МИНИСТЕРСТВОНАУКИИ ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение   
высшего образования

«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ   
УНИВЕРСИТЕТ»

Факультет информационных систем и технологий

Кафедра Информационные системы

Отчет по практике

защищен с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(оценка прописью)

Руководитель   
практики от  
университета \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(должность)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись) (ФИО)

**ОТЧЕТ по**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ практике

(вид практики)

Студента \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(ФИО)

Направление (специальность,

профиль) подготовки \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Группа \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Место прохождения практики \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(наименование профильной организации, подразделение университета)

Ульяновск 20\_\_\_ г.

Задание 1:

Создание приложения (консольное приложение), знакомство с языком.

* 1. Пользователь вводит в консоль два числа, определить, делится ли второе на первое, и вывести об этом сообщение в консоль.
  2. Пользователь вводит в консоль три числа. Определить, делится ли их сумма на разность первого и второго, и вывести об этом сообщение в консоль.

Код:

* 1. Program.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab1

{

class Program

{

static void Main(string[] args)

{

int i = 0;

while (true)

{

Console.Clear();

switch (i)

{

case 0:

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("----------------------------------");

Console.WriteLine("| Task 6 |");

Console.WriteLine("----------------------------------");

Console.ForegroundColor = ConsoleColor.Gray;

Console.WriteLine("----------------------------------");

Console.WriteLine("| Task 10 |");

Console.WriteLine("----------------------------------");

Console.WriteLine("----------------------------------");

Console.WriteLine("| Exit |");

Console.WriteLine("----------------------------------");

}

break;

case 1:

{

Console.ForegroundColor = ConsoleColor.Gray;

Console.WriteLine("----------------------------------");

Console.WriteLine("| Task 6 |");

Console.WriteLine("----------------------------------");

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("----------------------------------");

Console.WriteLine("| Task 10 |");

Console.WriteLine("----------------------------------");

Console.ForegroundColor = ConsoleColor.Gray;

Console.WriteLine("----------------------------------");

Console.WriteLine("| Exit |");

Console.WriteLine("----------------------------------");

}

break;

case 2:

{

Console.ForegroundColor = ConsoleColor.Gray;

Console.WriteLine("----------------------------------");

Console.WriteLine("| Task 6 |");

Console.WriteLine("----------------------------------");

Console.WriteLine("----------------------------------");

Console.WriteLine("| Task 10 |");

Console.WriteLine("----------------------------------");

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("----------------------------------");

Console.WriteLine("| Exit |");

Console.WriteLine("----------------------------------");

Console.ForegroundColor = ConsoleColor.Gray;

}

break;

}

ConsoleKeyInfo d = Console.ReadKey();

if (d.KeyChar == 'w') if (i - 1 < 0) i = 2; else i--;

if (d.KeyChar == 's') if (i + 1 > 2) i = 0; else i++;

if (d.Key == ConsoleKey.Enter)

{

bool anyExceptions = true;

switch(i)

{

case 0:

{

while (anyExceptions)

{

Console.Clear();

Console.WriteLine("Enter 2 numbers:");

try

{

int a = Convert.ToInt32(Console.ReadLine());

int b = Convert.ToInt32(Console.ReadLine());

if ((a != 0) && b % a == 0) Console.WriteLine("Divisible.");

else Console.WriteLine("Indivisible.");

anyExceptions = false;

Console.WriteLine("Press any key to return to menu...");

Console.ReadKey();

}

catch (Exception ex)

{

Console.WriteLine(ex.Message + "\nPress any key to retry...");

Console.ReadKey();

}

}

}

break;

case 1:

{

while (anyExceptions)

{

Console.Clear();

Console.WriteLine("Enter 3 numbers:");

try

{

int a = Convert.ToInt32(Console.ReadLine());

int b = Convert.ToInt32(Console.ReadLine());

int c = Convert.ToInt32(Console.ReadLine());

if ((b - a) != 0 && (a + b + c) % (b - a) == 0) Console.WriteLine("Divisible.");

else Console.WriteLine("Indivisible.");

anyExceptions = false;

Console.WriteLine("Press any key to return to menu...");

Console.ReadKey();

}

catch (Exception ex)

{

Console.WriteLine(ex.Message + "\nPress any key to retry...");

Console.ReadKey();

}

}

}

break;

case 2:

Environment.Exit(0);

break;

}

}

if (d.Key == ConsoleKey.Escape) Environment.Exit(0);

}

}

}

}

Задание 2:

Создание приложения (консольное приложение), работа с одномерным массивом.

Блок 1:

1.6. Дано N целых чисел. Требуется выбрать из них три таких числа, произведение которых максимально.

1.10. Дан целочисленный массив A размера 11. Вывести номер первого из тех его элементов A[i], которые удовлетворяют двойному неравенству: A[1] < A[i] < A[10]. Если таких элементов нет, то вывести 0.

Блок 2:

2.6. Дан массив размера N. Определить количество его промежутков монотонности (то есть участков, на которых его элементы возрастают или убывают) и переставить местами первый и последний из них.

2.10. Поменять местами первое простое и первое совершенное числа в массиве

Блок 3:

3.6. Вставить 0 после каждого элемента, большего по модулю среднего арифметического положительных элементов. Тестовый массив 4 5 6 -4 -5 -12 -34 3 4 Результат: среднее арифметическое равно 4,4, результат вывода: 4 5 0 6 0 -4 -5 0 -12 0 -34 0 3 4

3.10. Вставить максимальное положительное после каждого нечетного отрицательного. Тестовый пример -12 -3 -4 5 3 4 5 -12 -4 -5 6 5 4 Результат: максимальное положительное=6, результат вывода: -12 -3 6 -4 5 3 4 5 -12 -4 -5 6 6 5 4

Блок 4:

4.6. Удалить элементы массива НЕ кратные его последнему элементу (последний элемент при этом не рассматривать). Тестовый пример 4 3 4 5 -1 -2 -3 2. Результат: 4 4 -2 2.

4.10. Удалить элементы массива, которые будут делителями для суммы его первого и последнего элементов (элементы рассматривать по модулю). Тестовый пример 2 3 4 5 -1 -2 -3 -4. Результат: Сумма первого и последнего 6, результат 4 5 -4.

Код:

Program.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab2

{

class Program

{

static void Main(string[] args)

{

int i = 0;

Console.SetWindowSize(60, 50);

Menu menu = new Menu(8);

while (true)

{

Console.Clear();

menu.print(i);

ConsoleKeyInfo d = Console.ReadKey();

if (d.KeyChar == 'w' || d.KeyChar == 'ц') if (i - 1 < 0) i = 8; else i--;

if (d.KeyChar == 's' || d.KeyChar == 'ы') if (i + 1 > 8) i = 0; else i++;

if (d.Key == ConsoleKey.Enter)

{

Console.Clear();

bool anyExceptions = true;

switch(i)

{

case 0:

{

while (anyExceptions)

{

Console.Clear();

Console.WriteLine("Enter array size, min value, max value: ");

try

{

int n = Convert.ToInt32(Console.ReadLine());

int min = Convert.ToInt32(Console.ReadLine());

int max = Convert.ToInt32(Console.ReadLine());

Array1D a1d = new Array1D(n, min, max);

Console.WriteLine("\nAnswer: " + String.Join(" ", a1d.solve\_task1()) + "\n\nFrom generated array:");

a1d.print(a1d.n);

anyExceptions = false;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message + "\nPress any key to retry...");

}

Console.WriteLine("\nPress any key to return to menu...");

Console.ReadKey();

}

}

break;

case 1:

{

while (anyExceptions)

{

Console.Clear();

Console.WriteLine("Enter min value, max value of array elements: ");

try

{

int min = Convert.ToInt32(Console.ReadLine());

int max = Convert.ToInt32(Console.ReadLine());

Array1D a1d = new Array1D(11, min, max);

Console.WriteLine("\nAnswer: " + Convert.ToString(a1d.solve\_task2()) + "\n\nFrom generated array: ");

a1d.print(a1d.n);

anyExceptions = false;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message + "\nPress any key to retry...");

}

Console.WriteLine("\nPress any key to return to menu...");

Console.ReadKey();

}

}

break;

case 2:

{

while (anyExceptions)

{

Console.Clear();

