Пояснительная записка

Роженко Варвара бпи205

Практическое задание 2, вариант 187 (условие задач 5, обработка данных в контейнере 14)

Описание полученного задания

Задание: Разработка контейнера, содержащего квадратные матрицы с действительными числами, используя объектно-ориентированный подход.
Виды матриц: 1) обычный двумерный массив, 2) диагональная матрица, 3) нижняя треугольная матрица.
Диагональная матрица реализована на основе одномерного массива, содержащего элементы, стоящие на диагонали.
Нижняя треугольная матрица реализована на основе одномерного массива с формулой пересчета.
Общая для всех альтернатив переменная – размерность.
Общая функция для всех альтернатив функция – вычисление среднего арифметического.
Функция обработки данных в контейнере – упорядочивание по убыванию, используя сортировку Шелла.

Формат ввода команды: случайная генерация матриц (*./abc2 -n number outputData*, где outputData (файл для записи результата) файлы формата txt) или считывание из файла *(./abc2 -f inputData outputData*, где inputData (файл со входными данными) outputData (файл для записи результата) файлы формата txt).

Формат описания матриц: (вид матрицы) (размерность матрицы) (элементы матрицы в строку через пробел)

Виды матрицы соответствуют номерам: 1 – обычный двумерный массив,
2 – диагональная матрица, 3 – нижняя треугольная матрица.

Пример описания матрицы: 1 2 1 1 2 2 – двумерный массив размерности 2: $\begin{matrix} 1 & 1 \\ 2 & 2 \end{matrix}$

Структурная схема ВС с размещенной на ней разработанной программой

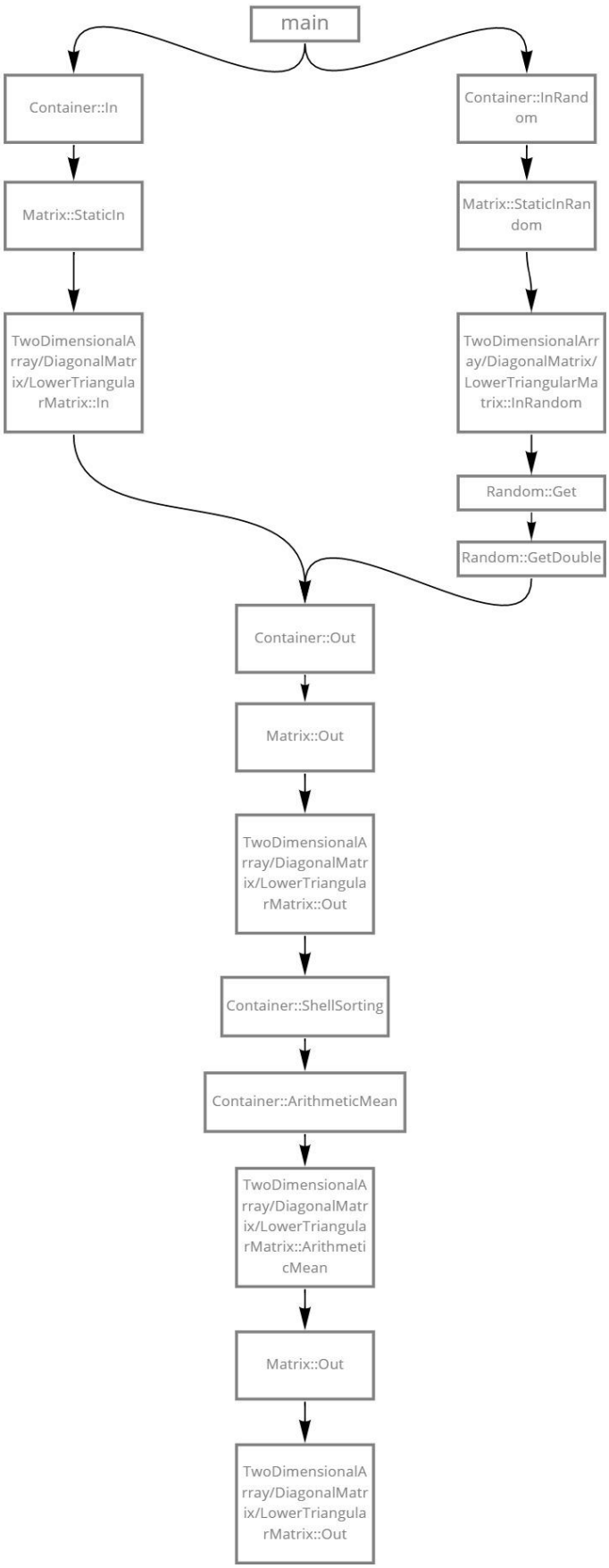
Таблица типов:

