

Практическая работа по теме:  
«Java Core(Types, NIO, Collections, Threads)»

выполнил:  
Бузов Артур

## 1. Особенности реализации.

Для умножения матриц использовал стандартный алгоритм умножения со сложностью  $O(n^3)$ .

Протестировал умножение матриц размерностью 1000x1000, используя для хранения чисел массивы `double [][]` и `ArrayList<ArrayList<Double>>` (табл. 1).

Таблица 1 – Временя умножения при использовании разных типов массивов

Размер матрицы 1000x1000		
	Тип массива	
Количество потоков	<code>double [][]</code>	<code>ArrayList&lt;ArrayList&lt;Double&gt;&gt;</code>
1	46с.	99с.
2	21с.	45с.

Как видно из таблицы умножение матриц с использованием `ArrayList` значительно медленнее, поэтому в дальнейшем все остальные тесты я проводил с `double [][]`.

Многопоточное умножение реализовал по следующей схеме:

- создается группа потоков, им передается умножаемые матрицы ( $A$ ,  $B$ ) и матрица ( $C$ ), в которую будет записываться результат;
- каждый поток берет одну строку матрицы  $C$  через синхронизированный метод и увеличивает общее число строк (*rowForTpread*) взятых в работу на 1;
- потоки перед тем, как взять новую строку матрицы  $C$  в работку получают значение переменной *rowForTpread*, если она равна числу строк матрицы  $C$  – то прекращают свою работу.

Провел эксперимент для потоков с количеством строк, которые они берут для себя в работу (табл. 2). Проводил по три замера для каждого случая.

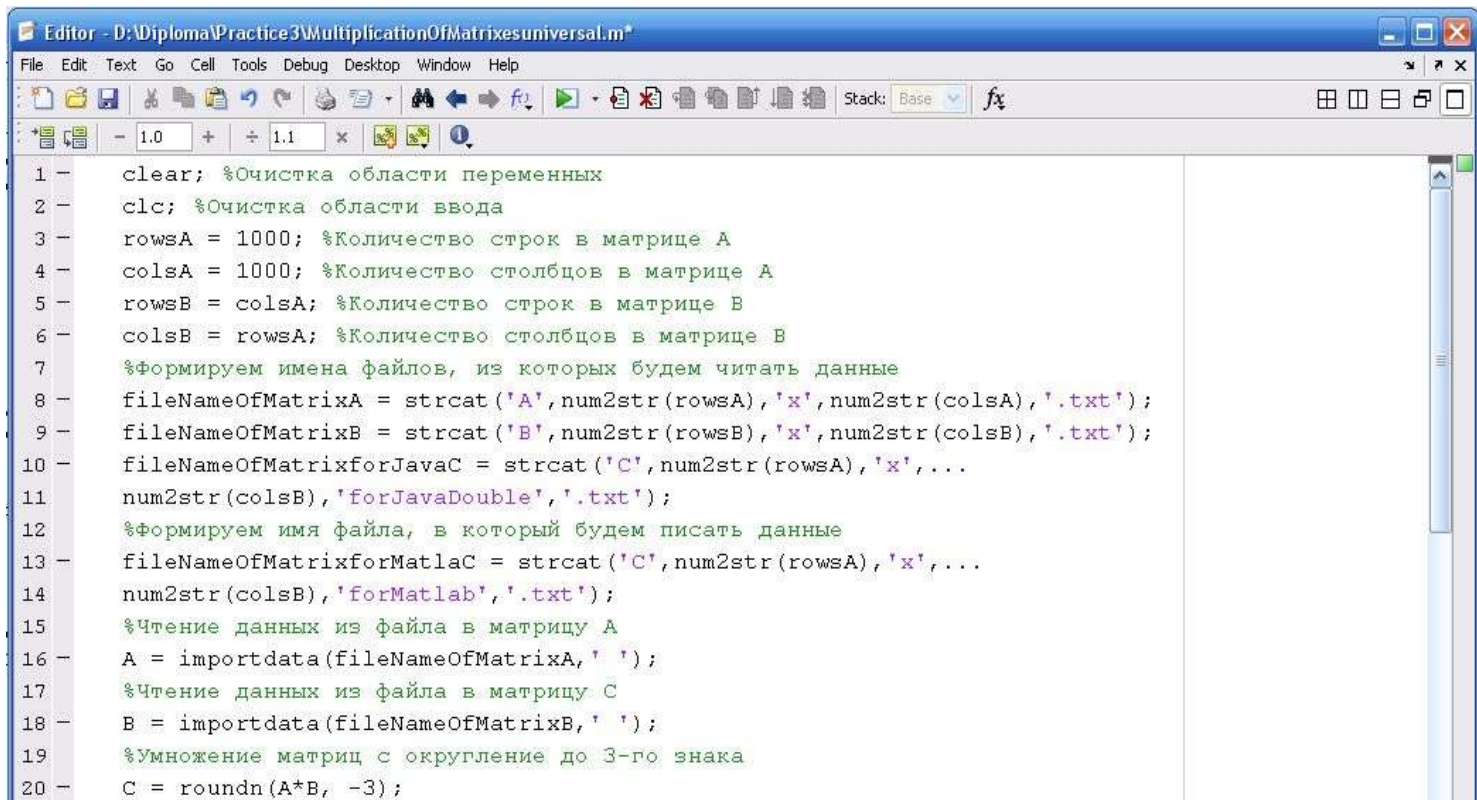
Как видно из таблицы 2, изменение количества отдаваемых одновременно строк (*stepRow*) потоку не особо влияет на быстродействие, для упрощения алгоритма управления потоками в дальнейшем использовал шаг 1.

