Федеральное государственное бюджетное образовательное учреждение высшего образования «Пермский национальный исследовательский политехнический университет»

Лабораторная работа №6

По дисциплине

“Объектно-ориентированное программирование”

Тема

“Выполнение программы простой структуры. Вычисление выражений с использованием стандартных функций”

Вариант 13

Выполнил работу

студент группы РИС-19-1б

Мазитов Т.Э.

Проверила

доцент кафедры ИТАС

Викентьева О.Л.

Работу выполнил:

Пермь 2020

**Постановка задачи**

Задача 1:

* Создать динамический массив (одномерный, двумерный, рваный) из элементов заданного типа.
* При заполнении массива использовать 2 способа (ручной и с помощью ДСЧ).Массив вывести на печать.
* Выполнить операции с массивом, указанные в варианте, используя, по возможности, методы класса Array.

|  |  |  |  |
| --- | --- | --- | --- |
| № | Тип массива | Тип элементов | Операция |
| 13 | Одномерный | char | Удалить из массива все цифры |
|  |  |  |  |

* Результаты обработки вывести на печать.

Задача 2:

* Ввести строку символов с клавиатуры. Строка состоит из слов, разделенных пробелами (пробелов может быть несколько) и знаками препинания (, ;:). В строке может быть несколько предложений, в конце каждого предложения стоит знак препинания (.!?).
* Выполнить обработку строки в соответствии с вариантом.
* Результаты обработки вывести на печать.

|  |  |
| --- | --- |
| № | Задание |
| 13 | Перевернуть все слова в предложении и отсортировать слова по убыванию. |

**Анализ задачи**

Пользовательское меню будет состоять из одного уровня и предлагать выбрать один из пунктов работы с одномерным массивом.

Пункты 2,3 не будут работать, пока массив не создан или пуст. Пункты 5,6 не будут работать, пока строка не создана. После успешного выполнения любого пункта будет выведено сообщение и меню.

Каждый пункт меню вызывает одну функцию. Если функция изменяет массив или строку, то в качестве аргумента ей нужно передать массив или строку и записать возвращаемое значение. Для создания массива или строки будет вызываться функция без аргументов, которая возвращает массив или строку. Пользователю будет дана возможность выбрать, как заполнить массив: с помощью генератора псевдослучайных чисел или самому ввести все значения с клавиатуры, и как заполнить строку: использовать готовую строку или ввести свою.

Взаимодействия пользователя с программой представляет собой ввод с клавиатуры чисел, которые должны лежать в определенном промежутке значений. Поэтому каждый такой этап включает в себя валидацию этих значений на соответствие критериям. Пока все критерии не будут соблюдены, программа будет предупреждать пользователя и просить ввести значение еще раз.

**Проектирование**

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Функция** | **Входные данные** | **Выходные данные** |
| **Основные функции** | | | |
| **Задание 1** | | | |
| 1 | CreateArray - выбор способа формирование массива |  | char []– созданный массив |
| 2 | CreateArrayRandom  - создание массива и заполнение случайными символами | int length – длина массива | char [] array – созданный массив |
| 3 | CreateArrayManual  - создание массива и заполнение символами с клавиатуры | int length – длина массива | char [] array – созданный массив |
| 4 | PrintArray - печать массива | char [] array - массив | Печать массива |
| 5 | DeleteDigits– удаление цифр из массива | char [] array - массив | char [] array - массив |
| **Задание 2** | | | |
| 6 | CreateString – создание строки и выбор способа создания |  | string – созданная строка |
| 7 | CreateStringManual – возвращате строку пользователя |  | string – введенная строка |
| 8 | ReverseOrderString – переворачивает слова в предложениях и расставляет их в порядке убывания | string text – строка | string newstring – новая строка |
| 9 | SplitString – разбивает строку на предложения и слова | string text – строка, которую необходимо разбить | string [][] words – отдельные члены предложений (слова, пробелы и знаки препинания) |
| 10 | ReverseWords – для каждого массива вызывает функцию ReverseWordsSentence | string [][] words – рваный массив членов предложений | string [][] - перевернутые члены предложений (слова, пробелы и знаки препинания) |
| 11 | ReverseWordsSentence – переворачивает все слова | string [] words – рваный массив членов предложения | string [] - перевернутые члены предложений (слова, пробелы и знаки препинания) |
| 12 | OrderDesc – для каждого массива вызывает функцию OrderDescSentence | string [][] words – рваный массив членов предложений | string [][] - отсортированные члены предложений (слова, пробелы и знаки препинания) |
| 13 | OrderDescSentence – расставить слова в предложении в порядке убывания длины (исключая пробелы и пунктуацию) | string [] words – рваный массив членов предложения | string [] - отсортированные члены предложения (слова, пробелы и знаки препинания) |
| 14 | AssembleString – создает строку из массивов элементов | string [][] words – рваный массив членов предложений | string – новая строка |
| **Вспомогательные функции** | | | |
| 16 | RandomOrManual – выбор способа заполнения массива / ввода строки | string option1, string option2 – строки, сообщающие пользователю способ ввода информации | bool – булева переменная:  true – заполнение случайными значениями  false – ввод с клавиатуры |
| 16 | IntInput – ввод чисел в консоль | string msg – сообщение для пользователя, что требуется ввести  int begin – начала диапазона значений  int end – конец диапазона значений | int elem – целое число, введенной пользователем |
| 17 | IntLetInput – ввод цифр и букв в консоль | string msg – сообщение для пользователя, что требуется ввести | char elem – символ, введенный пользователем |
| 18 | Menu – диалог для ввода команд пользователя |  | Вызов функций  Сообщение об ошибке |

