 Министерство образования Российской Федерации

Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЁТ**

по лабораторной работе №2

по курсу «Логика и основы алгоритмизации в ИЗ»

на тему «Оценка времени выполнения программ»

Выполнили:

студенты группы 22ВВВ2

Перфилов А.В.

Приняли:

доцент, профессор Митрохин М.А.

к.э.н., доцент Акифьев И.В.

Пенза 2023

**Название**

Оценка времени выполнения программ

**Цель работы**

Оценить время выполнения программы

**Лабораторное задание**

**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.

**Листинг**

#include <stdio.h>

#include <time.h>

#include <malloc.h>

#include <random>

#include <locale.h>

int summ(int len) {

setvbuf(stdin, NULL, \_IONBF, 0);

setvbuf(stdout, NULL, \_IONBF, 0);

clock\_t start, end; // объявляем переменные для определения времени выполнения

int i = 0, j = 0, r;

int a[200][200], b[200][200], c[200][200], elem\_c;

srand(time(NULL)); // инициализируем параметры генератора случайных чисел

while (i < len)

{

while (j < len)

{

a[i][j] = rand() % 100 + 1; // заполняем массив случайными числами

j++;

}

i++;

}

srand(time(NULL)); // инициализируем параметры генератора случайных чисел

i = 0; j = 0;

while (i < 200)

{

while (j < len)

{

b[i][j] = rand() % 100 + 1; // заполняем массив случайными числами

j++;

}

i++;

}

start = clock();

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

{

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

{

elem\_c = 0;

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

{

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

c[i][j] = elem\_c;

}

}

}

end = clock();

double answer = difftime(end, start);

printf("начальное время: %d\nконечное время: %d\nразница (в секундах): %f\n\n", start, end, answer / CLOCKS\_PER\_SEC);

return 0;

}

void shell(int\* items, int count) {

int i, j, gap, k;

int x, a[5];

a[0] = 9; a[1] = 5; a[2] = 3; a[3] = 2; a[4] = 1;

for (k = 0; k < 5; k++) {

gap = a[k];

for (i = gap; i < count; ++i) {

x = items[i];

for (j = i - gap; (x < items[j]) && (j >= 0); j = j - gap)

items[j + gap] = items[j];

items[j + gap] = x;

}

}

}

void qs(int\* items, int left, int right) {//вызов функции: qs(items, 0, count-1);

int i, j;

int x, y;

i = left; j = right;

/\* выбор компаранда \*/

x = items[(left + right) / 2];

do {

while ((items[i] < x) && (i < right)) i++;

while ((x < items[j]) && (j > left)) j--;

if (i <= j) {

y = items[i];

items[i] = items[j];

items[j] = y;

i++; j--;

}

} while (i <= j);

if (left < j) qs(items, left, j);

if (i < right) qs(items, i, right);

}

void randitems(int\* mas, int size) {

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

mas[i] = rand() % 100;

}

}

int main() {

setlocale(LC\_ALL, "Rus");

srand(time(NULL));

int sizes[7] = { 100, 250, 500, 1000, 2500, 5000, 10000 };

int mass[5] = { 10, 20, 40, 100, 200 };

int\* mas = (int\*)malloc(sizes[6] \* sizeof(int));

clock\_t start, end;

printf("\_\_\_\_\_\_\_задание 1\_\_\_\_\_\_\_\n");

for (int i = 0; i < 5; i++) {

summ(mass[i]);

}

printf("\_\_\_\_\_\_\_задание 2\_\_\_\_\_\_\_\n");

for (int i = 0; i < 7; i++){

randitems(mas, sizes[i]);

start = clock();

shell(mas,sizes[i]);

end = clock();

printf("Сортировка Шеллом\nначальное время: %d\nконечное время: %d\nВремя: %f\nЭлементов:%d\n",start, end, difftime(end, start) /CLOCKS\_PER\_SEC,sizes[i]);

randitems(mas, sizes[i]);

start = clock();

qs(mas,0,sizes[i]-1);

end = clock();

printf("Быстрая сортировка\nначальное время: %d\nконечное время: %d\nВремя: %f\nЭлементов:%d\n\n", start, end,difftime(end, start) / CLOCKS\_PER\_SEC, sizes[i]);

}

return 0;

}

**Результат работы программы**

Результат работы программы показан на рисунках 1 и 2.

