

Группа 191-726

Отчет

Основы алгоритмизации и программирования

Лабораторная работа №8

СПИСОК ЗАДАЧ ПО ЯЗЫКУ С#

Выполнил(а): Ремизов А. И.

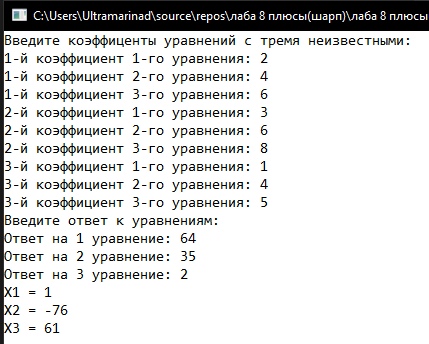
Проверил(а): Новичков И. К.

2019 г.

# Задание № 1

Решить систему уравнений методом Крамера.

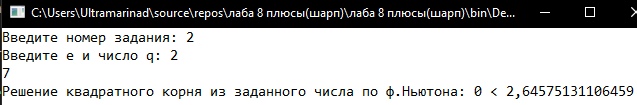
|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Threading;  using System.IO;  using System.Diagnostics;  namespace Код\_Лабораторной\_8\_плюсы\_шарпы  {  class Program  {  static void Main(string[] args)  {  int i, w, b1;  float x1, x2, x3, A, B, C, prop;  int[,] koef = new int[3, 3];  int[] ravno = new int[3];  Console.Write("Введите коэффиценты уравнений с тремя неизвестными: \n");  for (i = 0; i < 3; i++)  {  for (w = 0; w < 3; w++)  {  Console.Write("{0}-й коэффициент {1}-го уравнения: ", i + 1, w + 1);  koef[i, w] = int.Parse(Console.ReadLine());  }  }  Console.Write("Введите ответ к уравнениям: \n");  for (b1 = 0; b1 < 3; b1++)  {  Console.Write("Ответ на {0} уравнение: ", b1 + 1);  ravno[b1] = int.Parse(Console.ReadLine());  }  float full = koef[0, 0] \* koef[1, 1] \* koef[2, 2] + koef[0, 1]  \* koef[1, 2] \* koef[2, 0] + koef[1, 0] \* koef[2, 1] \* koef[0, 2]  - koef[0, 2] \* koef[1, 1] \* koef[2, 0] - koef[0, 1] \* koef[1, 0]  \* koef[2, 2] - koef[0, 0] \* koef[1, 2] \* koef[2, 1];  A = ravno[0] \* koef[1, 1] \* koef[2, 2] + koef[0, 1] \* koef[1, 2] \* ravno[2]  + ravno[1] \* koef[2, 1] \* koef[0, 2] - koef[0, 2] \* koef[1, 1] \* ravno[2]  - koef[0, 1] \* ravno[1] \* koef[2, 2] - ravno[0] \* koef[1, 2] \* koef[2, 1];  B = koef[0, 0] \* ravno[1] \* koef[2, 2] + ravno[0] \* koef[1, 2] \* koef[2, 0]  + koef[1, 0] \* ravno[2] \* koef[0, 2] - koef[0, 2] \* ravno[1] \* koef[2, 0]  - ravno[0] \* koef[1, 0] \* koef[2, 2] - koef[0, 0] \* koef[1, 2] \* ravno[2];  C = koef[0, 0] \* koef[1, 1] \* ravno[2] + koef[0, 1] \* ravno[1] \* koef[2, 0]  + koef[1, 0] \* koef[2, 1] \* ravno[0] - ravno[0] \* koef[1, 1] \* koef[2, 0]  - koef[0, 1] \* koef[1, 0] \* ravno[2] - koef[0, 0] \* ravno[1] \* koef[2, 1];  Console.Write("X1 = ");  x1 = A / full;  Console.Write(x1 + "\n");  Console.Write("X2 = ");  x2 = B / full;  Console.Write(x2 + "\n");  Console.Write("X3 = ");  x3 = C / full;  Console.Write(x3 + "\n");  Console.ReadKey();  }  }  } |



# Задание № 2

Разработать программу для вычисления квадратного корня из заданного числа q с точностью ε по формуле Ньютона: 𝑥𝑛 = 0,5 (𝑥𝑐 + 𝑞 𝑥𝑐 ), где xc и xn – соответственно старое и новое значения искомого корня на текущем шаге вычислений. На 1-м шаге вычислений считать xc = 1. Вычисления закончить при выполнении условия |𝑥𝑛 − 𝑥𝑐 | < 𝜀. Сравнить значение корня, вычисленное по формуле Ньютона, со значением, полученным по формуле x = sqrt(q).

