Пензенский государственный университет

Кафедра «Вычислительная техника»

**ОТЧЕТ**

по лабораторной работе №2

по дисциплине: «Логика и основы алгоритмизации в инженерных задачах.»

на тему: «Оценка времени выполнения программ.»

Выполнили:

студенты группы 21ВВ4

Колокольцева У. А.

Нагорная Д. А.

Принял:

Акифьев И. В.  
Юрова О. В.

Пенза, 2022

**Цель работы:** научиться применять средства, предоставляемые библиотекой time.h.

**Лабораторная работа:**

**Задание 1:**

Дана программа, вычисляющая произведение двух квадратных матриц:

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <iostream>

#include <locale>

using namespace std;

const unsigned int n = 100;

int a[n][n], b[n][n], c[n][n];

int main() {

setlocale(LC\_ALL, "rus");

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

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

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

int i = 0, j = 0, r;

start = time(NULL);

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

printf("Матрица A:\n\n");

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

{

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

{

a[i][j] = rand() % 10 + 1; // заполняем массив случайными числами

printf("%5d", a[i][j]);

}

printf("\n");

printf("\n");

}

start = time(NULL);

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

printf("Матрица B:\n\n");

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

{

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

{

b[i][j] = rand() % 10 + 5; // заполняем массив случайными числами

printf("%5d", b[i][j]);

}

printf("\n");

printf("\n");

}

printf("Матрица C = A \* B:\n\n");

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

{

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

{

c[i][j] = 0;

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

{

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

}

printf("%5d", c[i][j]); // Вывод значения элемента на экран:

}

printf("\n");

printf("\n");

}

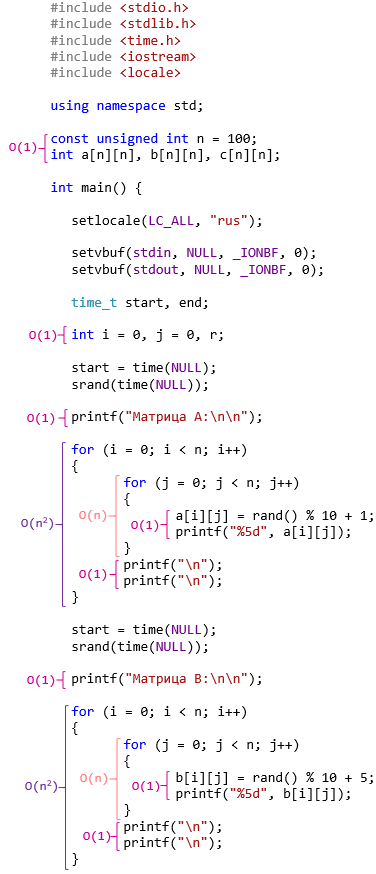
end = time(NULL);

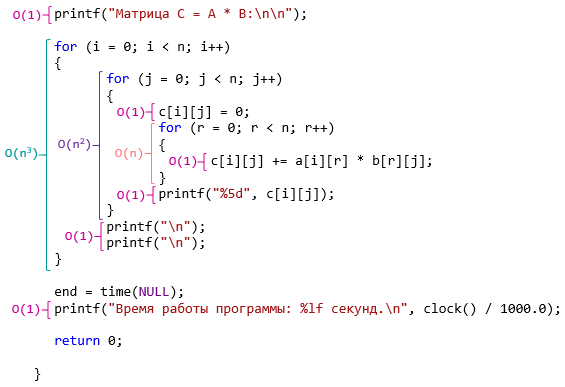
printf("Время работы программы: %lf секунд.\n", clock() / 1000.0);

return 0;

}

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



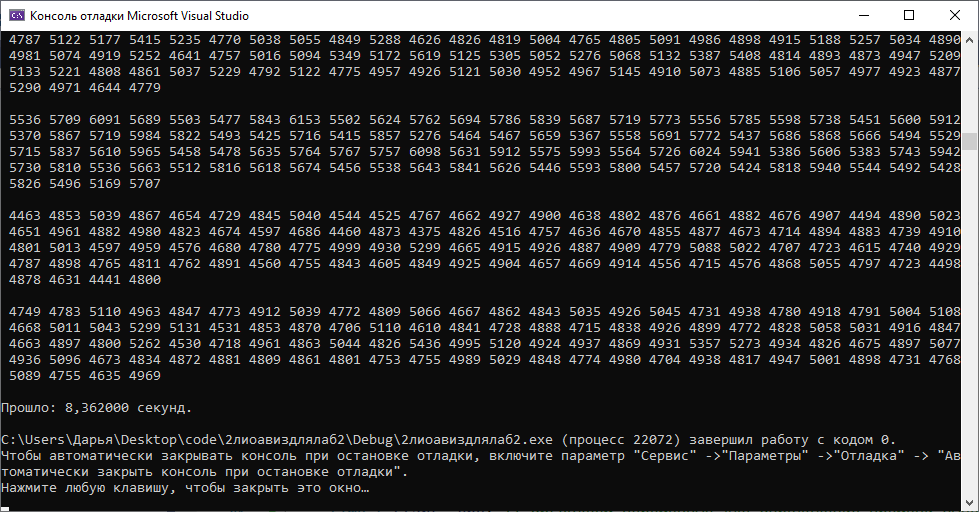


Сложность алгоритма равна O(n3).