Console.WriteLine("Enter array size, min value, max value: ");

try

{

int n = Convert.ToInt32(Console.ReadLine());

int min = Convert.ToInt32(Console.ReadLine());

int max = Convert.ToInt32(Console.ReadLine());

Array1D a1d = new Array1D(n, min, max);

Console.WriteLine("Generated array: ");

a1d.print(a1d.n);

a1d.solve\_task3();

Console.WriteLine("Solved array: ");

a1d.print(a1d.n);

anyExceptions = false;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message + "\nPress any key to retry...");

}

Console.WriteLine("\nPress any key to return to menu...");

Console.ReadKey();

}

}

break;

case 3:

{

while (anyExceptions)

{

Console.Clear();

Console.WriteLine("Enter array size, min value, max value: ");

try

{

int n = Convert.ToInt32(Console.ReadLine());

int min = Convert.ToInt32(Console.ReadLine());

int max = Convert.ToInt32(Console.ReadLine());

Array1D a1d = new Array1D(n, min, max);

Console.WriteLine("Generated array: ");

a1d.print(a1d.n);

a1d.solve\_task4();

Console.WriteLine("Changed array: ");

a1d.print(a1d.n);

anyExceptions = false;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message + "\nPress any key to retry...");

}

Console.WriteLine("\nPress any key to return to menu...");

Console.ReadKey();

}

}

break;

case 4:

{

while (anyExceptions)

{

Console.Clear();

Console.WriteLine("Enter array size, min value, max value: ");

try

{

int n = Convert.ToInt32(Console.ReadLine());

int min = Convert.ToInt32(Console.ReadLine());

int max = Convert.ToInt32(Console.ReadLine());

Array1D a1d = new Array1D(n \* 2, min, max);

Console.WriteLine("\nGenerated array: ");

a1d.print(a1d.n / 2);

a1d.solve\_task5();

anyExceptions = false;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message + "\nPress any key to retry...");

}

Console.WriteLine("\nPress any key to return to menu...");

Console.ReadKey();

}

}

break;

case 5:

{

while (anyExceptions)

{

Console.Clear();

Console.WriteLine("Enter array size, min value, max value: ");

try

{

int n = Convert.ToInt32(Console.ReadLine());

int min = Convert.ToInt32(Console.ReadLine());

int max = Convert.ToInt32(Console.ReadLine());

Array1D a1d = new Array1D(n \* 2, min, max);

Console.WriteLine("\nGenerated array: ");

a1d.print(a1d.n / 2);

Console.WriteLine("\nChanged array: ");

a1d.solve\_task6();

anyExceptions = false;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message + "\nPress any key to retry...");

}

Console.WriteLine("\nPress any key to return to menu...");

Console.ReadKey();

}

}

break;

case 6:

{

while (anyExceptions)

{

Console.Clear();

Console.WriteLine("Enter array size, min value, max value: ");

try

{

int n = Convert.ToInt32(Console.ReadLine());

int min = Convert.ToInt32(Console.ReadLine());

int max = Convert.ToInt32(Console.ReadLine());

Array1D a1d = new Array1D(n, min, max);

Console.WriteLine("\nGenerated array: ");

a1d.print(a1d.n);

Console.WriteLine("\nChanged array: ");

a1d.solve\_task7();

anyExceptions = false;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message + "\nPress any key to retry...");

}

Console.WriteLine("\nPress any key to return to menu...");

Console.ReadKey();

}

}

break;

case 7:

{

while (anyExceptions)

{

Console.Clear();

Console.WriteLine("Enter array size, min value, max value: ");

try

{

int n = Convert.ToInt32(Console.ReadLine());

int min = Convert.ToInt32(Console.ReadLine());

int max = Convert.ToInt32(Console.ReadLine());

Array1D a1d = new Array1D(n, min, max);

Console.WriteLine("\nGenerated array: ");

a1d.print(a1d.n);

Console.WriteLine("\nChanged array: ");

a1d.solve\_task8();

anyExceptions = false;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message + "\nPress any key to retry...");

}

Console.WriteLine("\nPress any key to return to menu...");

Console.ReadKey();

}

}

break;

case 8:

Environment.Exit(0);

break;

}

}

if (d.Key == ConsoleKey.Escape) Environment.Exit(0);

}

}

}

}

Array1D.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab2

{

class Array1D

{

protected int[] array;

public int n;

public Array1D(int n, int min, int max)

{

this.array = new int[n];

this.n = n;

Random rand = new Random();

for (int i = 0; i < n; i++) this.array[i] = rand.Next(min, max + 1);

}

public int[] solve\_task1()

{

int max\_case1 = 1;

int max\_case2 = 1;

int[] max\_nums\_1 = new int[3];

int[] max\_nums\_2 = new int[3];

bool[] flags = Enumerable.Repeat(false, array.Length).ToArray();

bool flag = false;

int max\_j = -1;

int cur;

bool has\_zero = false;

//Solve for case 1 (3 positives, should work with 3 negatives)

for (int i = 0; i < 3; i++)

{

flag = false;

cur = Int32.MinValue;

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

{

if (array[j] >= cur && !flags[j])

{

cur = array[j];

max\_j = j;

flag = true;

}

}

flags[max\_j] = true;

if (flag)

{

max\_nums\_1[i] = cur;

max\_case1 \*= cur;

}

}

//Solve for case 2 (2 negatives 1 positive)

flags = Enumerable.Repeat(false, array.Length).ToArray();

for (int i = 0; i < 2; i++)

{

flag = false;

cur = -1;

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

{

if (array[j] <= cur && !flags[j])

{

cur = array[j];

max\_j = j;

flag = true;

}

}

flags[max\_j] = true;

if (flag)

{

max\_case2 \*= cur;

max\_nums\_2[i] = cur;

}

}

cur = 0;

max\_j = 0;

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

{

if (array[j] == 0) has\_zero = true;

if (array[j] > cur && !max\_nums\_2.Contains(array[j]))

{

cur = array[j];

}

}

max\_nums\_2[2] = cur;

max\_case2 \*= cur;

if (has\_zero)

{

if (max\_case1 > max\_case2) return max\_nums\_1;

else return max\_nums\_2;

}

else if (max\_case1 == 0 || max\_case1 > max\_case2) return max\_nums\_1;

else return max\_nums\_2;

}

public int solve\_task2()

{

int i = 0;

while (i < 11)

{

if (array[i] > array[1] && array[i] < array[10]) return i;

i++;

}

return 0;

}

public void solve\_task3()

{

bool mnt = true; // true = growing, false = declining

int i = 0;

int first\_start = -1;

int first\_end = -1;

int last\_start = -1;

int last\_end = -1;

try

{

//Finding the 1st mnt zone

while (array[i] == array[i + 1] && i < array.Length - 1) i++;

if (i < array.Length - 1 && array[i] > array[i + 1]) mnt = false;

first\_start = i;

if (mnt) while (i < array.Length - 1 && array[i] < array[i + 1]) i++;

else while (i < array.Length - 1 && array[i] > array[i + 1]) i++;

first\_end = i;

//Finding the 2nd mnt zone

i = array.Length - 1;

while (array[i] == array[i - 1] && i > 0) i--;

if (i > 0 && array[i] > array[i - 1]) mnt = true;

else mnt = false;

last\_end = i;

if (mnt) while (i > 0 && array[i] > array[i - 1]) i--;

else while (i > 0 && array[i] < array[i - 1]) i--;

last\_start = i;

}

catch (Exception ex) {}

List<int> firstext = new List<int>();

List<int> lastext = new List<int>();

for (i = first\_start; i <= first\_end; i++)

{

firstext.Add(array[i]);

array[i] = Int32.MinValue;

}

for (i = last\_start; i <= last\_end; i++)

{

lastext.Add(array[i]);

array[i] = Int32.MinValue;

}

int ext\_count = 0;

i = 0;

while (i < array.Length - 1 && array[i] == array[i + 1]) i++;

if (i != array.Length - 1)

{

if (array[i] > array[i + 1]) mnt = false;

else mnt = true;

for (i = 0; i < array.Length - 1; i++)

{

if (array[i] == array[i + 1]) continue;

if (array[i] > array[i + 1] && mnt)

{

ext\_count++;

mnt = !mnt;

}

if (array[i] < array[i + 1] && !mnt)