Таблица 2 – Изменение шага потоков

Размер матрицы 1000x1000			
Количество потоков	Шаг строк 1	Шаг строк 50	Шаг строк 100
2	21,2 с.	21,5 с.	21,9 с.
4	21,1 с.	22,3 с.	22,9 с.
6	20,7 с.	21,0 с.	20,5 с.
8	20,9 с.	21,2 с.	21,0 с.
10	20,3 с.	20,5 с.	20,4 с.

Проверку результатов умножения матриц на Java провел в системе Matlab.

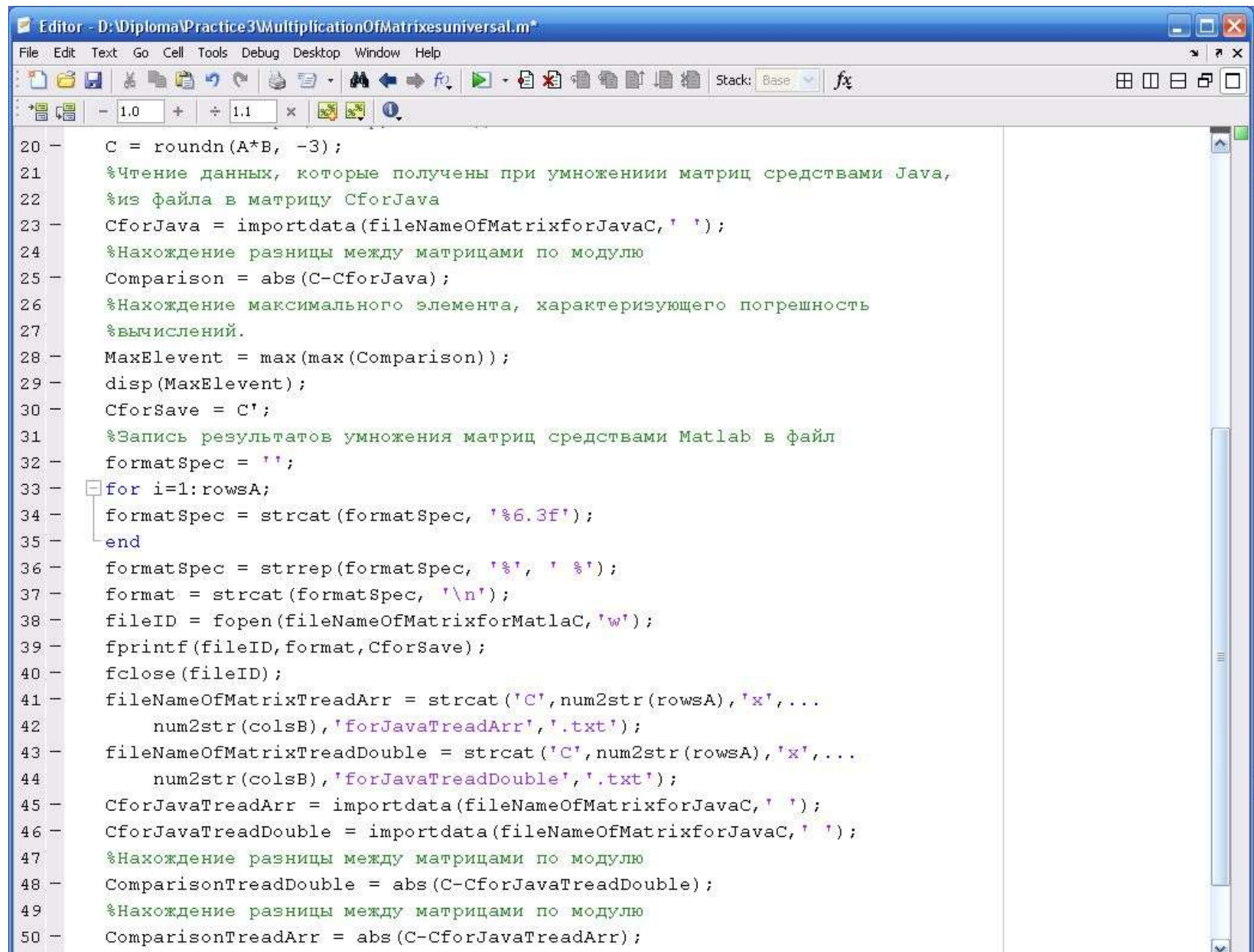
Листинг программы указан на рисунках 1-3.



```

1 - clear; %Очистка области переменных
2 - clc; %Очистка области ввода
3 - rowsA = 1000; %Количество строк в матрице A
4 - colsA = 1000; %Количество столбцов в матрице A
5 - rowsB = colsA; %Количество строк в матрице B
6 - colsB = rowsA; %Количество столбцов в матрице B
7 - %Формируем имена файлов, из которых будем читать данные
8 - fileNameOfMatrixA = strcat('A', num2str(rowsA), 'x', num2str(colsA), '.txt');
9 - fileNameOfMatrixB = strcat('B', num2str(rowsB), 'x', num2str(colsB), '.txt');
10 - fileNameOfMatrixforJavaC = strcat('C', num2str(rowsA), 'x', ...
11 - num2str(colsB), 'forJavaDouble', '.txt');
12 - %Формируем имя файла, в который будем писать данные
13 - fileNameOfMatrixforMatlaC = strcat('C', num2str(rowsA), 'x', ...
14 - num2str(colsB), 'forMatlab', '.txt');
15 - %Чтение данных из файла в матрицу A
16 - A = importdata(fileNameOfMatrixA, ' ');
17 - %Чтение данных из файла в матрицу C
18 - B = importdata(fileNameOfMatrixB, ' ');
19 - %Умножение матриц с округление до 3-го знака
20 - C = roundn(A*B, -3);
  
```

Рисунок 1 – Листинг программы в Matlab



