

ЛАБОРАТОРНАЯ РАБОТА №4

Курс «Основы программирования»

Тема: Работа со строками в C#. Знакомство с регулярными выражениями.

Цель: Научиться работать с типом string на языке C#, а также со строкой как с массивом символов; научиться писать простейшие регулярные выражения для поиска подстроки в строке.

Ход работы

Вариант 8

1. Ввести с клавиатуры текст предложения, завершить точкой. Вывести на консоль все символы, которые входят в этот текст ровно по одному разу.

Решить задачу 2 способами: через обработку строки как массива символов и с помощью методов класса string.

1 способ:

```
1  using System;
2
3  0 references
4  class Program
5  {
6      0 references
7      static void Main()
8      {
9          Console.WriteLine("Введите текст, завершённый точкой:");
10         string input = Console.ReadLine();
11
12         if (string.IsNullOrEmpty(input) || input[^1] != '.')
13         {
14             Console.WriteLine("Текст должен быть завершён точкой.");
15             return;
16         }
17
18         char[] characters = input[..^1].ToCharArray(); // отбрасываем последнюю точку
19         foreach (char c in characters)
20         {
21             if (Array.FindAll(characters, ch => ch == c).Length == 1)
22             {
23                 Console.Write(c);
24             }
25         }
26         Console.WriteLine();
27     }
28 }
```

```

Введите текст, завершённый точкой:
Какой-то текст без точки
Текст должен быть завершён точкой.
PS C:\Users\Mikhail\Documents\Repos\Uni
work\Base_Programmin_1\ЛР4> ^C
PS C:\Users\Mikhail\Documents\Repos\Uni
work\Base_Programmin_1\ЛР4>
PS C:\Users\Mikhail\Documents\Repos\Uni
work\Base_Programmin_1\ЛР4> & 'c:\User
scode\extensions\ms-dotnettools.csharp-
2-x64\.debugger\x86_64\vsdbg.exe' '--in
code' '--connection=78bfa6ee03454b9e9f7
3'
Введите текст, завершённый точкой:
Какой-то текст с точкой .
Ka-ech
PS C:\Users\Mikhail\Documents\Repos\Uni

```

2 способ:

```

1  using System;
2  using System.Linq;
3
4  0 references
5  class Program
6  {
7      0 references
8      static void Main()
9      {
10         Console.WriteLine("Введите текст, завершённый точкой:");
11         string input = Console.ReadLine();
12
13         if (string.IsNullOrEmpty(input) || !input.EndsWith('.'))
14         {
15             Console.WriteLine("Текст должен быть завершён точкой.");
16             return;
17         }
18
19         string content = input.TrimEnd('.');
20         string uniqueChars = new string(content.Where(c => content.Count(ch =>
21             ch == c) == 1).ToArray());
22         Console.WriteLine(uniqueChars);
23     }
24 }

```

```
Введите текст, завершённый точкой:  
Текст, пример, что-то.  
ТКСПИМЧ-  
PS C:\Users\Mikhail\Documents\Repos\University-Work\Base_Programmin_1\ЛР4> □
```

2. Ввести с клавиатуры текст предложения, завершить точкой. Сформировать новую строку на основе исходной, в которой после каждого слова в скобках указать номер слова в предложении (слова разделяются запятыми, пробелами или тире).

Например, если введено «Донецк – прекрасный город», результирующая строка должна выглядеть так: «Донецк(1) – прекрасный(2) город(3)».

Решить задачу 2 способами: через обработку строки как массива символов и с помощью методов класса string.

1 способ:

```
Console.WriteLine("Введите текст, завершённый точкой:");  
string input = Console.ReadLine();  
  
if (string.IsNullOrEmpty(input) || input[^1] != '.')  
{  
    Console.WriteLine("Текст должен быть завершён точкой.");  
    return;  
}  
  
input = input[..^1]; // убираем точку в конце  
char[] delimiters = { ' ', ',', '-', '.' };  
string[] words = input.Split(delimiters, StringSplitOptions.RemoveEmptyEntries);  
char[] inputChars = input.ToCharArray();  
  
int wordIndex = 0, charIndex = 0;
```

```

while (charIndex < inputChars.Length)
{
    if (!char.IsLetterOrDigit(inputChars[charIndex]) &&
        delimiters.Contains(inputChars[charIndex]))
    {
        Console.Write(inputChars[charIndex]);
        charIndex++;
        continue;
    }

    Console.Write(words[wordIndex] + $"({wordIndex + 1})");
    charIndex += words[wordIndex].Length;
    wordIndex++;

    while (charIndex < inputChars.Length && delimiters.Contains(inputChars[charIndex]))
    {
        Console.Write(inputChars[charIndex]);
        charIndex++;
    }
}
Console.WriteLine();