{

ext\_count++;

mnt = !mnt;

}

}

}

else ext\_count = -1;

Console.Write("Mnt count: " + (ext\_count + 1));

int offset = (last\_end - last\_start + 1) - (first\_end - first\_start + 1);

if (offset != 0 && first\_start != -1 && first\_end != -1 && last\_start != -1 && last\_end != -1)

{

if (offset > 0)

{

i = array.Length - 1;

while (array[i] != Int32.MinValue) i--;

while (array[i] == Int32.MinValue) i--;

while (array[i] != Int32.MinValue)

{

array[i + 1] = array[i];

array[i] = Int32.MinValue;

i--;

}

}

if (offset < 0)

{

i = 0;

while (array[i] != Int32.MinValue) i++;

while (array[i] == Int32.MinValue) i++;

while (array[i] != Int32.MinValue)

{

array[i - 1] = array[i];

array[i] = Int32.MinValue;

i++;

}

}

i = 0;

while (array[i] != Int32.MinValue) i++;

for (int j = 0; j < lastext.Count; j++)

{

array[i] = lastext[j];

i++;

}

i = array.Length - 1;

while (array[i] != Int32.MinValue) i--;

for (int j = 0; j < firstext.Count; j++)

{

array[i] = firstext[firstext.Count - j - 1];

i--;

}

}

else

{

for (i = 0; i < firstext.Count; i++)

{

array[first\_start] = lastext[i];

array[last\_start] = firstext[i];

first\_start++;

last\_start++;

}

}

}

public void solve\_task4()

{

int sum = 0;

int i = 0;

bool flag\_perfect = false;

for (i = 0; i < array.Length; i++)

{

sum = 0;

for (int j = 1; j < Math.Truncate(Math.Sqrt(array[i])) + 1; j++)

{

if (array[i] % j == 0) sum += j + (array[i] / j);

}

if (array[i] == (sum - array[i]) && Math.Abs(array[i]) >= 6)

{

flag\_perfect = true;

break;

}

}

int count = 0;

int k = 0;

bool flag\_prime = false;

for (k = 0; k < array.Length; k++)

{

count = 0;

for (int j = 1; j < Math.Truncate(Math.Sqrt(array[k])) + 1; j++)

{

if ((array[k] % j) == 0) count++;

}

if (count == 1)

{

flag\_prime = true;

break;

}

}

if (flag\_perfect) Console.WriteLine("\nPerfect number index: " + i);

else Console.WriteLine("\nNo perfect number found.");

if (flag\_prime) Console.WriteLine("Prime number index: " + k + '\n');

else Console.WriteLine("No prime number found." + '\n');

if (flag\_perfect && flag\_prime)

{

int temp = array[i];

array[i] = array[k];

array[k] = temp;

}

}

public void solve\_task5()

{

double average = -1;

int sum = 0;

int count = 0;

int new\_len = array.Length / 2;

for (int i = 0; i < new\_len; i++)

{

if (array[i] > 0)

{

sum += array[i];

count++;

}

}

if (count != 0) average = (double)sum / (double)count;

for (int i = 0; i < new\_len; i++)

{

if (Math.Abs(array[i]) > average)

{

for (int j = array.Length - 1; j > i; j--)

{

array[j] = array[j - 1];

}

array[i + 1] = 0;

i++;

new\_len++;

}

}

Console.WriteLine("Average: " + average);

for (int i = 0; i < new\_len; i++) Console.Write("" + array[i] + " ");

}

public void solve\_task6()

{

int new\_len = array.Length / 2;

int maxpos = -1;

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

{

if (array[i] > maxpos) maxpos = array[i];

}

Console.WriteLine("Max positive: " + maxpos);

if (maxpos == -1)

{

Console.WriteLine("No positive numbers found.");

return;

}

for (int i = 0; i < new\_len; i++)

{

if (array[i] % 2 == -1)

{

for (int j = array.Length - 1; j > i; j--)

{

array[j] = array[j - 1];

}

array[i + 1] = maxpos;

i++;

new\_len++;

}

}

for (int i = 0; i < new\_len; i++) Console.Write("" + array[i] + " ");

}

public void solve\_task7()

{

if (array[array.Length - 1] == 0)

{

Console.WriteLine("Last element is 0.");

return;

}

int last = array[array.Length - 1];

int new\_len = array.Length;

int k = array.Length - 2;

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

{

if (array[i] % last != 0)

{

for (int j = i; j < array.Length - 1; j++)

{

array[j] = array[j + 1];

}

new\_len--;

i--;

}

}

for (int i = 0; i < new\_len; i++)

{

Console.Write("" + array[i] + " ");

}

}

public void solve\_task8()

{

if (array[0] + array[array.Length - 1] == 0)

{

Console.WriteLine("Sum is 0.");

Console.Write(String.Join(" ", array));

return;

}

int sum = array[0] + array[array.Length - 1];

int new\_len = array.Length;

int k = array.Length - 2;

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

{

if (array[i] != 0 && sum % array[i] == 0)

{

for (int j = i; j < array.Length - 1; j++)

{

array[j] = array[j + 1];

}

new\_len--;

i--;

}

}

for (int i = 0; i < new\_len; i++)

{

Console.Write("" + array[i] + " ");

}

}

public void print(int len)

{

for (int i = 0; i < len; i++)

{

Console.Write(String.Format("{0, -4}", array[i]));

}

Console.WriteLine();

Console.ForegroundColor = ConsoleColor.Gray;

}

}

}

Menu.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab2

{

class Menu

{

int task\_count;

public Menu(int i)

{

this.task\_count = i;

}

public void print(int cur\_i)

{

for (int k = 0; k < task\_count; k++)

{

if (k == cur\_i) Console.ForegroundColor = ConsoleColor.Red;

else Console.ForegroundColor = ConsoleColor.Gray;

Console.WriteLine("----------------------------------");

Console.WriteLine("| Task {0} |", k);

Console.WriteLine("----------------------------------");

}

if (cur\_i == task\_count) Console.ForegroundColor = ConsoleColor.Red;

else Console.ForegroundColor = ConsoleColor.Gray;

Console.WriteLine("----------------------------------");

Console.WriteLine("| Exit |");

Console.WriteLine("----------------------------------");

}

}

}

Задание 3:

Создание приложения (консольное приложение), работа с двумерным массивом.

Блок 1:

1.6. Поменять местами столбец с первым отрицательным в последней строке и последним положительным в первой.

1.10. Найти сумму в первой половине массива и сумму во второй (по столбцам). Определить, какая из этих сумм больше. Если первая больше, заменить нулями элементы в первом столбце, иначе в первой строке.

Блок 2:

2.6. Добавить(дублировать) строку, содержащую максимальный элемент после строк с элементами массива, которые будут делителями для суммы его первого и последнего элементов (элементы рассматривать по модулю).

2.10. Добавить строку нулей после строки с элементами, которые будут делителями для суммы его первого, второго (в первой строке) и последнего элементов (элементы рассматривать по модулю).

Блок 3:

3.6. Упорядочить элементы в матрице по возрастанию (справа налево, снизу-вверх).

3.10. Отсортировать матрицу по возрастанию. Направление: слева направо, сверху вниз.