The image shows a MATLAB editor window titled "Editor - D:\Diploma\Practice3\MultiplicationOfMatrixesuniversal.m\*". The window contains a MATLAB script with the following code:

```
20 - C = roundn(A*B, -3);
21 - %Чтение данных, которые получены при умножении матриц средствами Java,
22 - %из файла в матрицу CforJava
23 - CforJava = importdata(fileNameOfMatrixforJavaC, ' ');
24 - %Нахождение разницы между матрицами по модулю
25 - Comparison = abs(C-CforJava);
26 - %Нахождение максимального элемента, характеризующего погрешность
27 - %вычислений.
28 - MaxElevent = max(max(Comparison));
29 - disp(MaxElevent);
30 - CforSave = C';
31 - %Запись результатов умножения матриц средствами Matlab в файл
32 - formatSpec = ' ';
33 - for i=1:rowsA;
34 -     formatSpec = strcat(formatSpec, '%6.3f');
35 - end
36 - formatSpec = strrep(formatSpec, '%', ' %');
37 - format = strcat(formatSpec, '\n');
38 - fileID = fopen(fileNameOfMatrixforMatlaC, 'w');
39 - fprintf(fileID, format, CforSave);
40 - fclose(fileID);
41 - fileNameOfMatrixTreadArr = strcat('C', num2str(rowsA), 'x', ...
42 -     num2str(colsB), 'forJavaTreadArr', '.txt');
43 - fileNameOfMatrixTreadDouble = strcat('C', num2str(rowsA), 'x', ...
44 -     num2str(colsB), 'forJavaTreadDouble', '.txt');
45 - CforJavaTreadArr = importdata(fileNameOfMatrixforJavaC, ' ');
46 - CforJavaTreadDouble = importdata(fileNameOfMatrixforJavaC, ' ');
47 - %Нахождение разницы между матрицами по модулю
48 - ComparisonTreadDouble = abs(C-CforJavaTreadDouble);
49 - %Нахождение разницы между матрицами по модулю
50 - ComparisonTreadArr = abs(C-CforJavaTreadArr);
```