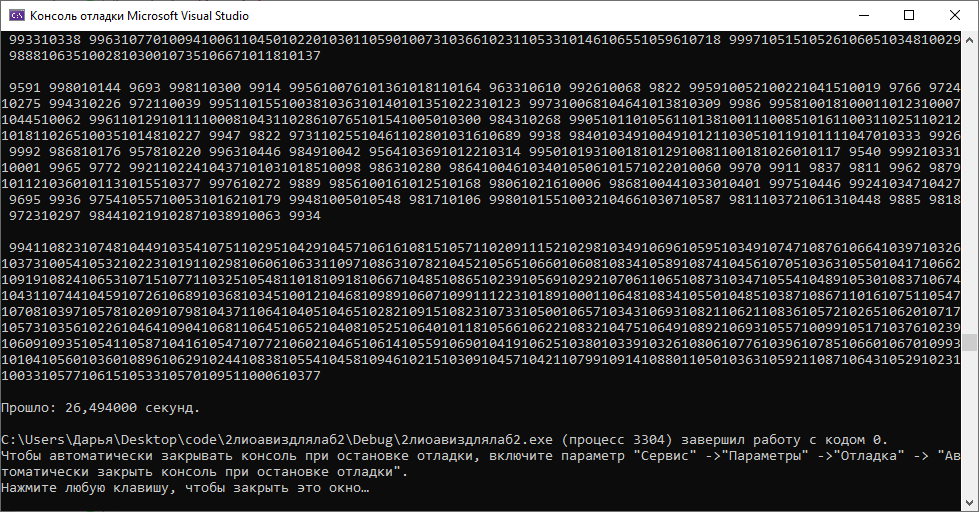
1. Оценить время выполнения программы и кода, выполняющего перемножение матриц, используя функции библиотеки time.h для матриц размерами от 100, 200, 400, 1000, 2000, 4000, 10000.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Количество элементов | 100 | 200 | 400 | 1000 | 2000 | 4000 | 10000 |
| Время, с | 8,362 | 26,494 | 78,945 | 729,282 | 2534,775 | 7378,439 | 21036,245 |

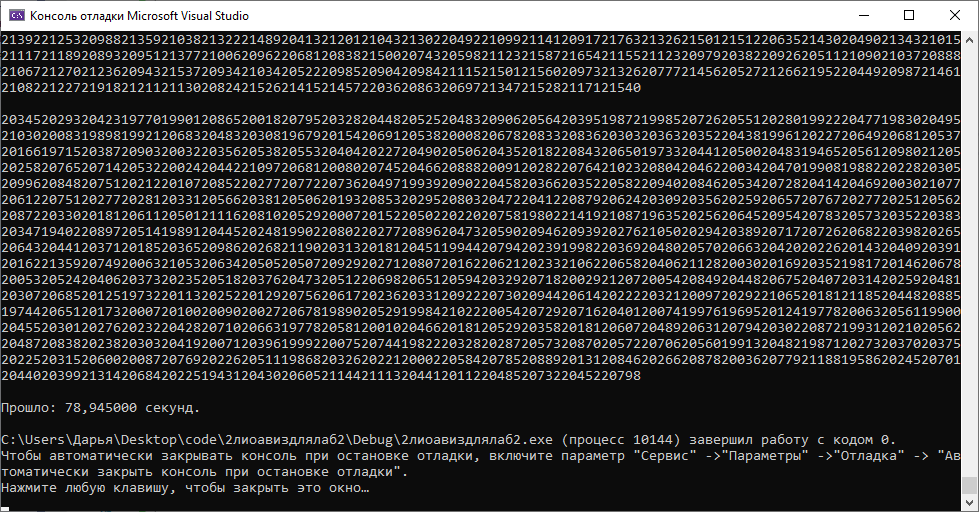
Матрицы A и B размером 100 × 100:



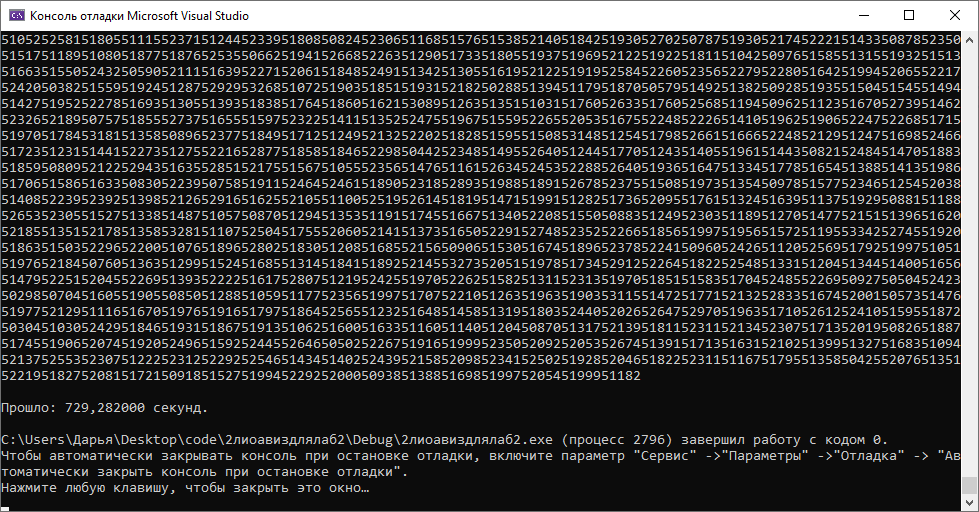
Матрицы A и B размером 200 × 200:



