Министерство и науки высшего образования

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»

Работу выполнил студент группы 21ВВ2:

Асянов И. К.

Приняли:

Юрова О.В.

Митрохин М.А.

Пенза 2022

Задание 1.

Была написана программа для вычисления времени перемножения матриц заданного размера.

Листинг

#include <iostream>

#include <cstdlib>

#include <ctime>

using namespace std;

const int SIZE = 100;

//const int SIZE = 200;

//const int SIZE = 400;

//const int SIZE = 1000;

//const int SIZE = 2000;

//const int SIZE = 4000;

//const int SIZE = 10000;

const int R\_MIN = -5;

const int R\_MAX = 5;

int arr1[SIZE][SIZE];

int arr2[SIZE][SIZE];

int arr3[SIZE][SIZE];

int elem\_c = 0;

int r;

int i, j;

int x, y;

int main()

{

setlocale(0, "rus");

srand(time(NULL));

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

{

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

{

arr1[i][j] = rand() % (R\_MAX - R\_MIN + 1) + R\_MIN; // или: arr[i][j] = rand()%101-50;

}

}

srand(time(NULL));

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

{

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

{

arr2[i][j] = rand() % (R\_MAX - R\_MIN + 1) + R\_MIN; // или: arr[i][j] = rand()%101-50;

}

}

//cout << "Первый массив:" << endl;

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

{

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

{

//cout << arr1[i][j] << '\t';

}

//cout << endl;

}

//cout << "Второй массив:" << endl;

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

{

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

{

//cout << arr2[i][j] << '\t';

}

//cout << endl;

}

cout << "Произведения матриц : "<< SIZE << " \* " << SIZE << endl;

unsigned int start\_time1 = clock();

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

{

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

{

elem\_c = 0;

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

{

elem\_c = elem\_c + arr1[i][r] \* arr2[r][j];

arr3[i][j] = elem\_c;

}

//cout << arr3[i][j]<<'\t';

}

//cout << endl;

}

unsigned int end\_time1 = clock();

cout << "Время выполнения кода, : " << (end\_time1 - start\_time1)/1000.0<<" c"<<endl;

system("pause");

return 0;

}

O(n3), так как содержит 3 вложенных цикла.

|  |  |  |
| --- | --- | --- |
| Размер матрицы(n\*n), n | 100 | 200 |
| Время, t |  |  |

|  |  |  |
| --- | --- | --- |
| Размер матрицы(n\*n), n | 400 | 1000 |
| Время, t |  |  |

|  |  |  |
| --- | --- | --- |
| Размер матрицы(n\*n), n | 2000 | 4000 |
| Время, t |  |  |

|  |  |
| --- | --- |
| Размер матрицы(n\*n), n | 10000 |
| Время, t |  |

Таким образом, график подтвердил О-символику.

При увеличении количества элементов, время обработки также увеличивается

Задание 2.

Были написаны программы реализаций следующих сортировок: сортировка Шелла, быстрая сортировка, сортировка функцией sort().

Листинг

Сортировка Шелла

#include<iostream>

#include<cstdlib>

using namespace std;

void Qs(int Arr[], int left, int right);

int i = 0;

int n;

int pr = 0;

int main()

{

setlocale(0, "rus");

cout << "Введите размер массива ";

cin >> n;

int\* Arr = new int[n];

srand(time(0));

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

{

Arr[i] = rand() % 101-50; //рандомный набор;

//cout << i + 1 << " элемент > "<<Arr[i]<<endl;

}

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

{

Arr[i] = (rand() % 5) + pr; //рандомный набор на возрастании;

pr = Arr[i];

//cout << i + 1 << " элемент > " << Arr[i] << endl;

}

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

{

Arr[i] = -(rand() % 5) + pr; // рандомный набор на убывании;

pr = Arr[i];

//cout << i + 1 << " элемент > " << Arr[i] << endl;

}

\*/for (i = 0; i < (n / 2); i++)

{

Arr[i] = (rand() % 5) + pr;

pr = Arr[i];

//cout << i + 1 << " элемент > " << Arr[i] << endl; // рандомный набор на возрастании и убывании;

}

for (i = (n - n / 2); i < n; i++)

{

Arr[i] = -(rand() % 5) + pr;

pr = Arr[i];

//cout << i + 1 << " элемент > "<<Arr[i]<<endl;

}

//cout << "До сортировки: ";

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

//cout << Arr[i] << " ";

}

unsigned int start\_time = clock();

cout << endl;

Qs(Arr, 0, (n - 1));

//cout << "После ";

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

//cout << Arr[i] << " ";

}

cout << endl;

delete[] Arr;

unsigned int end\_time = clock();

cout << "Время выполнения кода: " << (end\_time - start\_time)/1000.0 << " c";

}

void Qs(int Arr[], int left, int right)

{

int i = left;

int j = right;

int y = 0;

/\* опорный элемент или компаранда \*/

int z = Arr[(left + right) / 2];

