Министерство науки и высшего образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №2

по курсу «Логика и основы алгоритмизации в инженерных задачах»

на тему «оценка времени выполнения программ»

**Выполнили студенты группы 21вв1:**

Нечаев А.Д.

Киреев Д.А.

**Приняли**

Юрова О.В.

Акифьев И.В.

Пенза 2022

**Методические указания.**

**Задание 1**:

**1.** Вычислить порядок сложности программы (О-символику).

**2.**Оценить время выполнения программы и кода, выполняющего перемножение матриц, используя функции библиотеки time.h для матриц размерами от 100, 200, 400, 1000, 2000, 4000, 10000.

**3.** Построить график зависимости времени выполнения программы от размера матриц и сравнить полученный результат с теоретической оценкой.

**Задание 2**:

**1.** Оценить время работы каждого из реализованных алгоритмов на случайном наборе значений массива.

**2.** Оценить время работы каждого из реализованных алгоритмов на массиве, представляющем собой возрастающую последовательность чисел.

**3.** Оценить время работы каждого из реализованных алгоритмов на массиве, представляющем собой убывающую последовательность чисел.

**4.** Оценить время работы каждого из реализованных алгоритмов на массиве, одна половина которого представляет собой возрастающую последовательность чисел, а вторая, – убывающую.

**5.** Оценить время работы стандартной функции qsort, реализующей алгоритм быстрой сортировки на выше указанных наборах данных.

**Описание метода решения задачи:**

**Задание 1**:

При помощи функции **rand()** массив заполняется случайными числами. Далее программа перемножает 2 матрицы по формуле. Затем при помощи функций **clock\_t** и **time\_t** измерили время перемножение двух матриц. Потом программа при помощи функции **double time = (end\_time - start\_time)/1000** выводит время перемножения двух матриц в секундах.

**Задание 2**:

В начале пользователь вводит с клавиатуры размер массива, далее под него выделяется память при помощи функции **malloc()**. После пользователь вводит сам массив.

В программе мы описали 4 разных массива функциями (случайный, возрастающий, убывающий и треугольный где 1 половина возрастает, а 2 убывает). Далее мы написали 3 трассировки: шеллы, qs, qsort. Затем измерили время 3 трассировок на разных наборах данных и вывели результаты в таблицу.

**Листинг:**

**Задание 1:**

#include <iostream>

#include <locale.h>

#include <stdio.h>

#include <time.h>

#include <stdlib.h>

int main()

{

int i = 0, j = 0,r;

int a[100][100], b[100][100], c[100][100], d;

srand(time(NULL));

for(i = 0; i < 100; i++ )

{

for(j = 0; j < 100; j++)

{

a[i][j] = rand()% 100;

}

}

i = 0; j = 0;

for(i = 0; i < 100; i++)

{

for( j = 0; j < 100; j++)

{

b[i][j] = rand()% 100;

}

}

time\_t start\_time = clock();

for( i = 0; i < 100; i++)

{

for( j = 0; j < 100; j++)

{

d = 0;

for( r = 0; r < 100; r++)

{

d=d+a[i][r]\*b[r][j];

c[i][j]=d;

}

}

}

time\_t end\_time = clock();

/\*for (i = 0; i < 100; i++)

{

for( j = 0; j < 100; j++)

{

printf("%d ", c[i][j]);

}

}\*/

//printf("\n\n\n");

double time = (end\_time - start\_time)/1000.0;

printf("\nВремя выполнения программы %f",time);

}

**Задание 2:**

#include <iostream>

#include <locale.h>

#include <stdio.h>

#include <time.h>

#include <stdlib.h>

void rand(int\* m, int size)

{

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

{

m[i] = rand() % 100;

}

}

void vozr(int\* m, int size)

{

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

{

m[i] = i;

}

}

void ubiv(int\* m, int size)

{

for (int i = size; i > 0; i--)

{

m[i] = i;

}

}

void trigle(int\* m, int size)

{

for (int i = 0; i < size / 2; i++)

{

m[i] = i;

}

for (int i = size / 2; i < size; i++)

{

m[i] = m[i - 1] - 1;

}

}

void qs(int\* mass, int size)