int	4
double	8
class TwoDimensionalArray dimension: int tdaElements: double**	12 4[0] 8[4]
class DiagonalMatrix dimension: int tdaElements: double*	12 4[0] 8[4]
class LowerTriangularMatrix dimension: int tdaElements: double*	12 4[0] 8[4]
class Matrix rnd: Random	8 8[0]
class Container len: int storage [10000]: matrix*	80004 4[0] 8*10000[4]=80000
class Random first: int last: int	8 4[0] 4[4]

Память программы:

int main(int argc, char* argv[]) argc: int argv: char* c: container fileInput: FILE* size: int fileOutput: FILE*	80032 4[0] 4[4] 80004[8] 8[80012] 4[80020] 8[80024]
void Container::ShellSorting() step: int temp: Matrix*	12 4[0] 8[4]
Matrix* Matrix::StaticIn(FILE *file) file: FILE* k: int m: Matrix*	20 8[0] 4[8] 8[12]
Matrix* Matrix::StaticInRandom() k: int m: Matrix*	12 4[0] 8[4]

Stack


main
Container::In
Container::InRandom
Matrix::StaticIn
Matrix::StaticInRandom
TwoDimensionalArray/DiagonalMatrix/LowerTriangularMatrix::In
TwoDimensionalArray/DiagonalMatrix/LowerTriangularMatrix::InRandom
Random::Get
Random::GetDouble
Container::Out
Matrix::Out
TwoDimensionalArray/DiagonalMatrix/LowerTriangularMatrix::Out
Container::ShellSorting
Container::ArithmeticMean
TwoDimensionalArray/DiagonalMatrix/LowerTriangularMatrix::ArithmeticMean
Matrix::Out
TwoDimensionalArray/DiagonalMatrix/LowerTriangularMatrix::Out

Характеристики программы

Число интерфейсных модулей: 6

Число модулей реализации: 5+main.cpp

Общее число строк кода: 531

Время выполнения программы для различных тестовых наборов данных

1) время:

```
user@user-VirtualBox:~/CLionProjects/abc2/cmake-build-debug$ ./abc2 -f inData ou
tData56
Start
Total time: 0.000118 seconds
Stop
```

входные данные:

inData

```
1 1 2 1.4 2 3 4
2 2 2 1 2
3 3 3 1 2 3 4 5 6
```

выходные данные:

```
1 Container contains 3 elements.
2 0: It is two dimensional array: dimension = 2. Arithmetic mean = 2.60
3 1: It is diagonal matrix: dimension = 2. Arithmetic mean = 0.75
4 2: It is lower triangular matrix: dimension = 3. Arithmetic mean = 2.33
5
6
7 Sorted container:
8 Container contains 3 elements.
9 0: It is two dimensional array: dimension = 2. Arithmetic mean = 2.60
10 1: It is lower triangular matrix: dimension = 3. Arithmetic mean = 2.33
11 2: It is diagonal matrix: dimension = 2. Arithmetic mean = 0.75
```

2) время:

```
user@user-VirtualBox:~/CLionProjects/abc2/cmake-build-debug$ ./abc2 -f inData10
outData12
Start
Total time: 0.000197 seconds
Stop
```