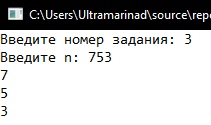
|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Threading;  using System.IO;  using System.Diagnostics;  namespace Код\_Лабораторной\_8\_плюсы\_шарпы  {  class Program  {  static void Main(string[] args)  {  float Xn, Xc = 1, q, e, x;  double ret;  Console.Write("Введите е и число q: ");  e = int.Parse(Console.ReadLine());  q = Convert.ToInt64(Console.ReadLine());  Xn = 0;  while (Math.Abs(Xn - Xc) >= e)  {  Xn = 1 / 2 \* (Xc + q / Xc);  Xc = Xn;  }  ret = q;  if (Xn > ret)  {  Console.Write("Решение квадратного корня из заданного числа по ф.Ньютона: ");  Console.WriteLine(Xn + " > " + Math.Sqrt(ret));  }  else if (Xn < ret)  {  Console.Write("Решение квадратного корня из заданного числа по ф.Ньютона: ");  Console.WriteLine(Xn + " < " + Math.Sqrt(ret));  }  else if (Xn == ret)  {  Console.Write("Решение квадратного корня из заданного числа по ф.Ньютона: ");  Console.WriteLine(Xn + " = " + Math.Sqrt(ret));  }  Console.ReadKey();  }}} |



# Задание № 3

Дано целое число N. Не используя строковые переменные, вывести в столбик это число.

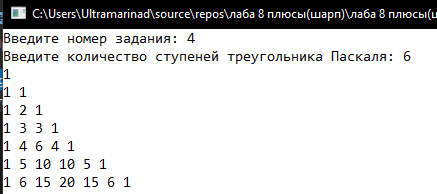
|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Threading;  using System.IO;  using System.Diagnostics;  namespace Код\_Лабораторной\_8\_плюсы\_шарпы  {  class Program  {  static void Main(string[] args)  {  static void Stolbik()  {  Console.Write("Введите n: ");  int n = int.Parse(Console.ReadLine());  int j = 0;  int[] chislo = new int[n];  while (n > 0)  {  j = j + 1;  chislo[j] = n % 10;  n /= 10;  }  for (int i = j; i > 0; i--)  {  Console.WriteLine(chislo[i]);  }  Console.ReadKey(); }  }  } |



# Задание № 4

Используя динамический массив, вывести на экран треугольник Паскаля произвольной длины N<12.

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Threading;  using System.IO;  using System.Diagnostics;  namespace Код\_Лабораторной\_8\_плюсы\_шарпы  {  class Program  {  static void Main(string[] args)  {  Console.Write("Введите количество ступеней треугольника Паскаля: ");  int strochki = int.Parse(Console.ReadLine());  int gran = 2;  int[] upper = new int[1] { 1 };  Console.WriteLine(upper[0]);  int[] lower = new int[2];  while (strochki > 0)  {  for (int i = 0; i < gran; i++)  {  if (i == 0 || i == gran - 1)  {  lower[i] = 1;  }  else { lower[i] = upper[i - 1] + upper[i];  }  Console.Write(lower[i]);  Console.Write(" ");  }  Console.Write("\n");  Array.Resize(ref upper, upper.Length + 1);  for (int i = 0; i < gran; i++)  {  upper[i] = lower[i];  }  Array.Resize(ref lower, lower.Length + 1);  gran++;  strochki--;  }  Console.ReadKey();  } }} |



# Задание № 5

Работа с текстовыми файлами. Программа должна уметь:  
• Записывать в текстовый файл информацию, вводимую с клавиатуры