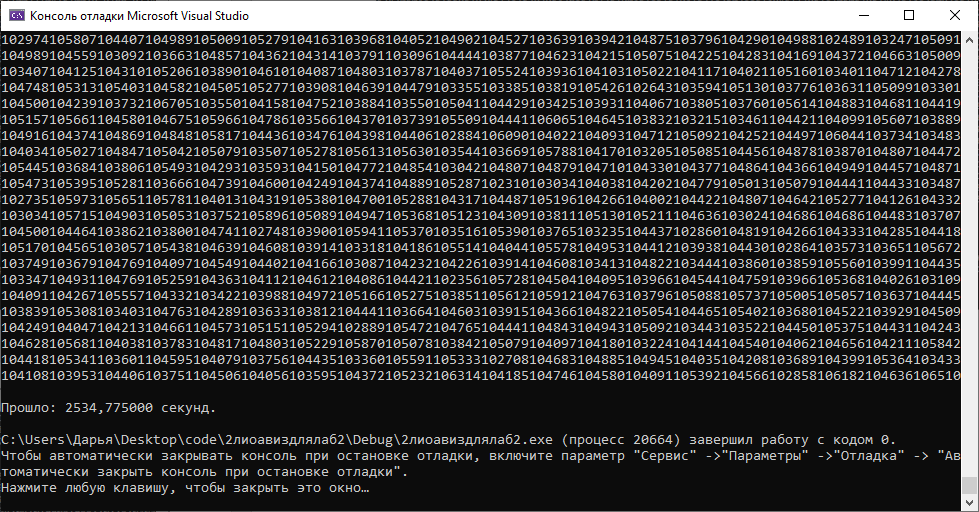
Матрицы A и B размером 400 × 400:



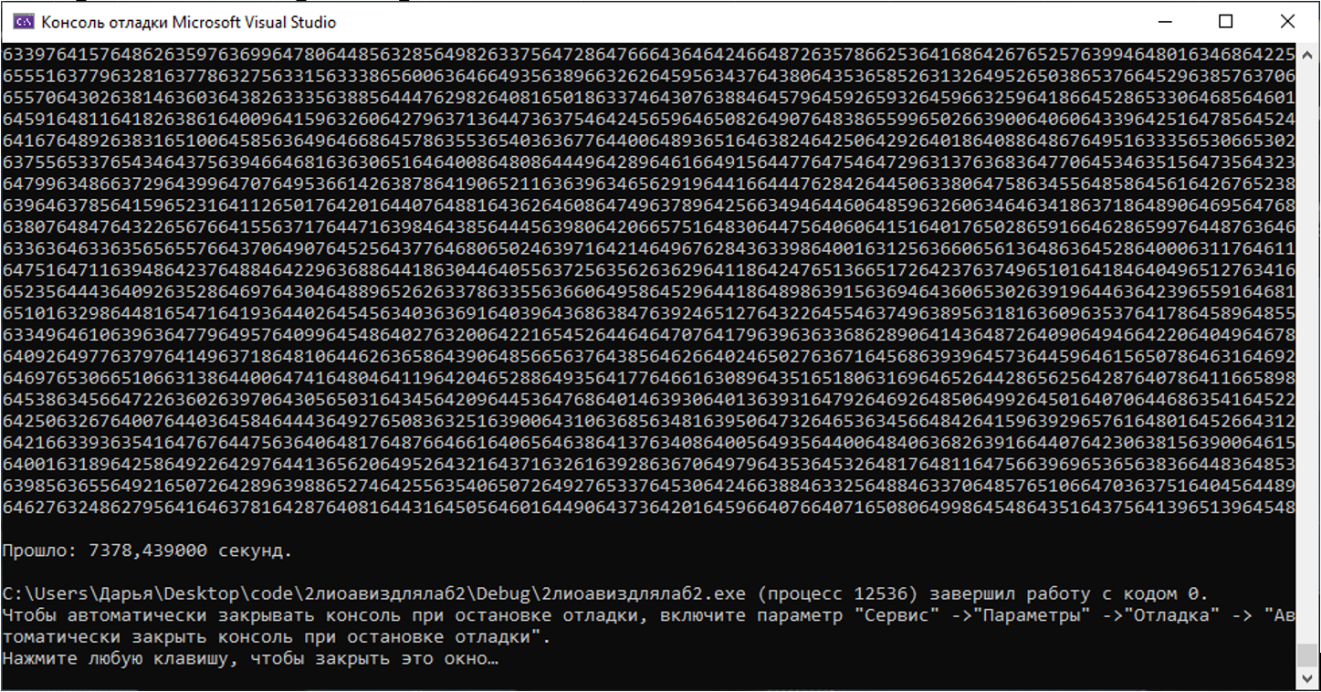
Матрицы A и B размером 1000 × 1000:



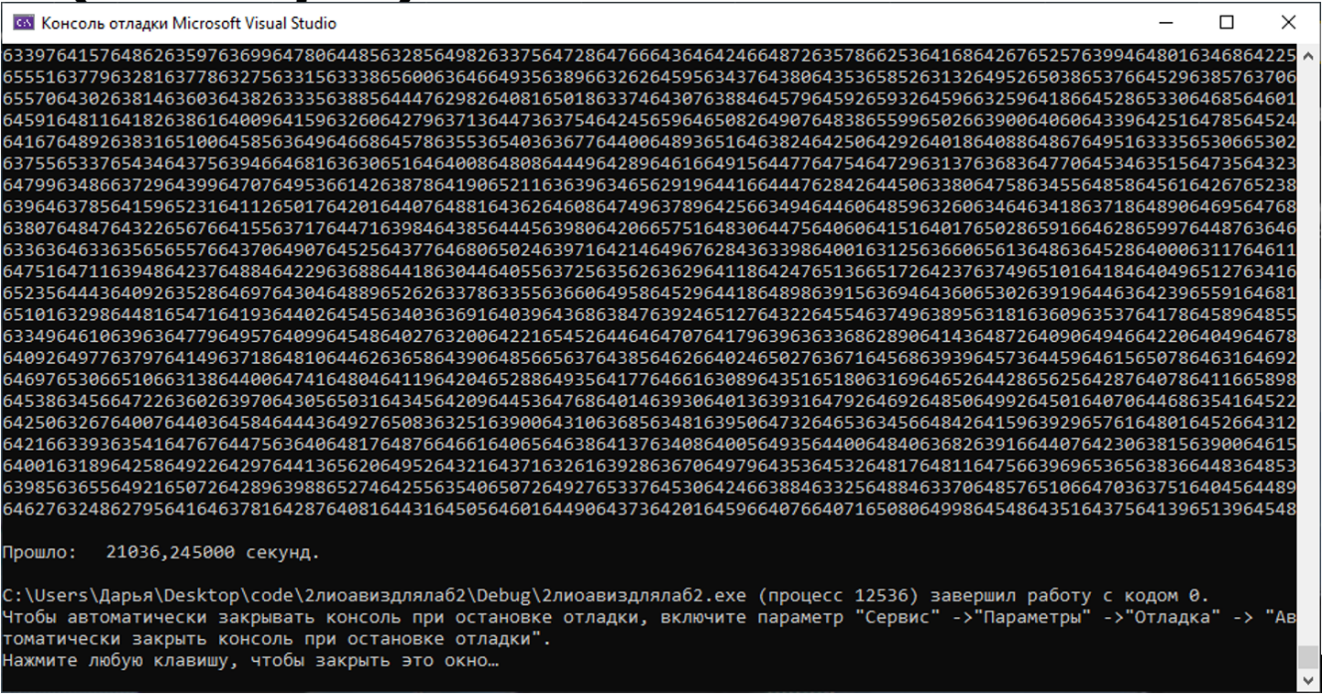
Матрицы A и B размером 2000 × 2000:



