Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное

учреждение высшего образования

ИРКУТСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ

ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Институт информационных технологий и анализа данных

Факультет Информационные системы и технологии

Отчёт по лабораторной работе № 1

по дисциплине Технология программирования

Выполнил студент ИСТб-20-3 М. Д. Ярков 15.09.2021

номер группы подпись И. О. Фамилия Дата

Принял З. А. Бахвалова . .2021

должность подпись И. О. Фамилия Дата

Иркутск 2021

### Постановка задачи

**Задача на If** – вариант 12**.** Дано натуральное число 1 ≤ n ≤ 9999, определяющее стоимость товара в копейках. Выразить стоимость в рублях и копейках, например, 3 рубля 21 копейка, 15 рублей 5 копеек, 1 рубль ровно и т. п.

### Словесное описание алгоритма

1. Создаём числовую переменную n для вводимого числа, добавляем в неё полученное от пользователя значение. Число получаем с помощью метода Custom.ReadInt("число n", 1, 9999):
   1. Методу были переданы значения переменным: string name = "число n", int min = 1, int max = 9999
   2. Создаём переменную для вывода ошибки при необходимости isInputGot, даём ей значение false – оно обозначает, что пользователь ещё ничего не вводил
   3. Создаём переменную для хранения введённого числа output, даём ей значение unused (константа, введённая для удобства в классе Custom) – оно обозначает, что пользователь ещё ничего не вводил
   4. Запускаем цикл do..while:
      1. Если в isInputGot уже true, выводим сообщение об ошибке
      2. Проверяем значения переменных min и max, которые были переданы методу на шаге 1.1. В зависимости от них выводим соответствующее сообщение, предлагающее пользователю ввести число
      3. Получаем введённую строку, помещаем её в новую переменную input
      4. Задаём переменной isInputGot значение true
      5. Если введённая строка содержит только число, добавляем его в output
   5. Выходим из цикла, если output изменён и входит в заданный промежуток значений
   6. Выводим output
2. Выводим строку, получаемую от функции GetMoney(n):
   1. Методу было передано значение переменной n, название осталось идентичным
   2. Инициализируем строковую переменную output со значением "", числовые переменные rubs со значением (n / 100) и kops со значением (n % 100)
   3. В случае наличия в переменной rubs числа больше нуля, выводим количество рублей:
      1. Добавляем к output значение rubs
      2. Если rubs в промежутке от 5 до 20 добавляем к output " рублей "
      3. Если rubs имеет на конце 1 добавляем к output " рубль "
      4. Если rubs имеет на конце 2, 3 или 4 добавляем к output " рубля "
      5. В остальных случаях добавляем к output " рублей "
   4. В случае наличия в переменной kops нуля добавляем к output "ровно", иначе:
      1. Добавляем к output значение kops
      2. Если kops в промежутке от 5 до 20 добавляем к output " копеек"
      3. Если kops имеет на конце 1 добавляем к output " копейка"
      4. Если kops имеет на конце 2, 3 или 4 добавляем к output " копейки"
      5. В остальных случаях добавляем к output " копеек"
   5. Выводим output
3. Вызываем Custom.Exit() для ожидания закрытия программы пользователем:
   1. Пишем в консоль «Для выхода из программы нажмите любую клавишу...»
   2. Ждём нажатия клавиши

### Таблица спецификаций

|  |  |  |
| --- | --- | --- |
| Имя | Назначение | Тип/Диапазон |
| Входные величины | | |
| n | Стоимость товара в копейках | Целое число от 1 до 9999 |

