|  |
| --- |
| Пермский филиал федерального государственного автономного образовательного учреждения высшего образования  «Национальный исследовательский университет  «Высшая школа экономики»  *Факультет социально-экономических и компьютерных наук* |
|  |
| Гуцол Степан Дмитриевич  **ЛАБОРАТОРНАЯ РАБОТА №5**  по направлению подготовки *38.03.05 Бизнес-информатика*  образовательная программа «Разработка информационных систем для бизнеса»   |  |  | | --- | --- | |  | Руководитель  Преподаватель кафедры ИТБ  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Найданов И.В. |   Пермь, 2022 |

**Оглавление**

[**Задача №1** 4](#_Toc120765635)

[**1. Постановка задачи** 4](#_Toc120765636)

[**2. Анализ** 4](#_Toc120765637)

[**3. Программа (листинг)** 5](#_Toc120765638)

[**4. Тестирование** 6](#_Toc120765639)

[**Задача №2** 7](#_Toc120765640)

[**1. Постановка задачи** 7](#_Toc120765641)

[**2. Анализ** 7](#_Toc120765642)

[**3. Программа (листинг)** 8](#_Toc120765643)

[**4. Тестирование** 9](#_Toc120765644)

[**Задача №3** 10](#_Toc120765645)

[**1. Постановка задачи** 10](#_Toc120765646)

[**2. Анализ** 10](#_Toc120765647)

[**3. Программа (листинг)** 11](#_Toc120765648)

[**4. Тестирование** 12](#_Toc120765649)

[**Задача №4** 13](#_Toc120765650)

[**1. Постановка задачи** 13](#_Toc120765651)

[**2. Анализ** 13](#_Toc120765652)

[**3. Программа (листинг)** 14](#_Toc120765653)

[**4. Тестирование** 15](#_Toc120765654)

[**Задача №5** 16](#_Toc120765655)

[**1. Постановка задачи** 16](#_Toc120765656)

[**2. Анализ** 16](#_Toc120765657)

[**3. Программа (листинг)** 17](#_Toc120765658)

[**4. Тестирование** 18](#_Toc120765659)

[**Задача №6** 19](#_Toc120765660)

[**1. Постановка задачи** 19](#_Toc120765661)

[**2. Анализ** 19](#_Toc120765662)

[**3. Программа (листинг)** 21](#_Toc120765663)

[**4. Тестирование** 22](#_Toc120765664)

[**Задача №7** 23](#_Toc120765665)

[**1. Постановка задачи** 23](#_Toc120765666)

[**2. Анализ** 23](#_Toc120765667)

[**3. Программа (листинг)** 25](#_Toc120765668)

[**4. Тестирование** 26](#_Toc120765669)

[**Задача №8** 27](#_Toc120765670)

[**1. Постановка задачи** 27](#_Toc120765671)

[**2. Анализ** 27](#_Toc120765672)

[**3. Программа (листинг)** 28](#_Toc120765673)

[**4. Тестирование** 29](#_Toc120765674)

[**Задача №9** 30](#_Toc120765675)

[**1. Постановка задачи** 30](#_Toc120765676)

[**2. Анализ** 30](#_Toc120765677)

[**3. Программа (листинг)** 31](#_Toc120765678)

[**4. Тестирование** 32](#_Toc120765679)

[**Задача №10** 33](#_Toc120765680)

[**1. Постановка задачи** 33](#_Toc120765681)

[**2. Анализ** 33](#_Toc120765682)

[**3. Программа (листинг)** 35](#_Toc120765683)

[**4. Тестирование** 36](#_Toc120765684)

[**Задача №11** 37](#_Toc120765685)

[**1. Постановка задачи** 37](#_Toc120765686)

[**2. Анализ** 37](#_Toc120765687)

[**3. Программа (листинг)** 38](#_Toc120765688)

[**4. Тестирование** 39](#_Toc120765689)

[**Задача №12** 40](#_Toc120765690)

[**1. Постановка задачи** 40](#_Toc120765691)

[**2. Анализ** 40](#_Toc120765692)

[**3. Программа (листинг)** 42](#_Toc120765693)

[**4. Тестирование** 43](#_Toc120765694)

[ПРИЛОЖЕНИЕ А **Программа** 45](#_Toc120765695)

**Задача №1**

**1. Постановка задачи**

Сформировать динамический одномерный массив, заполнить его случайными числами и вывести на печать.

**2. Анализ**

Исходные данные:

len – целое число большее нуля.

Выходные данные:

oneDimArr – массив целых чисел от -100 до 100 с len количеством элементов.

**Таблица 1 – Анализ классов исходных данных**

| **Задача** | **Исходные данные** | **Класс входных данных** | **Класс выходных данных** |
| --- | --- | --- | --- |
| Сформировать динамический одномерный массив, заполнить его случайными числами и вывести на печать. | len | len – целое число > 0 | массив целых чисел от -100 до 100 с длиной n |
| * len – строки (не числа) * len –отрицательные числа * len – нуль * len – вещественное число | сообщение об ошибке ввода |

**3. Программа (листинг)**

На выбор пользователю предоставляется возможность воспользоваться разными способами создания одномерного массива, в этом случае пользователь выбирает создание с помощью ДСЧ. Весь код программы представлен в приложении. См. Приложение А.

**4. Тестирование**

**Таблица 2 – Тестирование методом чёрного ящика**

| **№** | **Исходные данные** | **Ожидаемый результат** | **Полученный результат** |
| --- | --- | --- | --- |
| **1** | len = ‘0’ | Сообщение об ошибке | Сообщение об ошибке |
| **2** | len = ‘5’ | Массив из 5 случайных чисел | oneDimArr = ‘47 21 58 32 21’ |
| **3** | len = ‘-3’ | Сообщение об ошибке | Сообщение об ошибке |
| **4** | len = ‘три’ | Сообщение об ошибке | Сообщение об ошибке |
| **5** | len = ‘0.5’ | Сообщение об ошибке | Сообщение об ошибке |

**Задача №2**

**1. Постановка задачи**

Сформировать массив из len элементов с элементами, которые ввёл пользователь (Num).

**2. Анализ**

Исходные данные:

* len – целое число большее нуля,
* Num – целое число.

Выходные данные:

oneDimArr – одномерный динамический массив, состоящий из элементов(Num), с len количеством элементов.