/\* разделение \*/

do {

/\* находим элемент левее \*/

while ((Arr[i] < z) && (i < right)) i++;

/\* находим элемент правее \*/

while ((Arr[j] > z) && (j > left)) j--;

if (i <= j) {

/\* меняем местами 2 элемента \*/

y = Arr[i];

Arr[i] = Arr[j];

Arr[j] = y;

i++;

j--;

}

} while (i <= j);

/\*рекурсия\*/

if (left < j) {

Qs(Arr, left, j);

}

if (i < right) {

Qs(Arr, i, right);

}

}

Быстрая сортировка

#include<iostream>

#include<cstdlib>

using namespace std;

void Qs(int Arr[], int left, int right);

int i = 0;

int n;

int pr = 0;

int main()

{

setlocale(0, "rus");

cout << "Введите размер массива ";

cin >> n;

int\* Arr = new int[n];

srand(time(0));

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

{

Arr[i] = rand() % 101-50; //рандомный набор;

//cout << i + 1 << " элемент > "<<Arr[i]<<endl;

}

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

{

Arr[i] = (rand() % 5) + pr; //рандомный набор на возрастании;

pr = Arr[i];

//cout << i + 1 << " элемент > " << Arr[i] << endl;

}

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

{

Arr[i] = -(rand() % 5) + pr; // рандомный набор на убывании;

pr = Arr[i];

//cout << i + 1 << " элемент > " << Arr[i] << endl;

}

\*/for (i = 0; i < (n / 2); i++)

{

Arr[i] = (rand() % 5) + pr;

pr = Arr[i];

//cout << i + 1 << " элемент > " << Arr[i] << endl; // рандомный набор на возрастании и убывании;

}

for (i = (n - n / 2); i < n; i++)

{

Arr[i] = -(rand() % 5) + pr;

pr = Arr[i];

//cout << i + 1 << " элемент > "<<Arr[i]<<endl;

}

//cout << "До сортировки: ";

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

//cout << Arr[i] << " ";

}

unsigned int start\_time = clock();

cout << endl;

Qs(Arr, 0, (n - 1));

//cout << "После ";

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

//cout << Arr[i] << " ";

}

cout << endl;

delete[] Arr;

unsigned int end\_time = clock();

cout << "Время выполнения кода: " << (end\_time - start\_time)/1000.0 << " c";

}

void Qs(int Arr[], int left, int right)

{

int i = left;

int j = right;

int y = 0;

/\* опорный элемент или компаранда \*/

int z = Arr[(left + right) / 2];

/\* разделение \*/

do {

/\* находим элемент левее \*/

while ((Arr[i] < z) && (i < right)) i++;

/\* находим элемент правее \*/

while ((Arr[j] > z) && (j > left)) j--;

if (i <= j) {

/\* меняем местами 2 элемента \*/

y = Arr[i];

Arr[i] = Arr[j];

Arr[j] = y;

i++;

j--;

}

} while (i <= j);

/\*рекурсия\*/

if (left < j) {

Qs(Arr, left, j);

}

if (i < right) {

Qs(Arr, i, right);

}

}

Сортировка функцией sort()

#include <iostream>

#include <cstdlib>

using namespace std;

int compare(const void\* a, const void\* b)

{

const int\* x = (int\*)a;

const int\* y = (int\*)b;

if (\*x > \*y)

return 1;

else if(\*x < \*y)

return -1;

return 0;

}

int num;

int pr = 0;

int main()

{

setlocale(0, "rus");

srand(time(0));

cin>> num;

int\* Arr = new int[num];

/\*

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

{

Arr[i] = rand() % 101-50; //рандомный набор;

//cout << i + 1 << " элемент > "<<Arr[i]<<endl;

}

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

{

Arr[i] = (rand() % 5) + pr; //рандомный набор на возрастании;

pr = Arr[i];

//cout << i + 1 << " элемент > " << Arr[i] << endl;

}

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

{

Arr[i] = -(rand() % 5) + pr; // рандомный набор на убывании;

pr = Arr[i];

//cout << i + 1 << " элемент > " << Arr[i] << endl;

}

\*/

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

{

Arr[i] = (rand() % 5) + pr;

pr = Arr[i];

//cout << i + 1 << " элемент > " << Arr[i] << endl; // рандомный набор на возрастании и убывании;

}

for (int i = (num - num / 2); i < num; i++)

{

Arr[i] = -(rand() % 5) + pr;

pr = Arr[i];

//cout << i + 1 << " элемент > "<<A[i]<<endl;

}

unsigned int starttime = clock();

qsort(Arr, num, sizeof(int), compare);

cout << endl << endl;

cout << "сортировка" << endl;

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

//cout << Arr[i] << " ";