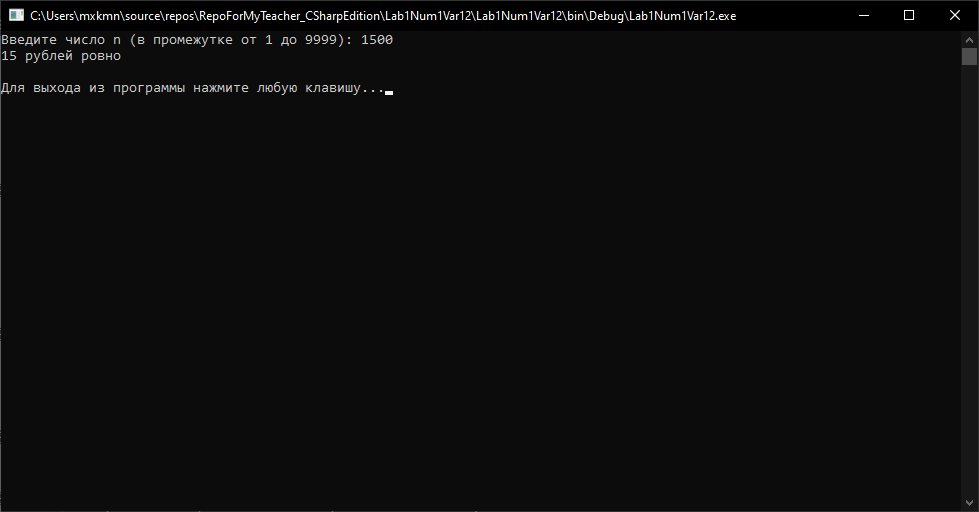
### Таблица тестов

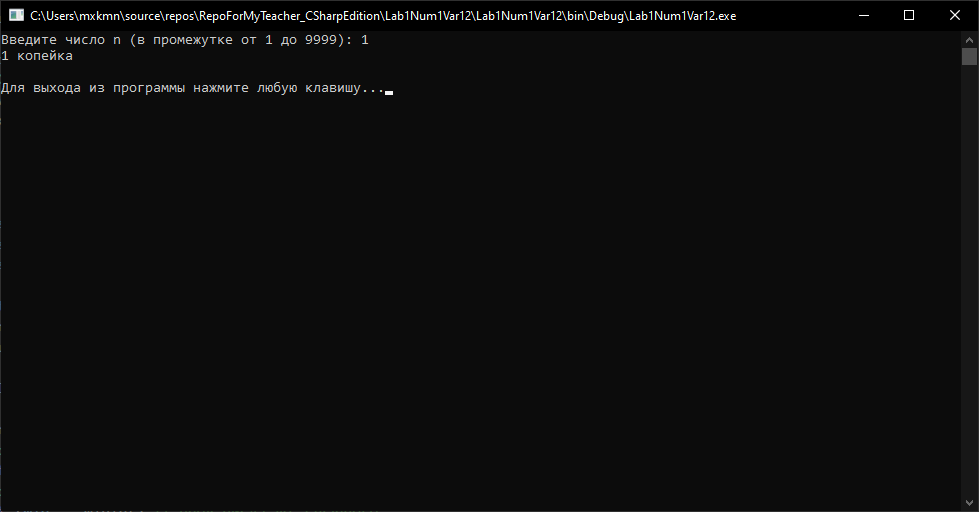
| Номер теста | Что проверяем | Входные данные | Выходные данные |
| --- | --- | --- | --- |
| 1 | Расчёт 1500 копеек (полная проверка программы) | 1500 | 15 рублей ровно  Для выхода из программы нажмите любую клавишу... |
| 2 | Расчёт 1 копейки (полная проверка программы) | 1 | 1 копейка  Для выхода из программы нажмите любую клавишу... |
| 3 | Расчёт 9999 копеек (полная проверка программы) | 9999 | 99 рублей 99 копеек  Для выхода из программы нажмите любую клавишу... |
| 4 | Проверка ввода числа, не входящего в нужный промежуток | 0 | Ошибка ввода! \*повторный ввод того же числа\* |
| 5 | Проверка ввода числа, не входящего в нужный промежуток | 10000 | Ошибка ввода! \*повторный ввод того же числа\* |
| 6 | Проверка ввода слова | Абв | Ошибка ввода! \*повторный ввод того же числа\* |
| 7 | Проверка ввода нескольких слов/чисел | 12 34 | Ошибка ввода! \*повторный ввод того же числа\* |