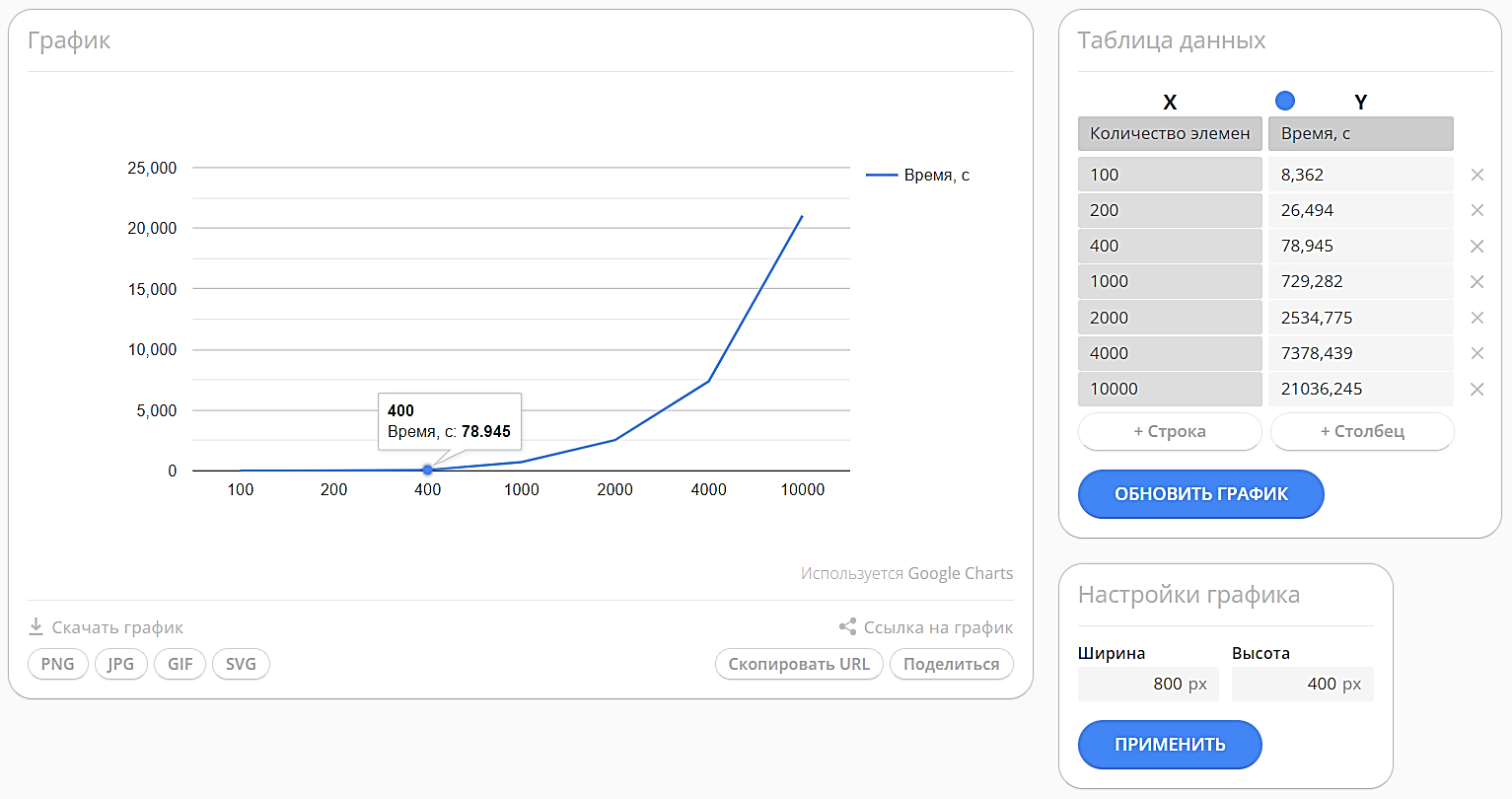
Матрицы A и B размером 4000 × 4000:



Матрицы A и B размером 10000 × 10000:



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



**Задание 2**:

Сортировка Шелла:

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <iostream>

#include <locale>

// Функция сортировки Шелла

void shell(int\* items, int count)

{

int x, a[8];

int i, j, gap, k;

a[0] = 3; a[1] = 7; a[2] = 19; a[3] = 35; a[4] = 64; a[5] = 91; a[6] = 128; a[7] = 547;

for (k = 0; k < 8; 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;

}

}

}

int main()

{

setlocale(LC\_ALL, "rus");

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

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

time\_t start, end;

int a[8];

a[0] = 3; a[1] = 7; a[2] = 19; a[3] = 35; a[4] = 64; a[5] = 91; a[6] = 128; a[7] = 547;

start = time(NULL);

printf("Исходный массив: ");

for (int i = 0; i < 8; i++)

{

printf("%2d ", a[i]);

}

shell(a, 8);

printf("\nПолученный массив: ");

for (int i = 0; i < 8; i++)

{

printf("%2d ", a[i]);

}

end = time(NULL);

printf("\nВремя работы программы: %lf секунд.\n", clock() / 1000.0);

return 0;

}

Быстрая сортировка:

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <iostream>

#include <locale>

using namespace std;

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);

}

int main()

{

setlocale(LC\_ALL, "rus");

srand(time(NULL));

int count = 8;

int a[] = {7, 35, 91, 547, 128, 64, 19, 3};

printf("Исходный массив:\n");

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

{

printf("%2d ", a[i]);

}

int first = 0, last = count - 1;

qs(a, first, last);

printf("\n\nРезультат:\n");

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

{

printf("%2d ", a[i]);

}

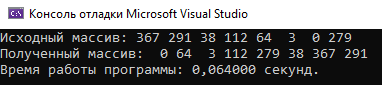
printf("\n\nВремя работы программы: %lf секунд.\n", clock() / 1000.0);

return 0;

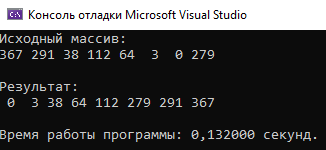
}

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

Сортировка Шелла:

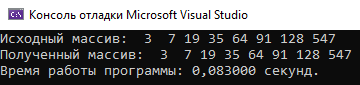


Быстрая сортировка:

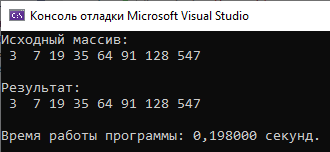


1. Оценить время работы каждого из реализованных алгоритмов на массиве, представляющем собой возрастающую последовательность чисел.

