Министерство высшего образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«Пермский национальный исследовательский политехнический университет» (ПНИПУ)**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

ОТЧЁТ

по лабораторной работе

Тема: «Сортировка Хоара и Шелла»

Выполнил

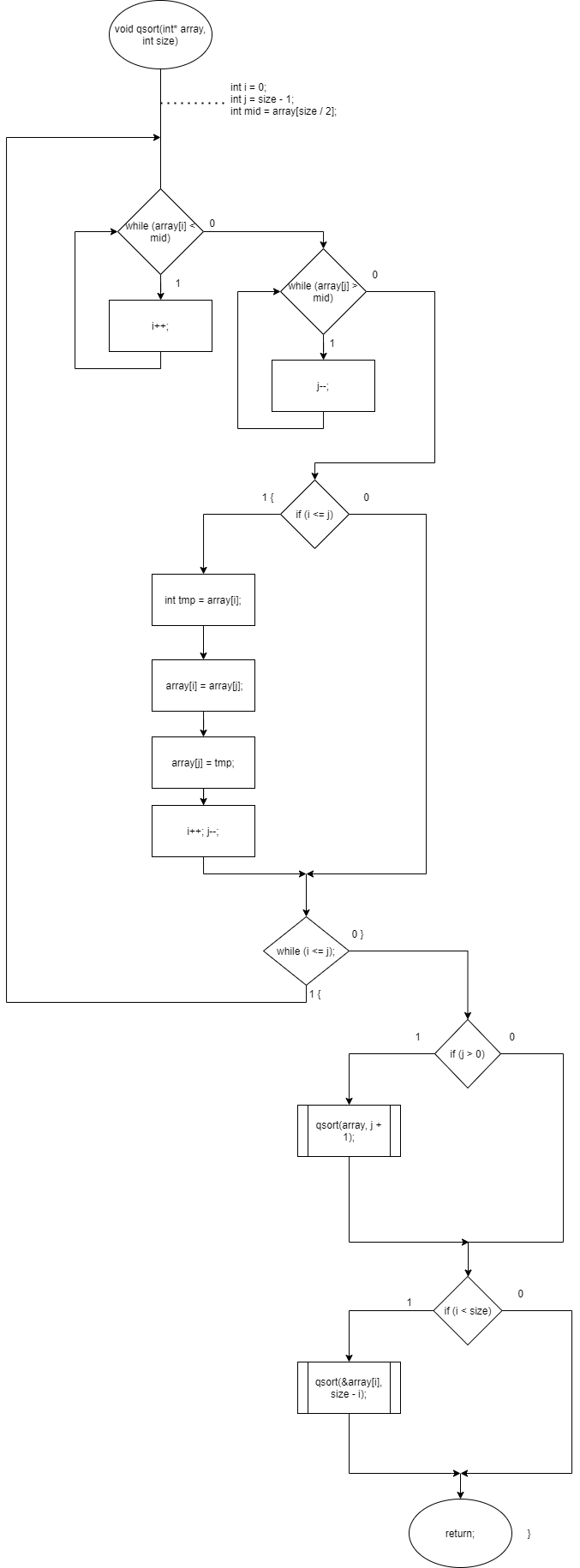
Студент группы РИС-22-2б

Будин Д.В.