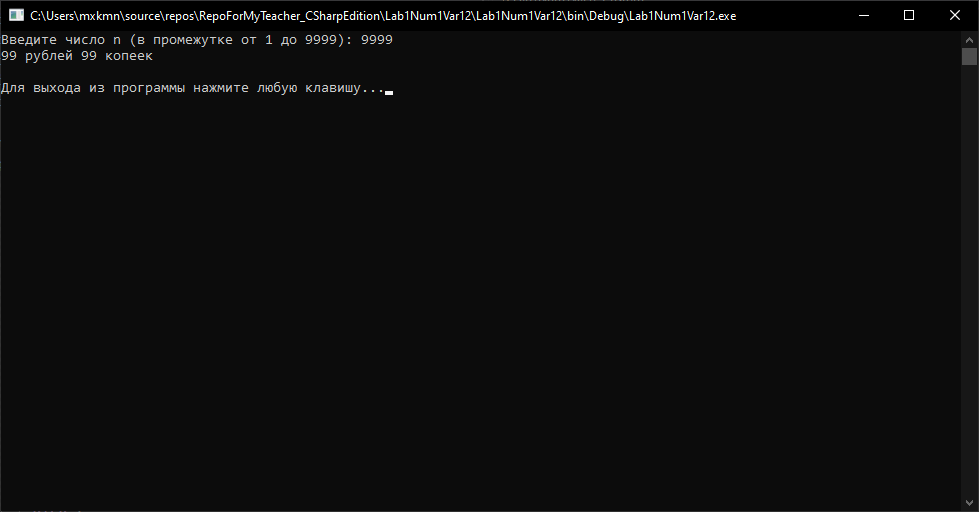
### Код программы

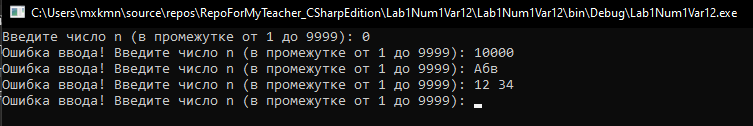
|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Lab1Num1Var12 {  public class Program {  public static string GetMoney(int n) { // отдаёт строку с количеством денег  string output = "";  int rubs = n / 100, kops = n % 100;  if (rubs > 0) { // если есть рубли  output += rubs;  if (5 <= rubs && rubs <= 20)  output += " рублей ";  else if (rubs % 10 == 1)  output += " рубль ";  else if (rubs % 10 == 2 || rubs % 10 == 3 || rubs % 10 == 4)  output += " рубля ";  else  output += " рублей ";  }  if (kops == 0) // если нет копеек  output += "ровно";  else {  output += kops;  if (5 <= kops && kops <= 20)  output += " копеек";  else if (kops % 10 == 1)  output += " копейка";  else if (kops % 10 == 2 || kops % 10 == 3 || kops % 10 == 4)  output += " копейки";  else  output += " копеек";  }  return output;  }  static void Main(string[] args) {  int n = Custom.ReadInt("число n", 1, 9999); // ввод числа  Console.WriteLine(GetMoney(n)); // вывод количества денег  Custom.Exit(); // ожидание закрытия консоли пользователем  }  }  public class Custom {  public const int unused = Int32.MinValue; // константа для неиспользованных переменных  public const int minInt = Int32.MinValue; // константа минимального числа для типа Int  public const int maxInt = Int32.MaxValue; // константа максмального числа для типа Int  public static int ReadInt(string name, int min = minInt, int max = maxInt) {  bool isInputGot = false; // переменная для вывода ошибки при необходимости  int output = unused; // тут будет храниться введённое число  do {  if (isInputGot) Console.Write("Ошибка ввода! "); // вывод ошибки при повторном вводе  if (min == minInt && max == maxInt) // ввод без ОДЗ  Console.Write("Введите " + name + ": ");  else if (min != minInt && max != maxInt) // ввод с ОДЗ  Console.Write("Введите " + name + " (в промежутке от " + min + " до " + max + "): ");  else if (min == minInt) // ввод числа до заданного  Console.Write("Введите " + name + " (до " + max + "): ");  else if (max == maxInt) // ввод числа от заданного  Console.Write("Введите " + name + " (от " + min + "): ");  string input = Console.ReadLine(); // ввод некоторой строки  isInputGot = true; // изменение переменной для вывода ошибки при необходимости  // перевод строки input в числовой output, если строка является числом  if (int.TryParse(input, out int trash)) output = int.Parse(input);  }  while (output == unused || output < min || output > max);  return output;  }  public static void Exit() { // ожидание закрытия консоли пользователем  Console.Write("\nДля выхода из программы нажмите любую клавишу...");  Console.ReadKey(true);  }  }  } |

### Тесты









### Постановка задачи

**Задача на For** – вариант 14**.** Дана последовательность натуральных чисел. Определить, есть ли в последовательности хотя бы одна n-ка одинаковых “соседних” чисел (n и элементы последовательности вводятся с клавиатуры). В случае положительного ответа определить порядковые номера чисел первой из таких пар.