{

int i = 0;

int j = size - 1;

int mid = mass[size / 2];

do

{

while (mass[i] < mid)

{

i++;

}

while (mass[j] > mid)

{

j--;

}

if (i <= j)

{

int tmp = mass[i];

mass[i] = mass[j];

mass[j] = tmp;

i++; j--;

}

} while (i <= j);

if (j > 0)

{

qs(mass, j + 1);

}

if (i < size)

{

qs(&mass[i], size - i);

}

}

int compare(const void\* x1, const void\* x2)

{

return (\*(int\*)x1 - \*(int\*)x2);

}

void shell(int\* arr, int n)

{

int i, j, zxc, k;

int x;

for (k = n / 2; k > 0; k /= 2)

{

for (i = k; i < n; i++)

{

for (j = i - k; (j >= 0) && arr[j] > arr[j + k]; j = j - k)

{

x = arr[j];

arr[j] = arr[j + k];

arr[j + k] = x;

}

}

}

}

int main()

{

//setlocale(LC\_ALL, "Rus");

srand(time(NULL));

system("chcp 1251");

int i, size;

scanf\_s("%d", &size);

**//РАНДОМНЫЙ**

int\* m;

m = (int\*)\_malloca(size \* sizeof(int));

rand(m, size);

time\_t start\_time = clock();

shell(m, size);

time\_t end\_time = clock();

double time = (end\_time - start\_time) / 1000.0;

rand(m, size);

time\_t start\_time1 = clock();

qs(m, size);

time\_t end\_time1 = clock();

double time1 = (end\_time1 - start\_time1) / 1000.0;

rand(m, size);

time\_t start\_time2 = clock();

qsort(m, size, sizeof(int), compare);

time\_t end\_time2 = clock();

double time2 = (end\_time2 - start\_time2) / 1000.0;

**//ВОЗРАСТАЮЩИЙ**

int\* m2;

m2 = (int\*)\_malloca(size \* sizeof(int));

vozr(m2, size);

time\_t start\_time3 = clock();

shell(m2, size);

time\_t end\_time3 = clock();

double time3 = (end\_time3 - start\_time3) / 1000.0;

vozr(m2, size);

time\_t start\_time4 = clock();

qs(m2, size);

time\_t end\_time4 = clock();

double time4 = (end\_time4 - start\_time4) / 1000.0;

vozr(m2, size);

time\_t start\_time5 = clock();

qsort(m2, size, sizeof(int), compare);

time\_t end\_time5 = clock();

double time5 = (end\_time5 - start\_time5) / 1000.0;

**//УБЫВАЮЩИЙ**

int\* m3;

m3 = (int\*)\_malloca(size \* sizeof(int));

ubiv(m3, size);

time\_t start\_time6 = clock();

shell(m3, size);

time\_t end\_time6 = clock();

double time6 = (end\_time6 - start\_time6) / 1000.0;

ubiv(m3, size);

time\_t start\_time7 = clock();

qs(m3, size);

time\_t end\_time7 = clock();

double time7 = (end\_time7 - start\_time7) / 1000.0;

ubiv(m3, size);

time\_t start\_time8 = clock();

qsort(m3, size, sizeof(int), compare);

time\_t end\_time8 = clock();

double time8 = (end\_time8 - start\_time8) / 1000.0;

**//ТРЕУГОЛЬНИК**

int\* m4;

m4 = (int\*)\_malloca(size \* sizeof(int));

trigle(m4, size);

time\_t start\_time9 = clock();

shell(m4, size);

time\_t end\_time9 = clock();

double time9 = (end\_time9 - start\_time9) / 1000.0;

trigle(m4, size);

time\_t start\_time10 = clock();

qs(m4, size);

time\_t end\_time10 = clock();

double time10 = (end\_time10 - start\_time10) / 1000.0;

trigle(m4, size);

time\_t start\_time11 = clock();

qsort(m4, size, sizeof(int), compare);

time\_t end\_time11 = clock();

double time11 = (end\_time11 - start\_time11) / 1000.0;

printf("\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n");

printf("| Массив | Шелл | Qs | Qs(функция) |\n");

printf("|\_\_\_\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_|\n");

printf("|Рандомный |%f |%f|%17f |\n", time, time1, time2);

printf("|Возрастающий|%f |%f|%17f |\n", time3, time4, time5);

printf("|Убывающий |%f |%f|%17f |\n", time6, time7, time8);

printf("|Треугольный |%f |%f|%17f |\n", time9, time10, time11);

