|  |
| --- |
| Пермский филиал федерального государственного автономного образовательного учреждения высшего образования  «Национальный исследовательский университет  «Высшая школа экономики»  *Факультет экономики, менеджмента и бизнес-информатики* |
|  |
| Меньшиков Олег Владимирович  *Лабораторная 6*  по направлению подготовки *09.03.04 Программная инженерия*  образовательная программа «Программная инженерия»   |  |  | | --- | --- | |  | Доцент к[афедры информационных технологий в бизнесе](https://perm.hse.ru/bi/infobis/), кандидат технических наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  О.Л. Викентьева | |

# **Анализ**

**Постановка задачи 1.**

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

|  |  |  |  |
| --- | --- | --- | --- |
| 10 | Одномерный | int | Найти сумму всех четных элементов, которые находятся между первым минимальным элементом и последним максимальным элементом массива. Минимальных и максимальных элементов в массиве может быть несколько. |

1. Результаты обработки вывести на печать.

**Постановка задачи 2.**

1. 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 (.!?).
2. Выполнить обработку строки в соответствии с вариантом.
3. Результаты обработки вывести на печать.

|  |  |
| --- | --- |
| 10 | Определить есть ли в строке идентификаторы, если есть, то напечатать самый короткий идентификатор. |

Строка является идентификатором, когда:

1. Не содержит знаков препинания.
2. Начинается с @ и продолжение является ключевым словом.
3. Не начинается с цифры.
4. Не содержит только цифры.
5. Может начинаться с \_.
6. Не является ключевым словом.

# **Проектирование**

Таблица 1 – Функции

|  |  |  |  |
| --- | --- | --- | --- |
| **Название функции** | **Назначение** | **Входные параметры** | **Выходные параметры** |
| ParsingAll | Функция конвертирует строку в тип integer |  | Число типа integer(d) |
| Parsing | Функция конвертирует строку в тип integer, если число неотрицательно |  | Неотрицательное число типа integer(d) |
| CreateArrRandom | Функция создает одномерный массив с помощью датчика случайных чисел |  | Одномерный массив(mas) |
| CreateArr | Функция создает одномерный массив, элементы которого вводятся пользователем |  | Одномерный массив(mas) |
| OutputArray | Функция выводит на консоль все элементы одномерного массива | Одномерный массив(mas) |  |
| OutputMenu1 | Функция выводит меню 1 на экран |  |  |
| OutputMenu2 | Функция выводит меню 2 на экран |  |  |
| FirstMin | Функция возвращает индекс первого минимального элемента | Одномерный массив(mas) | int indexMin |
| LastMax | Функция возвращает индекс последнего максимального элемента | Одномерный массив(mas) | int indexMax |
| SumEvenBetweenMaxAndMin | Функция возвращает сумму четных элементов между indexMin и indexMax | Одномерный массив(mas) | int sum |
| SeparateWords | Функция разбивает предложения на массив слов | String s | String[] arr |
| ListOfKeyWords | Функция возвращает массив ключевых слов в С# |  | String[] arr |
| ListOfChar | Функция возвращает массив символов, недопустимых для идентификаторов |  | Char[] arr |
| HasNoSigns | Функция возвращает, имеет ли строка знаки из массива символов | String s | Bool ok |
| StartsWithDog | Функция возвращает, начинается ли строка с @ | String s | Bool ok |
| ContainsKeyWord | Функция возвращает, содержит ли строка ключевое слово C# | String s | Bool ok |
| IsKeyWord | Функция возвращает, является ли строка ключевым словом C# | String s | Bool ok |
| StartsWithSignOrDigit | Функция возвращает, начинается ли строка со знака из массива символов или цифры | String s | Bool ok |
| FindTheShortestIdent | Функция возвращает кратчайший(-е) идентификаторы | String[] arr | String[]mas |
| CountIdent | Функция возвращает количество идентификаторов в строке | String[] arr | Int s |
| GetIdent | Функция создает массив идентификаторов из элементов заданного массива | String[] arr | String[] mas |
| OutputIdent | Функция выводит на экран массив | String[] arr |  |

# **Листинг**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace лаба6

{

class Program

{

static int Parsing()

{

bool ok;

int n;

do

{

string buf = Console.ReadLine();

ok = int.TryParse(buf, out n);

if ((!ok) || (n < 0)) Console.WriteLine("Значение введено неверно! Введите заново: ");

}

while ((!ok) || (n < 0));

return n;

}

static int ParsingAll()

{

bool ok;

int n;

do

{

string buf = Console.ReadLine();

ok = int.TryParse(buf, out n);

if (!ok) Console.WriteLine("Значение введено неверно! Введите заново: ");

}

while (!ok);

return n;

}

static int[] CreateArr()

{

Console.WriteLine("Введите количество чисел в массиве: ");

int size = Parsing();

int[] mas;

if (size != 0)

{

mas = new int[size];