**Тестирование**

**Навигация по меню**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Ввод | ER | AR | Примечание |
| 1 | 0 | Сообщение об ошибке | Некорректные данные.  Повторите ввод. |  |
| 2 | 8 | Сообщение об ошибке | Некорректные данные. Повторите ввод. |
| 3 | -5 | Сообщение об ошибке | Некорректные данные. Повторите ввод. |
| 4 | “ ” | Сообщение об ошибке | Некорректные данные. Повторите ввод. |
| 5 | abc | Сообщение об ошибке | Некорректные данные. Повторите ввод. |
| 7 | 1 | Меню пункта 1 | Меню пункта 1 |
| 8 | 4 | Меню пункта 4 | Меню пункта 4 |
| 9 | 7 | Выход из программы | Выход из программы |

**Создание массива**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Ввод | ER | AR | Примечание |
| 1 | 1 | 1 число | 6 | Ввод длины массива |
| 2 | 5 | 5 чисел | 1 2 3 4 5 |
| 3 | 10 | 10 чисел | 10 чисел |
| 4 | 0 | Сообщение об ошибке | Некорректная длина. Повторите ввод. |
| 5 | -5 | Сообщение об ошибке | Некорректная длина. Повторите ввод. |
| 6 | 1.23 | Сообщение об ошибке | Некорректная длина. Повторите ввод. |
| 7 | abc | Сообщение об ошибке | Некорректная длина. Повторите ввод. |
| 8 | 100 | Сообщение об ошибке | Некорректная длина. Повторите ввод. |
| 9 | 1 | 1 | 1 | Ввод элементов массива с клавиатуры |
| 10 | a | a | a |
| 11 | ф | ф | ф |
| 12 | 43567897656, 1.4, abc | Сообщение об ошибке | Введите только один символ |
| 13 | 10 | Сообщение об ошибке | Введите только один символ |

**Удаление цифр из массива**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Ввод | ER | AR |
| 1 | 1 a 2 b | a b | a b |
| 2 | 1 2 3 | Пустой массив | Пустой массив |
| 3 | a b c | a b c | a b c |

**Разделение строки на предложения и слова**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Ввод | ER | AR |
| 1 | abcd abc, ab. abcd : ab, a! | "abcd", " ", "abc", ",", " ", "ab"  "."  " ", "abcd", " ", ":", " ", "ab", ",", " ", "a"  "!" | "abcd", " ", "abc", ",", " ", "ab"  "."  " ", "abcd", " ", ":", " ", "ab", ",", " ", "a"  "!" |
| 2 | "abcd: ab, cd. abc ab" | "abcd", ":", " ", "ab", ",", " ", "cd"  "."  " ", "abc", " ", "ab" | "abcd", ":", " ", "ab", ",", " ", "cd"  "."  " ", "abc", " ", "ab" |
| 3 | ": ab, cd! abc " | ":", " ", "ab", ",", " ", "cd"  "!"  " ", "abc", " " | ":", " ", "ab", ",", " ", "cd"  "!"  " ", "abc", " " |

**Переворот слов в предложении**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Ввод | ER | AR |
| 1 | "abcd", " ", "ab", ",", " ", "abc" | "dcba", " ", "ba", ",", " ", "cba" | "dcba", " ", "ba", ",", " ", "cba" |
| 2 | "abc", "ab", "a" | "cba", "ba", "a" | "cba", "ba", "a" |
| 3 | "" | "" | "" |
| 4 | " ", ".", ",", ":", "!" | " ", ".", ",", ":", "!" | " ", ".", ",", ":", "!" |