### Словесное описание алгоритма

1. Создаём числовую переменную n для вводимого количества чисел в последовательности, добавляем в неё полученное от пользователя значение. Число получаем с помощью метода Custom.ReadInt("количество натуральных чисел", 0):
   1. Методу были переданы значения переменным: string name = "количество натуральных чисел", int min = 0, int max = maxInt
   2. Создаём переменную для вывода ошибки при необходимости isInputGot, даём ей значение false – оно обозначает, что пользователь ещё ничего не вводил
   3. Создаём переменную для хранения введённого числа output, даём ей значение unused (константа, введённая для удобства в классе Custom) – оно обозначает, что пользователь ещё ничего не вводил
   4. Запускаем цикл do..while:
      1. Если в isInputGot уже true, выводим сообщение об ошибке
      2. Проверяем значения переменных min и max, которые были переданы методу на шаге 1.1. В зависимости от них выводим соответствующее сообщение, предлагающее пользователю ввести число
      3. Получаем введённую строку, помещаем её в новую переменную input
      4. Задаём переменной isInputGot значение true
      5. Если введённая строка содержит только число, добавляем его в output
   5. Выходим из цикла, если output изменён и входит в заданный промежуток значений
   6. Выводим output
2. Создаём числовую переменную nums для вводимого количества чисел, проверяемых на соседство, заполняем его через Custom.ReadInt("сколько чисел проверять на соседство"
3. Пишем в консоль значение, получаемое из GetN(n, nums):
   1. Методу были переданы n и nums, названия переменных внутри метода идентично
   2. Инициализируем переменные lastNum для последнего числа и sameNums для их количества, задаём значение Custom.Unused обоим
   3. Запускаем цикл, который сработает n раз для перебора чисел, получаемых от пользователя:
      1. В новую переменную newNum добавляем натуральное число с помощью Custom.ReadInt(i + "-ое число", 1)
      2. Проверяем, является ли новое число отличающимся от предыдущего. Если не является, добавляем к sameNums единицу, иначе устанавливаем lastNum значение newNum и sameNums значение 1
      3. Если sameNums равно nums, то искомая последовательность найдена – выводим соотвествующее сообщение
   4. Если вывода не произошло до этого шага (n-ка не найдена), выводим соотвествующее сообщение
4. Вызываем Custom.Exit() для ожидания закрытия программы пользователем:
   1. Пишем в консоль «Для выхода из программы нажмите любую клавишу…»
   2. Ждём нажатия клавиши

### Таблица спецификаций

|  |  |  |
| --- | --- | --- |
| Имя | Назначение | Тип/Диапазон |
| Входные величины | | |
| n | Количество чисел в последовательности | Целое число от нуля |
| nums | Количество последовательных чисел, которые необходимо проверить на схожесть | Натуральное число |