```

```

:\Users\Mikhail\.vscode\extensions\ms-dotn
ettools.csharp-2.55.29-win32-x64\.debugger
\x86_64\vsdbg.exe' '--interpreter=vscode'
'--connection=2985e7eb74be43d6a1c2755a556c
9daf'
Введите текст, завершённый точкой:
Чай - это круто.
Чай(1) - это(2) круто(3)
PS C:\Users\Mikhail\Documents\Repos\Univer
sity-Homework\Base_Programmin_1\ЛР4>

```

2 способ:

```

Console.WriteLine("Введите текст, завершённый точкой:");
string input = Console.ReadLine();

if (string.IsNullOrEmpty(input) || !input.EndsWith('.'))
{
    Console.WriteLine("Текст должен быть завершён точкой.");
    return;
}

string content = input.TrimEnd('.');
char[] delimiters = { ' ', ',', '-', '.' };
string[] words = content.Split(delimiters, StringSplitOptions.RemoveEmptyEntries);

int wordIndex = 0;
string result = string.Empty;
int currentWordStart = 0;

```

```
foreach (string word in words)
{
    int wordPosition = content.IndexOf(word, currentWordStart);

    result += content.Substring(currentWordStart, wordPosition - currentWordStart);

    // Добавляем слово с индексом
    result += $"{word}({++wordIndex})";

    // Обновляем текущую позицию
    currentWordStart = wordPosition + word.Length;
}

if (currentWordStart < content.Length)
{
    result += content.Substring(currentWordStart);
}

Console.WriteLine(result + ".");
```

```
content.IndexOf(word, currentWordStart);
substring(currentWordStart, wordPosition - currentWordStart);

индексом
++wordIndex}");

) позицию
wordPosition + word.Length;

content.Length)
```

```
PS C:\Users\Mikhail\Documents\Repos\University-Homework\Base_Programmin_1\LP4> & 'c:\Users\Mikhail\.vscode\extensions\ms-dotnettools.csharp-2.55.29-win32-x64\debugger\x86_64\vsdbg.exe' '--interpreter=vscode' '--connection=91960cc914b04ec09b57473222109fbe'
Введите текст, завершённый точкой:
Я очень люблю чай.
Я(1) очень(2) люблю(3) чай(4).
PS C:\Users\Mikhail\Documents\Repos\University-Homework\Base_Programmin_1\LP4>
```

3. Ввести текст из нескольких слов, завершить точкой. Сформировать новую строку на основе исходной путем перестановки слов в обратной последовательности.

Решить задачу 2 способами: через обработку строки как массива символов и с помощью методов классов string и StringBuilder.

1 способ:

```
static void Main()
{
    Console.WriteLine("Введите текст, завершённый точкой:");
    string input = Console.ReadLine();

    if (string.IsNullOrEmpty(input) || input[^1] != '.')
    {
        Console.WriteLine("Текст должен быть завершён точкой.");
        return;
    }

    char[] delimiters = { ' ', ',', '-', '.' };
    char[] inputChars = input[..^1].ToCharArray(); // Убираем точку
    string[] words = SplitToWords(inputChars, delimiters);
    Array.Reverse(words);
    Console.WriteLine(string.Join(" ", words) + ".");
}
```

```

static string[] SplitToWords(char[] input, char[] delimiters)
{
    var words = new System.Collections.Generic.List<string>();
    int start = -1;

    for (int i = 0; i < input.Length; i++)
    {
        if (Array.Exists(delimiters, d => d == input[i]) || i == input.Length - 1)
        {
            if (start != -1)
            {
                int length;
                if (i == input.Length - 1 && !Array.Exists(delimiters, d => d ==
input[i]))
                {
                    length = i - start + 1;
                }
                else
                {
                    length = i - start;
                }

                words.Add(new string(input, start, length));
                start = -1;
            }
        }
        else if (start == -1)
        {
            start = i;
        }
    }

    return words.ToArray();
}