Сортировка Шелла:

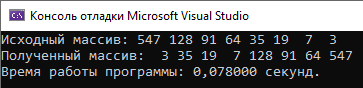


Быстрая сортировка:

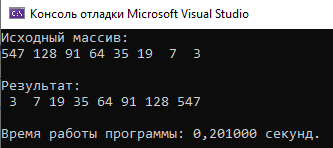


1. Оценить время работы каждого из реализованных алгоритмов на массиве, представляющем собой убывающую последовательность чисел.

Сортировка Шелла:

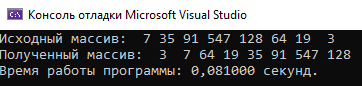


Быстрая сортировка:

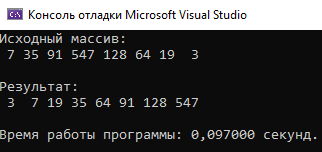


1. Оценить время работы каждого из реализованных алгоритмов на массиве, одна половина которого представляет собой возрастающую последовательность чисел, а вторая, – убывающую.

Сортировка Шелла:



Быстрая сортировка:



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

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <iostream>

#include <locale>

using namespace std;

int a[] = { 3,7,19,35,64,91,128,547 };

int compare(const void\* x1, const void\* x2) // функция сравнения элементов массива

{

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

}

int main()

{

setlocale(LC\_ALL, "rus");

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

int count = 8;

printf("Исходный массив: ");

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

{

printf("%2d ", a[i]);

}

printf("\n\nРезультат: ");

int first = 0, last = count - 1;

qsort(a, count, sizeof(int), compare);

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

{

printf("%2d ", a[i]);

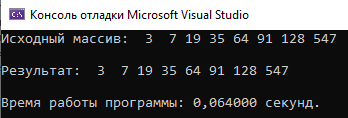
}

printf("\n\nВремя работы программы: %lf секунд.\n", clock() / 1000.0);

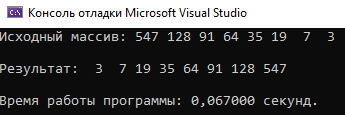
return 0;

}

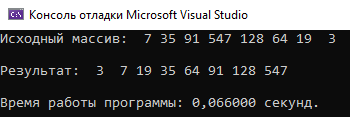
Возрастающая последовательность чисел:



Убывающая последовательность чисел:



Треугольная последовательность чисел:



Запись результатов в файл в виде таблицы:

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <iostream>

#include <locale>

// ФУНКЦИЯ СОРТИРОВКИ ШЕЛЛА

void shell(int\* items1, int count1)

{

int x1;

int i1, j1, gap, k;

int a[8];

a[0] = 3; a[1] = 7; a[2] = 19; a[3] = 35; a[4] = 64; a[5] = 91; a[6] = 128; a[7] = 547;

int a1[8];

a1[0] = 547; a1[1] = 128; a1[2] = 91; a1[3] = 64; a1[4] = 35; a1[5] = 19; a1[6] = 7; a1[7] = 3;

int a2[8];

a2[0] = 7; a2[1] = 35; a2[2] = 91; a2[3] = 547; a2[4] = 128; a2[5] = 64; a2[6] = 19; a2[7] = 3;

for (k = 0; k < 8; k++) {

gap = a[k];

for (i1 = gap; i1 < count1; ++i1) {

x1 = items1[i1];

for (j1 = i1 - gap; (x1 < items1[j1]) && (j1 >= 0); j1 = j1 - gap)

items1[j1 + gap] = items1[j1];

items1[j1 + gap] = x1;

}

}

}

// ФУНКЦИЯ БЫСТРОЙ СОРТИРОВКИ

void qs(int\* items2, int left, int right) //вызов функции: qs(items, 0, count-1);

{

int a[8];

a[0] = 3; a[1] = 7; a[2] = 19; a[3] = 35; a[4] = 64; a[5] = 91; a[6] = 128; a[7] = 547;

int a1[8];

a1[0] = 547; a1[1] = 128; a1[2] = 91; a1[3] = 64; a1[4] = 35; a1[5] = 19; a1[6] = 7; a1[7] = 3;

int a2[8];

a2[0] = 7; a2[1] = 35; a2[2] = 91; a2[3] = 547; a2[4] = 128; a2[5] = 64; a2[6] = 19; a2[7] = 3;

int i2, j2;

int x2, y2;

i2 = left; j2 = right;

/\* выбор компаранда \*/

x2 = items2[(left + right) / 2];

do {

while ((items2[i2] < x2) && (i2 < right)) i2++;

while ((x2 < items2[j2]) && (j2 > left)) j2--;

if (i2 <= j2) {

y2 = items2[i2];

items2[i2] = items2[j2];

items2[j2] = y2;

i2++; j2--;

}

} while (i2 <= j2);

if (left < j2) qs(items2, left, j2);

if (i2 < right) qs(items2, i2, right);