**Сортировка слова по убыванию длины**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Ввод | ER | AR |
| 1 | "abcd", " ", "ab", ",", " ", "abc" | "abcd", " ", "abc", ",", " ", "ab" | "abcd", " ", "abc", ",", " ", "ab" |
| 2 | "ab", " ", "ab", ",", " ", "ab" | "ab", " ", "ab", ",", " ", "ab" | "ab", " ", "ab", ",", " ", "ab" |
| 3 | "abcd", "ab", "abc" ," ", ",", " " | "abcd", "abc", "ab", " ", ",", " " | "abcd", "abc", "ab", " ", ",", " " |
| 4 | "a","abcd", "abc", "abcde","ab" | "abcde", "abcd", "abc", "ab", "a" | "abcde", "abcd", "abc", "ab", "a" |

**Сборка строки из слов**

|  |  |  |  |
| --- | --- | --- | --- |
| № | ER | Ввод | AR |
| 1 | "abcd", " ", "abc", ",", " ", "ab"  "."  " ", "abcd", " ", ":", " ", "ab", ",", " ", "a"  "!" | abcd abc, ab. abcd : ab, a! | abcd abc, ab. abcd : ab, a! |
| 2 | "abcd", ":", " ", "ab", ",", " ", "cd"  "."  " ", "abc", " ", "ab" | "abcd: ab, cd. abc ab" | "abcd: ab, cd. abc ab" |
| 3 | ":", " ", "ab", ",", " ", "cd"  "!"  " ", "abc", " " | ": ab, cd! abc " | ": ab, cd! abc " |
| 4 | "abcd", " ", "abc", ",", " ", "ab"  "."  " ", "abcd", " ", ":", " ", "ab", ",", " ", "a"  "!" | "abcd abc, ab. abcd : ab, a!" | "abcd abc, ab. abcd : ab, a!" |

**Интеграционное тестирование (функция ReverseOrderString)**

|  |  |  |  |
| --- | --- | --- | --- |
| № | ER | Ввод | AR |
| 1 | "a abcd, ab. abcd : ab, abc!" | "dcba ba, a. dcba : cba, ba!" | "dcba ba, a. dcba : cba, ba!" |
| 2 | "a a, a. a : a, a!" | "a a, a. a : a, a!" | "a a, a. a : a, a!" |
| 3 | ",ab!" | ",ba!" | ",ba!" |
| 4 | "abc!" | "cba!" | "cba!" |

**Листинг программы**

using System;

using System.Collections;

using System.Collections.Generic;

using System.Collections.Immutable;

using System.ComponentModel.DataAnnotations.Schema;

using System.Linq;

using System.Text.RegularExpressions;

namespace LR6

{

class Program

{

public static void Main(string[] args) => Menu();

static void Menu()

{

char[] array = new char[0];

string stroke = "";

int choice;

bool exit = false;

while (!exit)

{

Console.WriteLine("\nМеню:" +

"\n\t1. Создать одномерный массив;" +

"\n\t2. Распечатать массив;" +

"\n\t3. Удалить цифры;" +

"\n\t4. Создать строку;" +

"\n\t5. Перевернуть слова и расположить в порядке убыванию длины;" +

"\n\t6. Распечатать строку;" +

"\n\t7. Выход из программы.");

choice = IntInput("Ваш выбор: ",1,7);

if (array.Length == 0 && choice >= 2 && choice <= 3)

Console.WriteLine("\nОШИБКА. Создайте массив!");

else if (stroke == "" && choice >= 5 && choice <= 6)

Console.WriteLine("\nОШИБКА. Создайте строку!");

else

{

switch (choice)

{

case 1:

array = CreateArray();

break;

case 2:

Console.WriteLine("\nВывод элементов массива.\n");

PrintArray(array);

break;

case 3:

Console.WriteLine("\nУдаление цифр из массива.\n");

array = DeleteDigits(array);

break;

case 4:

stroke = CreateString();

break;

case 5:

stroke = ReverseString(stroke);

break;

case 6:

Console.WriteLine(stroke);

break;

case 7:

exit = true;

break;

}

}

}

Console.WriteLine("\nЗавершение работы программы");

}

//---------- МЕТОДЫ ОБРАБОТКИ ВВОДА ----------

static bool RandomOrManual(string option1, string option2)

{

Console.WriteLine($"\n1. {option1}\n2. {option2}");

int choice = IntInput("Ваш выбор: ", 1, 2);

return (choice == 1) ? true : false;

}

static int IntInput(string msg, int beg, int end)

{

bool flag;

int elem = 0;

do

{

flag = false;

Console.WriteLine();

