Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное

учреждение

высшего образования

«**Пермский национальный исследовательский**

**политехнический университет»**

Кафедра «Информационные технологии и автоматизированные

системы»

**ОТЧЁТ**

**по лабораторной работе №15**

Дисциплина: «Основы алгоритмизации и программирования»

Тема: Методы внутренней сортировки массивов:

быстрая сортировка, сортировка подсчётом

Вариант 17

Выполнил:

Студент группы ИВТ-20-2б

Сафронов Владислав Владиславович

Проверила:

Доцент кафедры ИТАС

Полякова О.А.

**Пермь, 2021**

**Цель задачи**

Научиться программировать различные методы сортировки одномерных массивов.

**Постановка задачи**

Дано железнодорожное расписание. Упорядочить его по алфавитному порядку станций следования поездов.

**Анализ задачи**

**Какие типы данных будут использованы:**

* Целочисленный тип данных integer используется для хранения и ввода размера массива, индексов элементов, параметра в циклах.
* Строковый тип данных string для ввода и хранения названия станций.

**Какие с этими данными выполняются действия:**

* integer – ввод с клавиатуры, инкремент, присваивание, сравнение;
* String – ввод с клавиатуры, сравнение.

**В каком виде эти данные будут представлены:**

1. Структура Time для хранения времени прибытия поездов на станцию. Имеет 2 поля типа int:

struct Time

{

int hr = -1, min = -1;

};

1. Структура Station. Имеет 2 поля: строку name для хранения названия станции и экземпляр структуры Time для хранения времени прибытия поезда:

struct Station

{

string name;

Time arr\_time;

* 1. Метод структуры InputData для заполнения экземпляра структуры данными:

void InputData()

{

cout << "\nВведите название станции: ";

getline(cin, name);

while (name == "")

{

cout << "\nВведена пустая строка.";

cout << "\nВведите название станции: ";

getline(cin, name);

}

cout << "\nВведите время прибытия (чч мм): ";

cin >> arr\_time.hr >> arr\_time.min;

while (arr\_time.hr < 0 || arr\_time.hr > 23 ||

arr\_time.min < 0 || arr\_time.min > 59)

{

cout << "\nВремя прибытия введено некорректно. ";

cout << "\nВведите время прибытия (чч мм): ";

cin >> arr\_time.hr >> arr\_time.min;

}

cin.get();

}

* 1. Метод Print для вывода в консоль экземпляра структуры:

void Print()

{

cout << "Название станции: " << name;

cout << "\nВремя прибытия: " << arr\_time.hr << ":" << arr\_time.min;

cout << "\n--------------------------------------------------------\n";

}

1. Функция PrintArr для вывода массива структуры:

void PrintArr(vector<Station> lst)

{

cout << "\n--------------------------------------------------------\n";

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

{

lst[i].Print();

}

}

1. Функция ShellSort, реализующая метод сортировки Шелла. В качестве входного параметра функция принимает ссылку на массив элементов структуры. Сортировка происходит по названию станции:

void ShellSort(vector<Station>& lst)

{

int j;

string tmpname;

Time tmptime;

for (int step = lst.size() / 2; step > 0; step /= 2)

{

for (int i = step; i < lst.size(); i++)

{

tmpname = lst[i].name;

tmptime = lst[i].arr\_time;

for (j = i; j >= step; j -= step)

{

if (tmpname < lst[j - step].name)

{

lst[j].name = lst[j - step].name;

lst[j].arr\_time = lst[j - step].arr\_time;

}

else break;

}

lst[j].name = tmpname;

lst[j].arr\_time = tmptime;

}

}

}

1. Функция QuickSort, реализующая метод быстрой сортировки Хоара. В качестве входных параметров функция принимает ссылку на массив элементов структур, индекс первого элемента и индекс последнего элемента. Функция вызывается рекурсивно.

void QuickSort(vector<Station>& lst, int first, int last)

