Министерство образования и науки РФ

Федеральное государственное автономное образовательное учреждение высшего образования

«Омский государственный технический университет»

|  |  |
| --- | --- |
| Факультет (институт) | *Информационных технологий и компьютерных систем* |
|  |  |
| Кафедра | *Прикладная математика и фундаментальная информатика* |
|  |  |

**Расчетно-графическая работа**

|  |  |
| --- | --- |
| по дисциплине | ***Алгоритмизация и программирование*** |
|  |  |
| на тему | Разработка программы |

Пояснительная записка

|  |  |
| --- | --- |
| **Шифр проекта** | 020-РГР-02.03.02-№ 1-ПЗ |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | **Студента** | | Есакова Кирилла Николаевича и Дмитрия Дудикова Евгеньевича | | | | | |
|  |  |  |  | | фамилия, имя, отчество полностью | | | | | |
|  |  |  | Курс | *1* |  | Группа | | ФИТ-232 | | |
|  |  |  |  |  |  | |  |  | |  |
|  | | | **Направление (специальность)** | | | | | ***02.03.02*** | | |
|  | | | *Фундаментальная информатика и информационные технологии* | | | | | | | |
|  |  |  | код, наименование | | | | | | | |
|  |  |  | Руководитель | | ***ст. преподаватель*** | | | | | |
|  |  |  | ученая степень, звание | | | | | |
|  |  |  | ***Федотова И.В.*** | | | | | | | |
|  |  |  | фамилия, инициалы | | | | | | | |
|  |  |  | Выполнил | | 04.01.2024 | | | | | |
|  |  |  | дата, подпись студента | | | | | |
|  |  |  | **Работа защищена с количеством баллов** | | | | | | | |
|  |  |  |  | | | | | |  | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | дата, подпись руководителя |  |  |  |

Омск 2023

Содержание

[**Введение** 3](#_Toc155264781)

[**Теоретическая часть** 4](#_Toc155264782)

[**Разработка кода** 6](#_Toc155264783)

[**Результаты программы** 10](#_Toc155264784)

[**Заключение** 12](#_Toc155264785)

[**Список используемой литературы** 13](#_Toc155264786)

# **Введение**

В ходе работы нам предстоит написать две программы. Первая программа состоит в том, чтобы с помощью датчика случайных чисел выбирается число в диапазоне от 0 до N. Требуется угадать это число с ограничением числа попыток и без ограничения. После каждой попытки сообщается, больше или меньше названное число задуманного.

Вторая программа представляет собой игру в слова. Программа выбирает слово и рисует на экране столько прочерков (прямоугольников), сколько букв в этом слове. Требуется отгадать, какое слово загадано программой. В каждый ход играющий указывает одну букву. Если названа буква, входящая в состав слова, она подставляется вместо соответствующего прочерка (прямоугольника). В противном случае играющий теряет одно очко. В начальный момент у играющего 15 очков.

# **Теоретическая часть**

Для написания программы мы использовали такие как if..else. Условные конструкции — один из базовых компонентов многих языков программирования, которые направляют работу программы по одному из путей в зависимости от определенных условий. Одной из таких конструкций в языке программирования C# является конструкция if..else.

Конструкция if/else проверяет истинность некоторого условия и в зависимости от результатов проверки выполняет определенный код

После ключевого слова if ставится условие. Условие должно представлять значение типа bool. Это может быть непосредственно значение типа bool или результат условного выражения или другого выражения, которое возвращает значение типа bool. И если это условие истинно (равно true), то срабатывает код, который помещен далее после условия внутри фигурных скобок.

Если мы захотим, чтобы при несоблюдении условия также выполнялись какие-либо действия, то этом случае мы можем добавить блок else. Блок else выполняется, если условие после if ложно, то есть равно false. Если блок else содержит толко одну инструкцию, то оять же мы можем его сократить, убрав фигурные скобки.

Также при работе с программой мы использовали цикл while, for, foreach. Циклы являются управляющими конструкциями, позволяя в зависимости от определенных условий выполнять некоторое действие множество раз.

