Пензенский государственный университет

Кафедра «Вычислительная техника»

**Отчёт**

По лабораторной работе №2

по дисциплине: «Логика и основы оптимизации в инженерных задачах»

на тему «Оценка времени выполнения программ»

Выполнили студенты группы 19ВВ2:

Муромский Д.А.

Кобзев М.И.

Приняли:

Митрохин М. А.

Юрова О.В.

Пенза 2020

**Практическая часть**

Дана программа, вычисляющая произведение двух матриц:

**#include** <stdio.h>

**#include** <stdlib.h>

**#include** <time.h>

**int** **main**(**void**)

{

**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<200)

{

**while**(j<200)

{

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<200)

{

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

j++;

}

i++;

}

**for**(i=0;i<200;i++)

{

**for**(j=0;j<200;j++)

{

elem\_c=0;

**for**(r=0;r<200;r++)

{

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

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

}

}

}

**return**(0);

}

**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.
4. O- символика данной программы: O(n3 + 2n2)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| N | 100 | 200 | 400 | 1000 | 2000 | 4000 | 10000 |
| 1 | 3 | 24 | 213 | 4012 | 57390 | 12.4 мин | 3 часа |
| 2 | 3 | 23 | 214 | 4027 | 57886 |
| 3 | 3 | 28 | 218 | 3959 | 53760 |



Даны реализации алгоритмов сортировки Шелла и быстрой сортировки:

**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);

}

**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.
6. Оценка производилась на массиве из 100 000 элементов заполненном случайными числами.

|  |  |  |  |
| --- | --- | --- | --- |
| Шелл | 743 | 830 | 781 |
| Qsort | 6 | 5 | 6 |

1. Оценка производилась на массиве из 100 000 элементов отсортированного по возрастанию.

|  |  |  |  |
| --- | --- | --- | --- |
| Шелл | 919 | 761 | 780 |
| Qsort | 6 | 6 | 5 |

1. Оценка производилась на массиве из 100 000 элементов отсортированного по убыванию.

|  |  |  |  |
| --- | --- | --- | --- |
| Шелл | 1448 | 1442 | 1641 |
| Qsort | 5 | 5 | 5 |

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

|  |  |  |  |
| --- | --- | --- | --- |
| Шелл | 796 | 791 | 756 |
| Qsort | 7 | 8 | 7 |

1. Оценка времени работы стандартной функции qsort.

|  |  |  |  |
| --- | --- | --- | --- |
| rand | 11 | 11 | 11 |
| >> | 10 | 11 | 11 |
| << | 11 | 11 | 12 |
| >><< | 23 | 22 | 24 |