{

int i = first, j = last;

string tmpname = lst[(first + last) / 2].name;

Time tmptime = lst[(first + last) / 2].arr\_time;

do

{

while (lst[i].name < tmpname) i++;

while (lst[j].name > tmpname) j--;

if (i <= j)

{

if (lst[i].name > lst[j].name) swap(lst[i], lst[j]);

i++;

j--;

}

} while (i <= j);

if (i < last) QuickSort(lst, i, last);

if (first < j) QuickSort(lst, first, j);

}

**Какими операторами будет организован ввод и вывод:**

* Ввод данных осуществляется через консоль с помощью функции cin и getline();
* Вывод данных в консоль осуществляется с помощью функции cout.

**Описание функции main():**

Подключение русского языка:

setlocale(LC\_ALL, "Russian");

system("chcp 1251>nul");

Резервирование памяти под переменные и ввод размерности массива:

int n;

cout << "Введите количество элементов списка (1-100): ";

cin >> n;

while (n < 1 || n > 100)

{

cout << "\nКоличество введено некорректно.\n";

cout << "Введите количество элементов списка (1-100): ";

cin >> n;

}

cin.get();

Создание и ввод данных элементов структуры:

vector<Station> arr(n);

for (int i = 0; i < n; i++)

{

arr[i].InputData();

}

Выбор метода сортировки массива:

cout << "\n--------------------------------------------------------\n";

cout << "| Выберите метод сортировки\n";

cout << "| 1 - Сортировка Шелла\n";

cout << "| 2 - Сортировка Хоара\n";

cout << "Введите число (1-2): ";

cin >> n;

while (n < 1 || n > 2)

{

cout << "\nВвод некорректен.\n";

cout << "Введите число (1-2): ";

cin >> n;

} Освобождение памяти:

for (int i = 0; i < n; i++) {

delete[] arr[i];

}

delete[] arr;

Сортировка массива соответствующей функцией и вывод отсортированного массива:

switch (n) {

case 1: ShellSort(arr); break;

case 2: QuickSort(arr, 0, (arr.size() - 1)); break;

}

cout << "\nОтсортированный массив: ";

PrintArr(arr);

**Блок-схема программы**

**Код программы**

#include <iostream>

#include <vector>

#include <string>

using namespace std;

struct Time

{

int hr = -1, min = -1;

};

struct Station

{

string name;

Time arr\_time;

void InputData()

{

cout << "\nВведите название станции: ";

getline(cin, name);

while (name == "")

{

cout << "\nВведена пустая строка.";

cout << "\nВведите название станции: ";

getline(cin, name);

}

cout << "\nВведите время прибытия (чч мм): ";

cin >> arr\_time.hr >> arr\_time.min;

while (arr\_time.hr < 0 || arr\_time.hr > 23 ||

arr\_time.min < 0 || arr\_time.min > 59)

{

cout << "\nВремя прибытия введено некорректно. ";

cout << "\nВведите время прибытия (чч мм): ";

cin >> arr\_time.hr >> arr\_time.min;

}

cin.get();

}

void Print()

{

cout << "Название станции: " << name;

cout << "\nВремя прибытия: " << arr\_time.hr << ":" << arr\_time.min;

cout << "\n--------------------------------------------------------\n";

}

};

void PrintArr(vector<Station> lst)

{

cout << "\n--------------------------------------------------------\n";

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

{

lst[i].Print();

}

}

void ShellSort(vector<Station>& lst)

{

int j;

string tmpname;

Time tmptime;

for (int step = lst.size() / 2; step > 0; step /= 2)

{

for (int i = step; i < lst.size(); i++)

{

tmpname = lst[i].name;

tmptime = lst[i].arr\_time;

for (j = i; j >= step; j -= step)

{

if (tmpname < lst[j - step].name)

{

lst[j].name = lst[j - step].name;

lst[j].arr\_time = lst[j - step].arr\_time;

}

else break;