Код:

Array2D.cs (реализация)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab3

{

class Array2D

{

protected int[][] array;

public int n;

public Array2D(int n, int m, int min, int max)

{

this.array = new int[n][];

this.n = n;

Random rand = new Random();

for (int i = 0; i < n; i++)

{

this.array[i] = new int[m];

for (int j = 0; j < array[i].Length; j++)

{

array[i][j] = rand.Next(min, max + 1);

}

}

}

public void solve\_106()

{

int first\_neg\_i = -1;

int last\_pos\_i = -1;

int i = 0;

while (first\_neg\_i == -1 && i < array[array.Length - 1].Length)

{

if (array[array.Length - 1][i] < 0) first\_neg\_i = i;

i++;

}

i = array[0].Length - 1;

while (last\_pos\_i == -1 && i >= 0)

{

if (array[0][i] > 0) last\_pos\_i = i;

i--;

}

if (first\_neg\_i == -1 || last\_pos\_i == -1)

{

Console.WriteLine("Could not find one of the two numbers.");

return;

}

i = 0;

for (i = 0; i < array.Length; i++)

{

int temp = array[i][first\_neg\_i];

array[i][first\_neg\_i] = array[i][last\_pos\_i];

array[i][last\_pos\_i] = temp;

}

}

public void solve\_110()

{

int fh = 0, sh = 0;

int c = array.Length / 2 + array.Length % 2;

for (int i = 0; i < array[0].Length / 2; i++)

{

Console.WriteLine("c = " + c + " i = " + i);

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

{

fh += array[j][i];

sh += array[j][c];

}

c++;

}

if (array[0].Length % 2 != 0)

{

for (int i = 0; i < array.Length; i++) sh += array[i][array.Length / 2];

}

Console.WriteLine("First half: " + fh);

Console.WriteLine("Second half: " + sh);

if (fh > sh)

{

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

{

array[i][0] = 0;

}

}

else

{

for (int i = 0; i < array[0].Length; i++)

{

array[0][i] = 0;

}

}

}

public void solve\_206()

{

int sum = Math.Abs(array[0][0]) + Math.Abs(array[array.Length - 1][array[0].Length - 1]);

int max = Int32.MinValue;

int max\_i = -1;

int new\_len = array.Length / 2;

for (int i = 0; i < new\_len; i++)

{

for (int j = 0; j < array[i].Length; j++) if (array[i][j] > max)

{

max = array[i][j];

max\_i = i;

}

}

Console.WriteLine("Max: " + max + " Sum: " + sum);

for (int i = 0; i < new\_len; i++)

{

for (int j = 0; j < array[i].Length; j++)

{

if (array[i][j] != 0 && sum % array[i][j] == 0)

{

Console.WriteLine("Found at row " + i);

for (int k = array.Length - 1; k > i; k--) array[k] = array[k - 1];

if (i < max\_i) max\_i++;

array[i + 1] = array[max\_i];

i++;

new\_len++;

break;

}

}

}

Console.WriteLine("New len: " + new\_len);

for (int i = 0; i < new\_len; i++)

{

for (int j = 0; j < array[i].Length; j++)

{

Console.Write(String.Format("{0, 3} ", array[i][j]));

}

Console.Write(Environment.NewLine + Environment.NewLine);

}

}

public void solve\_210()

{

int sum = Math.Abs(array[0][0]) + Math.Abs(array[0][1]) + Math.Abs(array[array.Length - 1][array[0].Length - 1]);

int new\_len = array.Length / 2;

Console.WriteLine("Sum: " + sum);

for (int i = 0; i < new\_len; i++)

{

for (int j = 0; j < array[i].Length; j++)

{

if (array[i][j] != 0 && sum % array[i][j] == 0)

{

Console.WriteLine("Found at row " + i);

for (int k = array.Length - 1; k > i; k--) array[k] = array[k - 1];

for (int m = 0; m < array[i + 1].Length; m++) array[i + 1][m] = 0;

i++;

new\_len++;

break;

}

}

}

for (int i = 0; i < new\_len; i++)

{

for (int j = 0; j < array[i].Length; j++)

{

Console.Write(String.Format("{0, 3} ", array[i][j]));

}

Console.Write(Environment.NewLine + Environment.NewLine);

}

}

public void solve\_306()

{

int i = 0;

int size = array.Length \* array[1].Length;

while (i < size)

{

int cur\_row = i / array.Length;

int cur\_col = i % array.Length;

int max = array[cur\_row][cur\_col];

int max\_i = -1;

int max\_j = -1;

for (int j = cur\_row; j < array.Length; j++)

{

int k;

if (j == cur\_row) k = cur\_col;

else k = 0;

for (; k < array[j].Length; k++)

{

if (array[j][k] > max)

{

max = array[j][k];

max\_i = j;

max\_j = k;

}

}

}

if (max\_i != -1)

{

int temp = array[cur\_row][cur\_col];

array[cur\_row][cur\_col] = array[max\_i][max\_j];

array[max\_i][max\_j] = temp;

}

i++;

}

}

public void solve\_310()

{

int i = 0;

int size = array.Length \* array[1].Length;

while (i < size)

{

int cur\_row = i / array.Length;

int cur\_col = i % array.Length;

int min = array[cur\_row][cur\_col];

int min\_i = -1;

int min\_j = -1;

for (int j = cur\_row; j < array.Length; j++)

{

int k;

if (j == cur\_row) k = cur\_col;

else k = 0;

for (; k < array[j].Length; k++)

{

if (array[j][k] < min)

{

min = array[j][k];

min\_i = j;

min\_j = k;

}

}

}

if (min\_i != -1)

{

int temp = array[cur\_row][cur\_col];

array[cur\_row][cur\_col] = array[min\_i][min\_j];

array[min\_i][min\_j] = temp;

}

i++;

}

}

public void print(int rows)

{

for (int i = 0; i < rows; i++)

{

for (int j = 0; j < array[i].Length; j++)

{

Console.Write(string.Format("{0, 3} ", array[i][j]));

}

Console.Write(Environment.NewLine + Environment.NewLine);

}

}

}

}

Program.cs (меню, интерфейс)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab3

{

class Program

{

static void Main(string[] args)

{

int i = 0;

Console.SetWindowSize(50, 50);

Menu menu = new Menu(6);

while (true)

{

Console.Clear();

menu.print(i);

ConsoleKeyInfo d = Console.ReadKey();

if (d.KeyChar == 'w') if (i - 1 < 0) i = 6; else i--;

if (d.KeyChar == 's') if (i + 1 > 6) i = 0; else i++;

if (d.Key == ConsoleKey.Enter)

{

Console.Clear();

bool anyExceptions = true;

switch (i)

{

case 0:

{

while (anyExceptions)

{

Console.Clear();

Console.WriteLine("Enter 2D array dimensions (rows, cols), min value, max value: ");

try

{

int n = Convert.ToInt32(Console.ReadLine());

int m = Convert.ToInt32(Console.ReadLine());

int min = Convert.ToInt32(Console.ReadLine());

int max = Convert.ToInt32(Console.ReadLine());

Array2D a2d = new Array2D(n, m, min, max);

Console.WriteLine("Generated array: ");

a2d.print(a2d.n);

a2d.solve\_106();

Console.WriteLine("Solved array:");

a2d.print(a2d.n);

anyExceptions = false;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message + "\nPress any key to retry...");

}

Console.WriteLine("\nPress any key to return to menu...");

Console.ReadKey();

}

}

break;

case 1:

{

while (anyExceptions)

{

Console.Clear();

Console.WriteLine("Enter 2D array dimensions (rows, cols), min value, max value: ");

try

{

int n = Convert.ToInt32(Console.ReadLine());

int m = Convert.ToInt32(Console.ReadLine());

int min = Convert.ToInt32(Console.ReadLine());

int max = Convert.ToInt32(Console.ReadLine());

Array2D a2d = new Array2D(n, m, min, max);

Console.WriteLine("Generated array: ");

a2d.print(a2d.n);

a2d.solve\_110();

Console.WriteLine("Solved array:");

a2d.print(a2d.n);

anyExceptions = false;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message + "\nPress any key to retry...");

}

Console.WriteLine("\nPress any key to return to menu...");

Console.ReadKey();

}

}

break;

case 2:

{

while (anyExceptions)

{

Console.Clear();

Console.WriteLine("Enter 2D array dimensions (rows, cols), min value, max value: ");

try

{

int n = Convert.ToInt32(Console.ReadLine());

int m = Convert.ToInt32(Console.ReadLine());

int min = Convert.ToInt32(Console.ReadLine());

int max = Convert.ToInt32(Console.ReadLine());

Array2D a2d = new Array2D(n \* 2, m, min, max);

Console.WriteLine("Generated array: ");

a2d.print(a2d.n / 2);

Console.WriteLine("Solved array:");

a2d.solve\_206();

anyExceptions = false;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message + "\nPress any key to retry...");

}

Console.WriteLine("\nPress any key to return to menu...");

Console.ReadKey();

}

}

break;

case 3:

{

while (anyExceptions)

{

Console.Clear();

Console.WriteLine("Enter 2D array dimensions (rows, cols), min value, max value: ");

try

{

int n = Convert.ToInt32(Console.ReadLine());

int m = Convert.ToInt32(Console.ReadLine());

int min = Convert.ToInt32(Console.ReadLine());

int max = Convert.ToInt32(Console.ReadLine());