Объявление цикла for состоит из трех частей. Первая часть объявления цикла - некоторые действия, которые выполняются один раз до выполнения цикла. Обычно здесь определяются переменные, которые будут использоваться в цикле. Вторая часть — условие, при котором будет выполняться цикл. Пока условие равно true, будет выполняться цикл. И третья часть - некоторые действия, которые выполняются после завершения блока цикла. Эти действия выполняются каждый раз при завершении блока цикла.

Оператор while выполняет оператор или блок операторов, пока определенное логическое выражение равно значению true. Так как это выражение оценивается перед каждым выполнением цикла, цикл while выполняется ноль или несколько раз.

Цикл foreach предназначен для перебора набора или коллекции элементов. После оператора foreach в скобках сначала идет определение переменной. Затем ключевое слово in и далее коллекция, элементы которой надо перебрать.

При выполнении цикл последовательно перебирает элементы коллекции и помещает их в переменную, и таким образом в блоке цикла мы можем выполнить с ними некоторые действия.

Иногда возникает ситуация, когда требуется выйти из цикла, не дожидаясь его завершения. В этом случае мы можем воспользоваться оператором break.

Помимо условных опреаторов и циклов, мы обращались к классу Random. Этот класс представляет генератор псевдослучайных чисел. То есть, в работе этого класса заложен алгоритм, который генерирует последовательность чисел, отвечающую определенным статистическим критериям случайности.

# **Разработка кода**

В первой программе для начала задаем диапазон чисел, а также выбираем режим игра: с ограничением попыток или без. Если выбран режим с ограничениями, то выбираем, сколько у нас будет попыток. Далее запускаем цикл с условием, пока количество попыток больше нуля. Если пользователь отгадал число, выводим соответствующее сообщение и выходим из цикла. Если нет, то количество попыток уменьшается и в зависимости от того, больше или меньше загаданного введенное пользователем число, дается подсказка. Если количество попыток равно нулю и пользователь не отгадал число, появляется сообщение о проигрыше. Если выбран режим без ограничения попыток, то создаем цикл до тех пор, пока пользователь не отгадает число. Если пользователь отгадал число, выводим соответствующее сообщение и выходим из цикла. Если нет, то в зависимости от того, больше или меньше загаданного введенное пользователем число, дается подсказка.

using System;

class Program

{

static void Main()

{

Console.WriteLine("Введите диапазон от 0 до N");

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

Random rand = new Random();

int x = rand.Next(N);

Console.WriteLine("Играем с ограничеснием попыток? д/н");

char choice = Convert.ToChar(Console.ReadLine());

if (choice == 'д')

{

Console.WriteLine("Сколько попыток будет?");

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

Console.WriteLine($"У вас {attempts} попыток отгадать задуманное число");

while (attempts > 0)

{

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

if (x == number)

{

Console.WriteLine("Вы отгадали число");

break;

}

else

{

attempts--;

if (number > x) Console.WriteLine("Нужно меньше");

if (number < x) Console.WriteLine("Нужно больше");

}

if(x!= number && attempts == 0) Console.WriteLine("Вы проиграли!!!!!!!!!");

}

}

if (choice == 'н')

{

Console.WriteLine("Введите число:");

while (true)

{

int number1 = Convert.ToInt32(Console.ReadLine());

if (x == number1)

{

Console.WriteLine("Вы отгадали число");

break;

}

else

{

if (number1 > x) Console.WriteLine("Нужно меньше");

if (number1 < x) Console.WriteLine("Нужно больше");

}

}

}

}

}

Во второй программе в начале мы создаём переменную для выбора случайного числа в указанном позже диапазоне, сразу после устанавливаем счётчик попыток, а также строку из слов, разделённых пробелами для дальнейшего преобразования этой строки в массив из слов и из него выбора при помощи генератора случайных чисел. Мы также преобразуем выбранное слово в массив с символами для дальнейшего сравнения с вводимым символом пользователя, сразу после создаём и заполняем выводимый на экран массив с условными обозначениями "неизвестной буквы". Также создаём пару вспомогательных булевых переменных, первая нужна для выхода из цикла "while" вторая для изменения счёта(кол-ва попыток), оставшегося у игрока. Заходим в цикл "while", этот цикл действует до тех пор, пока игрок не отгадает слово целиком или же не проиграет, потратив все 15 очков(попыток). Каждую итерацию цикла мы сопровождаем очисткой консоли в начале, вывода слова, счёта и объявления следующей буквы пользователем. Выясняем есть ли буква в слове и меняем массив, который выводим на экран. Выход из цикла сопровождается соответствующим оповещением о победе или поражении для пользователя, выводом счёта и раскрытием угадываемого слова.