```

```

s, d => d ==
s\University-Homework\Base_Program
min_1\ЛР4> & 'c:\Users\Mikhail\.v
scode\extensions\ms-dotnettools.cs
harp-2.55.29-win32-x64\.debugger\x
86_64\vsdbg.exe' '--interpreter=vs
code' '--connection=c93020a8833a42
1bb4cc9c2b056df4e0'
Введите текст, завершённый точкой:
текст, который завершен .
завершен который текст.
PS C:\Users\Mikhail\Documents\Repo
s\University-Homework\Base_Program
min_1\ЛР4>

```

2 способ:

```
static void Main()
{
    Console.WriteLine("Введите текст, завершённый точкой:");
    string input = Console.ReadLine();

    if (string.IsNullOrEmpty(input) || !input.EndsWith('.'))
    {
        Console.WriteLine("Текст должен быть завершён точкой.");
        return;
    }

    string content = input.TrimEnd('.');
    string[] words = content.Split(new[] { ' ', ',', '-', '.' },
        StringSplitOptions.RemoveEmptyEntries);
    Array.Reverse(words);

    StringBuilder reversedString = new StringBuilder();
    reversedString.Append(string.Join(" ", words));
    reversedString.Append('.');

    Console.WriteLine(reversedString.ToString());
}
```

4. Ввести с клавиатуры 7 строк, занести их в массив. Вывести все строки, в которых содержится хотя бы одно слово, оканчивающееся на “.com” (регистр символов не важен; слова разделяются пробелами, запятыми или точками). Также вывести номер строки, содержащей наименьшее число пробелов.

Решить 2 способами: через обработку строки как массива символов и с помощью методов класса string.

1 способ:

```
int minSpacesIndex = -1;
int minSpacesCount = int.MaxValue;

for (int i = 0; i < lines.Length; i++)
{
    int spaces = 0;
    foreach (char c in lines[i])
    {
        if (c == ' ') spaces++;
    }

    if (spaces < minSpacesCount)
    {
        minSpacesCount = spaces;
        minSpacesIndex = i;
    }
}

Console.WriteLine($"Наименьшим количеством пробелов находится под номером {minSpacesIndex} строка");
```

```

string[] lines = new string[7];
Console.WriteLine("Введите 7 строк:");
for (int i = 0; i < 7; i++)
{
    lines[i] = Console.ReadLine();
}

Console.WriteLine("\nСтроки, содержащие слова, оканчивающиеся на '.com':");
for (int i = 0; i < lines.Length; i++)
{
    char[] chars = lines[i].ToCharArray();
    bool hasCom = false;

    for (int j = 0; j < chars.Length - 3; j++)
    {
        if (chars[j] == '.' &&
            (chars[j + 1] == 'c' || chars[j + 1] == 'C') &&
            (chars[j + 2] == 'o' || chars[j + 2] == 'O') &&
            (chars[j + 3] == 'm' || chars[j + 3] == 'M') &&
            (j + 4 == chars.Length || chars[j + 4] == ' ' || chars[j + 4] ==
             ',' || chars[j + 4] == '.'))
        {
            hasCom = true;
            break;
        }
    }

    if (hasCom)
    {
        Console.WriteLine(lines[i]);
    }
}

```

2 способ:

```

string[] lines = new string[7];
Console.WriteLine("Введите 7 строк:");
for (int i = 0; i < 7; i++)
{
    lines[i] = Console.ReadLine();
}

Console.WriteLine("\nСтроки, содержащие слова, оканчивающиеся на '.com':");
for (int i = 0; i < lines.Length; i++)
{
    // Разбиваем строку на слова, игнорируя пробелы, точки и запятые
    string[] words = lines[i].Split(new[] { ' ', ',', '.' },
    StringSplitOptions.RemoveEmptyEntries);
    if (words.Any(word => word.EndsWith("com", StringComparison.OrdinalIgnoreCase)))
    {
        Console.WriteLine(lines[i]);
    }
}