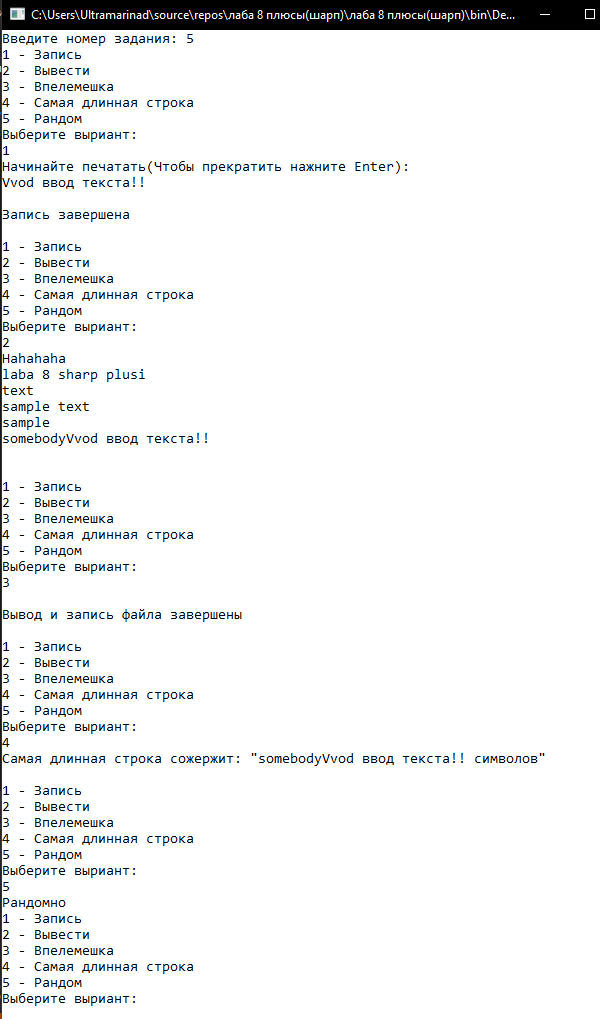
• Считывать на экран информацию из текстового файла

• Организовать построчное считывание файла и запись строк задом наперед

• Найти самую длинную строку в файле. Вывести ее.

• Перемешать строки в файле случайным образом.