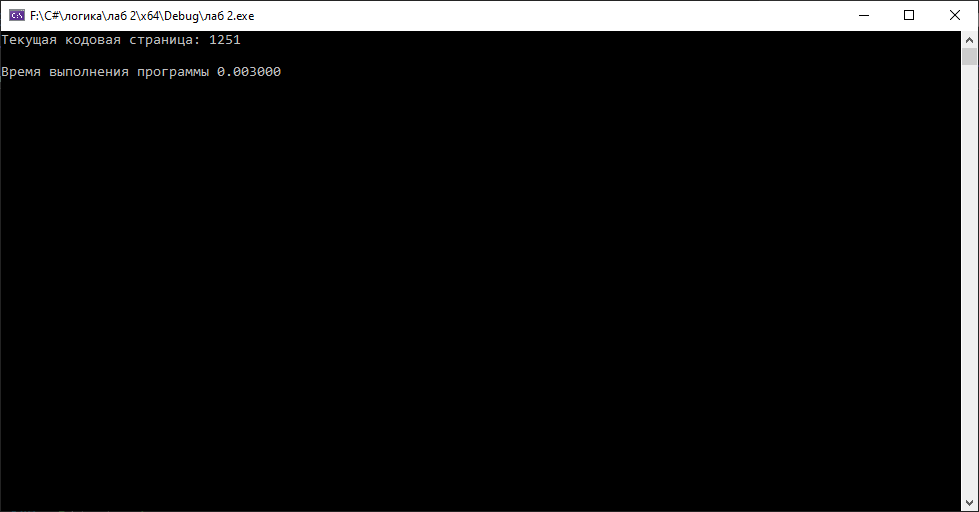
//getchar();

//getchar();

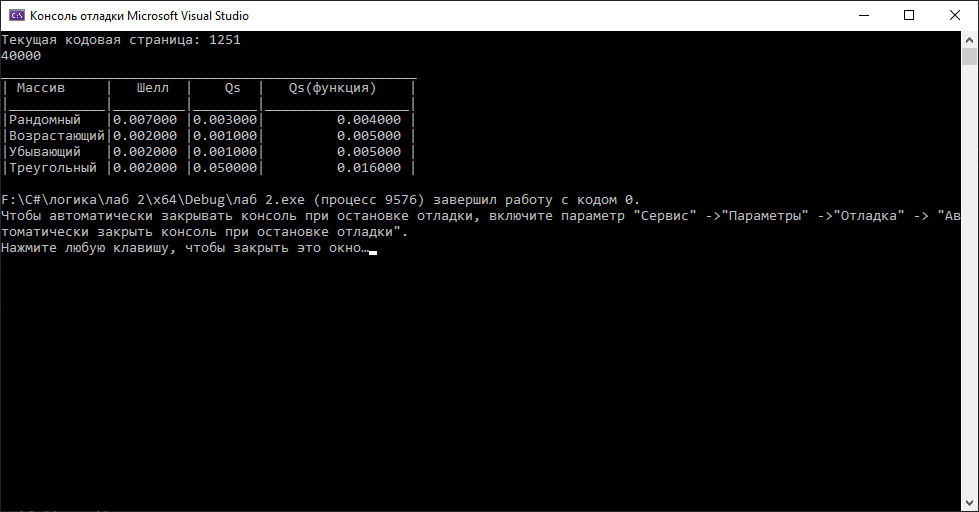
}

**Результаты работы программы:**

**Задание 1:**



**Задание 2:**



**Вывод:** в ходе выполнения лабораторной работы были разработаны программа, выполняющая функции с применением статичных и динамических массивов, а также функций. Результаты работы программ совпали с результатами трассировки, следовательно программы работают без ошибок. Также научились измерять время работы отдельных частей программы . Получили опыт в создании проектов в среде MicrosoftVisualStudio, научились писать и отлаживать программы с использованием функций и массивов на языке Си.