**Таблица 3 – Анализ классов исходных данных**

| **Задача** | **Исходные данные** | **Класс входных данных** | **Класс выходных данных** |
| --- | --- | --- | --- |
| Сформировать массив из len элементов с элементами, которые ввёл пользователь (Num). | len | len – целое число > 0 | ввод элементов массива |
| * len – строки (не числа) * len – отрицательные числа * len – нуль * len – вещественное число | сообщение об ошибке ввода |
| Num | Num – целое число | массив целых чисел с длиной len |
| * Num– строки (не числа) * Num – вещественные числа | сообщение об ошибке ввода |

**3. Программа (листинг)**

На выбор пользователю предоставляется возможность воспользоваться разными способами создания одномерного массива, в этом случае пользователь выбирает создание с помощью ручного ввода. Весь код программы представлен в приложении. См. Приложение А.

**4. Тестирование**

**Таблица 4 – Тестирование методом чёрного ящика**

| **№** | **Исходные данные** | **Ожидаемый результат** | **Полученный результат** |
| --- | --- | --- | --- |
| **1** | len = ‘0’  Num = ’’ | Сообщение об ошибке | Сообщение об ошибке |
| **2** | len = ‘6’  Num = ’-47 21 58 32 21 0’ | Одномерный массив из 6 чисел | oneDimArr = ‘-47 21 58 32 21 0’ |
| **3** | len = ‘-3’  Num = ’’ | Сообщение об ошибке | Сообщение об ошибке |
| **4** | len = ‘три’  Num = ‘’ | Сообщение об ошибке | Сообщение об ошибке |
| **5** | len = ‘3’  Num = ‘2 -5 четыре’ | Сообщение об ошибке | Сообщение об ошибке |
| **6** | len = ‘2.75’  Num = ‘’ | Сообщение об ошибке | Сообщение об ошибке |

**Задача №3**

**1. Постановка задачи**

Распечатать одномерный массив.

**2. Анализ**

Исходные данные:

oneDimArr – одномерный массив с длиной len, состоящий из целых чисел.

Выходные данные:

Symbol – каждый элемент массива, который будет выведен на экран.

**Таблица 5 – Анализ классов исходных данных**

| **Задача** | **Исходные данные** | **Класс входных данных** | **Класс выходных данных** |
| --- | --- | --- | --- |
| Распечатать одномерный массив. | oneDimArr | oneDimArr – массив с количеством элементов больше нуля | вывод элементов массива |

**3. Программа (листинг)**

На выбор пользователю предоставляется возможность воспользоваться разными функциями меню, в этом случае пользователь выбирает печать одномерного массива. Вывод осуществляется путём использования цикла foreach. Весь код программы представлен в приложении. См. Приложение А.

**4. Тестирование**

**Таблица 6 – Тестирование методом чёрного ящика**

| **№** | **Исходные данные** | **Ожидаемый результат** | **Полученный результат** |
| --- | --- | --- | --- |
| **1** | oneDimArr = ‘-2 3 0’ | Symbol = ‘-2 3 0’ | Symbol = ‘-2 3 0’ |
| **2** | oneDimArr = ‘-32 3 10 5 2’ | Symbol = ‘-32 3 10 5 2’ | Symbol = ‘-32 3 10 5 2’ |

**Задача №4**

**1. Постановка задачи**

Выполнить удаление всех элементов с четными индексами

**2. Анализ**

Исходные данные:

oneDimArr – одномерный массив, состоящий из целых чисел.

Выходные данные:

oneDimArr – одномерный массив, с изменённым размером и без элементов с чётными индексами.

**Таблица 7 – Анализ классов исходных данных**

| **Задача** | **Исходные данные** | **Класс входных данных** | **Класс выходных данных** |
| --- | --- | --- | --- |
| Выполнить удаление всех элементов с четными индексами | oneDimArr | oneDimArr – одномерный массив с количеством элементов больше единицы | Вывод массива без элементов с чётными индексами |
| oneDimArr – одномерный массив с единственным элементом | Вывод массива без изменений |

**3. Программа (листинг)**

На выбор пользователю предоставляется возможность воспользоваться разными функциями меню, в этом случае пользователь выбирает удаление элементов одномерного массива с чётными индексами. В этой функции использовался метод класса Array, а именно Array.Resize(). Этот метод позволяет заменить несколько строчек кода. Он создаёт новый массив с указанным размером, копирует элементы старого массива в новый, а затем заменяет старый массив новым. Этот метод работает только для одномерных массивов. Весь код программы представлен в приложении. См. Приложение А.

**4. Тестирование**

**Таблица 8 – Тестирование методом чёрного ящика**

| **№** | **Исходные данные** | **Ожидаемый результат** | **Полученный результат** |
| --- | --- | --- | --- |
| **1** | oneDimArr = ‘2’ | oneDimArr = ‘2’ | oneDimArr = ‘2’ |
| **2** | oneDimArr = ‘2 5 7 9 11’ | oneDimArr = ‘2 7 11’ | oneDimArr = ‘2 7 11’ |

**Задача №5**

**1. Постановка задачи**

Сформировать двумерный массив и заполнить его случайными числами.

**2. Анализ**

Исходные данные:

* lenH – целое число, большее нуля (количество строчек),
* lenW – целое число, большее нуля(количество столбцов),

Выходные данные:

twoDimArr – двумерный массив, с lenH количеством строчек и lenW количеством столбцов с элементами от -100 до 100.

**Таблица 9 – Анализ классов исходных данных**

| **Задача** | **Исходные данные** | **Класс входных данных** | **Класс выходных данных** |
| --- | --- | --- | --- |
| Сформировать двумерный массив и заполнить его случайными числами. | lenH | lenH – целое число, большее нуля | создание двумерного массива(twoDimArr), если lenW тоже подходит |
| * lenH – строки (не числа) * lenH – отрицательные числа * lenH – нуль * lenH – вещественные числа | сообщение об ошибке |
| lenW | lenW – целое число, большее нуля | создание двумерного массива(twoDimArr), если lenH тоже подходит |
| * lenW – строки (не числа) * lenW – отрицательные числа * lenW – нуль * lenW – вещественные числа | сообщение об ошибке |

**3. Программа (листинг)**

На выбор пользователю предоставляется возможность воспользоваться разными способами создания двумерного массива, в этом случае пользователь выбирает создание двумерного массива с помощью датчика случайных чисел. Весь код программы представлен в приложении. См. Приложение А.