}

lst[j].name = tmpname;

lst[j].arr\_time = tmptime;

}

}

}

void QuickSort(vector<Station>& lst, int first, int last)

{

int i = first, j = last;

string tmpname = lst[(first + last) / 2].name;

Time tmptime = lst[(first + last) / 2].arr\_time;

do

{

while (lst[i].name < tmpname) i++;

while (lst[j].name > tmpname) j--;

if (i <= j)

{

if (lst[i].name > lst[j].name) swap(lst[i], lst[j]);

i++;

j--;

}

} while (i <= j);

if (i < last) QuickSort(lst, i, last);

if (first < j) QuickSort(lst, first, j);

}

int main()

{

setlocale(LC\_ALL, "Russian");

system("chcp 1251>nul");

int n;

cout << "Введите количество элементов списка (1-100): ";

cin >> n;

while (n < 1 || n > 100)

{

cout << "\nКоличество введено некорректно.\n";

cout << "Введите количество элементов списка (1-100): ";

cin >> n;

}

cin.get();

vector<Station> arr(n);

for (int i = 0; i < n; i++)

{

arr[i].InputData();

}

cout << "\n--------------------------------------------------------\n";

cout << "| Выберите метод сортировки\n";

cout << "| 1 - Сортировка Шелла\n";

cout << "| 2 - Сортировка Хоара\n";

cout << "Введите число (1-2): ";

cin >> n;

while (n < 1 || n > 2)

{

cout << "\nВвод некорректен.\n";

cout << "Введите число (1-2): ";

cin >> n;

}

switch (n) {

case 1: ShellSort(arr); break;

case 2: QuickSort(arr, 0, (arr.size() - 1)); break;

}

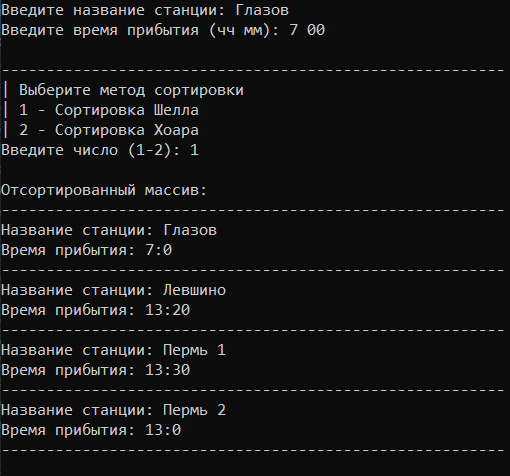
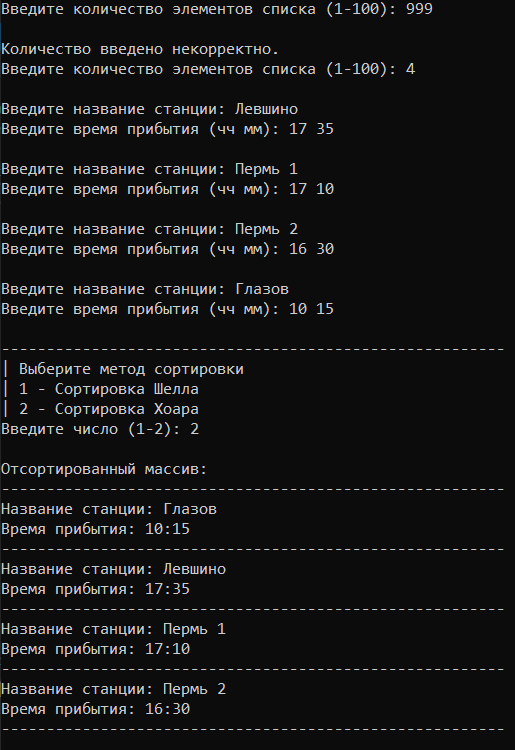
cout << "\nОтсортированный массив: ";

PrintArr(arr);

return 0;

}

**Скриншоты результатов работы программы**

****