Проверил доц. Кафедры ИТАС

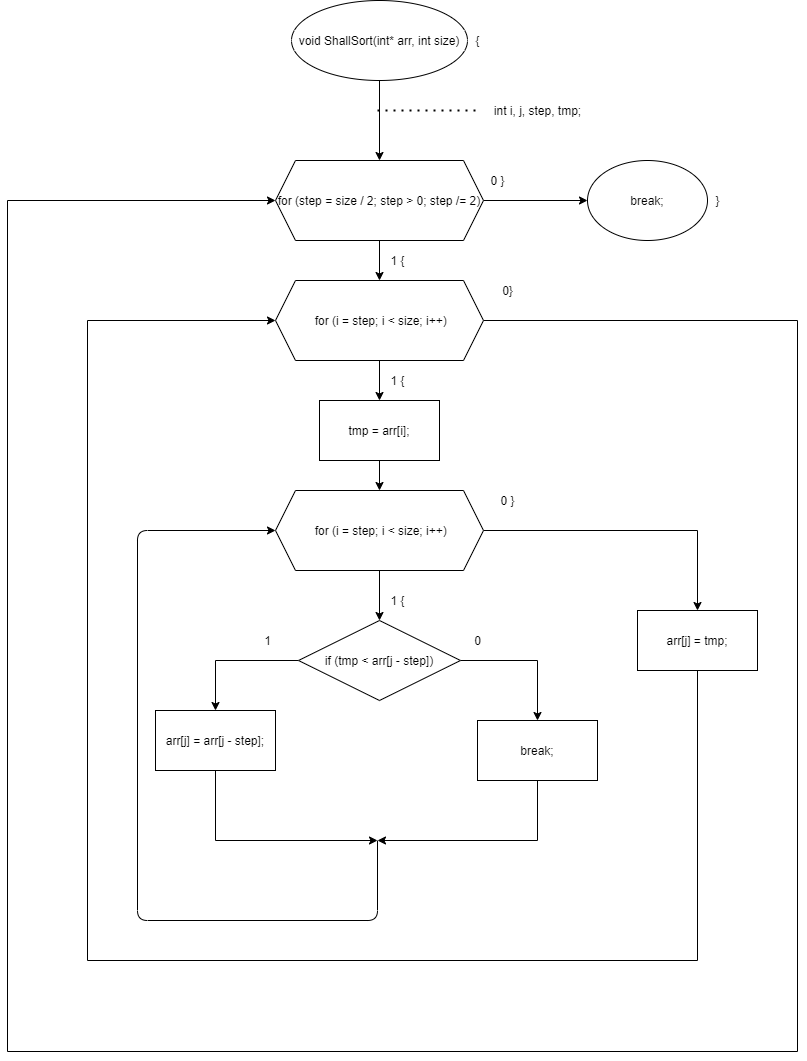
Полякова Ольга Андреевна

Пермь 2023

****Постановка задачи

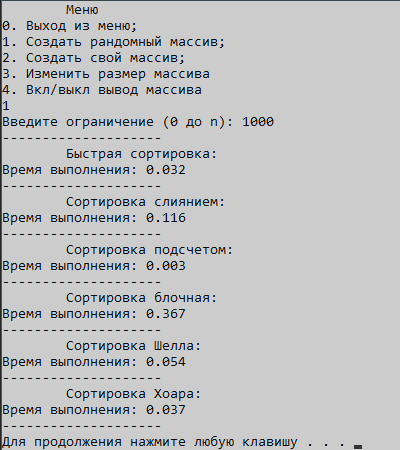
Разработать сортировки: Хоара, Шелла

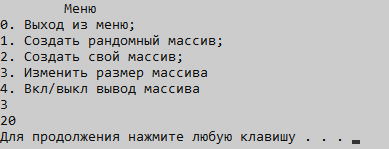
**Алгоритм программы**

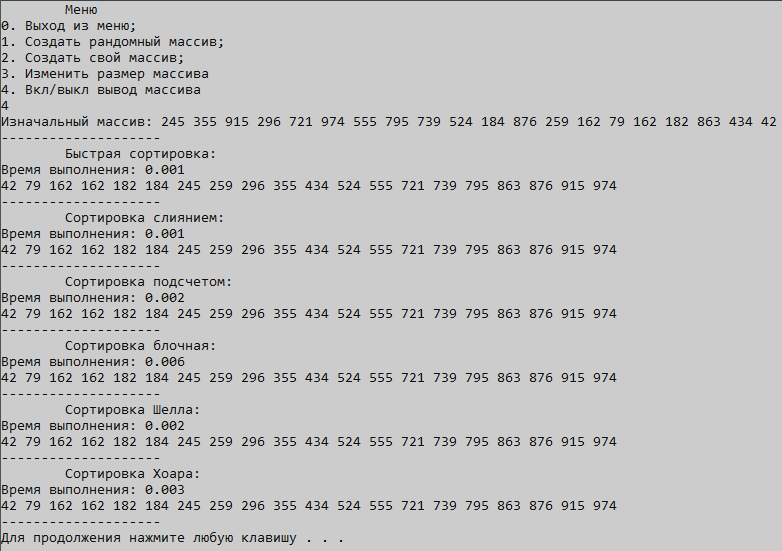
****

**Скриншоты результатов:**









**Код программы:**

#include <iostream>

#include <windows.h>

#include <stdlib.h>

#include <list>

#include <vector>

#include <algorithm>

using namespace std;

void show\_menu(); // визуал меню

void menu(); // само меню

void qsort(int\* array, int size); // быстрая сортировка

void merge\_sort(int\* arr, int size);// сортировка слиянием

int size\_arr; // размерность массива, задается в menu()

int maxValue = 99;

int minValue = 0;

void ShallSort(int\* arr, int size)

{

int i, j, step, tmp;

for (step = size / 2; step > 0; step /= 2)

for (i = step; i < size; i++)

{

tmp = arr[i];

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

{

if (tmp < arr[j - step])

arr[j] = arr[j - step];

else

break;

}

arr[j] = tmp;

}

}

void BucketSort(int\* array)

{

list<int>\*\* buckets = new list<int>\*[maxValue];

for (int i = 0; i < maxValue; i++) buckets[i] = new list<int>;

for (int i = 0; i < size\_arr; i++)

buckets[(int)floor(array[i] / 10)]->push\_back(array[i]);

int k = 0;

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

buckets[i]->sort();

for (auto iter = buckets[i]->begin(); iter != buckets[i]->end(); iter++)