Рисунок 2 – Листинг программы в Matlab



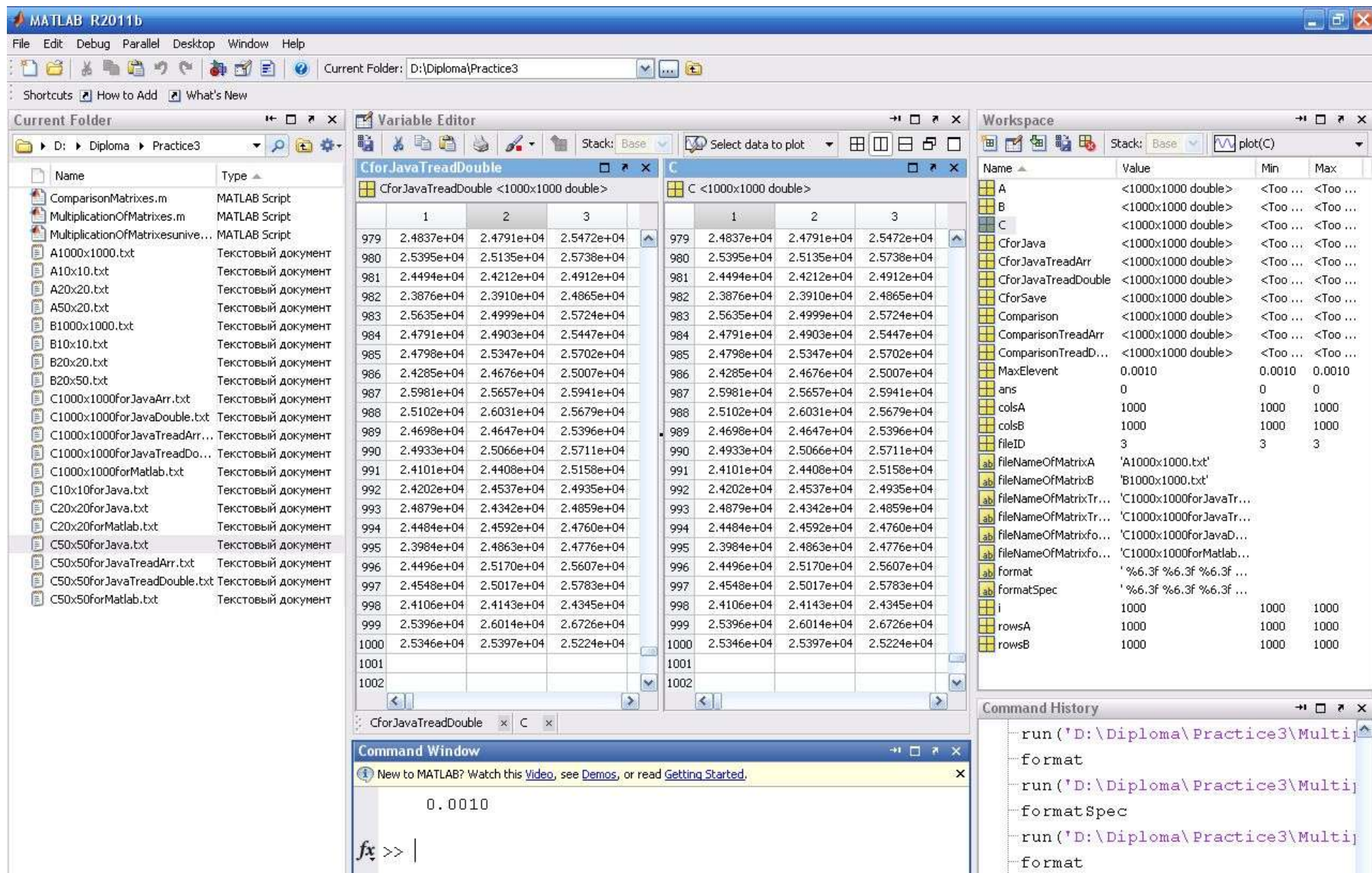


Рисунок 3 – Окно программы в Matlab

При умножении матриц размером меньше 100x100 использование многопоточности показывало худший результат по времени в сравнении с однопоточным умножением. Эту логику отразил в коде, при вызове многопоточного умножения проверяется размер матрицы, если он меньше 100, то вызывается один поток, количество вызываемых потоков зависит от количества ядер процессора на компьютере:

```
quantityOfStreams = Runtime.getRuntime().availableProcessors();
```

На моем компьютере двухядерный процессор, я пробовал создать потоков больше количества ядер, эксперимент провел на умножении матриц размерностью 2000x2000 и 5000x5000 (табл. 3).

Таблица 3 – Использование различного количества потоков

размер матрицы 2000x2000		размер матрицы 5000x5000	
кол-во потоков	время, с	кол-во потоков	время, с
1	385	1	6415 (~1 час 47 мин.)
2	192	2	3446 (~57 мин.)
4	194	4	3440 (~57 мин.)
8	185	8	3485 (~58 мин.)

Умножение матриц 10000x10000 на моем компьютере провести не удалось из-за нехватки памяти (рис. 4).

d:\Documents and Settings\RT\Мои документы\NetBeansProjects\Practice3\*.*				*	▼
Имя	Тип	Размер	Дата		
..			<Папка> 05.11.2014 16:59		
.git			<Папка> 05.11.2014 16:19		
build			<Папка> 05.11.2014 14:11		