Console.WriteLine("Введите элементы массива через Enter: ");

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

mas[i] = ParsingAll();

Console.WriteLine("Полученный массив:");

foreach (int x in mas) Console.Write(x + " ");

Console.WriteLine();

}

else

{

mas = null;

Console.WriteLine("Полученный массив:\nМассив пуст");

}

return mas;

}

static int[] CreateArrRandom()

{

Random rand = new Random();

Console.WriteLine("Введите количество чисел в массиве: ");

int size = Parsing();

int[] mas;

if (size != 0)

{

mas = new int[size];

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

mas[i] = rand.Next(-100, 100);

Console.WriteLine("Полученный массив:");

foreach (int x in mas) Console.Write(x + " ");

Console.WriteLine();

}

else

{

mas = null;

Console.WriteLine("Полученный массив:\nМассив пуст");

}

return mas;

}

static void OutputArr(int[] mas)

{

Console.WriteLine("Текущий массив: ");

if (mas != null)

{

foreach (int x in mas)

Console.Write(x + " ");

Console.WriteLine();

}

else Console.WriteLine("Массив пуст.");

} //

static void OutputMenu1()

{

Console.ForegroundColor = ConsoleColor.Green;

Console.WriteLine(@" МЕНЮ 1

1. Создать массив через ДСЧ

2. Создать массив вручную

3. Вывести массив на экран

4. Выполнить задачу из варианта #10

5. Переход в меню 2

6. Выход из программы");

Console.ResetColor();

Console.ForegroundColor = ConsoleColor.Black;

Console.BackgroundColor = ConsoleColor.Green;

Console.WriteLine("Введите выбранный пункт меню:");

Console.ResetColor();

}

static void OutputMenu2()

{

Console.ForegroundColor = ConsoleColor.DarkRed;

Console.WriteLine(@" МЕНЮ 2

1. Найти идентификаторы в строке

2. Переход в меню 1

3. Выход из программы");

Console.ResetColor();

Console.ForegroundColor = ConsoleColor.White;

Console.BackgroundColor = ConsoleColor.DarkRed;

Console.WriteLine("Введите выбранный пункт меню: ");

Console.ResetColor();

}

static int FirstMin(int[] mas)

{

int min = mas[0];

int indexMin = 0;

for (int i = 1; i < mas.Length; i++)

if (mas[i] < min)

{

min = mas[i];

indexMin = i;

}

return indexMin;

}

static int LastMax(int[] mas)

{

int max = mas[0];

int indexMax = 0;

for (int i = 1; i < mas.Length; i++)

if (mas[i] >= max)

{

max = mas[i];

indexMax = i;

}

return indexMax;

}

static void SumEvenBetweenMaxAndMin(int[] mas)

{

int sum = 0;

if (mas == null) Console.WriteLine("Массив пуст. Действие невозможно.");

else if ((mas.Length == 1) || (mas.Length == 2)) Console.WriteLine("Действие невозможно, необходимо минимум 3 числа в последовательности!");

else

{

int left = FirstMin(mas);

int rigth = LastMax(mas);

if (left > rigth)

{

int temp = left;

left = rigth;

rigth = temp;

}

for (int i = left + 1; i < rigth; i++)

if (mas[i] % 2 == 0)

sum += mas[i];

Console.WriteLine($"Полученная сумма = {sum}");

}

}

//строки

static string[] SeparateWords (string a)

{

char[] da = { ' ', '.' };

string[] arr = a.Split(da, StringSplitOptions.RemoveEmptyEntries);

return arr;

}

static string[] ListOfKeyWords()

{

string[] keyWords = {"abstract", "as", "base", "bool", "break","byte", "case","catch", "char", "checked","class",

"const","continue","decimal","default","delegate","do","double","else","enum","event","explicit","extern","false","finally",

"fixed","float","for","foreach","goto","if","implicit","in","int","interface","internal","is","lock","long","namespace",

"new","null","object","operator","out","override","params","private","protected","public","readonly","ref","return", "sbyte",

"sealed","short","sizeof","stackalloc","static","string","struct","switch","this","throw","true","try","typeof","uint","ulong",

"unchecked","unsafe","ushort","using","virtual","void","while", "add","ascending","async","await","by","descending","dynamic","equals",

"from","get","global","group","in","into","join","let","on","orderby","partial","remove","select","set","value","var","where","yield"};

return keyWords;

}

static char[] ListOfChar()

{

char[] keySigns = { '.', ',', '<', '>', '?', '/', ':', ';', '"', ']', '[', '{', '}', '=', '+', '-', '—', '@', '!',

')', '(', '\*', '&', '\\', '^', '%', '$', '#', '№', '`', '~'};

return keySigns;

}

static char[] ListOfCharAndDigits()

{

char[] keyCharAndDigits = { '.', ',', '<', '>', '?', '/', ':', ';', '"', ']', '[', '{', '}', '=', '+', '-', '—', '@', '!',

