch (mode) {

case 1:

AddTovar(thing);

break;

case 2:

ShowTovar(thing);

break;

case 3:

Choose(thing);

break;

case 4:

Paste(thing);

break;

case 5:

Shell(thing);

break;

case 6:

Fast(thing, 0, end);

break;

case 0:

return 0;

default:

cout << "Ввод неверный!\n";

}

}

}

Пример работы:

1. Ввод данных товара (только на английском)

2. Вывод списка товаров

3. Сортировка выбором

4. Сортировка вставками

5. Сортировка Шелла

6. Быстрая сортировка

0. Выход

1

Введите название товара:

Baget

Введите стоимость товара:

129

Введите код товара:

12

1. Ввод данных товара (только на английском)

2. Вывод списка товаров

3. Сортировка выбором

4. Сортировка вставками

5. Сортировка Шелла

6. Быстрая сортировка

0. Выход

1

Введите название товара:

Marmelad

Введите стоимость товара:

304

Введите код товара:

640

1. Ввод данных товара (только на английском)

2. Вывод списка товаров

3. Сортировка выбором

4. Сортировка вставками

5. Сортировка Шелла

6. Быстрая сортировка

0. Выход

1

Введите название товара:

juice

Введите стоимость товара:

110

Введите код товара:

14

1. Ввод данных товара (только на английском)

2. Вывод списка товаров

3. Сортировка выбором

4. Сортировка вставками

5. Сортировка Шелла

6. Быстрая сортировка

0. Выход

1

Введите название товара:

arbuzzzz

Введите стоимость товара:

1

Введите код товара:

1

1. Ввод данных товара (только на английском)

2. Вывод списка товаров

3. Сортировка выбором

4. Сортировка вставками

5. Сортировка Шелла

6. Быстрая сортировка

0. Выход

1

Введите название товара:

water

Введите стоимость товара:

32

Введите код товара:

2

1. Ввод данных товара (только на английском)

2. Вывод списка товаров

3. Сортировка выбором

4. Сортировка вставками

5. Сортировка Шелла

6. Быстрая сортировка

0. Выход

1

Введите название товара:

pirozhok

Введите стоимость товара:

59

Введите код товара:

45

1. Ввод данных товара (только на английском)

2. Вывод списка товаров

3. Сортировка выбором

4. Сортировка вставками

5. Сортировка Шелла

6. Быстрая сортировка

0. Выход

2

Название | Цена | Код

Baget | 129 | 12

Marmelad | 304 | 640

juice | 110 | 14

arbuzzzz | 1 | 1

water | 32 | 2

pirozhok | 59 | 45

1. Ввод данных товара (только на английском)

2. Вывод списка товаров

3. Сортировка выбором

4. Сортировка вставками

5. Сортировка Шелла

6. Быстрая сортировка

0. Выход

5

Отсортировано

1. Ввод данных товара (только на английском)

2. Вывод списка товаров

3. Сортировка выбором

4. Сортировка вставками

5. Сортировка Шелла

6. Быстрая сортировка

0. Выход

2

Название | Цена | Код

Baget | 129 | 1

Marmelad | 304 | 2

juice | 110 | 12

arbuzzzz | 1 | 14

water | 32 | 45

pirozhok | 59 | 640

1. Ввод данных товара (только на английском)

2. Вывод списка товаров

3. Сортировка выбором

4. Сортировка вставками

5. Сортировка Шелла

6. Быстрая сортировка

0. Выход

0

Вывод: выполнив данную лабораторную работу я узнал о быстрой сортировке и сортировке Шелла и узнал о принципе работы этих сортировок. Теперь я могу использовать данные сортировки в своих проектах и задачах.