using System.Threading.Tasks.Sources;

namespace Game

{

class Program

{

static void Main()

{

Random rnd = new Random();

int score = 15;

string[] words = "я и в к у на не ну мы ты нас вас таз зуб дуб клуб друг враг трак этот целый жулик искра майор назад".Split(' ');

int Num\_of\_Word = rnd.Next(0, words.Length);

char[] word = words[Num\_of\_Word].ToCharArray();

char[] player\_word = new char[words[Num\_of\_Word].Length];

bool yes = false;

for (int i = 0; i < player\_word.Length; i++)

player\_word[i] = '?';

bool lose = true;

char letter = '?';

while (!yes)

{

Console.Clear();

foreach (char c in player\_word)

Console.Write(c);

Console.WriteLine($"\nВаш счёт = {score} \nВаша буква - ");

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

for (int i = 0; i < player\_word.Length; i++)

{

if (letter == word[i])

{

player\_word[i] = word[i];

lose = false;

}

}

if (lose == true)

{

score--;

}

else lose = true;

foreach (char c in player\_word)

{

if (c == '?')

{

yes = false; break;

}

yes = true;

}

if (score == 0)

{

break;

}

}

Console.Clear();

if (score == 0) Console.WriteLine("Вы проиграли");

else Console.WriteLine($"Вы победили со счётом - {score}");

Console.Write("\nВаше слово - ");

foreach (char c in word)

{

Console.Write(c);

}

}

}

}

# **Результаты программы**

Примеры работы первой программы:

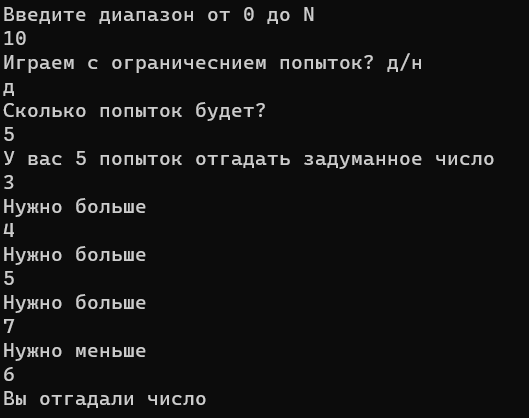


Рисунок 1 – Результат первой программы с ограничением попыток

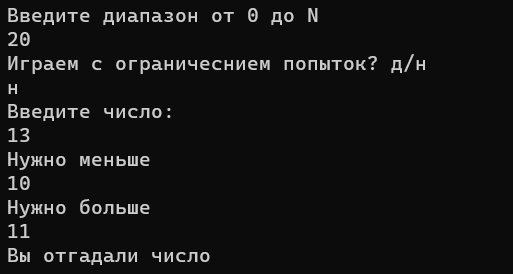


Рисунок 2 – Результат первой программы без ограничения попыток

Примеры работы второй программы:

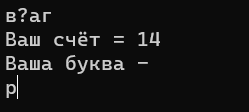


Рисунок 4 – Пример работы программы

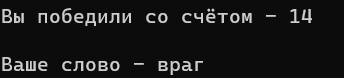


Рисунок 5 – Результат работы второй программы

# **Заключение**

В конечном итоге было разработано две программы. Эти программы помогли закрепить знания, полученные во время изучения языка программирования C#.

# **Список используемой литературы**

1. Metanit, [Язык программирования C# и платформа .NET (metanit.com)](https://metanit.com/sharp/?ysclid=lqytxqnrhe40364687), Дата обращения: 04.01.2024
2. Microsoft Learn, [Документация по C#. Начало работы, руководства, справочные материалы. | Microsoft Learn](https://learn.microsoft.com/ru-ru/dotnet/csharp/), Дата обращения: 04.01.2024