Console.Write(msg);

try

{

elem = Convert.ToInt32(Console.ReadLine());

if (elem < beg || elem > end) throw new IndexOutOfRangeException();

}

catch

{

flag = true;

Console.WriteLine("\nВы ввели некорректные данные. Повторите ввод.");

}

} while (flag);

return elem;

}

static char IntLetInput(string msg)

{

bool flag;

char elem = '1';

do

{

flag = false;

Console.WriteLine();

Console.Write(msg);

try

{

elem = Convert.ToChar(Console.ReadLine());

if (!char.IsDigit(elem) && !char.IsLetter(elem)) throw new IndexOutOfRangeException();

}

catch

{

flag = true;

Console.WriteLine("\nВы ввели некорректные данные. Повторите ввод.");

}

} while (flag);

return elem;

}

//---------- ЗАДАНИЕ 1 ----------

static char[] CreateArray()

{

int length = IntInput("Введите длину массива [1, 10]: ", 1, 10);

if (RandomOrManual("Заполнение массива случайными числами от 0 до 9.", "Ввод элементов массива с клавиатуры."))

return CreateArrayRandom(length);

else

return CreateArrayManual(length);

}

static char[] CreateArrayRandom(int length)

{

Random rnd = new Random();

return Enumerable.Range(0, length).Select(index => rnd.Next(0, 2) == 0 ? (char)rnd.Next('a', 'z') : (char)rnd.Next('0', '9')).ToArray();

}

static char[] CreateArrayManual(int length)

{

return Enumerable.Range(0, length).Select(index => IntLetInput("Введите цифру или букву: ")).ToArray();

}

static void PrintArray(char[] array)

{

Array.ForEach(array, x => Console.Write(" {0} ", x));

Console.WriteLine();

foreach (var i in Enumerable.Range(1, array.Length)) Console.Write("[{0}] ", i);

Console.WriteLine();

}

static char[] DeleteDigits(char[] array)

{

return array.Where(x => !new Regex(@"[0-9]").IsMatch(x.ToString())).Select(x => x).ToArray();

}

//---------- ЗАДАНИЕ 2 ----------

static string CreateString()

{

if (RandomOrManual("Использовать готовую строку.","Ввести строку."))

return

"Сегодня, несмотря на погоду, был чудесный день! " +

"Я увидел много нового: как размышляют люди; как зарождается жизнь; как стареют поколения. " +

"Возможно, такое со мной никогда больше не случится? Но никто точно не знает.";

else

return CreateStringManual();

}

static string CreateStringManual()

{

Console.WriteLine("\nВведите строку: ");

return Console.ReadLine();

}

static string ReverseString(string text)

{

string [][] words = SplitString(text);

words = OrderDesc(ReverseWords(words));

string reversed = AssembleString(words);

return reversed;

}

static string[][] SplitString(string text)

{

string[] sentences = Regex.Split(text, @"(!)|(\.)|(\?)").Where(s => s != "").ToArray();

string[][] words = sentences.Select(sentence => Regex.Split(sentence, @"(;)|(:)|(,)|( )").Where(w => w != "").ToArray()).ToArray();

return words;

}

static string[][] ReverseWords(string[][] words)

{

return words.Select(row => row.Select(word =>

{

var array = word.ToCharArray();

Array.Reverse(array);

return string.Join("", array);

}

).ToArray()).ToArray();

}

static string[][] OrderDesc(string[][] words)

{

return words.Select(row =>

{

var temp = row.ToArray();

Array.Sort(temp);

Stack<string> st = new Stack<string>(temp

.SkipWhile(word => !char.IsLetterOrDigit(word[0]))

.OrderBy(word => word.Length));

foreach (var i in Enumerable.Range(0, row.Length))

{

if (char.IsLetterOrDigit(row[i][0]))

row[i] = st.Pop();

}

return row;

}).ToArray();

}

static string AssembleString(string[][] words)

{

var elem = string.Join("", words.Select(row => string.Join("", row)));

return elem;

}

//---------- НЕИСПОЛЬЗУЕМЫЕ МЕТОДЫ ----------

static void PrintWords(string[][] words)

{

Array.ForEach(words, row =>

{

Array.ForEach(row, word => Console.Write(word));

Console.WriteLine();

});

Console.WriteLine();

}

static string[][] SSplitString(string text)

{

string[] sentences = text.Split(new char[] { '!', '?', '.' }, StringSplitOptions.RemoveEmptyEntries);

string[][] words = sentences.Select(sentence => sentence.Split(new char[] { ' ', ';', ':', ',' }, StringSplitOptions.RemoveEmptyEntries)).ToArray();

return words;

}

}

}