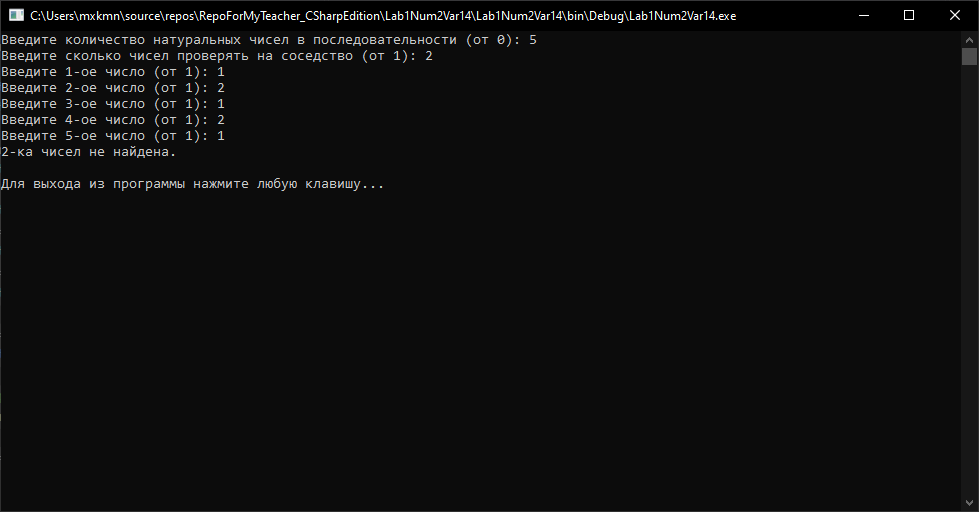
### Таблица тестов

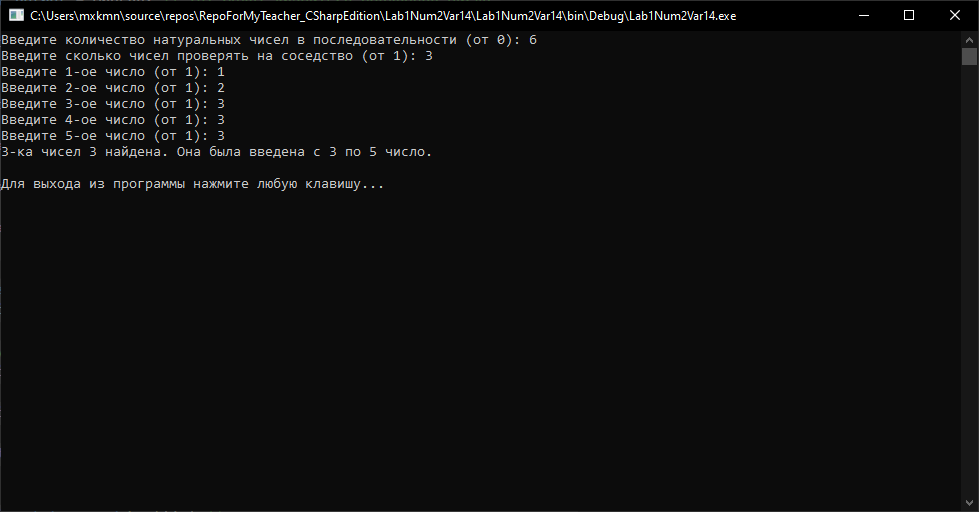
| Номер теста | Что проверяем | Входные данные | Выходные данные |
| --- | --- | --- | --- |
| 1 | Проверка несоседства (полная проверка программы) | 5  2  1 2 1 2 1 | 2-ка чисел не найдена.  Для выхода из программы нажмите любую клавишу... |
| 2 | Проверка соседства (полная проверка программы) | 6  3  1 2 3 3 3 3 | 3-ка чисел 3 найдена. Она была введена с 3 по 5 число.  Для выхода из программы нажмите любую клавишу... |
| 3 | Проверка ввода числа, входящего в нужный промежуток | 5 | \*продолжение ввода\* |
| 4 | Проверка ввода числа, не входящего в нужный промежуток | -5 | Ошибка ввода! \*повторный ввод того же числа\* |
| 5 | Проверка ввода слова | Абв | Ошибка ввода! \*повторный ввод того же числа\* |
| 6 | Проверка ввода нескольких слов/чисел | 12 34 | Ошибка ввода! \*повторный ввод того же числа\* |

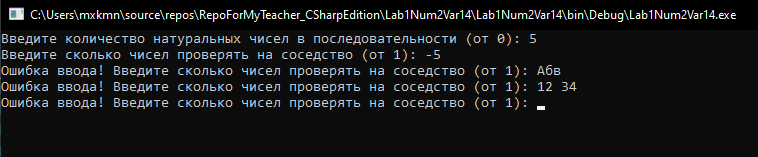
### Код программы

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Lab1Num2Var14 {  public class Program {  public static string GetN(int n, int nums) {  int lastNum = Custom.Unused;  int sameNums = Custom.Unused;  for (int i = 1; i <= n; i++) {  int newNum = Custom.ReadInt(i + "-ое число", 1);  if (lastNum == newNum) // число идентично прошлому  sameNums++;  else { // число не идентично прошлому  lastNum = newNum;  sameNums = 1;  }  if (sameNums == nums) // найдена последовательность из n одинаковых чисел  return (nums + "-ка чисел " + lastNum + " найдена. Она была введена с " + (i - nums + 1) + " по " + i + " число.");  }  return (nums + "-ка чисел не найдена."); // исключение при ненахождении  }  static void Main(string[] args) {  int n = Custom.ReadInt("количество натуральных чисел в последовательности", 0);  int nums = Custom.ReadInt("сколько чисел проверять на соседство", 1);  Console.WriteLine(GetN(n, nums));  Custom.Exit();  }  }  public class Custom {  public const int Unused = Int32.MinValue; // константа для неиспользованных переменных  public const int MinInt = Int32.MinValue; // константа минимального числа для типа Int  public const int MaxInt = Int32.MaxValue; // константа максмального числа для типа Int  public static int ReadInt(string name, int min = MinInt, int max = MaxInt) {  bool isInputGot = false; // переменная для вывода ошибки при необходимости  int output = Unused; // тут будет храниться введённое число  do {  if (isInputGot) // вывод ошибки при повторном вводе  Console.Write("Ошибка ввода! ");  if (min == MinInt && max == MaxInt) // ввод без ОДЗ  Console.Write("Введите " + name + ": ");  else if (min != MinInt && max != MaxInt) // ввод с ОДЗ  Console.Write("Введите " + name + " (в промежутке от " + min + " до " + max + "): ");  else if (min == MinInt) // ввод числа до заданного  Console.Write("Введите " + name + " (до " + max + "): ");  else if (max == MaxInt) // ввод числа от заданного  Console.Write("Введите " + name + " (от " + min + "): ");  string input = Console.ReadLine(); // ввод некоторой строки  isInputGot = true; // изменение переменной для вывода ошибки при необходимости  // перевод строки input в числовой output, если строка является числом  if (int.TryParse(input, out int trash)) output = int.Parse(input);  }  while (output == Unused || output < min || output > max);  return output;  }  public static void Exit() { // ожидание закрытия консоли пользователем  Console.Write("\nДля выхода из программы нажмите любую клавишу...");  Console.ReadKey(true);  }  }  } |