**4. Тестирование**

**Таблица 10 – Тестирование методом чёрного ящика**

| **№** | **Исходные данные** | **Ожидаемый результат** | **Полученный результат** |
| --- | --- | --- | --- |
| **1** | lenH = ‘0’  lenW = ‘’ | Сообщение об ошибке | Сообщение об ошибке |
| **2** | lenH = ‘-2’  lenW = ‘’ | Сообщение об ошибке | Сообщение об ошибке |
| **3** | lenH = ‘2.3’  lenW = ‘’ | Сообщение об ошибке | Сообщение об ошибке |
| **4** | lenH = ‘три’  lenW = ‘’ | Сообщение об ошибке | Сообщение об ошибке |
| **5** | lenH = ‘2’  lenW = ‘0’ | Сообщение об ошибке | Сообщение об ошибке |
| **6** | lenH = ‘2’  lenW = ‘-1’ | Сообщение об ошибке | Сообщение об ошибке |
| **7** | lenH = ‘2’  lenW = ‘три’ | Сообщение об ошибке | Сообщение об ошибке |
| **8** | lenH = ‘2’  lenW = ‘1.5’ | Сообщение об ошибке | Сообщение об ошибке |
| **9** | lenH = ‘2’  lenW = ‘3’ | двумерный массив(twoDimArr) со случайными цифрами от -100 до 100 | twoDimArr = ‘75 -5 42  -64 16 -92’ |

**Задача №6**

**1. Постановка задачи**

Сформировать двумерный массив и заполнить его элементами с помощью ручного ввода.

**2. Анализ**

Исходные данные:

* lenH – целое число, большее нуля (количество строчек),
* lenW – целое число, большее нуля(количество столбцов),
* Num – целое число,

Выходные данные:

twoDimArr – двумерный массив, с lenH количеством строчек и lenW количеством столбцов с элементами Num.

**Таблица 11 – Анализ классов исходных данных**

| **Задача** | **Исходные данные** | **Класс входных данных** | **Класс выходных данных** |
| --- | --- | --- | --- |
| Сформировать двумерный массив и заполнить его элементами с помощью ручного ввода. | lenH | lenH – целое число, большее нуля | создание двумерного массива(twoDimArr), если lenW тоже подходит |
| * lenH – строки (не числа) * lenH – отрицательные числа * lenH – нуль   lenH – вещественные числа | сообщение об ошибке |
| lenW | lenW – целое число, большее нуля | создание двумерного массива(twoDimArr), если lenH тоже подходит |
| * lenW – строки (не числа) * lenW – отрицательные числа * lenW – нуль * lenW – вещественные числа | сообщение об ошибке |
| Num | * Num – целое число | двумерный массив(twoDimArr) целых чисел с lenH количеством строчек и lenW количеством столбцов |
| * Num– строки (не числа) * Num – вещественные числа | сообщение об ошибке ввода |

**3. Программа (листинг)**

На выбор пользователю предоставляется возможность воспользоваться разными способами создания двумерного массива, в этом случае пользователь выбирает создание двумерного массива с помощью ручного ввода. Весь код программы представлен в приложении. См. Приложение А.

**4. Тестирование**

**Таблица 12 – Тестирование методом чёрного ящика**

| **№** | **Исходные данные** | **Ожидаемый результат** | **Полученный результат** |
| --- | --- | --- | --- |
| **1** | lenH = ‘0’  lenW = ‘’  Num = ‘’ | Сообщение об ошибке | Сообщение об ошибке |
| **2** | lenH = ‘-2’  lenW = ‘’  Num = ‘’ | Сообщение об ошибке | Сообщение об ошибке |
| **3** | lenH = ‘2.3’  lenW = ‘’  Num = ‘’ | Сообщение об ошибке | Сообщение об ошибке |
| **4** | lenH = ‘три’  lenW = ‘’  Num = ‘’ | Сообщение об ошибке | Сообщение об ошибке |
| **5** | lenH = ‘2’  lenW = ‘0’  Num = ‘’ | Сообщение об ошибке | Сообщение об ошибке |
| **6** | lenH = ‘2’  lenW = ‘-1’  Num = ‘’ | Сообщение об ошибке | Сообщение об ошибке |
| **7** | lenH = ‘2’  lenW = ‘три’  Num = ‘’ | Сообщение об ошибке | Сообщение об ошибке |
| **8** | lenH = ‘2’  lenW = ‘1.5’  Num = ‘’ | Сообщение об ошибке | Сообщение об ошибке |
| **9** | lenH = ‘1’  lenW = ‘2’  Num = ‘2.5 3’ | Сообщение об ошибке | Сообщение об ошибке |
| **10** | lenH = ‘1’  lenW = ‘2’  Num = ‘3 четыре ’ | Сообщение об ошибке | Сообщение об ошибке |
| **11** | lenH = ‘2’  lenW = ‘3’  Num = ‘1 2 3 4 5 6’ | twoDimArr = ‘1 2 3  4 5 6’ | twoDimArr = ‘1 2 3  4 5 6’ |

**Задача №7**

**1. Постановка задачи**

Добавить K столбцов, начиная со столбца с номером N в двумерный массив

**2. Анализ**

Исходные данные:

* twoDimArray – двумерный динамический массив, состоящий из целых чисел,
* k – целое число, большее нуля,
* n – целое число, большее нуля, но меньшее или равное lenW+1.

Выходные данные:

tmpArray – временный двумерный массив, с которым будут производится действия.

**Таблица 13 – Анализ классов исходных данных**

| **Задача** | **Исходные данные** | **Класс входных данных** | **Класс выходных данных** |
| --- | --- | --- | --- |
| Добавить K столбцов, начиная со столбца с номером N в двумерный массив | twoDimArr | twoDimArr – массив с количеством элементов больше нуля | tmpArray – старый массив, но с дополнительными столбцами |
| k | k – целое число, большее нуля | tmpArray – старый массив, но с дополнительными столбцами |
| * k – строки (не числа) * k – вещественные числа * k – нулю * k – отрицательные целые числа | сообщение об ошибке |
| n | * n – целое число, большее нуля, но меньшее или равное lenW+1 | tmpArray – старый массив, но с дополнительными столбцами |
| * n – строки (не числа) * n – вещественные числа * n – меньше или равно нулю * n >lenW+1 * n – отрицательные целые числа | Сообщение об ошибке |

**3. Программа (листинг)**

На выбор пользователю предоставляется возможность воспользоваться разными функциями меню, в этом случае пользователь выбирает добавление K столбцов с N номера. Весь код программы представлен в приложении. См. Приложение А.