Array2D a2d = new Array2D(n \* 2, m, min, max);

Console.WriteLine("Generated array: ");

a2d.print(a2d.n / 2);

Console.WriteLine("Solved array:");

a2d.solve\_210();

anyExceptions = false;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message + "\nPress any key to retry...");

}

Console.WriteLine("\nPress any key to return to menu...");

Console.ReadKey();

}

}

break;

case 4:

{

while (anyExceptions)

{

Console.Clear();

Console.WriteLine("Enter 2D array dimensions (rows, cols), min value, max value: ");

try

{

int n = Convert.ToInt32(Console.ReadLine());

int m = Convert.ToInt32(Console.ReadLine());

int min = Convert.ToInt32(Console.ReadLine());

int max = Convert.ToInt32(Console.ReadLine());

Array2D a2d = new Array2D(n, m, min, max);

Console.WriteLine("Generated array: ");

a2d.print(a2d.n);

a2d.solve\_306();

Console.WriteLine("Solved array:");

a2d.print(a2d.n);

anyExceptions = false;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message + "\nPress any key to retry...");

}

Console.WriteLine("\nPress any key to return to menu...");

Console.ReadKey();

}

}

break;

case 5:

{

while (anyExceptions)

{

Console.Clear();

Console.WriteLine("Enter 2D array dimensions (rows, cols), min value, max value: ");

try

{

int n = Convert.ToInt32(Console.ReadLine());

int m = Convert.ToInt32(Console.ReadLine());

int min = Convert.ToInt32(Console.ReadLine());

int max = Convert.ToInt32(Console.ReadLine());

Array2D a2d = new Array2D(n, m, min, max);

Console.WriteLine("Generated array: ");

a2d.print(a2d.n);

a2d.solve\_310();

Console.WriteLine("Solved array:");

a2d.print(a2d.n);

anyExceptions = false;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message + "\nPress any key to retry...");

}

Console.WriteLine("\nPress any key to return to menu...");

Console.ReadKey();

}

}

break;

}

}

if (d.Key == ConsoleKey.Escape) Environment.Exit(0);

}

}

}

}

Menu.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab3

{

class Menu

{

int task\_count;

public Menu(int i)

{

this.task\_count = i;

}

public void print(int cur\_i)

{

for (int k = 0; k < task\_count; k++)

{

if (k == cur\_i) Console.ForegroundColor = ConsoleColor.Red;

else Console.ForegroundColor = ConsoleColor.Gray;

Console.WriteLine("----------------------------------");

Console.WriteLine("| Task {0} |", k);

Console.WriteLine("----------------------------------");

}

if (cur\_i == task\_count) Console.ForegroundColor = ConsoleColor.Red;

else Console.ForegroundColor = ConsoleColor.Gray;

Console.WriteLine("----------------------------------");

Console.WriteLine("| Exit |");

Console.WriteLine("----------------------------------");

}

}

}

Задание 4:

Создание приложения (консольное приложение), работа со строками и файлами.

1.6. Из файла считывается строка, содержащая несколько круглых и квадратных скобок. Если скобки расставлены правильно (то есть каждой открывающей соответствует одна закрывающая), то вывести число 0. В противном случае вывести или номер позиции, в которой расположена первая ошибочная закрывающая скобка, или, если закрывающих скобок не хватает, число –1. Уточнить, каких именно скобок не хватает. Вывод осуществить в файл. Выбор файлов осуществить через диалоги.

1.13. Из файла считывается текст на кириллице. Вывести в другой файл этот же текст на латинице (каждая буква кириллицы заменяется латинской по звучанию или сочетаниями типа ш – sh, ч – ch), инвертируем регистр и располагаем слова в случайном порядке. Выбор файлов осуществить через диалоги.

1.20. Из файла считывается n строк на кириллице. Инвертировать строки, чья длина больше среднего арифметического длин всех строк и инвертировать регистр повторяющихся гласных в них. Вывести гласные и количество их в этих строках в другой файл, отсортированные по выбору пользователя - по алфавиту, по длине слов или по количеству повторов. Выбор файлов осуществить через диалоги.

Код:

Form1.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

using System.Diagnostics;

namespace Lab4Forms

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

timer1.Interval = 1000;

}

private void Form1\_Load(object sender, EventArgs e)

{

}

private void button1\_Click(object sender, EventArgs e)

{

if (openFileDialog1.ShowDialog() == DialogResult.OK)

{

using (StreamReader sr = new StreamReader(openFileDialog1.FileName, Encoding.GetEncoding("utf-8")))

{

string s = sr.ReadLine();

Debug.WriteLine(s);

int balance\_round = 0;

int balance\_square = 0;

Queue<int> opBrckts = new Queue<int>();

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

{

if (s[i] == '(') balance\_round++;

if (s[i] == '[') balance\_square++;

if (s[i] == ')') balance\_round--;

if (s[i] == ']') balance\_square--;

if (balance\_round < 0 || balance\_square < 0)

{

label1.Text = "First inappropriate closing bracket at: " + i + '\n';

return;

}

}

if (balance\_round > 0)

{

string l = "-1\nMissing closing circle brackets for opening brackets at: ";

for (int i = s.Length - 1; i >= 0; i--)

{

if (s[i] == ')')

{

int j = i - 1;

while (s[j] != '(' || opBrckts.Contains(j)) j--;

opBrckts.Enqueue(j);

}

if (s[i] == '(' && !opBrckts.Contains(i)) l += Convert.ToString(i) + " ";

}

label1.Text = l;

}

opBrckts.Clear();

if (balance\_square > 0)

{

string l;

if (label1.Text.Contains("-1")) l = "\n";

else l = "\n-1";

l += "\nMissing closing square brackets for opening brackets at: ";

for (int i = s.Length - 1; i >= 0; i--)

{

if (s[i] == ']')

{

int j = i - 1;

while (s[j] != '[' || opBrckts.Contains(j)) j--;

opBrckts.Enqueue(j);

}

if (s[i] == '[' && !opBrckts.Contains(i)) l += Convert.ToString(i) + " ";

}

label1.Text += l;

}

if (balance\_round == 0 && balance\_square == 0) label1.Text = "0";

}

}

}

private void button2\_Click(object sender, EventArgs e)

{

string text = "";

if (openFileDialog1.ShowDialog() == DialogResult.OK)

{

using (StreamReader sr = new StreamReader(openFileDialog1.FileName))

{

string s;

string[] latin = new string[] { "a", "b", "v", "g", "d", "e", "yo", "zh", "z", "i", "ii", "k", "l", "m", "n", "o", "p", "r", "s", "t", "u", "f", "h", "c", "ch", "sh", "sh", "i", "y", "i", "a", "yu", "ya", " ",

"A", "B", "V", "G", "D", "E", "YO", "ZH", "Z", "I", "II", "K", "L", "M", "N", "O", "P", "R", "S", "T", "U", "F", "H", "C", "CH", "SH", "SH", "I", "Y", "I", "A", "YU", "YA", " "};

char[] cyrillic = new char[] { 'а', 'б', 'в', 'г', 'д', 'е', 'ё', 'ж', 'з', 'и', 'й', 'к', 'л', 'м', 'н', 'о', 'п', 'р', 'с', 'т', 'у', 'ф', 'х', 'ц', 'ч', 'ш', 'щ', 'ъ', 'ы', 'ь', 'э', 'ю', 'я', ' ',

'А', 'Б', 'В', 'Г', 'Д', 'Е', 'Ё', 'Ж', 'З', 'И', 'Й', 'К', 'Л', 'М', 'Н', 'О', 'П', 'Р', 'С', 'Т', 'У', 'Ф', 'Х', 'Ц', 'Ч', 'Ш', 'Щ', 'Ъ', 'Ы', 'Ь', 'Э', 'Ю', 'Я', ' '};

while ((s = sr.ReadLine()) != null)

{

Debug.WriteLine(s);

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

{

bool flag = false;

for (int j = 0; j < cyrillic.Length; j++) if (cyrillic[j] == s[i]) flag = true;

if (flag) text += latin[(Array.FindIndex(cyrillic, item => item == s[i]) + (cyrillic.Length / 2)) % cyrillic.Length];

else text += s[i];