### Тесты







### Постановка задачи

**Задача на строки** – вариант 11**.** Даны два слова. Для каждой буквы первого слова определить, входит ли она во второе слово. Повторяющиеся буквы первого слова не рассматривать. Например, если заданные слова процессор и информация, то для букв первого из них ответом должно быть: нет да да да нет нет.

### Словесное описание алгоритма

1. Создаём строковую переменную firstWord для первого вводимого слова, добавляем в неё полученное от пользователя значение
2. Добавляем в переменную justChars все входящие в firstWord символы с помощью метода GetCharsFromWord(firstWord):
   1. Методу были передано значение firstWord как string word
   2. Создаём переменную для хранения будущего текстового вывода output
   3. Запускаем цикл foreach, который будет перебирать все символы в word: если outWord не содержит символ, то добавляем его в outWord
   4. Выводим outWord
3. Создаём строковую переменную secondWord для второго вводимого слова, добавляем в неё полученное от пользователя значение
4. Запускаем цикл foreach, который будет перебирать все символы в justChars: если secondWord содержит символ, то выводим «да », иначе «нет »
5. Вызываем Custom.Exit() для ожидания закрытия программы пользователем:
   1. Пишем в консоль «Для выхода из программы нажмите любую клавишу…»
   2. Ждём нажатия клавиши

### Таблица спецификаций

|  |  |  |
| --- | --- | --- |
| Имя | Назначение | Тип/Диапазон |
| Входные величины | | |
| firstWord | Первое слово | Строка |
| justChars | Все символы первого слова | Строка |
| secondWord | Второе слово | Строка |

### Таблица тестов

| Номер теста | Что проверяем | Входные данные | Выходные данные |
| --- | --- | --- | --- |
| 1 | Тестовый пример (полная проверка программы) | процессор  информация | нет да да да нет нет  Для выхода из программы нажмите любую клавишу... |
| 2 | Тестовый пример с различным регистром символов (полная проверка программы) | процеСсор  информация | нет да да да нет нет нет  Для выхода из программы нажмите любую клавишу... |
| 3 | Ввод строки с несколькими словами | Проц Проц | \*повторный ввод\* |
| 4 | Пустая строка |  | \*повторный ввод\* |

### Код программы

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Text;  using System.Threading.Tasks;  namespace Lab1Num3Var11 {  public class Program {  public static string GetCharsFromWord(string word) {  string outWord = "";  foreach (char c in word.ToCharArray()) { // перебор всех символов  // добавление символа в строку, если ещё не встречался:  if (!outWord.Contains(c)) outWord += c;  }  return outWord;  }  static void Main(string[] args) {  // получаем первое слово  string firstWord = "";  do {  Console.Write("Введите первое слово: ");  firstWord = Console.ReadLine();  } while (firstWord.Contains(' ') || firstWord == "");  // добываем из слова все входящие в него символы  string justChars = GetCharsFromWord(firstWord);    // получаем второе слово  string secondWord = "";  do {  Console.Write("Введите второе слово: ");  secondWord = Console.ReadLine();  } while (secondWord.Contains(' ') || secondWord == "");  // перебираем символы, ищем во второй строке  foreach (char c in justChars.ToCharArray()) { // перебор всех символов  if (secondWord.Contains(c)) Console.Write("да "); // символ найден  else Console.Write("нет "); // символ не найден  }  Console.WriteLine();  Custom.Exit(); // ждём закрытия окна пользователем  }  }  public class Custom {  public static void Exit() { // ожидание закрытия консоли пользователем  Console.Write("\nДля выхода из программы нажмите любую клавишу...");  Console.ReadKey(true);  }  }  } |

### Тесты