}

// ФУНКЦИЯ QSORT

int aq[] = { 3,7,19,35,64,91,128,547 };

int a1q[] = { 547,128,91,64,35,19,7,3 };

int a2q[] = { 7,35,91,547,128,64,19,3 };

int compare(const void\* x3\_1, const void\* x3\_2) // функция сравнения элементов массива

{

return (\*(int\*)x3\_1 - \*(int\*)x3\_2);

}

int main()

{

setlocale(LC\_ALL, "rus");

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

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

char inputFile[255];

FILE\* input;

int a[8];

a[0] = 3; a[1] = 7; a[2] = 19; a[3] = 35; a[4] = 64; a[5] = 91; a[6] = 128; a[7] = 547;

int a1[8];

a1[0] = 547; a1[1] = 128; a1[2] = 91; a1[3] = 64; a1[4] = 35; a1[5] = 19; a1[6] = 7; a1[7] = 3;

int a2[8];

a2[0] = 7; a2[1] = 35; a2[2] = 91; a2[3] = 547; a2[4] = 128; a2[5] = 64; a2[6] = 19; a2[7] = 3;

/\*-----------------------------------------------------------------\*/

do {

printf("Введите имя файла, в который необходимо записать результаты сравнения времени работы всех сортировок: ");

scanf("%s", inputFile);

input = fopen(inputFile, "w");

if (input != NULL)

break;

printf("Ошибка! Не удалось открыть файл! Повторите ввод\n\n");

} while (true);

/\*-----------------------------------------------------------------\*/

//----------------------------------------------------------------------

printf("\n\n1.1 Сортировка Шелла:\n");

clock\_t start11, end11;

start11 = clock();

printf("Исходный массив при возрастающей последовательности: ");

for (int i = 0; i < 8; i++)

{

printf("%2d ", a[i]);

}

shell(a, 8);

printf("\nПолученный массив: ");

for (int i = 0; i < 8; i++)

{

printf("%2d ", a[i]);

}

end11 = clock();

double extime11 = (double)(end11 - start11) / clock();

printf("\n\tВремя работы программы: %lf секунд.\n\n", extime11);

//----------------------------------------------------------------------

//----------------------------------------------------------------------

printf("\n\n1.2 Сортировка Шелла:\n");

clock\_t start12, end12;

start12 = clock();

printf("Исходный массив при убывающей последовательности: ");

for (int i = 0; i < 8; i++)

{

printf("%2d ", a1[i]);

}

shell(a1, 8);

printf("\nПолученный массив: ");

for (int i = 0; i < 8; i++)

{

printf("%2d ", a[i]);

}

end12 = clock();

double extime12 = (double)(end12 - start12) / clock();

printf("\n\tВремя работы программы: %lf секунд.\n\n", extime12);

//----------------------------------------------------------------------

//----------------------------------------------------------------------

printf("\n\n1.3 Сортировка Шелла:\n");

clock\_t start13, end13;

start13 = clock();

printf("Исходный массив при треугольной последовательности: ");

for (int i = 0; i < 8; i++)

{

printf("%2d ", a2[i]);

}

shell(a2, 8);

printf("\nПолученный массив: ");

for (int i = 0; i < 8; i++)

{

printf("%2d ", a[i]);

}

end13 = clock();

double extime13 = (double)(end13 - start13) / clock();

printf("\n\tВремя работы программы: %lf секунд.\n\n", extime13);

//----------------------------------------------------------------------

int first1 = 0, last1 = 8 - 1;

//----------------------------------------------------------------------

printf("\n\n2.1 Быстрая сортировка:\n");

clock\_t start21, end21;

start21 = clock();

printf("Исходный массив при возрастающей последовательности: : ");

for (int i = 0; i < 8; i++)

{

printf("%2d ", a[i]);

}

qs(a, first1, last1);

printf("\nПолученный массив: ");

for (int i = 0; i < 8; i++)

{

printf("%2d ", a[i]);

}

end21 = clock();

double extime21 = (double)(end21 - start21) / clock();

printf("\n\tВремя работы программы: %lf секунд.\n\n", extime21);

//----------------------------------------------------------------------

//----------------------------------------------------------------------

printf("\n\n2.2 Быстрая сортировка:\n");

clock\_t start22, end22;

start22 = clock();

printf("Исходный массив при убывающей последовательности: ");

for (int i = 0; i < 8; i++)

{

printf("%2d ", a1[i]);

}

qs(a1, first1, last1);

printf("\nПолученный массив: ");

for (int i = 0; i < 8; i++)

{

printf("%2d ", a1[i]);

}

end22 = clock();

double extime22 = (double)(end22 - start22) / clock();

printf("\n\tВремя работы программы: %lf секунд.\n\n", extime22);

//----------------------------------------------------------------------

//----------------------------------------------------------------------

printf("\n\n2.3 Быстрая сортировка: \n");

clock\_t start23, end23;

start23 = clock();

printf("Исходный массив при треугольной последовательности: ");

for (int i = 0; i < 8; i++)

{

printf("%2d ", a2[i]);

}

qs(a2, first1, last1);

printf("\nПолученный массив: ");

for (int i = 0; i < 8; i++)