}

text += Environment.NewLine;

}

label1.Text = text;

}

}

if (saveFileDialog1.ShowDialog() == DialogResult.OK)

{

using (StreamWriter sw = File.CreateText(saveFileDialog1.FileName))

{

//Forming a list of words

List<string> words = new List<string>();

string word;

for (int i = 0; i < text.Length - 1; i++)

{

word = "";

while (i < text.Length - 1 && text[i] != ' ' && Convert.ToString(text[i]) != Environment.NewLine)

{

word += text[i];

i++;

}

words.Add(word);

}

//Shuffle list

Random rand = new Random();

int n = words.Count;

while (n > 1)

{

n--;

int k = rand.Next(n + 1);

string temp = words[k];

words[k] = words[n];

words[n] = temp;

}

//Reforming text

text = "";

foreach (var item in words) text += item + ' ';

sw.Write(text);

}

}

}

private void button3\_Click(object sender, EventArgs e)

{

int n = 0;

string the\_text = "";

HashSet<Dictionary<char, int>> vowel\_string\_set = new HashSet<Dictionary<char, int>>();

try

{

n = Convert.ToInt32(textBox1.Text);

if (openFileDialog1.ShowDialog() == DialogResult.OK)

{

using (StreamReader sr = new StreamReader(openFileDialog1.FileName))

{

string[] text = new string[n];

int i = 0;

string s;

double sum = 0;

while ((s = sr.ReadLine()) != null && i < n)

{

sum += s.Length;

text[i] = s;

i++;

}

sum /= (double)n;

string remaining\_text = Environment.NewLine + s;

while ((s = sr.ReadLine()) != null)

{

remaining\_text += s + Environment.NewLine;

}

Dictionary<char, int> vowels = new Dictionary<char, int>();

string init\_text\_string = String.Join("\n", text);

Debug.WriteLine(init\_text\_string);

for (i = 0; i < n; i++)

{

if (text[i].Length <= sum) text[i] = null;

if (text[i] != null)

{

//Saving the old string

string init\_row = text[i];

StringBuilder sb = new StringBuilder(text[i]);

vowels = new Dictionary<char, int> { { 'а', 0 }, { 'е', 0 }, { 'о', 0 }, { 'у', 0 }, { 'ё', 0 }, { 'ы', 0 }, { 'э', 0 }, { 'я', 0 }, { 'и', 0 }, { 'ю', 0 } };

for (int j = 0; j < text[i].Length; j++)

{

if (vowels.ContainsKey(char.ToLower(text[i][j]))) vowels[char.ToLower(text[i][j])]++;

}

foreach (KeyValuePair<char, int> entry in vowels)

{

if (entry.Value > 1)

{

for (int j = 0; j < text[i].Length; j++)

if (char.ToLower(text[i][j]) == entry.Key)

{

sb[j] = char.IsUpper(text[i][j]) ? char.ToLower(text[i][j]) : char.ToUpper(text[i][j]);

text[i] = sb.ToString();

}

vowel\_string\_set.Add(vowels);

}

}

//Inverting the string

for (int j = 0; j < (text[i].Length / 2); j++)

{

sb = new StringBuilder(text[i]);

char temp = sb[j];

sb[j] = sb[sb.Length - j - 1];

sb[sb.Length - j - 1] = temp;

text[i] = sb.ToString();

}

init\_text\_string = init\_text\_string.Replace(init\_row, text[i]);

}

}

the\_text = init\_text\_string + remaining\_text;

Debug.WriteLine(the\_text);

}

File.WriteAllLines(openFileDialog1.FileName, the\_text.Split('\n'));

}

if (saveFileDialog1.ShowDialog() == DialogResult.OK)

{

using (StreamWriter sw = File.CreateText(saveFileDialog1.FileName))

{

foreach (var item in vowel\_string\_set)

{

List<char> keys = new List<char>(item.Keys);

List<int> values = new List<int>(item.Values);

int temp = 0;

char temp\_char;

if (Convert.ToInt32(textBox2.Text) == 1)

{

// Sort by keys

for (int i = 0; i < values.Count; i++)

{

for (int j = 0; j < values.Count - 1; j++)

{

if (values[j] > values[j + 1])

{

temp = values[j + 1];

values[j + 1] = values[j];

values[j] = temp;

temp\_char = keys[j + 1];

keys[j + 1] = keys[j];

keys[j] = temp\_char;

}

}

}

}

else // Sort by values

{

for (int i = 0; i < values.Count; i++)

{

for (int j = 0; j < values.Count - 1; j++)

{

if (keys[j] > keys[j + 1])

{

temp = values[j + 1];

values[j + 1] = values[j];

values[j] = temp;

temp\_char = keys[j + 1];

keys[j + 1] = keys[j];

keys[j] = temp\_char;

}

}

}

}

for (int i = 0; i < keys.Count; i++)

{

sw.Write("" + keys[i] + " = " + values[i] + " ");

}

sw.Write(Environment.NewLine);

}

}

}

}

catch(Exception ex)

{

label2.ForeColor = Color.Red;

label2.Text = ex.Message;

timer1.Start();

}

}

private void timer1\_Tick(object sender, EventArgs e)

{

label2.ForeColor = Color.White;

label2.Text = "<- number of rows to read";

timer1.Stop();

}

private void label2\_Click(object sender, EventArgs e)

{

}

}

}

Program.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Lab4Forms

{

static class Program

{

/// <summary>

/// The main entry point for the application.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

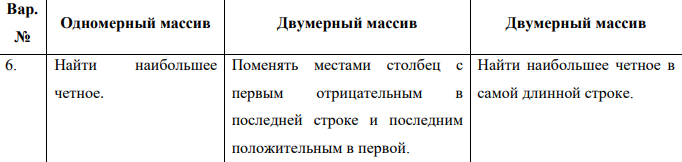
Application.Run(new Form1());

}

}

}

Задание 5:

Написать класс, имеющий защищенное поле одномерный массив, а также метод его обработки (см. вариант). Разработать класс, наследник от первого класса, который имеет закрытое поле двумерный массив и метод его обработки (см. вариант). Массивы сгенерировать рандомно в конструкторе. В основной программе запросить у пользователя размер массива и передать это число в конструктор класса. У базового класса разработать метод записи элементов массива в строку и возврат этой строки для вывода на экран. Отдельные строки двумерного массива рассматривать как одномерные массивы и обрабатывать их методами базового класса. Продемонстрировать работу всех методов. 

Код:

Program.cs:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab5

{

class Program

{

static void Main(string[] args)

{

int i = 0;

while (true)

{

Console.Clear();

switch (i)

{

case 0:

{

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("----------------------------------");

Console.WriteLine("| Task 1 |");

Console.WriteLine("----------------------------------");

Console.ForegroundColor = ConsoleColor.Gray;

Console.WriteLine("----------------------------------");

Console.WriteLine("| Task 2 |");

Console.WriteLine("----------------------------------");

Console.WriteLine("----------------------------------");

Console.WriteLine("| Task 3 |");

Console.WriteLine("----------------------------------");

Console.WriteLine("----------------------------------");

Console.WriteLine("| Exit |");

Console.WriteLine("----------------------------------");

}

break;

case 1:

{

Console.ForegroundColor = ConsoleColor.Gray;

Console.WriteLine("----------------------------------");

Console.WriteLine("| Task 1 |");

Console.WriteLine("----------------------------------");

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("----------------------------------");

Console.WriteLine("| Task 2 |");

Console.WriteLine("----------------------------------");

Console.ForegroundColor = ConsoleColor.Gray;

Console.WriteLine("----------------------------------");

Console.WriteLine("| Task 3 |");

Console.WriteLine("----------------------------------");

Console.WriteLine("----------------------------------");

Console.WriteLine("| Exit |");

Console.WriteLine("----------------------------------");

}

break;

case 2:

{

Console.ForegroundColor = ConsoleColor.Gray;

Console.WriteLine("----------------------------------");

Console.WriteLine("| Task 1 |");

Console.WriteLine("----------------------------------");

Console.WriteLine("----------------------------------");

Console.WriteLine("| Task 2 |");

Console.WriteLine("----------------------------------");

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("----------------------------------");

Console.WriteLine("| Task 3 |");

Console.WriteLine("----------------------------------");