**4. Тестирование**

**Таблица 14 – Тестирование методом чёрного ящика**

| **№** | **Исходные данные** | **Ожидаемый результат** | **Полученный результат** |
| --- | --- | --- | --- |
| **1** | twoDimArr = ‘1 2  3 4’  k = ‘два’  n = ‘’ | Сообщение об ошибке | Сообщение об ошибке |
| **2** | twoDimArr = ‘1 2  3 4’  k = ‘-2’  n = ‘’ | Сообщение об ошибке | Сообщение об ошибке |
| **3** | twoDimArr = ‘1 2  3 4’  k = ‘0’  n = ‘’ | Сообщение об ошибке | Сообщение об ошибке |
| **4** | twoDimArr =‘1 2  3 4’  k = ‘2.5’  n = ‘’ | Сообщение об ошибке | Сообщение об ошибке |
| **5** | twoDimArr =‘1 2  3 4’  k = ‘2’  n = ‘0’ | Сообщение об ошибке | Сообщение об ошибке |
| **6** | twoDimArr =‘1 2  3 4’  k = ‘2’  n = ‘-2’ | Сообщение об ошибке | Сообщение об ошибке |
| **7** | twoDimArr =‘1 2  3 4’  k = ‘2’  n = ‘2.5’ | Сообщение об ошибке | Сообщение об ошибке |
| **8** | twoDimArr =‘1 2  3 4’  k = ‘2’  n = ‘3’ | Сообщение об ошибке | Сообщение об ошибке |
| **9** | twoDimArr =‘1 2  3 4’  k = ‘2’  n = ‘два’ | Сообщение об ошибке | Сообщение об ошибке |
| **10** | twoDimArr =‘1 2  3 4’  k = ‘2’  n = ‘2’ | Новый массив(tmpArray) с двумя новыми столбцами справа | tmpArray = ‘1 2 77 -94  3 4 78 -39 ‘ |

**Задача №8**

**1. Постановка задачи**

Распечатать двумерный динамический массив

**2. Анализ**

Исходные данные:

twoDimArr – двумерный массив, состоящий из целых чисел,

Выходные данные:

symbol– элемент массива.

**Таблица 15 – Анализ классов исходных данных**

| **Задача** | **Исходные данные** | **Класс входных данных** | **Класс выходных данных** |
| --- | --- | --- | --- |
| Распечатать двумерный динамический массив | twoDimArr | twoDimArr – массив с положительном количеством элементов | symbol – каждый элемент массива |

**3. Программа (листинг)**

На выбор пользователю предоставляется возможность воспользоваться разными функциями меню, в этом случае пользователь выбирает распечатать двумерный массив. Весь код программы представлен в приложении. См. Приложение А.

**4. Тестирование**

**Таблица 16 – Тестирование методом чёрного ящика**

| **№** | **Исходные данные** | **Ожидаемый результат** | **Полученный результат** |
| --- | --- | --- | --- |
| **1** | twoDimArr = ‘1  2’ | symbol = ‘1  2’ | symbol = ‘1  2’ |
| **2** | twoDimArr = ‘1 2  3 4’ | symbol = ‘1 2  3 4’ | symbol = ‘1 2  3 4’ |
| **3** | twoDimArr = ‘1 2’ | symbol = ‘1 2’ | symbol = ‘1 2’ |

**Задача №9**

**1. Постановка задачи**

Сформировать динамический рваный массив со случайными числами.

**2. Анализ**

Исходные данные:

* lenH – целое число, большее нуля(количество строчек),
* width – целое число, большее нуля(длина строки).

Выходные данные:

jagDimArray – временный двумерный массив, с которым будут производится действия.

**Таблица 17 – Анализ классов исходных данных**

| **Задача** | **Исходные данные** | **Класс входных данных** | **Класс выходных данных** |
| --- | --- | --- | --- |
| Сформировать динамический рваный массив со случайными числами. | lenH | lenH – целое число, большее нуля | рваный массив(jagDimArr) с lenH количеством строк |
| * lenH – строки (не числа) * lenH – вещественные числа * lenH – нуль * lenH – отрицательные целые числа | сообщение об ошибке |
| width | width – целое число, большее нуля | рваный массив(jagDimArr) с lenH количеством строк |
| * width – строки (не числа) * width – вещественные числа * width – нуль * width – отрицательные целые числа | сообщение об ошибке |

**3. Программа (листинг)**

На выбор пользователю предоставляется возможность создать рваный массив разными способами, в этом случае пользователь выбирает создать рваный массив и заполнить его случайными числами от -100 до 100. Весь код программы представлен в приложении. См. Приложение А.

**4. Тестирование**

**Таблица 18 – Тестирование методом чёрного ящика**

| **№** | **Исходные данные** | **Ожидаемый результат** | **Полученный результат** |
| --- | --- | --- | --- |
| **1** | lenH = ‘-2’  width = ‘’ | сообщение об ошибке | сообщение об ошибке |
| **2** | lenH = ‘2.5’  width = ‘’ | сообщение об ошибке | сообщение об ошибке |
| **3** | lenH = ‘0’  width = ‘’ | сообщение об ошибке | сообщение об ошибке |
| **4** | lenH = ‘три’  width = ‘’ | сообщение об ошибке | сообщение об ошибке |
| **5** | lenH = ‘3’  width = ‘2 2.5 3’ | сообщение об ошибке | сообщение об ошибке |
| **6** | lenH = ‘3’  width = ‘два 3 4’ | сообщение об ошибке | сообщение об ошибке |
| **7** | lenH = ‘3’  width = ‘-2 3 4’ | сообщение об ошибке | сообщение об ошибке |
| **8** | lenH = ‘3’  width = ‘0 3 4’ | сообщение об ошибке | сообщение об ошибке |
| **9** | lenH = ‘3’  width = ‘2 3 4’ | рваный массив с тремя строчками с длинами 2, 3 и 4 | jagDimArr = ‘-63 -37  53 76 -82  38 5 -74 36’ |

**Задача №10**

**1. Постановка задачи**

Сформировать динамический рваный массив с помощью ручного ввода.