{

array[k++] = \*iter;

}

}

for (int i = 0; i < maxValue; i++) delete buckets[i];

delete[] buckets;

}

void merge\_sort(int\* arr, int size)

{

if (size == 1) return;

merge\_sort(arr, size / 2);

merge\_sort(&arr[size / 2], size - size / 2);

int liter = 0;

int riter = 0;

int\* tmp = new int[size];

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

if (arr[liter] < arr[size / 2 + riter])

tmp[i] = arr[liter++];

else

tmp[i] = arr[size / 2 + riter++];

if (liter == size / 2) {

while (riter < size - size / 2)

tmp[++i] = arr[size / 2 + riter++];

break;

}

else if (riter == size - size / 2) {

while (liter < size / 2)

tmp[++i] = arr[liter++];

break;

}

}

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

arr[i] = tmp[i];

delete[] tmp;

}

void countingSort(int\* array, int size) {

int max, min;

max = min = array[0];

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

if (array[i] < min) {

min = array[i];

}

if (array[i] > max) {

max = array[i];

}

}

int\* temp = new int[max - min +1]();

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

++temp[array[i] - min];

}

int index = 0;

for (int i = 0; i < max - min + 1; ++i) {

for (int j = 0; j < temp[i]; ++j) {

array[index++] = i + min;

}

}

delete[] temp;

}

void qsort(int\* array, int size)

{

int i = 0;

int j = size - 1;

int mid = array[size / 2];

do {

while (array[i] < mid) i++;

while (array[j] > mid) j--;

if (i <= j) {

int tmp = array[i];

array[i] = array[j];

array[j] = tmp;

i++; j--;

}

} while (i <= j);

if (j > 0) {

qsort(array, j + 1);

}

if (i < size) {

qsort(&array[i], size - i);

}

}

int\* new\_array(int\* array, int k)

{

cout << "Введите ограничение (0 до n): ";

cin >> maxValue;

if (k == 1)

{

for (int i = 0; i < size\_arr; i++)

{

array[i] = rand() % maxValue;

}

}

else

{

cout << "Вводите числа (0 до " << maxValue << "): ";

for (int i = 0; i < size\_arr; i++)

{

cin >> array[i];

}

}

return array;

}

void output(int\* array)

{

for (int i = 0; i < size\_arr; i++)

{

cout << array[i] << " ";

} cout << endl;

}

void menu()

{

bool flag = 0;

int n;

unsigned int start, end;

cout << "Задайте размерность массива: ";

cin >> size\_arr;

int\* array = new int[size\_arr];

int\* sort\_array = new int[size\_arr];

do

{

system("pause");

system("cls");

show\_menu();

cin >> n;

switch (n)

{

case 0: {

cout << "Выход из меню!" << endl;

return;

}

case 1: {

array = new\_array(array, n);

break;

}

case 2: {

array = new\_array(array, n);

break;

}

case 3: {

cin >> size\_arr;

int\* array = new int[size\_arr];

int\* sort\_array = new int[size\_arr];

continue;

break;

}

case 4: {flag = !flag; break; }

default: cout << ">>???>>" << endl;

}

if (flag) {

cout << "Изначальный массив: "; output(array);

}

for (int i = 1; i <= 6; i++)

{

memcpy(sort\_array, array, sizeof(int) \* size\_arr);

cout << "--------------------" << endl;

start = clock();

switch (i)

{

case 1: {

cout << "\tБыстрая сортировка:" << endl;

qsort(sort\_array, size\_arr);

break;

}

case 2: {

cout << "\tСортировка слиянием:" << endl;

merge\_sort(sort\_array, size\_arr);

break;

}

case 3: {

cout << "\tСортировка подсчетом:" << endl;

countingSort(sort\_array, size\_arr);

break;

}

case 4: {

cout << "\tСортировка блочная:" << endl;

BucketSort(sort\_array);

break;

}

case 5: {

cout << "\tСортировка Шелла:" << endl;

ShallSort(sort\_array, size\_arr);

break;

}

case 6: {

cout << "\tСортировка Хоара:" << endl;

qsort(sort\_array, size\_arr);

break;

}

}

end = clock() - start;

cout << "Время выполнения: " << (float)end / CLOCKS\_PER\_SEC << endl;

if (flag) output(sort\_array);

}

cout << "--------------------" << endl;

} while (n != 0);

delete[] array;

delete[] sort\_array;

}

void show\_menu()

{

cout << "\tМеню\t" << endl;

cout << "0. Выход из меню;" << endl;

cout << "1. Создать рандомный массив;" << endl;

cout << "2. Создать свой массив;" << endl;

cout << "3. Изменить размер массива" << endl;

cout << "4. Вкл/выкл вывод массива" << endl;

}

int main()

{

srand(time(NULL));

setlocale(LC\_ALL, "Rus");

menu();

return 0;

}