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Threading;  using System.IO;  using System.Diagnostics;  namespace Код\_Лабораторной\_8\_плюсы\_шарпы  {  class Program  {  static void Text()  {  int ch = 0;  string choose;  string pathInput = "C:\\Users\\Ultramarinad\\Desktop\\fkfkfkfkfkf.txt";  string pathOutput = "C:\\Users\\Ultramarinad\\Desktop\\fkfkfkfkfkf.txt";  while (ch != 6)  {  do  {  Console.WriteLine("1 - Запись");  Console.WriteLine("2 - Вывести");  Console.WriteLine("3 - Впелемешка");  Console.WriteLine("4 - Самая длинная строка");  Console.WriteLine("5 - Рандом");  Console.WriteLine("Выберите выриант: ");  choose = Console.ReadLine();  } while (choose == "" || !int.TryParse(choose, out ch));  switch (ch)  {  case 1:  Zapis(pathOutput);  break;  case 2:  Vivod(pathInput);  break;  case 3:  ZadPer(pathInput);  break;  case 4:  Longest(pathInput);  break;  case 5:  Random(pathInput);  break;  default:  Console.WriteLine("Error");  break;  }  }  }  static void Zapis(string path)  {  string writing;  FileStream file1 = new FileStream(path, FileMode.Append);  StreamWriter sw = new StreamWriter(file1);  Console.WriteLine("Начинайте печатать(Чтобы прекратить нажните Enter):");  do  {  writing = Console.ReadLine();  sw.WriteLine(writing);  } while (writing != "");  sw.Close();  Console.WriteLine("Запись завершена\n");  }  static void Vivod(string path)  {  string writing;  FileStream file1 = new FileStream(path, FileMode.Open);  StreamReader sr = new StreamReader(file1);  writing = sr.ReadToEnd();  Console.WriteLine(writing);  sr.Close();  }  static void ZadPer(string path)  {  string writing;  int count = 0;  FileStream file1 = new FileStream(path, FileMode.Open);  StreamReader strr = new StreamReader(file1);  do  {  writing = strr.ReadLine();  count++;  } while (strr.Peek() >= 0);  strr.Close();  file1.Close();  string[] text = new string[count];  file1 = new FileStream(path, FileMode.Open);  strr = new StreamReader(file1);  for (int i = 0; i < count; i++)  {  text[i] = strr.ReadLine();  }  strr.Close();  file1 = new FileStream(path, FileMode.Truncate);  StreamWriter sw = new StreamWriter(file1);  for (int i = count - 1; i >= 0; i--)  {  sw.WriteLine(text[i]);  }  sw.Close();  Console.WriteLine("\nВывод и запись файла завершены\n");  }  static void Longest(string path)  {  string longest = "", str;  FileStream file1 = new FileStream(path, FileMode.Open);  StreamReader sr = new StreamReader(file1);  do  {  str = sr.ReadLine();  longest = str.Length > longest.Length ? str : longest;  } while (sr.Peek() >= 0);  Console.WriteLine("Самая длинная строка сожержит: " + "\"" + longest + " символов" + "\"\n");  sr.Close();  }  static void Random(string path)  {  Random rand = new Random();  int count = 0;  FileStream file1 = new FileStream(path, FileMode.Open);  StreamReader sr = new StreamReader(file1);  do  {  sr.ReadLine();  count++;  } while (sr.Peek() >= 0);  sr.Close();  file1.Close();  string[] text = new string[count];  file1 = new FileStream(path, FileMode.Open);  sr = new StreamReader(file1);  for (int i = 0; i < count; i++)  {  text[i] = sr.ReadLine();  }  sr.Close();  file1.Close();  List<int> rnd = new List<int>();  int number;  for (int i = 0; i < count; i++)  {  do  {  number = rand.Next(0, count);  } while (rnd.Contains(number));  rnd.Add(number);  }  file1 = new FileStream(path, FileMode.Truncate);  StreamWriter sw = new StreamWriter(file1);  for (int i = 0; i < count; i++)  {  sw.WriteLine(text[rnd[i]]);  }  Console.WriteLine("Рандомно");  sw.Close();  } }  } |



# Задача № 6

Алгоритмы сортировка. Реализовать приведенные ниже алгоритмы в одной программе на следующих массивах данных: 1 тыс., 10 тыс., 100 тыс., 1 млн., 10 млн. Посчитать количество итераций и затраченное время, вывести эти данные о производительности и сравнить между собой.

• Сортировка Пузырьком

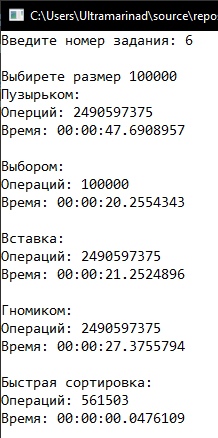
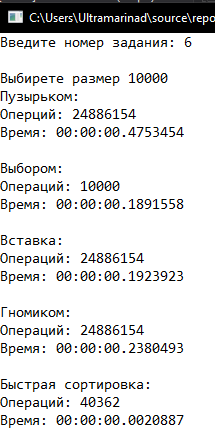
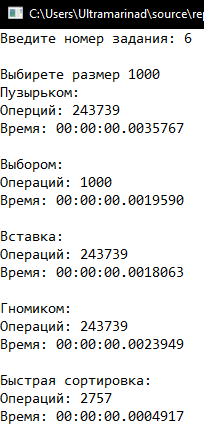
• Сортировка Выбором