dist			<Папка> 05.11.2014 14:11		
nbproject			<Папка> 30.10.2014 13:44		
src			<Папка> 05.11.2014 11:48		
.gitignore		6	05.11.2014 14:07		
A50x20	txt	6 107	05.11.2014 14:48		
A2000x2000	txt	24 004 791	05.11.2014 14:54		
A5000x5000	txt	150 011 297	04.11.2014 00:50		
A10000x10000	txt	600 025 183	04.11.2014 02:03		
B20x50	txt	6 047	05.11.2014 14:48		
B2000x2000	txt	24 004 788	05.11.2014 14:54		
B5000x5000	txt	150 011 287	04.11.2014 00:51		
B10000x10000	txt	600 024 872	04.11.2014 02:03		
build	xml	3 612	30.10.2014 13:44		
C50x50forJavaDouble	txt	15 107	05.11.2014 14:48		
C50x50forJavaThreadDouble	txt	20 107	05.11.2014 14:48		
C2000x2000forJavaDouble	txt	24 004 011	05.11.2014 15:03		
C2000x2000forJavaThreadDouble	txt	40 004 011	05.11.2014 15:16		
C5000x5000forJavaDouble	txt	150 011 287	04.11.2014 00:51		
C5000x5000forJavaThreadDouble	txt	150 011 287	04.11.2014 00:51		
manifest	mf	85	30.10.2014 13:45		

Рисунок 4 – Результаты создания и умножения матриц

2. При запуске программы выбираем тип умножения матриц (рис. 5):

- однопоточное (вводим 1);
- многопоточное (вводим 2).



Рисунок 5 – Выбор способа умножения матриц

После переходим к выбору меню (рис. 6):

- работа в демонстрационном режиме (вводим 0);
- работа с файлами, которые содержат матрицы (вводим 1).