**2. Анализ**

Исходные данные:

* lenH – целое число, большее нуля(количество строчек),
* Num – целое число(элемент массива),
* width – целое число, большее нуля(длина строки).

Выходные данные:

jagDimArr – временный двумерный массив, с которым будут производится действия.

**Таблица 19 – Анализ классов исходных данных**

| **Задача** | **Исходные данные** | **Класс входных данных** | **Класс выходных данных** |
| --- | --- | --- | --- |
| Сформировать динамический рваный массив с помощью ручного ввода. | lenH | lenH – целое число, большее нуля | рваный массив(jagDimArr) с lenH количеством строк |
| * lenH – строки (не числа) * lenH – вещественные числа * lenH – нуль * lenH – отрицательные целые числа | сообщение об ошибке |
| width | width – целое число, большее нуля | рваный массив(jagDimArr) с lenH количеством строк |
| * width – строки (не числа) * width – вещественные числа * width – нуль * width – отрицательные целые числа | сообщение об ошибке |
| Num | * Num – целое число | рваный массив(jagDimArr) с lenH количеством строк |
| * Num– строки (не числа) * Num – вещественные числа | сообщение об ошибке ввода |

**3. Программа (листинг)**

На выбор пользователю предоставляется возможность создать рваный массив разными способами, в этом случае пользователь выбирает создать рваный массив и заполнить ручным вводом. Весь код программы представлен в приложении. См. Приложение А.

**4. Тестирование**

**Таблица 20 – Тестирование методом чёрного ящика**

| **№** | **Исходные данные** | **Ожидаемый результат** | **Полученный результат** |
| --- | --- | --- | --- |
| **1** | lenH = ‘-2’  width = ‘’  Num = ‘’ | сообщение об ошибке | сообщение об ошибке |
| **2** | lenH = ‘2.5’  width = ‘’  Num = ‘’ | сообщение об ошибке | сообщение об ошибке |
| **3** | lenH = ‘0’  width = ‘’  Num = ‘’ | сообщение об ошибке | сообщение об ошибке |
| **4** | lenH = ‘три’  width = ‘’  Num = ‘’ | сообщение об ошибке | сообщение об ошибке |
| **5** | lenH = ‘3’  width = ‘2 2.5 3’  Num = ‘’ | сообщение об ошибке | сообщение об ошибке |
| **6** | lenH = ‘3’  width = ‘два 3 4’  Num = ‘’ | сообщение об ошибке | сообщение об ошибке |
| **7** | lenH = ‘3’  width = ‘-2 3 4’  Num = ‘’ | сообщение об ошибке | сообщение об ошибке |
| **8** | lenH = ‘3’  width = ‘0 3 4’  Num = ‘’ | сообщение об ошибке | сообщение об ошибке |
| **9** | lenH = ‘2’  width = ‘2 1’  Num = ‘2 один 3’ | сообщение об ошибке | сообщение об ошибке |
| **10** | lenH = ‘3’  width = ‘2 3 4’  Num = ‘2 1.5 3’ | сообщение об ошибке | сообщение об ошибке |
| **11** | lenH = ‘3’  width = ‘2 3 4’  Num = ‘1 2 3 4 5 6 7 8 9’ | jagDimArr = ‘1 2  3 4 5  6 7 8 9’ | jagDimArr = ‘1 2  3 4 5  6 7 8 9’ |

**Задача №11**

**1. Постановка задачи**

Распечатать рваный массив

**2. Анализ**

Исходные данные:

jagDimArr – рваный массив с lenH количеством строчек,

Выходные данные:

symbol – целое число, элемент массива.

**Таблица 21 – Анализ классов исходных данных**

| **Задача** | **Исходные данные** | **Класс входных данных** | **Класс выходных данных** |
| --- | --- | --- | --- |
| Распечатать рваный массив | jagDimArr | jagDimArr – рваный массив с положительным числом элементов | каждый элемент рваного массива (symbol) |

**3. Программа (листинг)**

На выбор пользователю предоставляется возможность выбрать несколько функций, в этом случае пользователь выбирает распечатать рваный массив. Весь код программы представлен в приложении. См. Приложение А.

**4. Тестирование**

**Таблица 22 – Тестирование методом чёрного ящика**

| **№** | **Исходные данные** | **Ожидаемый результат** | **Полученный результат** |
| --- | --- | --- | --- |
| **1** | jagDimArr = ‘1 2  3 4 5’ | symbol = ‘1 2  3 4 5’ | symbol = ‘1 2  3 4 5’ |
| **2** | jagDimArr = ‘1 2’ | symbol = ‘1 2’ | symbol = ‘1 2’ |
| **3** | jagDimArr = ‘1 2  3 4 5  6’ | symbol = ‘1 2  3 4 5  6’ | symbol = ‘1 2  3 4 5  6’ |

**Задача №12**

**1. Постановка задачи**

Добавить строку с заданным номером в рваный динамический массив

**2. Анализ**

Исходные данные:

* jagDimArr – рваный массив, с lenH количеством строчек,
* numberOfString – целое число, меньшее или равно lenH+1, но больше нуля(номер новой строки),
* lenOfString – целое число, большее нуля(длина новой строки),

Выходные данные:

tmpArray – временный рваный массив, с которым произведены действия, в результате заменит старый массив.

**Таблица 23 – Анализ классов исходных данных**

| **Задача** | **Исходные данные** | **Класс входных данных** | **Класс выходных данных** |
| --- | --- | --- | --- |
| Добавить строку с заданным номером в рваный динамический массив | jagDimArr | jagDimArr – рваный массив с положительным числом элементов | новый рваный массив с дополнительной строкой(tmparray) |
| numberOfString | numberOfString – целое число, большее нуля, но меньшее или равно lenH+1 | новый рваный массив с дополнительной строкой(tmparray) |
| * numberOfString – строки (не числа) * numberOfString – вещественные числа * numberOfString – нуль * numberOfString – отрицательные целые числа * numberOfString – положительные целые числа больше, чем lenH+1 | сообщение об ошибке |
| lenOfString | lenOfString – целое число большее нуля | новый рваный массив с дополнительной строкой(tmparray) |
| * lenOfString – строки (не числа) * lenOfString – вещественные числа * lenOfString – нуль * lenOfString – отрицательные целые числа | сообщение об ошибке |

**3. Программа (листинг)**

На выбор пользователю предоставляется возможность выбрать несколько функций, в этом случае пользователь выбирает добавить строку с определённым номером в рваный массив. Весь код программы представлен в приложении. См. Приложение А.