Console.ForegroundColor = ConsoleColor.Gray;

Console.WriteLine("----------------------------------");

Console.WriteLine("| Exit |");

Console.WriteLine("----------------------------------");

}

break;

case 3:

{

Console.ForegroundColor = ConsoleColor.Gray;

Console.WriteLine("----------------------------------");

Console.WriteLine("| Task 1 |");

Console.WriteLine("----------------------------------");

Console.WriteLine("----------------------------------");

Console.WriteLine("| Task 2 |");

Console.WriteLine("----------------------------------");

Console.WriteLine("----------------------------------");

Console.WriteLine("| Task 3 |");

Console.WriteLine("----------------------------------");

Console.ForegroundColor = ConsoleColor.Red;

Console.WriteLine("----------------------------------");

Console.WriteLine("| Exit |");

Console.WriteLine("----------------------------------");

Console.ForegroundColor = ConsoleColor.Gray;

}

break;

}

ConsoleKeyInfo d = Console.ReadKey();

if (d.KeyChar == 'w') if (i - 1 < 0) i = 3; else i--;

if (d.KeyChar == 's') if (i + 1 > 3) i = 0; else i++;

if (d.Key == ConsoleKey.Enter)

{

bool anyExceptions = true;

switch (i)

{

case 0:

{

while (anyExceptions)

{

Console.Clear();

Console.WriteLine("Enter array size:");

try

{

int n = Convert.ToInt32(Console.ReadLine());

Array1D array = new Array1D(n);

Console.WriteLine("\nArray:\n" + array.ToString());

array.solve1();

Console.WriteLine("\nPress any key to return...");

Console.ReadKey();

anyExceptions = false;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message + "\nPress any key to retry...");

Console.ReadKey();

}

}

}

break;

case 1:

{

while (anyExceptions)

{

Console.Clear();

Console.WriteLine("Enter array dimensions:");

try

{

int n = Convert.ToInt32(Console.ReadLine());

int m = Convert.ToInt32(Console.ReadLine());

Array2D array = new Array2D(n, m, true);

Console.WriteLine("\nArray:\n" + array.ToString());

//array.solve2();

Console.WriteLine("\nSolved array:\n" + array.ToString());

Console.WriteLine("\nPress any key to return...");

Console.ReadKey();

anyExceptions = false;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message + "\nPress any key to retry...");

Console.ReadKey();

}

}

}

break;

case 2:

{

while (anyExceptions)

{

Console.Clear();

Console.WriteLine("Enter array dimensions:");

try

{

int n = Convert.ToInt32(Console.ReadLine());

int m = Convert.ToInt32(Console.ReadLine());

Array2D array = new Array2D(n, m, false);

Console.WriteLine("\nArray:\n" + array.ToString());

array.solve3();

Console.WriteLine("\nPress any key to return...");

Console.ReadKey();

anyExceptions = false;

}

catch (Exception ex)

{

Console.WriteLine(ex.Message + "\nPress any key to retry...");

Console.ReadKey();

}

}

}

break;

case 3:

Environment.Exit(0);

break;

}

}

if (d.Key == ConsoleKey.Escape) Environment.Exit(0);

}

}

}

}

Array1D.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab5

{

class Array1D

{

protected int[] a;

public int length;

public Array1D(int n)

{

this.a = new int[n];

Random rand = new Random();

for (int i = 0; i < n; i++) a[i] = rand.Next(-10, 11);

this.length = n;

}

public void solve1()

{

int max\_even = int.MinValue;

for (int i = 0; i < a.Length; i++) if (a[i] % 2 == 0 && a[i] > max\_even) max\_even = a[i];

if (max\_even != int.MinValue) Console.WriteLine("Max even: " + max\_even);

else Console.WriteLine("No even numbers found.");

}

public override string ToString()

{

return String.Join(" ", a);

}

public void randomize()

{

Random rand = new Random();

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

{

int j = rand.Next(0, this.length);

int temp = a[i];

a[i] = a[j];

a[j] = temp;

}

}

}

}

Array2D.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab5

{

class Array2D : Array1D

{

int[][] array;

public Array2D(int n, int m, bool evenRows) : base (n)

{

this.array = new int[n][];

Random rand = new Random();

if (evenRows)

{

for (int i = 0; i < n; i++) array[i] = new int[m];

}

else

{

for (int i = 0; i < n; i++) array[i] = new int[rand.Next(1, m + 1)];

}

for (int i = 0; i < n; i++)

{

for (int j = 0; j < array[i].Length; j++) array[i][j] = rand.Next(-10, 11);

}

}

public void solve2()

{

int first\_neg\_i = -1;

int last\_pos\_i = -1;

int i = 0;

while (first\_neg\_i == -1 && i < array[a.Length - 1].Length)

{

if (array[a.Length - 1][i] < 0) first\_neg\_i = i;

i++;

}

i = array[0].Length - 1;

while (last\_pos\_i == -1 && i >= 0)

{

if (array[0][i] > 0) last\_pos\_i = i;

i--;

}

if (first\_neg\_i == -1 || last\_pos\_i == -1)

{

Console.WriteLine("Could not find one of the two numbers.");

return;

}

i = 0;

for (i = 0; i < a.Length; i++)

{

int temp = array[i][first\_neg\_i];

array[i][first\_neg\_i] = array[i][last\_pos\_i];

array[i][last\_pos\_i] = temp;

}

}

public void solve3()

{

int max\_len\_i = 0;

int max\_even = int.MinValue;

for (int i = 0; i < a.Length; i++) if (array[i].Length > array[max\_len\_i].Length) max\_len\_i = i;

a = array[max\_len\_i];

solve1();

if (max\_even == int.MinValue) Console.WriteLine("No even numbers found in row {0}.", max\_len\_i);

else Console.WriteLine("Max even in row {0}: {1}", max\_len\_i, max\_even);

}

public override string ToString()

{

string s = "";

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

{

for (int j = 0; j < array[i].Length; j++) s += String.Format("{0, 4}", array[i][j]);

s += "\n";

}

return s;

}

}

}

Задание 6:

Создать объектную модель предметной области, содержащую 2 класса. Один класс – объект по варианту с 3мя полями. Поля должны быть как со строковыми значениями, так и с числовыми, и должны однозначно идентифицировать объект (т.е. чтобы по полям было понятно, что это стол, а не человек и не собака). Второй класс – список объектов. Реализовать хранение в памяти при помощи коллекций (List, ArrayList). Функции системы: 0) Отображение списка объектов на форме. 1) добавление и удаление элементов в коллекцию при помощи элементов формы и обработка возможных ошибок. 2) вывод информации о всех полях выбранного объекта и редактирование его. 3) поиск и фильтрация элементов внутри коллекции по любому полю на выбор пользователя.

Предметная область: студент.

Код:

Student.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab6

{

class Student

{

public int age;

public string name;

public double median;

public Student(int age, string name, double median)

{

this.age = age;

this.name = name;

this.median = median;

}

public override string ToString()

{

return String.Format("{0}, {1} y.o., median: {2}", name, age, median);

}

}

}

StudentGroup.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace Lab6

{

class StudentGroup

{

public List<Student> students = new List<Student>();

public List<Student> StreamSort(int filter)

{

switch(filter)

{

case 1:

return students.OrderBy(student => student.age).ToList();

case 2:

return students.OrderBy(student => student.name).ToList();

case 3:

return students.OrderBy(student => student.median).ToList();

default:

return students;

}

}

public List<Student> StreamFilter(int filter, string text)

{

switch (filter)

{

case 1:

return students.Where(student => student.name.StartsWith(text)).ToList();

case 2:

return students.Where(student => student.age > Convert.ToInt32(text)).ToList();

case 3:

return students.Where(student => student.median > Convert.ToDouble(text)).ToList();

default:

return students;

}

}

}

}

Form1.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Diagnostics;

namespace Lab6

{

public partial class Form1 : Form

{

StudentGroup pibd13;

public Form1()

{

InitializeComponent();

pibd13 = new StudentGroup();

comboBox1.Items.Add("Name");

comboBox1.Items.Add("Age");

comboBox1.Items.Add("Median");

comboBox2.Items.Add("Name");

comboBox2.Items.Add("Age");

comboBox2.Items.Add("Median");