Рисунок 6 – Выбор способа умножения матриц

В режиме демонстрации мы вводим количество рядков и столбцов первой матрицы, вторая матрица для умножения создается исходя из математического соответствия. Обе исходные матрицы записываются в текстовые файлы, после матрицы перемножаются и результат записывается в текстовый файл. При генерации имен файлов, в которые будут записаны матрицы, учитывается размер матриц, поэтому файлы матриц разного размера не будут перезаписывать друг друга. Программа выводит длительность записи матриц в файлы и длительность операции умножения (рис. 7-8).

После мы можем выйти из программы (вводим 0), вернуться в меню на шаг выше (вводим 1) и повторить генерацию матриц (вводим любую строку).



```

G:\ D:\WINDOWS\system32\cmd.exe
Hello! It is matrix multiplier=>
To multiply in one thread input "1"
To multiply in more threads input "2"
2
To look at demonstration of work input "0".
To multiply your matrixes input "1".
0
Input quantity of rows of the matrix.
100
Input quantity of columns of the matrix.
100
Recording of the file lasted 46 ms.
Recording of the file lasted 47 ms.
Multitread multiplication of matrixes A 100x100 and B100x100.
Type of data is DOUBLE.
The tread number 1 started.
The stream number 1 stopped.
Multiplication of matrixes lasted 31 ms.
Recording of the file lasted 31 ms.

To continue press "Enter"
To return to the menu input "1"
To stop the running program input "0".
_

```

Рисунок 7 – Демонстрационный режим

A2x2	txt	33
A3x3	txt	65
A10x10	txt	627
A12x12	txt	895
A100x100	txt	60 210
B2x2	txt	0
B3x3	txt	65
B10x10	txt	826
B10x10	txt	0
B12x12	txt	895
B100x100	txt	60 209
C2x2forJavaDouble	txt	39
C3x3forJavaDouble	txt	77
C10x10forJavaDouble	txt	825
C12x12forJavaDouble	txt	1 183
C100x100forJavaDouble	txt	90 209
cv	txt	825
Practice3	bat	78
Practice3	jar	96 871

Рисунок 8 – Демонстрационный режим

При выборе меню умножения наших матриц, которые находятся в фалах, необходимо указать пути к ним и нужно указать путь к файлу, в который сохраняться результаты умножения (рис. 9).

```
CA D:\WINDOWS\system32\cmd.exe
Hello! It is matrix multiplier=>
To multiply in one thread input "1"
To multiply in more threads input "2"
2
To look at demonstration of work input "0".
To multiply your matrixes input "1".
1
Specify the path to the matrix A.
Example: "c:\A10X10.txt".
A100x100.txt
Specify the way to the matrix B.
B100x100.txt
Specify the path to the matrix in which to save result.
C100x100.txt
Multitread multiplication of matrixes A 100x100 and B100x100.
Type of data is DOUBLE.
The tread number 1 started.
The stream number 1 stopped.
Multiplication of matrixes lasted 31 ms.
Recording of the file lasted 79 ms.

To continue press "Enter"
To return to the menu input "1"
To stop the running program input "0".
-
```

Рисунок 9 – Режим умножения матриц, находящихся в текстовых файлах

При многопоточном умножении выводится информация о состоянии потоков.

При умножении матриц размером больше 1000x1000, выводится информация о прогрессе умножения (рис. 10).

```
To look at demonstration of work input "0".
To multiply your matrixes input "1".
0
Input quantity of rows of the matrix.
1500
Input quantity of columns of the matrix.
1500
Recording of the file lasted 7047 ms.
Recording of the file lasted 6531 ms.
Multitread multiplication of matrixes A 1500x1500 and B1500x1500.
Type of data is DOUBLE.
The tread number 1 started.
Multiplication progress 0%.
The tread number 3 started.
The tread number 2 started.
The tread number 4 started.
Multiplication progress 10%.
Multiplication progress 20%.
Multiplication progress 30%.
Multiplication progress 40%.
Multiplication progress 50%.
Multiplication progress 60%.
Multiplication progress 70%.
Multiplication progress 80%.
Multiplication progress 90%.
The stream number 1 stopped.
The stream number 4 stopped.
The stream number 3 stopped.
The stream number 2 stopped.
Multiplication of matrixes lasted 126453 ms.
Recording of the file lasted 6672 ms.
```

Рисунок 10 – Прогресс умножения крупных матриц