входные данные:

```
1 1 2 1 2 1 2
2 2 3 1 2 3
3 3 2 1 2 3
4 1 3 1 1 1 2 2 2 3 3 3
5 2 5 1 2 3 4 5
6 3 4 1 1 1 1 1 1 1 1 1 1
7 1 3 1 10 1 2 6 2 0 0 0
8 2 10 1 2 3 4 5 6 3 8 9 10
9 3 4 1 1 1 1 1 1 1 1 1 1
10 1 1 1
```

выходные данные:

```
Container contains 10 elements.
0: It is two dimensional array: dimension = 2. Arithmetic mean = 1.50
1: It is diagonal matrix: dimension = 3. Arithmetic mean = 0.67
2: It is lower triangular matrix: dimension = 2. Arithmetic mean = 1.50
3: It is two dimensional array: dimension = 3. Arithmetic mean = 2.00
4: It is diagonal matrix: dimension = 5. Arithmetic mean = 0.60
5: It is lower triangular matrix: dimension = 4. Arithmetic mean = 0.62
6: It is two dimensional array: dimension = 3. Arithmetic mean = 2.44
7: It is diagonal matrix: dimension = 10. Arithmetic mean = 0.51
8: It is lower triangular matrix: dimension = 4. Arithmetic mean = 0.62
9: It is two dimensional array: dimension = 1. Arithmetic mean = 1.00


Sorted container:
Container contains 10 elements.
0: It is two dimensional array: dimension = 3. Arithmetic mean = 2.44
1: It is two dimensional array: dimension = 3. Arithmetic mean = 2.00
2: It is two dimensional array: dimension = 2. Arithmetic mean = 1.50
3: It is lower triangular matrix: dimension = 2. Arithmetic mean = 1.50
4: It is two dimensional array: dimension = 1. Arithmetic mean = 1.00
5: It is diagonal matrix: dimension = 3. Arithmetic mean = 0.67
6: It is lower triangular matrix: dimension = 4. Arithmetic mean = 0.62
7: It is lower triangular matrix: dimension = 4. Arithmetic mean = 0.62
8: It is diagonal matrix: dimension = 5. Arithmetic mean = 0.60
9: It is diagonal matrix: dimension = 10. Arithmetic mean = 0.51
```

3) время для 20 элементов:

```
user@user-VirtualBox:~/CLionProjects/abc2/cmake-build-debug$ ./abc2 -n 20 outDat
a20
Start
Total time: 0.000526 seconds
Stop
```