```



```
{
    Console.WriteLine(lines[i]);
}

int minSpacesIndex = -1;
int minSpacesCount = int.MaxValue;

for (int i = 0; i < lines.Length; i++)
{
    // считаем пробелы в строке
    int spaces = lines[i].Count(c => c == ' ');
    if (spaces < minSpacesCount)
    {
        minSpacesCount = spaces;
        minSpacesIndex = i;
    }
}

Console.WriteLine($"Строка с минимальным количеством пробелов находится под номером {minSpacesIndex}");
```

Строки, содержащие слова, оканчивающиеся на '.com':
afjlkasdljksdaf.com
fdlkjadsfklsdj.com
fdas fads dsfa afsdasd.com sdfad
s
sdajfklj dfafds.com .dasfasd

Строка с минимальным количеством пробелов находится под номером: 1
PS C:\Users\Mikhail\Documents\Repository\University-Homework\Base_Programmin_1\JP4>

DEBUG CONSOLE
Filter (e.g. text, lexclude, \escape)

5. Ввести с клавиатуры текст. Программно найти в нем и вывести отдельно все слова, которые начинаются с большого латинского символа (от А до Z) и заканчиваются 2 цифрами, например «Petr93», «Johnny70», «Service02».

Решить 2 способами: через обработку строки как массива и с помощью регулярных выражений.

1 способ:

```
//обработка строки как массива символов
static void Main()
{
    Console.WriteLine("Введите текст:");
    string input = Console.ReadLine();

    Console.WriteLine("\nСлова, начинающиеся с заглавной латинской буквы и заканчивающиеся на 2 цифры:");
    string[] words = input.Split(new[] { ' ', ',', '.', ';', ':', '!', '?', '\n' }, StringSplitOptions.RemoveEmptyEntries);

    foreach (string word in words)
    {
        if (IsMatchingWord(word))
        {
            Console.WriteLine(word);
        }
    }
}
```

```
static bool IsMatchingWord(string word)
{
    if (word.Length < 3) return false; // Слово должно быть минимум длиной 3 символа

    // Первый символ должен быть заглавной латинской буквой (A-Z)
    if (word[0] < 'A' || word[0] > 'Z') return false;

    // Последние два символа должны быть цифрами
    int length = word.Length;
    if (!char.IsDigit(word[length - 1]) || !char.IsDigit(word[length - 2]))
        return false;

    return true;
}
```

```
ode' '--connection=55557f27f72142e6ab39819abdbb49ad'
Введите текст:
Я сегодня увидел jkflkja12 и Fadljk78

Слова, начинающиеся с заглавной латинской буквы и заканчивающиеся на 2 цифры:
Fadljk78
PS C:\Users\Mikhail\Documents\Repos\University-Homework\Base_Programmin_1\ЛР4>
```

2 способ:

```
//решение, с использованием регулярных выражений
static void Main()
{
    Console.WriteLine("Введите текст:");
    string input = Console.ReadLine();

    Console.WriteLine("\nСлова, начинающиеся с заглавной латинской буквы и заканчивающиеся на 2 цифры:");
    Regex regex = new Regex(@"^[A-Z][a-zA-Z]*\d{2}$");

    MatchCollection matches = regex.Matches(input);
    foreach (Match match in matches)
    {
        Console.WriteLine(match.Value);
    }
}
```

```
gger\x86_64\vsdbg.exe' '--interpreter=vsc
ode' '--connection=5f73ed9260584ae8b605f9a6dabb32cb'
Введите текст:
олдо длодл Kfasdfj12 Ljkljl12

Слова, начинающиеся с заглавной латинской буквы и заканчивающиеся на 2 цифры:
Kfasdfj12
Ljkljl12
PS C:\Users\Mikhail\Documents\Repos\University-Homework\Base_Programmin_1\ЛР4>
```