}

private void button1\_Click(object sender, EventArgs e)

{

try

{

pibd13.students.Add(new Student(Convert.ToInt32(textBox2.Text), textBox1.Text, Convert.ToDouble(textBox3.Text)));

listBox1.Items.Clear();

listBox1.Items.AddRange(pibd13.students.ToArray());

}

catch (Exception ex)

{

label4.Text = ex.Message;

}

}

private void button2\_Click(object sender, EventArgs e)

{

Debug.WriteLine(comboBox1.SelectedItem.ToString());

switch(comboBox1.SelectedItem.ToString())

{

case "Name":

{

listBox1.Items.Clear();

listBox1.Items.AddRange(pibd13.StreamSort(2).ToArray());

}

break;

case "Age":

{

listBox1.Items.Clear();

listBox1.Items.AddRange(pibd13.StreamSort(1).ToArray());

}

break;

case "Median":

{

listBox1.Items.Clear();

listBox1.Items.AddRange(pibd13.StreamSort(3).ToArray());

}

break;

}

}

private void button3\_Click(object sender, EventArgs e)

{

switch (comboBox2.SelectedItem.ToString())

{

case "Name":

{

try

{

listBox1.Items.Clear();

listBox1.Items.AddRange(pibd13.StreamFilter(1, textBox4.Text).ToArray());

}

catch (Exception ex)

{

label4.Text = ex.Message;

listBox1.Items.Clear();

listBox1.Items.AddRange(pibd13.students.ToArray());

}

}

break;

case "Age":

{

try

{

listBox1.Items.Clear();

listBox1.Items.AddRange(pibd13.StreamFilter(2, textBox4.Text).ToArray());

}

catch (Exception ex)

{

label4.Text = ex.Message;

listBox1.Items.Clear();

listBox1.Items.AddRange(pibd13.students.ToArray());

}

}

break;

case "Median":

{

try

{

listBox1.Items.Clear();

listBox1.Items.AddRange(pibd13.StreamFilter(3, textBox4.Text).ToArray());

}

catch (Exception ex)

{

label4.Text = ex.Message;

listBox1.Items.Clear();

listBox1.Items.AddRange(pibd13.students.ToArray());

}

}

break;

}

}

private void button4\_Click(object sender, EventArgs e)

{

listBox1.Items.Clear();

listBox1.Items.AddRange(pibd13.students.ToArray());

}

private void Form1\_Load(object sender, EventArgs e)

{

}

}

}

Program.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Lab6

{

static class Program

{

/// <summary>

/// The main entry point for the application.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Form1());

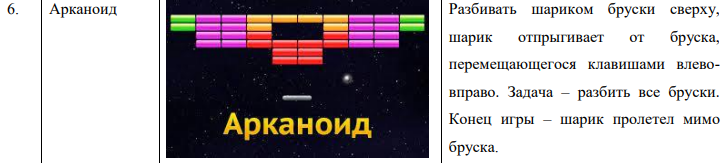
}

}

}

Задание 7:

Реализовать игровой процесс на графических примитивах (кружочки, квадратики) или (если игра это допускает), просто куче кнопок.



Код:

Form1.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Diagnostics;

namespace Game

{

public partial class Form1 : Form

{

const int WIDTH = 1;

const int HEIGHT = 1;

Rectangle[,] blocks;

int player\_x;

int player\_y;

int player\_speed;

int ball\_x;

int ball\_y;

int[] ball\_speed = new int[] { 4, 4 };

int score = 0;

bool gameover = false;

bool game\_win = false;

Bitmap bm;

Graphics g;

Timer movementTimer = new Timer();

public Form1()

{

InitializeComponent();

gameoverlabel.Hide();

win\_label.Hide();

blocks = new Rectangle[WIDTH, HEIGHT];

for (int i = 0; i < WIDTH; i++)

{

for (int j = 0; j < HEIGHT; j++) blocks[i, j] = new Rectangle(i \* (pictureBox1.Width / WIDTH), j \* 20, pictureBox1.Width / WIDTH - 2, 18);

}

player\_x = 200;

player\_y = 700;

ball\_x = 200;

ball\_y = 350;

bm = new Bitmap(pictureBox1.Width, pictureBox1.Height);

g = Graphics.FromImage(bm);

this.KeyDown += new KeyEventHandler(processInputDown);

this.KeyUp += new KeyEventHandler(processInputUp);

timer1.Interval = 10;

timer1.Tick += new EventHandler(timerUpdate);

timer1.Enabled = true;

movementTimer.Interval = 10;

movementTimer.Tick += new EventHandler(movementTick);

}

private void DrawEverything()

{

bm = new Bitmap(pictureBox1.Width, pictureBox1.Height);

g = Graphics.FromImage(bm);

for (int i = 0; i < WIDTH; i++)

{

for (int j = 0; j < HEIGHT; j++) g.FillRectangle(Brushes.Gray, blocks[i, j]);

}

g.FillRectangle(Brushes.Black, new Rectangle(player\_x, player\_y, 100, 20));

g.FillEllipse(Brushes.Black, new Rectangle(ball\_x, ball\_y, 20, 20));

pictureBox1.Image = bm;

scorelabel.Text = "Score: " + score;

}

private void Form1\_Load(object sender, EventArgs e)

{

}

private void checkCollisions()

{

//Walls

if (ball\_x + ball\_speed[0] + 20 > pictureBox1.Width || ball\_x + ball\_speed[0] < 0) ball\_speed[0] \*= -1;

if (ball\_y + ball\_speed[1] < 0) ball\_speed[1] \*= -1;

if (ball\_y + ball\_speed[1] > 740) gameover = true;

//Player

if ((player\_y - ball\_y) < 20 && ball\_x + 10 < player\_x + 100 && ball\_x + 10 > player\_x) ball\_speed[1] \*= -1;

//Blocks

for (int i = 0; i < WIDTH; i++)

{

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

{

if (blocks[i, j] != null && ball\_x + 10 > blocks[i, j].X && ball\_x + 10 < blocks[i, j].X + blocks[i, j].Width && Math.Abs(ball\_y - blocks[i, j].Y) < 10)

{

blocks[i, j] = Rectangle.Empty;

ball\_speed[1] \*= -1;

score += 50;

}

}

}

ball\_x += ball\_speed[0];

ball\_y += ball\_speed[1];

}

private void checkWin()

{

for (int i = 0; i < WIDTH; i++)

{

for (int j = 0; j < HEIGHT; j++) if (blocks[i, j] != Rectangle.Empty) return;

}

game\_win = true;

}

private void timerUpdate(object sender, EventArgs e)

{

if (!gameover && !game\_win)

{

checkCollisions();

checkWin();

DrawEverything();

}

else if (gameover)

{

gameoverlabel.Show();

}

else

{

win\_label.Show();

}

}

private void movementTick(object sender, EventArgs e)

{

if (!gameover)

{

if (!(player\_x + player\_speed + 100 > pictureBox1.Width) && !(player\_x + player\_speed < 0))

player\_x += player\_speed;

DrawEverything();

}

}

private void processInputDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.A)

{

player\_speed = -5;

}

if (e.KeyCode == Keys.D)

{

player\_speed = +5;

}

if ((gameover || game\_win) && e.KeyCode == Keys.R)

{

ball\_x = 200;

ball\_y = 350;

score = 0;

for (int i = 0; i < WIDTH; i++)

{

for (int j = 0; j < HEIGHT; j++) blocks[i, j] = new Rectangle(i \* (pictureBox1.Width / WIDTH), j \* 20, pictureBox1.Width / WIDTH - 2, 18);

}

gameover = false;

game\_win = false;

gameoverlabel.Hide();

win\_label.Hide();

ball\_speed = new int[] { 4, 4 };

}

UpdateTimer();

DrawEverything();

}

private void processInputUp(object sender, KeyEventArgs e)

{

player\_speed = 0;

UpdateTimer();

DrawEverything();

}

private void OnPaint(object sender, PaintEventArgs e)

{

DrawEverything();

}

private void UpdateTimer()

{

movementTimer.Enabled = player\_speed != 0;

}

private void gameoverlabel\_Click(object sender, EventArgs e)

{

}

}

}

Program.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Game

{

static class Program

{

/// <summary>

/// The main entry point for the application.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Form1());

}

}

}