{

printf("%2d ", a2[i]);

}

end23 = clock();

double extime23 = (double)(end23 - start23) / clock();

printf("\n\tВремя работы программы: %lf секунд.\n\n", extime23);

//----------------------------------------------------------------------

int count2 = 8;

int first2 = 0, last2 = count2 - 1;

//----------------------------------------------------------------------

printf("\n\n3.1 Сортировка с использованием функции qsort:\n");

clock\_t start31, end31;

start31 = clock();

printf("Исходный массив при возрастающей последовательности: ");

for (int i = 0; i < 8; i++)

{

printf("%2d ", aq[i]);

}

qsort(aq, count2, sizeof(int), compare);

printf("\nПолученный массив: ");

for (int i = 0; i < 8; i++)

{

printf("%2d ", aq[i]);

}

end31 = clock();

double extime31 = (double)(end31 - start31) / clock();

printf("\n\tВремя работы программы: %lf секунд.\n\n", extime31);

//----------------------------------------------------------------------

//----------------------------------------------------------------------

printf("\n\n3.2 Сортировка с использованием функции qsort:\n");

clock\_t start32, end32;

start32 = clock();

printf("Исходный массив при убывающей последовательности: ");

for (int i = 0; i < 8; i++)

{

printf("%2d ", a1q[i]);

}

qsort(a1q, count2, sizeof(int), compare);

printf("\nПолученный массив: ");

for (int i = 0; i < 8; i++)

{

printf("%2d ", a1q[i]);

}

end32 = clock();

double extime32 = (double)(end32 - start32) / clock();

printf("\n\tВремя работы программы: %lf секунд.\n\n", extime32);

//----------------------------------------------------------------------

//----------------------------------------------------------------------

printf("\n\n3.3 Сортировка с использованием функции qsort:\n");

clock\_t start33, end33;

start33 = clock();

printf("Исходный массив при треугольной последовательности: ");

for (int i = 0; i < 8; i++)

{

printf("%2d ", a2q[i]);

}

qsort(a2q, count2, sizeof(int), compare);

printf("\nПолученный массив: ");

for (int i = 0; i < 8; i++)

{

printf("%2d ", a2q[i]);

}

end33 = clock();

double extime33 = (double)(end33 - start33) / clock();

printf("\n\tВремя работы программы: %lf секунд.\n\n", extime33);

//----------------------------------------------------------------------

fprintf(input, "Результат:\n\n");

fprintf(input, " ----------------------------------------------------------------------------------\n");

fprintf(input, " | | Сортировка Шелла | Быстрая Сортировка | Функция qsort |\n");

fprintf(input, " |------------------|--------------------|----------------------|-----------------|\n");

fprintf(input, " | Возраст. послед. | %lf с | %lf с | %lf с |\n", extime11, extime21, extime31);

fprintf(input, " |------------------|--------------------|----------------------|-----------------|\n");

fprintf(input, " | Убывающ. послед. | %lf c | %lf c | %lf c |\n", extime12, extime22, extime32);

fprintf(input, " |------------------|--------------------|----------------------|-----------------|\n");

fprintf(input, " | Треугол. послед. | %lf c | %lf c | %lf c |\n", extime13, extime23, extime33);

fprintf(input, " ----------------------------------------------------------------------------------\n");

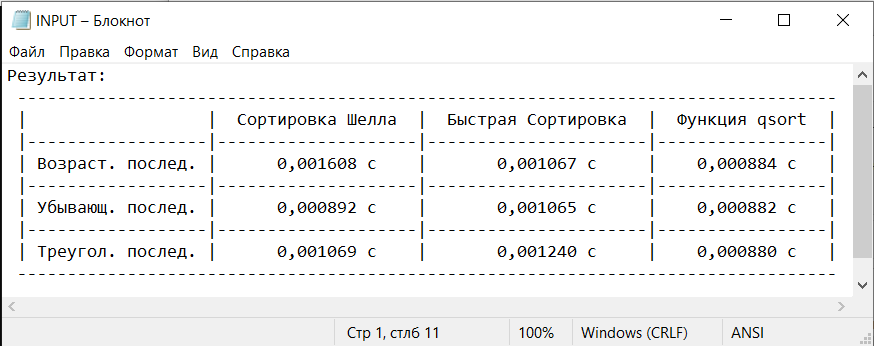
fclose(input);

return 0;

}

Приложение к заданию 2:

|  |  |  |  |
| --- | --- | --- | --- |
|  | Сортировка Шелла | Быстрая сортировка | Функция qsort |
| Возрастающая | 0,083 с | 0,198 с | 0,064 с |
| Убывающая | 0,078 с | 0,201 с | 0,067 с |
| Треугольная | 0,081 с | 0,097 с | 0,066 с |



Вывод: мы научились применять средства, предоставляемые библиотекой time.h. Научились определять сложность алгоритма, рассмотрели зависимость времени от количества элементов в массиве с помощью графика. Пронаблюдали, что алгоритм быстрой сортировки работает медленнее, чем алгоритм Шелла, но самым быстрым в выполнении показала функция qsort.