6. Ввести строку вида « 15 + 36 = 51 » (количество пробелов может быть разным, числа – целые и могут быть отрицательны). С помощью регулярных выражений разобрать эту строку и занести в переменные типа int оба операнда и сумму. Вывести все переменные на консоль.

```
Console.WriteLine("Введите строку с математической операцией (например,");
string input = Console.ReadLine();

// Регулярное выражение для поиска чисел, оператора и результата
Regex regex = new Regex
(
    @"^(\s*(-?\d+)\s*)([+/*-])\s*(\s*(-?\d+)\s*)(=)\s*(\s*(-?\d+)\s*)$");

Match match = regex.Match(input);

if (match.Success)
{
    // Извлекаем операнды и результат
    int operand1 = int.Parse(match.Groups[1].Value);
    string operatorSign = match.Groups[2].Value;
    int operand2 = int.Parse(match.Groups[3].Value);
    int result = int.Parse(match.Groups[4].Value);

    // Выводим на консоль все извлеченные значения
    Console.WriteLine($"Операнд 1: {operand1}");
    Console.WriteLine($"Оператор: {operatorSign}");
    Console.WriteLine($"Операнд 2: {operand2}");
    Console.WriteLine($"Результат: {result}");
}
else
{
    Console.WriteLine("Строка не соответствует формату.");
}
```

```
Введите строку с математической операцией (например, '17 + 40 = 57'):  
64 - 90 = -26  
Операнд 1: 64  
Оператор: -  
Операнд 2: 90  
Результат: -26  
PS C:\Users\Mikhail\Documents\Repos\University-Homework\Bas
```

7. Дан треклист – массив из 10 строк следующего вида:

1. Gentle Giant – Free Hand [6:15]
2. Supertramp – Child Of Vision [07:27]
3. Camel – Lawrence [10:46]
4. Yes – Don't Kill The Whale [3:55]
5. 10CC – Notell Hotel [04:58]
6. Nektar – King Of Twilight [4:16]
7. The Flower Kings – Monsters & Men [21:19]
8. Focus – Le Clochard [1:59]
9. Pendragon – Fallen Dream And Angel [5:23]
10. Kaipa – Remains Of The Day (08:02)

Написать программу, которая обрабатывает весь треклист, суммирует время звучания песен и выводит результат на экран, а также отображает самую длинную и самую короткую песню в списке и пару песен с минимальной разницей во времени звучания.

```
static TimeSpan ParseTime(string track)
{
    // Обрабатываем оба варианта разделителей
    var startIdx = track.IndexOfAny(new char[] { '[', '(' });
    var endIdx = track.IndexOfAny(new char[] { ']', ')' });

    var timeStr = track.Substring(startIdx + 1, endIdx - startIdx - 1);
    var parts = timeStr.Split(':');
    int minutes = int.Parse(parts[0]);
    int seconds = int.Parse(parts[1]);
    return new TimeSpan(0, minutes, seconds);
}
```

```

static void Main()
{
    string[] tracklist = new string[]
    {
        "Gentle Giant [ Free Hand [6:15]",
        "Supertramp [ Child Of Vision [07:27]",
        "Camel [ Lawrence [10:46]",
        "Yes [ Don't Kill The Whale [3:55]",
        "10CC [ Notell Hotel [04:58]",
        "Nektar [ King Of Twilight [4:16]",
        "The Flower Kings [ Monsters & Men [21:19]",
        "Focus [ Le Clochard [1:59]",
        "Pendragon [ Fallen Dream And Angel [5:23]",
        "Kaipa [ Remains Of The Day [08:02]"
    };

    TimeSpan[] durations = tracklist
        .Select(t => ParseTime(t))
        .ToArray();

    TimeSpan totalDuration = durations.Aggregate((t1, t2) => t1 + t2);
    TimeSpan longest = durations.Max();
    TimeSpan shortest = durations.Min();

    string longestSong = tracklist[Array.IndexOf(durations, longest)];
    string shortestSong = tracklist[Array.IndexOf(durations, shortest)];

    var minDiffPair = FindMinTimeDifference(durations, tracklist);

    Console.WriteLine($"Общая продолжительность: {totalDuration}");
    Console.WriteLine($"Наиболее длинная песня: {longestSong}");
    Console.WriteLine($"Наиболее короткая песня: {shortestSong}");
    Console.WriteLine($"Песни с минимальной разницей по продолжительности: [{minDiffPair.Item1} and {minDiffPair.Item2}]");
}