**4. Тестирование**

**Таблица 24 – Тестирование методом чёрного ящика**

| **№** | **Исходные данные** | **Ожидаемый результат** | **Полученный результат** |
| --- | --- | --- | --- |
| **1** | jagDimArr = ‘1 2  3 4 5’  numberOfString = ‘три’  lenOfString = ‘’ | сообщение об ошибке | сообщение об ошибке |
| **2** | jagDimArr = ‘1 2  3 4 5’  numberOfString = ‘-2’  lenOfString = ‘’ | сообщение об ошибке | сообщение об ошибке |
| **3** | jagDimArr = ‘1 2  3 4 5’  numberOfString = ‘0’  lenOfString = ‘’ | сообщение об ошибке | сообщение об ошибке |
| **4** | jagDimArr = ‘1 2  3 4 5’  numberOfString = ‘2.5’  lenOfString = ‘’ | сообщение об ошибке | сообщение об ошибке |
| **5** | jagDimArr = ‘1 2  3 4 5’  numberOfString = ‘4’  lenOfString = ‘’ | сообщение об ошибке | сообщение об ошибке |
| **6** | jagDimArr = ‘1 2  3 4 5’  numberOfString = ‘2’  lenOfString = ‘0’ | сообщение об ошибке | сообщение об ошибке |
| **7** | jagDimArr = ‘1 2  3 4 5’  numberOfString = ‘2’  lenOfString = ‘-1’ | сообщение об ошибке | сообщение об ошибке |
| **8** | jagDimArr = ‘1 2  3 4 5’  numberOfString = ‘2’  lenOfString = ‘два’ | сообщение об ошибке | сообщение об ошибке |
| **9** | jagDimArr = ‘1 2  3 4 5’  numberOfString = ‘2’  lenOfString = ‘1.5’ | сообщение об ошибке | сообщение об ошибке |
| **10** | jagDimArr = ‘1 2  3 4 5’  numberOfString = ‘2’  lenOfString = ‘3’ | новый массив со второй строкой с длиной 3 и случайными числами от -100 до 100,а вторая строка станет третьей | tmpArray = ‘1 2  -10 54 -50  3 4 5’ |

# ПРИЛОЖЕНИЕ А **Программа**

using System.Diagnostics.Metrics;

using System.Reflection;

using System.Security.Cryptography.X509Certificates;

namespace Labo5

{

internal class Program

{

// 23 вариант

static void Main(string[] args)

{

bool exit = false;

do

{

Console.ForegroundColor = ConsoleColor.White;

Console.Clear();

Console.WriteLine("Меню:");

Console.WriteLine("");

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("[1] - Создать одномерный массив");

Console.ForegroundColor = ConsoleColor.Magenta;

Console.WriteLine("[2] - Создать двумерный массив");

Console.ForegroundColor = ConsoleColor.Green;

Console.WriteLine("[3] - Создать рваный массив");

Console.WriteLine();

Console.ForegroundColor = ConsoleColor.White;

Console.WriteLine("[0] - Выход из программы");

int enter = EnterNumberZero();

switch (enter)

{

case 1: // одномерный массив

Console.ForegroundColor = ConsoleColor.Red;

bool exitOneDim = false;

do

{

enter = EnterMenuArr(enter);

switch (enter)

{

case 1: //ДСЧ

Console.Clear();

Console.WriteLine("Введите целое число - количество элементов массива:");

int len = EnterNumberMoreZero();

int[] oneDimArr = new int[len];

CreateRandomArray(oneDimArr);

ActionWithOneDim(enter, oneDimArr);

break;

case 2: // Ручной ввод

Console.Clear();

Console.WriteLine("Введите целое число - количество элементов массива:");

len = EnterNumberMoreZero();

oneDimArr = new int[len];

CreateArray(oneDimArr);

ActionWithOneDim(enter, oneDimArr);

break;

case 0:

exitOneDim = true;

break;

}

} while (!exitOneDim);

break;

case 2: //двумерный массив

Console.ForegroundColor = ConsoleColor.Magenta;

bool exitTwoDim = false;

do

{

enter = EnterMenuArr(enter);

switch (enter)

{

case 1: //ДСЧ

Console.Clear();

Console.WriteLine("Введите целое число - количество строчек двумерного массива:");

int lenH = EnterNumberMoreZero();

Console.WriteLine("Введите целое число - количество столбцов двумерного массива:");

int lenW = EnterNumberMoreZero();

int[,] twoDimArr = new int[lenH, lenW];

CreateRandomArray(twoDimArr);

ActionWithTwoDim(enter, twoDimArr);

break;

case 2: // Ручной ввод

Console.Clear();

Console.WriteLine("Введите целое число - количество строчек двумерного массива:");

lenH = EnterNumberMoreZero();

Console.WriteLine("Введите целое число - количество столбцов двумерного массива:");

lenW = EnterNumberMoreZero();

twoDimArr = new int[lenH, lenW];

CreateArray(twoDimArr);

ActionWithTwoDim(enter, twoDimArr);

break;

case 0:

exitTwoDim = true;

break;

}

} while (!exitTwoDim);

break;

case 3: //рваный массив

Console.ForegroundColor = ConsoleColor.Green;

bool exitJagDim = false;

do

{

enter = EnterMenuArr(enter);

switch (enter)

{

case 1: // ДСЧ

Console.Clear();

Console.WriteLine("Введите целое число - количество строчек рваного массива:");

int lenH = EnterNumberMoreZero();

int[][] JagDimArr = new int[lenH][];

CreateRandomArray(JagDimArr);

ActionWithJagDim(enter, JagDimArr);

break;

case 2: // Ручной ввод

Console.Clear();

Console.WriteLine("Введите целое число - количество строчек рваного массива:");

lenH = EnterNumberMoreZero();

JagDimArr = new int[lenH][];

CreateArray(JagDimArr);