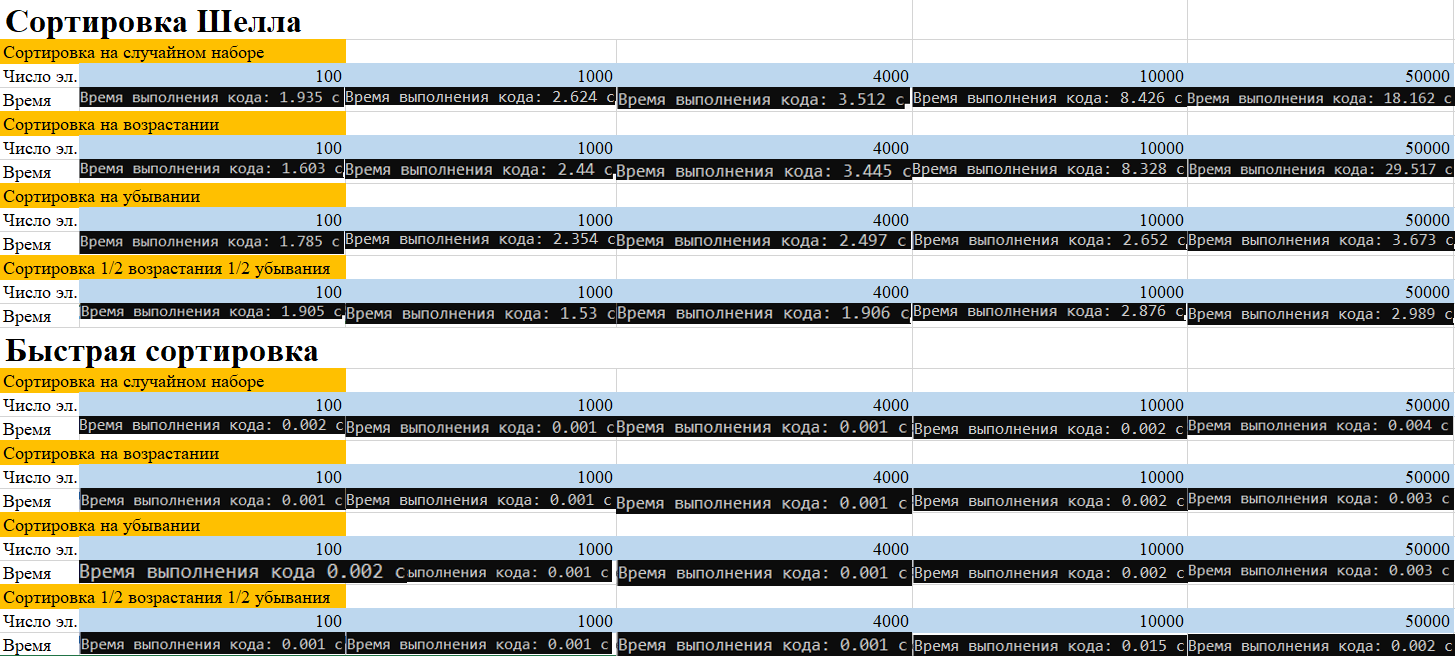
}

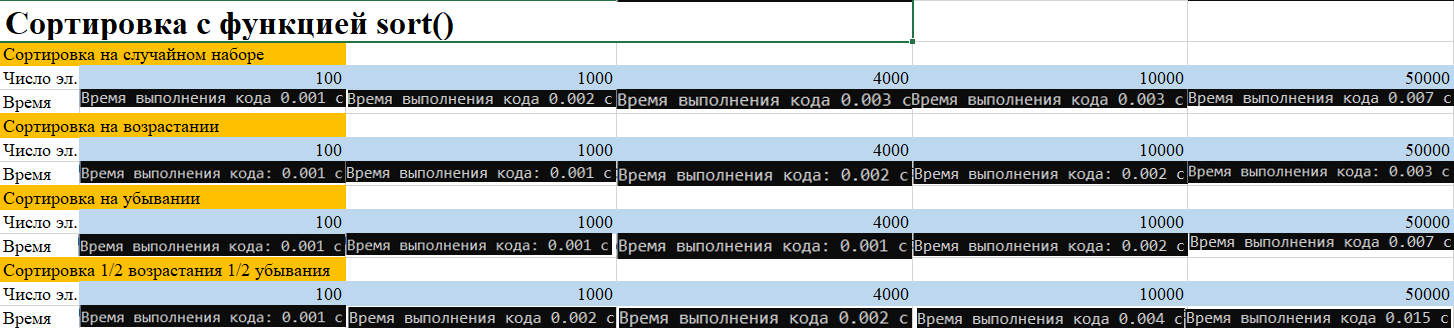
unsigned int endtime = clock();

cout <<"Время выполнения кода "<< (endtime - starttime) / 1000.0<<" c";

return 0;

}





**Вывод:**

Среди сортировок: Шелла, быстрой и qsort , самой долгой оказалась

сортировка Шелла. Быстрая сортировка показала наилучший результат по времени. Во всех случаях сортировки быстрее справлялись на уже упорядоченном по возрастанию или убыванию массиве.