```

```

static Tuple<string, string> FindMinTimeDifference(TimeSpan[] durations, string[] tracklist)
{
    TimeSpan minDiff = TimeSpan.MaxValue;
    string song1 = string.Empty;
    string song2 = string.Empty;

    for (int i = 0; i < durations.Length; i++)
    {
        for (int j = i + 1; j < durations.Length; j++)
        {
            TimeSpan diff = durations[i] - durations[j];
            if (diff < TimeSpan.Zero)
                diff = -diff;

            if (diff < minDiff)
            {
                minDiff = diff;
                song1 = tracklist[i];
                song2 = tracklist[j];
            }
        }
    }

    return new Tuple<string, string>(song1, song2);
}

```

6a483188e1eded9311cb2b'

Общая продолжительность: 01:14:20

Наиболее длинная песня: The Flower Kings - Monsters & Men [21:19]

Наиболее короткая песня: Focus - Le Clochard [1:59]

Песни с минимальной разницей по продолжительности: Yes - Don't Kill The Whale [3:55] и Nektar - King of Twilight [4:16]

PS C:\Users\Mikhail\Documents\Repos\University-Homework\Base_Programmin_1\LP4>

Индивидуальные задания:

1. Написать программу, позволяющую шифровать и расшифровывать строки символов на основе 3 симметричных алгоритмов шифрования (прилож. А).

8	5, 6, 10
---	----------

5

Шифр Гронсфельда

(http://ru.wikipedia.org/wiki/Полиалфавитный_шифр)

6

Книжный шифр

(http://ru.wikipedia.org/wiki/Книжный_шифр)

10

Шифр Тритемиуса

(http://ru.wikipedia.org/wiki/Шифр_Тритемиуса)

```
static void Main()
{
    // Ввод от пользователя
    Console.WriteLine("Введите текст для шифрования (только русские заглавные буквы:");
    string plaintext = Console.ReadLine().ToUpper();

    Console.WriteLine("Введите ключ для шифра Гронсфельда (только цифры:");
    string keyGronfeld = Console.ReadLine();

    Console.WriteLine("Введите ключ для шифра Тритемиуса (только русские заглавные буквы:");
    string keyTritemiusa = Console.ReadLine().ToUpper();

    Console.WriteLine("Выберите шифр:");
    Console.WriteLine("1. Шифр Гронсфельда");
    Console.WriteLine("2. Книжный шифр");
    Console.WriteLine("3. Шифр Тритемиуса");
    int choice = int.Parse(Console.ReadLine());

    string encryptedText = "";
    string decryptedText = "";
```

```

switch (choice)
{
    case 1:
        encryptedText = EncryptGronfeld(plaintext, keyGronfeld);
        decryptedText = DecryptGronfeld(encryptedText, keyGronfeld);
        Console.WriteLine("\nШифр Гронсфельда:");
        break;

    case 2:
        encryptedText = EncryptBook(plaintext);
        decryptedText = DecryptBook(encryptedText);
        Console.WriteLine("\nКнижный шифр:");
        break;

    case 3:
        encryptedText = EncryptTritemiusa(plaintext, keyTritemiusa);
        decryptedText = DecryptTritemiusa(encryptedText, keyTritemiusa);
        Console.WriteLine("\nШифр Тритемиуса:");
        break;

    default:
        Console.WriteLine("Неверный выбор.");
        return;
}

Console.WriteLine("Зашифрованный текст: " + encryptedText);
Console.WriteLine("Расшифрованный текст: " + decryptedText);
}

```

```

// Функция для шифрования с помощью шифра Гронсфельда (для русского алфавита)
static string EncryptGronfeld(string text, string key)
{
    string result = "";
    for (int i = 0; i < text.Length; i++)
    {
        int shift = key[i % key.Length] - '0'; // Преобразуем цифру из ключа в сдвиг
        char encryptedChar = (char)((text[i] - 'А' + shift) % 32 + 'А');
        // Используем диапазон 'А'-'Я'
        result += encryptedChar;
    }
    return result;
}

// Функция для расшифрования с помощью шифра Гронсфельда (для русского алфавита)
static string DecryptGronfeld(string text, string key)
{
    string result = "";
    for (int i = 0; i < text.Length; i++)
    {
        int shift = key[i % key.Length] - '0'; // Преобразуем цифру из ключа в сдвиг
        char decryptedChar = (char)((text[i] - 'А' - shift + 32) % 32 + 'А');
        // Используем диапазон 'А'-'Я'
        result += decryptedChar;
    }
    return result;
}