ActionWithJagDim(enter, JagDimArr);

break;

case 0:

exitJagDim = true;

break;

}

} while (!exitJagDim);

break;

case 0:

exit = true;

break;

}

} while (!exit);

Console.BackgroundColor = ConsoleColor.White;

Console.ForegroundColor = ConsoleColor.Black;

Console.WriteLine("ББ");

Console.BackgroundColor = ConsoleColor.Black;

Console.ForegroundColor = ConsoleColor.White;

}

static void CreateRandomArray(int[] oneDimArr) //Создание случайного одномерного массива

{

Random Rand = new Random();

for (int i = 0; i < oneDimArr.Length; i++)

{

oneDimArr[i] = Rand.Next(-100, 101);

}

}

static void CreateRandomArray(int[,] twoDimArr) //Создание случайного двумерного массива

{

Random Rand = new Random();

for (int i = 0; i < twoDimArr.GetLength(0); i++)

{

for (int j = 0; j < twoDimArr.GetLength(1); j++)

{

twoDimArr[i, j] = Rand.Next(-100, 101);

}

}

}

static void CreateRandomArray(int[][] jagDimArr) //Создание случайного рваного массива

{

Random Rand = new Random();

for (int i = 0; i < jagDimArr.Length; i++)

{

Console.WriteLine("Введите длину строки");

int width = EnterNumberMoreZero();

jagDimArr[i] = new int[width];

for (int j = 0; j < width; j++)

{

jagDimArr[i][j] = Rand.Next(-100, 101);

}

}

}

static void CreateArray(int[] oneDimArr) //Создание одномерного массива со своими числами

{

Random Rand = new Random();

for (int i = 0; i < oneDimArr.Length; i++)

{

oneDimArr[i] = EnterNumberArr();

}

}

static void CreateArray(int[,] twoDimArr) //Создание двумерного массива со своими числами

{

Random Rand = new Random();

for (int i = 0; i < twoDimArr.GetLength(0); i++)

{

for (int j = 0; j < twoDimArr.GetLength(1); j++)

{

twoDimArr[i, j] = EnterNumberArr();

}

}

}

static void CreateArray(int[][] jagDimArr) //Создание рваного массива со своими числами

{

Random Rand = new Random();

for (int i = 0; i < jagDimArr.Length; i++)

{

Console.WriteLine("Введите длину строки");

int width = EnterNumberMoreZero();

jagDimArr[i] = new int[width];

for (int j = 0; j < width; j++)

{

jagDimArr[i][j] = EnterNumberArr();

}

}

}

static int EnterNumberArr() // Ввод целого числа

{

bool ok;

int Num;

do

{

Console.WriteLine("Введите пожалуйста элемент массива, он должен являться целым числом:");

ok = int.TryParse(Console.ReadLine(), out Num);

if (!ok)

{

Console.WriteLine("Вы ввели некорректное число!");

}

} while (!ok);

return Num;

}

static int EnterNumber() // Ввод целого числа

{

bool ok;

int Num;

do

{

Console.WriteLine("Введите пожалуйста целое число:");

ok = int.TryParse(Console.ReadLine(), out Num);

if (!ok)

{

Console.WriteLine("Вы ввели некорректное число!");

}

} while (!ok);

return Num;

}

static int EnterNumberZero() // Ввод целого числа

{

bool ok;

int Num;

do

{

ok = int.TryParse(Console.ReadLine(), out Num);

if (!ok)

{

Console.WriteLine("Вы ввели некорректное число!");

}

} while (!ok);

return Num;

}

static int EnterNumberMoreZero() // Ввод целого большего нуля числа

{

bool ok;

int Num;

do

{

Console.WriteLine("Введите пожалуйста целое число:");

ok = ((int.TryParse(Console.ReadLine(), out Num)) && (Num > 0));

if (!ok)

{

Console.WriteLine("Вы ввели некорректное число!");

}

} while (!ok);

return Num;

}

static void PrintArray(int[] oneDimArr) // Печать одномерного массива

{

foreach (int Symbol in oneDimArr)

{

Console.WriteLine(Symbol);

}

}

static void PrintArray(int[,] twoDimArr) // Печать двумерного массива

{

int Counter = 0;

foreach (int symbol in twoDimArr)

{

Counter++;

if (Counter % twoDimArr.GetLength(1) == 0)

{

Console.WriteLine(symbol);

}

else

{

Console.Write(symbol + " ");

}

}

}

static void PrintArray(int[][] jagDimArr) // Печать рваного массива

{

for (int i = 0; i < jagDimArr.GetLength(0); i++)

{

foreach (int symbol in jagDimArr[i])

{

Console.Write(symbol + " ");

}

Console.WriteLine("");

}

}

static void DeleteEvenOneDim(ref int[] oneDimArr) // Удаление элементов с чётным индексом в одномерном массиве

{

int Counter = 0;

for (int i = 0; i < oneDimArr.Length; i++)

{

if (i % 2 == 0)

{

oneDimArr[Counter] = oneDimArr[i];

Counter++;

}

}

Array.Resize(ref oneDimArr, Counter);

}

static void AddNewColumnsTwoDim(ref int[,] twoDimArr) //Добавление K столбцов с номера N столбца

{

Console.WriteLine("Введите целое число К большее нуля - количество новых столбцов");

Console.WriteLine();

int k = EnterNumberMoreZero();

Console.WriteLine("Введите целое число N - номер столбца, с какого будет происходить добавление");

Console.WriteLine();

int n = EnterNumberPosition(twoDimArr);

AddAlgorythm(ref twoDimArr, k, n);

}

static int EnterNumberPosition(int[,] twoDimArr) // Ввод номера столбца, с которого пойдут действия над двумерным массивом

{

bool ok;

int Num;

do

{

Console.WriteLine("Введите пожалуйста целое число:");

ok = ((int.TryParse(Console.ReadLine(), out Num)) && (Num > 0) && (Num <= twoDimArr.GetLength(1)));

if (!ok)

{

Console.WriteLine("Вы ввели некорректное число!");

}

} while (!ok);

return Num;

}

static void AddAlgorythm(ref int[,] twoDimArr, int k, int n) // Алгоритм добалвения новых столбцов

{

Random Rand = new Random();

int[,] tmpArray = new int[twoDimArr.GetLength(0), twoDimArr.GetLength(1) + k];

for (int j = 0; j <= n - 1; j++) // до новых столбцов

{

for (int i = 0; i < tmpArray.GetLength(0); i++)

