ФИО: Ребдев Павел Александрович
Группа: 5130904/30008
Лабораторная работа: «Двумерные массивы. Указатели. Метод «сверху вниз»»

Постановка задачи

Разработать детальные требования и тест план для следующей задачи:

getNumberOfOrderedRows. Написать программу, принимающую на вход количество строк и столбцов в матрице и саму матрицу и выводящую в стандартный поток вывода количество строк в стандартной и введённой матрицах, элементы которых упорядочены по возрастанию

Детальные требования

1. Количество строк и столбцов в матрице (rows и colums) должно быть задано корректно:
   1.1. rows и colums — целые числа. Если rows или colums не являютмя целыми числами, то выводится сообщение об ошибке и программа завершается
   1.2. rows и colums — положительные числа. Если rows или colums не являютмя положительные числами, то выводится сообщение об ошибке и программа завершается
2. Элементы матрицы должны быть заданы корректно:
   2.1. Каждый элемент матрицы — целое число. Если элемент матрицы не является целым числом, то выводится сообщение об ошибке и программа завершается
3. rows, colums и элементы матрицы заданы корректно:
   3.1. Если rows, colums и элементы матрицы заданы корректно, то программа должна завершится с выводом в консоль сообщения, содержащего количество строк в стандартной и введённой матрицах, элементы которых упорядочены по возрастанию и кодом возврата 0

Тест-план

Проверка детальных требований с помощью тест-плана:

#	**Описание**	**Результат**

1.1	rows и colums — целые числа. Если rows или colums не являютмя целыми числами, то выводится сообщение об ошибке и программа завершается	**Input:**
		`12.2 34`
		Expected:
		`Incorrect input of rows or colums!`

1.2	rows и colums — положительные числа. Если rows или colums не являютмя положительные числами, то выводится сообщение об ошибке и программа завершается	**Input:**
		`34 -79`
		Expected:
		`std::bad_array_new_lengt`

2.1	Каждый элемент матрицы — целое число. Если элемент матрицы не является целым числом, то выводится сообщение об ошибке и программа завершается	**Input:**
		`2 2`
		`12 23`
		`9.3 18`
		Expected:
		`Input error!`

		Input:
		`3 3`
		`12 47 94`
		`-14 57 24`
		`94 abc 09`
		Expected:
		`Input error!`

3.1	Если rows, colums и элементы матрицы заданы корректно, то программа должна завершится с выводом в консоль сообщения, содержащего количество строк в стандартной и введённой матрицах, элементы которых упорядочены по возрастанию и кодом возврата 0	**Input:**
		`2 2`
		`2 1`
		`1 2`
		Expected:
		`Dynamic: 1`
		`Standart: 2`

		Input:
		`3 4`
		`1 2 3 4`
		`5 4 3 2`
		`1 11 111 1111`

		Expected: Dynamic: 2 Standart: 2

Исходные тексты программы

Файлы с исходными текстами лабораторной работы (полагаем <ROOT> для папки в котором располагаются исходные тексты):

./<ROOT>/main.cpp

```
#include <iostream>
#include <cstddef>
#include "matrixFunction.hpp"

int main()
{
  size_t rows = 0, colums = 0;
  std::cin >> rows >> colums;
  if (!std::cin)
  {
    std::cerr << "Incorrect input of rows or colums! \n";
    return 1;
  }

  int * matrix = nullptr;
  try
  {
    matrix =  new int[rows * colums];
  }
  catch (const std::exception & e)
  {
    delete [] matrix;
    std::cerr << e.what();
    return 2;
  }
  size_t standartRows = 3, standartColums = 3;
  int standartMatrix[] = {
  1, 2, 3,
  6, 5, 4,
  7, 8, 9
  };

  try
  {
    inArray(matrix, rows, colums);
  }
  catch (const std::exception & e)
  {
    delete [] matrix;
    std::cerr << e.what();
    return 1;
  }


    std::cout  <<  "Dynamic:  "  <<  getNumberOfOrderedRows(matrix,  rows,
colums) << '\n';
```

```
   std::cout << "Standart: " << getNumberOfOrderedRows(standartMatrix,
standartRows, standartColums) << '\n';
  delete [] matrix;
  return 0;
}
```

./<ROOT>/matrixFunction.cpp

```
#include "matrixFunction.hpp"
#include <iostream>

void inArray (int * matrix, size_t rows, size_t colums)
{
  for (size_t i = 0; i < (rows * colums); ++i)
  {
    std::cin >> matrix[i];
    if (!std::cin)
    {
      throw std::logic_error("Input error!");
    }
  }
};

void outArray (int * matrix, size_t rows, size_t colums)
{
  for (size_t i = 0; i < rows; ++i)
  {
    for (size_t j = 0; j < colums; ++j)
    {
      std::cout << matrix[i * colums + j] << " ";
    }
    std::cout << '\n';
  }
};

size_t getNumberOfOrderedRows (int * matrix, size_t rows, size_t colums)
{
  size_t num = 0;
  for (size_t i = 0; i < rows; ++i)
  {
    bool prot = 1;
    for (size_t j = 0; j < (colums - 1); ++j)
    {
      if (matrix[i * colums + j] >= matrix[i * colums + j + 1])
      {
        prot = 0;
        break;
      }
    }
    num += prot;
  }

  return num;
};
```

./<ROOT>/matrixFunction.hpp

```
#ifndef MATRIXFUNCTION_HPP
#define MATRIXFUNCTION_HPP
#include <cstddef>

void inArray (int * matrix, size_t rows, size_t colums);
void outArray (int * matrix, size_t rows, size_t colums);
size_t getNumberOfOrderedRows (int * matrix, size_t rows, size_t
colums);
#endif
```