выходные данные:

```
Container contains 20 elements.
0: It is two dimensional array: dimension = 9. Arithmetic mean = 1.48
1: It is diagonal matrix: dimension = 2. Arithmetic mean = 0.68
2: It is lower triangular matrix: dimension = 2. Arithmetic mean = 1.28
3: It is two dimensional array: dimension = 6. Arithmetic mean = 1.54
4: It is diagonal matrix: dimension = 14. Arithmetic mean = 0.10
5: It is lower triangular matrix: dimension = 20. Arithmetic mean = 0.78
6: It is lower triangular matrix: dimension = 19. Arithmetic mean = 0.78
7: It is diagonal matrix: dimension = 18. Arithmetic mean = 0.09
8: It is two dimensional array: dimension = 10. Arithmetic mean = 1.48
9: It is lower triangular matrix: dimension = 10. Arithmetic mean = 0.82
10: It is two dimensional array: dimension = 15. Arithmetic mean = 1.49
11: It is diagonal matrix: dimension = 18. Arithmetic mean = 0.08
12: It is lower triangular matrix: dimension = 17. Arithmetic mean = 0.80
13: It is diagonal matrix: dimension = 19. Arithmetic mean = 0.09
14: It is diagonal matrix: dimension = 14. Arithmetic mean = 0.10
15: It is two dimensional array: dimension = 4. Arithmetic mean = 1.59
16: It is diagonal matrix: dimension = 7. Arithmetic mean = 0.21
17: It is diagonal matrix: dimension = 7. Arithmetic mean = 0.22
18: It is diagonal matrix: dimension = 16. Arithmetic mean = 0.09
19: It is two dimensional array: dimension = 7. Arithmetic mean = 1.48


Sorted container:
Container contains 20 elements.
0: It is two dimensional array: dimension = 4. Arithmetic mean = 1.59
1: It is two dimensional array: dimension = 6. Arithmetic mean = 1.54
2: It is two dimensional array: dimension = 15. Arithmetic mean = 1.49
3: It is two dimensional array: dimension = 10. Arithmetic mean = 1.48
4: It is two dimensional array: dimension = 9. Arithmetic mean = 1.48
5: It is two dimensional array: dimension = 7. Arithmetic mean = 1.48
6: It is lower triangular matrix: dimension = 2. Arithmetic mean = 1.28
7: It is lower triangular matrix: dimension = 10. Arithmetic mean = 0.82
8: It is lower triangular matrix: dimension = 17. Arithmetic mean = 0.80
9: It is lower triangular matrix: dimension = 20. Arithmetic mean = 0.78
10: It is lower triangular matrix: dimension = 19. Arithmetic mean = 0.78
11: It is diagonal matrix: dimension = 2. Arithmetic mean = 0.68
12: It is diagonal matrix: dimension = 7. Arithmetic mean = 0.22
13: It is diagonal matrix: dimension = 7. Arithmetic mean = 0.21
14: It is diagonal matrix: dimension = 14. Arithmetic mean = 0.10
15: It is diagonal matrix: dimension = 14. Arithmetic mean = 0.10
16: It is diagonal matrix: dimension = 16. Arithmetic mean = 0.09
```

4) время для 8000 элементов:

```
user@user-VirtualBox:~/CLionProjects/abc2/cmake-build-debug$ ./abc2 -n 8000 outD
ata8000
Start
Total time: 0.166795 seconds
Stop
```

5) время для 10000 элементов:

```
user@user-VirtualBox:~/CLionProjects/abc2/cmake-build-debug$ ./abc2 -n 10000 out
Data10000
Start
Total time: 0.203922 seconds
Stop
```

6) переполнение:

```
user@user-VirtualBox:~/CLionProjects/abc2/cmake-build-debug$ ./abc2 -n 10001 out
Data10001
Start
incorrect number of matrixes = 10001. Set 0 < number <= 10000
```

Сравнение с характеристиками раннее разработанных программ

Время работы программы из первого задания на таких же тестовых данных:

```
user@user-VirtualBox:~/CLionProjects/abc/cmake-build-debug$ ./abc -f inData outD
ata90
Start
Total time for program: 0.000172 seconds
Stop
```

```
user@user-VirtualBox:~/CLionProjects/abc/cmake-build-debug$ ./abc -f inData10 ou
tData10
Start
Total time for program: 0.000240 seconds
Stop
```

```
user@user-VirtualBox:~/CLionProjects/abc/cmake-build-debug$ ./abc -n 20 outData2
0
Start
Total time for program: 0.000667 seconds
Stop
```

```
user@user-VirtualBox:~/CLionProjects/abc/cmake-build-debug$ ./abc -n 8000 outDat
a800
Start
Total time for program: 0.166133 seconds
Stop
```

```
user@user-VirtualBox:~/CLionProjects/abc/cmake-build-debug$ ./abc -n 10000 outDa
ta10000
Start
Total time for program: 0.220338 seconds
Stop
```

Из полученных данных можно сделать вывод, что программа из задания 1 имеет немного большее время исполнения, чем программа из задания 2. (к примеру для первого примера время уменьшилось на 0,000054, для второго примера на 0,000043).
Количество модулей осталось прежним, число строк уменьшилось (было 558 в 1 задании, во втором стало 531).