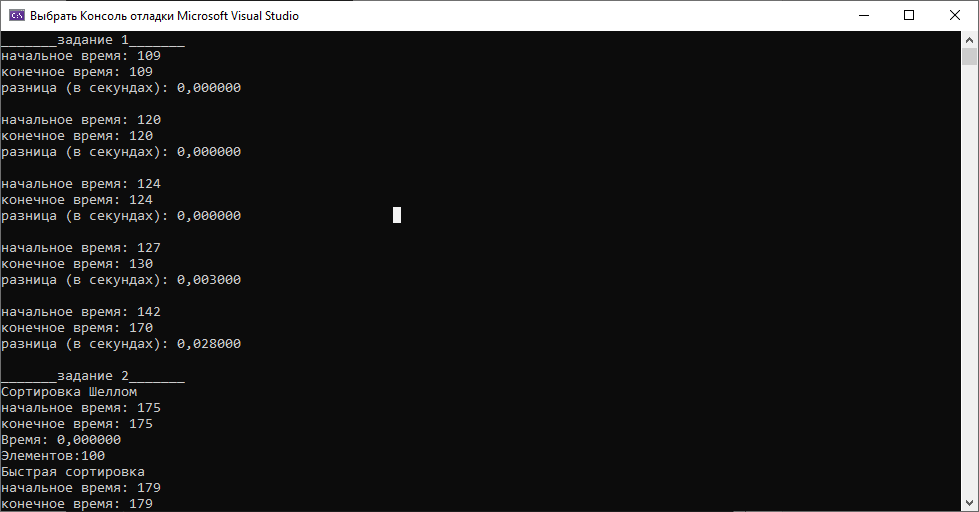


Рис. 1

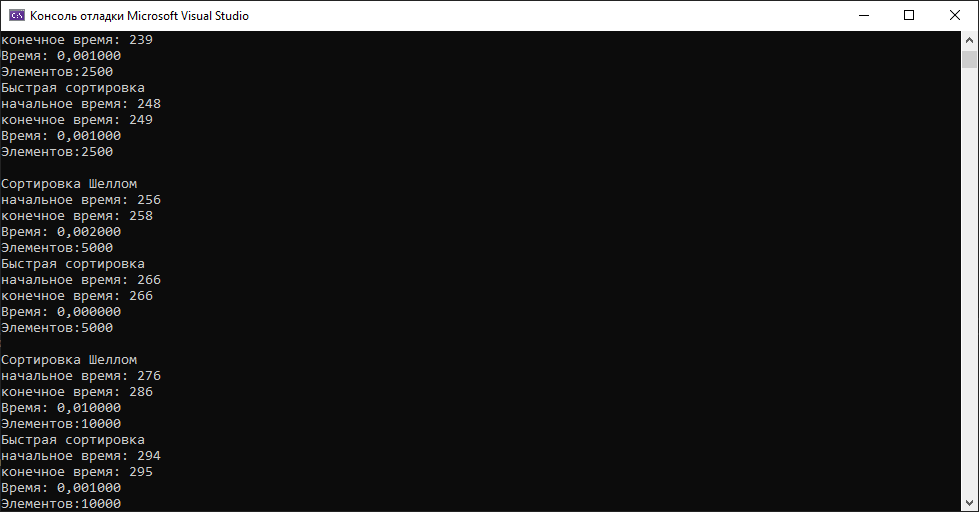


Рис.2

**Вывод:**

В ходе выполнения лабораторной работы мы научились оценивать время выполнения программы.

Ссылка на репозиторий: https://github.com/chif45/laba2\_logic.git