• Сортировка Вставками

• Сортировка Гномья

• Сортировка Быстрая

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  using System.Threading;  using System.IO;  using System.Diagnostics;  namespace Код\_Лабораторной\_8\_плюсы\_шарпы  {  class Program  {  static UInt64 countIterationsQsort = 0;  static void Sortirovka()  {  long size = 0;  string size\_s;  do  {  Console.Write("\nВыбирете размер ");  size\_s = Console.ReadLine();  } while (size\_s == "" || !int.TryParse(size\_s, out \_));  size = Math.Abs(Convert.ToInt64(size\_s));  int[] arr = CreateArray(size), arrt = new int[size];  Array.Copy(arr, arrt, size);  Bubble(arrt, size);  Array.Copy(arr, arrt, size);  Choose(arrt, size);  Array.Copy(arr, arrt, size);  Paste(arrt, size);  Array.Copy(arr, arrt, size);  Gnome(arrt, size);  Array.Copy(arr, arrt, size);  Stopwatch time = new Stopwatch();  time.Start();  Qsort(arrt, 0, size - 1);  time.Stop();  Console.WriteLine("Быстрая сортировка: ");  Console.WriteLine("Операций: " + countIterationsQsort);  Console.WriteLine("Время: " + time.Elapsed + "\n");  Console.ReadKey();  }  static int[] CreateArray(long size)  {  Random rand = new Random();  int[] arr = new int[size];  for (int i = 0; i < size; i++)  {  arr[i] = rand.Next(500);  }  return arr;  }  static void Bubble(int[] arr, long size)  {  Stopwatch time = new Stopwatch();  time.Start();  UInt64 countIterations = 0;  for (int i = 0; i < size - 1; i++)  {  for (int j = (int)size - 2; j >= i; j--)  {  if (arr[j] > arr[j + 1])  {  int temp = arr[j + 1];  arr[j + 1] = arr[j];  arr[j] = temp;  countIterations++;  }  }  }  time.Stop();  Console.WriteLine("Пузырьком: ");  Console.WriteLine("Оперций: " + countIterations);  Console.WriteLine("Время: " + time.Elapsed + "\n");  }  static void Choose(int[] arr, long size)  {  Stopwatch time = new Stopwatch();  time.Start();  UInt64 countIterations = 0;  for (int i = (int)size - 1; i >= 0; i--)  {  int max = 0;  for (int j = 0; j <= i; j++)  {  max = arr[j] > arr[max] ? j : max;  }  int temp = arr[max];  arr[max] = arr[i];  arr[i] = temp;  countIterations++;  }  time.Stop();  Console.WriteLine("Выбором: ");  Console.WriteLine("Операций: " + countIterations);  Console.WriteLine("Время: " + time.Elapsed + "\n");  }  static void Paste(int[] arr, long size)  {  Stopwatch time = new Stopwatch();  time.Start();  UInt64 countIterations = 0;  for (int i = 0; i < size; i++)  {  for (int j = i; j > 0 && arr[j] < arr[j - 1]; j--)  {  int temp = arr[j];  arr[j] = arr[j - 1];  arr[j - 1] = temp;  countIterations++;  }  }  time.Stop();  Console.WriteLine("Вставка: ");  Console.WriteLine("Операций: " + countIterations);  Console.WriteLine("Время: " + time.Elapsed + "\n");  }  static void Gnome(int[] arr, long size)  {  Stopwatch time = new Stopwatch();  time.Start();  UInt64 countIterations = 0;  int last = -1;  for (int i = 1; i < size;)  {  if (arr[i] < arr[i - 1])  {  if (last == -1) last = i;  int temp = arr[i];  arr[i] = arr[i - 1];  arr[i - 1] = temp;  countIterations++;  if (i > 1)  {  i--;  }  else  {  i = last + 1;  last = -1;  }  }  else  {  if (last != -1)  {  i = last + 1;  last = -1;  }  else i++;  }  }  time.Stop();  Console.WriteLine("Гномиком: ");  Console.WriteLine("Операций: " + countIterations);  Console.WriteLine("Время: " + time.Elapsed + "\n");  }  static void Qsort(int[] arr, long a, long b)  {  long x = a, y = b;  int piv = arr[(x + y) / 2];  while (x <= y)  {  while (arr[x] < piv) x++;  while (arr[y] > piv) y--;  if (x <= y)  {  int temp = arr[x];  arr[x] = arr[y];  arr[y] = temp;  x++; y--;  countIterationsQsort++;  }  }  if (a < y) Qsort(arr, a, y);  if (b > x) Qsort(arr, x, b);  }  }  } |



6