')', '(', '\*', '&', '\\', '^', '%', '$', '#', '№', '`', '~', '1', '2', '3', '4', '5', '6', '7', '8', '9', '0'};

return keyCharAndDigits;

}

static bool HasNoSigns(string str)

{

int c = 0;

char[] keySigns = ListOfChar();

for (int i = 1; i < str.Length; i++) //начинаем со 2го символа - 1ый символ будет рассматриваться в другой из-за правил си шарпа

for (int j = 0; j < keySigns.Length; j++)

{

if (str[i].Equals(keySigns[j])) c++;

}

if (c == 0) return true;

else return false;

}

static bool StartsWithDog(string str)

{

if (str[0].Equals('@')) return true;

else return false;

}

static bool ContainsKeyWord(string str)

{

int c = 0;

string[] keyWords = ListOfKeyWords();

for (int j = 0; j < keyWords.Length; j++)

if (str.Substring(1, str.Length - 1).Equals(keyWords[j]))

c++;

if (c == 0) return false;

else return true;

}

static bool IsKeyWord(string str)

{

int c = 0;

string[] keyWords = ListOfKeyWords();

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

if (str.Equals(keyWords[i])) c++;

if (c == 0) return false;

else return true;

}

static bool StartsWithSignOrDigit(string str)

{

int c = 0;

char[] keySigns = ListOfCharAndDigits();

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

if (str[0].Equals(keySigns[i])) c++;

if (c == 0) return false;

else return true;

}

static string[] FindTheShortestIdent(string[] ident)

{

string[] arr = null;

if (ident == null) return arr;

else

{

int minStr = ident[0].Length;

string shortStr = ident[0];

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

if (ident[i].Length < minStr)

minStr = ident[i].Length;

int c = 0;

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

if (minStr == ident[i].Length) c++;

arr = new string[c];

c = 0;

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

if (minStr == ident[i].Length)

{

arr[c] = ident[i];

c++;

}

return arr;

}

}

static int CountIdent(string[] arr)

{

int s = 0;

for (int k = 0; k < arr.Length; k++)

{

string str = arr[k];

if (HasNoSigns(str) && (StartsWithDog(str) && ContainsKeyWord(str) || !StartsWithSignOrDigit(str) && !IsKeyWord(str)))

s++;

}

return s;

}

static string[] GetIdent(int s, string[] arr)

{

string[] ident;

if (s == 0) ident = null;

else

{

ident = new string[s];

int d = 0;

for (int k = 0; k < arr.Length; k++)

{

string str = arr[k];

if (HasNoSigns(str) && (StartsWithDog(str) && ContainsKeyWord(str) || !StartsWithSignOrDigit(str) && !IsKeyWord(str)))

{

ident[d] = arr[k];

d++;

}

}

}

return ident;

}

static void OutputShortestIdent(string[] r)

{

if (r == null) Console.WriteLine("Идентификторов не найдено!");

else

{

Console.Write("Кратчайший(-e) идентификатор(-ы): ");

foreach (string x in r)

Console.Write(x + " ");

}

Console.WriteLine();

}

static void OutputIdent(string[] ident)

{

if (ident == null) Console.WriteLine();

else

{

Console.Write("Найденные идентификаторы: ");

foreach (string x in ident)

Console.Write(x + " ");

Console.WriteLine();

}

}

static void Main(string[] args)

{

Console.WriteLine("----------ЛАБОРАТОРНАЯ №6-----------");

bool ok, goOn, dalshe; // переменные-флажки

int[] mas = null;

do

{

goOn = false;

do

{

ok = false;

dalshe = false;

OutputMenu1();

int k = Parsing();

switch (k)

{

case 1:

mas = CreateArrRandom();

break;

case 2:

mas = CreateArr();

break;

case 3:

OutputArr(mas);

break;

case 4:

SumEvenBetweenMaxAndMin(mas);

break;

case 5:

ok = true;

goOn = true;

break;

case 6:

ok = true;

goOn = true;

dalshe = true;

break;

default:

Console.WriteLine("Данного пункта нет в меню 1");

break;

}

}

while (!ok);

while (!dalshe)

{

dalshe = false;

OutputMenu2();

int i = Parsing();

switch (i)

{

case 1:

Console.WriteLine("Введите строку:");

string a = Console.ReadLine();

string[] arr = SeparateWords(a);

if (arr == null) Console.WriteLine("В строке не найдено слов!");

else

{

int s = CountIdent(arr);

string[] ident = GetIdent(s, arr);

OutputIdent(ident);

string[] r = FindTheShortestIdent(ident);

OutputShortestIdent(r);

Console.WriteLine();

}

break;

case 2:

dalshe = true;

goOn = false;

break;

case 3:

dalshe = true;

goOn = true;

break;

}

}

}

while (!goOn);

}

}

}

# **Тесты**

См. файл Тесты.xlsx