{

tmpArray[i, j] = twoDimArr[i, j];

}

}

for (int j = n; j <= n + k - 1; j++) // новые столбцы

{

for (int i = 0; i < tmpArray.GetLength(0); i++)

{

tmpArray[i, j] = Rand.Next(-100, 101);

}

}

for (int j = n + k; j <= tmpArray.GetLength(1) - 1; j++) // После новых столбцов

{

for (int i = 0; i < tmpArray.GetLength(0); i++)

{

tmpArray[i, j] = twoDimArr[i, j - k];

}

}

twoDimArr = tmpArray;

}

static void AddNewStringJag(ref int[][] jagDimArr) // Добавление новой строчки в рваный массив

{

int numberOfString = EnterNumberPosition(jagDimArr)-1;

Console.WriteLine("Введите длину строки:");

int lenOfString = EnterNumberMoreZero();

AddAlgorythm(ref jagDimArr, numberOfString, lenOfString);

}

static int EnterNumberPosition(int[][] jagDimArr) // Ввод числа - номера новой строки

{

Console.WriteLine("Пожалуйста введите целое число большее нуля - номер строки, которую вы хотите добавить:");

bool ok;

int Num;

do

{

Console.WriteLine("Введите пожалуйста целое число:");

ok = ((int.TryParse(Console.ReadLine(), out Num)) && (Num > 0) && (Num <= jagDimArr.GetLength(0) + 1));

if (!ok)

{

Console.WriteLine("Вы ввели некорректное число!");

}

} while (!ok);

return Num;

}

static void AddAlgorythm(ref int[][]jagDimArr, int numberOfString, int lenOfString) // Алгоритм добалвения новых столбцов

{

Random Rand = new Random();

int[][] tmpArray = new int[jagDimArr.GetLength(0) + 1][];

for (int i = 0; i <= numberOfString - 1; i++) // До номера новой строки

{

tmpArray[i] = jagDimArr[i];

}

tmpArray[numberOfString] = new int[lenOfString];

for (int i =0;i<lenOfString;i++ ) // новая строка

{

tmpArray[numberOfString][i] = Rand.Next(-100, 101);

}

for (int i = numberOfString+1; i <= tmpArray.GetLength(0)-1; i++) //После новой строки

{

tmpArray[i] = jagDimArr[i-1];

}

jagDimArr = tmpArray;

}

static int EnterMenuArr(int enter) // функция меню, которая отвечает за печать выбора вида создания массивов

{

Console.Clear();

Console.WriteLine("Каким образом создать массив?");

Console.WriteLine("");

Console.WriteLine("[1] - Создать с помощью ДСЧ");

Console.WriteLine("[2] - Создать вручную");

Console.WriteLine();

Console.WriteLine("[0] - Назад");

enter = EnterNumberZero();

return enter;

}

static void Return(int enter) // функция меню, отвечающая за пункт возврата

{

Console.WriteLine("[0] - Вернуться назад");

do

{

enter = EnterNumberZero();

} while (enter != 0);

}

static void ActionWithOneDim(int enter, int[] oneDimArr) // Функция, которая распечатывает меню и даёт пользователю выбор для одномерного массива

{

bool exitOneDimAct = false;

do

{

Console.Clear();

Console.WriteLine("Что вы хотите сделать с массивом?");

Console.WriteLine("");

Console.WriteLine("[1] - Распечатать массив");

Console.WriteLine("[2] - Удалить все элементы с чётными индексами");

Console.WriteLine();

Console.WriteLine("[0] - Назад");

enter = EnterNumberZero();

switch (enter)

{

case 1: // печать массива

Console.Clear();

Console.WriteLine("Ваш массив:");

PrintArray(oneDimArr);

Return(enter);

break;

case 2: // Удаление элементов с чётными индексами

Console.Clear();

DeleteEvenOneDim(ref oneDimArr);

Console.WriteLine("Ваш массив:");

PrintArray(oneDimArr);

Return(enter);

break;

case 0:

exitOneDimAct = true;

break;

}

} while (!exitOneDimAct);

}

static void ActionWithTwoDim(int enter, int[,] twoDimArr) // Функция, которая распечатывает меню и даёт пользователю выбор для двумерного массива

{

bool exitTwoDimAct = false;

do

{

Console.Clear();

Console.WriteLine("Что вы хотите сделать с массивом?");

Console.WriteLine("");

Console.WriteLine("[1] - Распечатать массив");

Console.WriteLine("[2] - Добавить К столбцов, начиная со столбца с номером N");

Console.WriteLine();

Console.WriteLine("[0] - Назад");

enter = EnterNumberZero();

switch (enter)

{

case 1: // печать массива

Console.Clear();

Console.WriteLine("Ваш массив:");

PrintArray(twoDimArr);

Return(enter);

break;

case 2: // Действие с двумерным массивом, добавление новых столбцов

Console.Clear();

AddNewColumnsTwoDim(ref twoDimArr);

Console.WriteLine("Ваш массив:");

PrintArray(twoDimArr);

Return(enter);

break;

case 0:

exitTwoDimAct = true;

break;

}

} while (!exitTwoDimAct);

}

static void ActionWithJagDim(int enter, int[][]jagDimArr) // Функция, которая распечатывает меню и даёт пользователю выбор для рваного массива

{

bool exitJagDimAct = false;

do

{

Console.Clear();

Console.WriteLine("Что вы хотите сделать с массивом?");

Console.WriteLine("");

Console.WriteLine("[1] - Распечатать массив");

Console.WriteLine("[2] - Добавить строку с заданным номером");

Console.WriteLine();

Console.WriteLine("[0] - Назад");

enter = EnterNumberZero();

switch (enter)

{

case 1: // Печать массива

Console.Clear();

Console.WriteLine("Ваш массив:");

PrintArray(jagDimArr);

Return(enter);

break;

case 2: // Функция добавление новой строчки в рваный массив

Console.Clear();

AddNewStringJag(ref jagDimArr);

Console.Clear();

Console.WriteLine("Ваш массив:");

PrintArray(jagDimArr);

Return(enter);

break;

case 0:

exitJagDimAct = true;

break;

}

} while (!exitJagDimAct);

}

}

}