```

```
// Функция для шифрования с помощью книжного шифра (для русского алфавита)
static string EncryptBook(string text)
{
    string result = "";
    foreach (var ch in text)
    {
        char encryptedChar = (char)((ch - 'А' + 3) % 32 + 'А'); // Сдвиг на 3 символа
        result += encryptedChar;
    }
    return result;
}

// Функция для расшифрования с помощью книжного шифра
static string DecryptBook(string text)
{
    string result = "";
    foreach (var ch in text)
    {
        char decryptedChar = (char)((ch - 'А' - 3 + 32) % 32 + 'А'); // Обратный сдвиг на 3
        result += decryptedChar;
    }
    return result;
}
```

```
// Функция для шифрования с помощью шифра Тригемииуса
static string EncryptTritemiusa(string text, string key)
{
    string result = "";
    for (int i = 0; i < text.Length; i++)
    {
        int shift = key[i % key.Length] - 'А';
        // Преобразуем букву ключа в сдвиг
        char encryptedChar = (char)((text[i] - 'А' + shift) % 32 + 'А');
        // Используем диапазон 'А'-'Я'
        result += encryptedChar;
    }
    return result;
}

// Функция для расшифрования с помощью шифра Тригемииуса
static string DecryptTritemiusa(string text, string key)
{
    string result = "";
    for (int i = 0; i < text.Length; i++)
    {
        int shift = key[i % key.Length] - 'А'; // Преобразуем букву ключа в сдвиг
        char decryptedChar = (char)((text[i] - 'А' - shift + 32) % 32 + 'А');
        // Используем диапазон 'А'-'Я'
        result += decryptedChar;
    }
    return result;
}
}
```

496e965a5aba3a95430b'

Введите текст для шифрования (только русские заглавные буквы):

СЛОВО

Введите ключ для шифра Гронсфельда (только цифры):

56

Введите ключ для шифра Тритемиуса (только русские заглавные буквы):

КЛЮЧ

Выберите шифр:

1. Шифр Гронсфельда
2. Книжный шифр
3. Шифр Тритемиуса

2

Книжный шифр:

Зашифрованный текст: Ф0СЕС

Расшифрованный текст: СЛОВО

PS C:\Users\Mikhail\Documents\Repos\University-Homework\Base_Programmin_1\LP4> █

4d9183d35ef3a2029dae'

Введите текст для шифрования (только русские заглавные буквы):

ТЕКСТИК

Введите ключ для шифра Гронсфельда (только цифры):

23

Введите ключ для шифра Тритемиуса (только русские заглавные буквы):

КЛЮЧИК

Выберите шифр:

1. Шифр Гронсфельда
2. Книжный шифр
3. Шифр Тритемиуса

3

Шифр Тритемиуса:

Зашифрованный текст: ЪРИИЪТФ

Расшифрованный текст: ТЕКСТИК

PS C:\Users\Mikhail\Documents\Repos\University-Homework\Base_Programmin_1\LP4> █


```

Введите текст для шифрования (только русские заглавные буквы):
ТЕКСТИК
Введите ключ для шифра Гронсфельда (только цифры):
4565
Введите ключ для шифра Тритемиуса (только русские заглавные буквы):
КЛЮЧ
Выберите шифр:
1. Шифр Гронсфельда
2. Книжный шифр
3. Шифр Тритемиуса
1

Шифр Гронсфельда:
Зашифрованный текст: ЦКРЦЦНР
Расшифрованный текст: ТЕКСТИК
PS C:\Users\Mikhail\Documents\Repos\University-Homework\Bas

```

2. Написать программу обработки текста, в соответствии с вариантом.

Решить задачу 2 способами: через обработку строки как массива символов и с помощью методов классов string и/или StringBuilder.

Ввести с клавиатуры текст. Сформировать новую строку из исходной путем замены всех цифр от 1 до 5 соответствующими словами – one, two, three, four, five.

1 способ:

```

// Использование массива символов
static void Main(string[] args)
{
    Console.WriteLine("Введите текст:");
    string inputText = Console.ReadLine();

    string result = ReplaceDigitsWithWordsArray(inputText);
    Console.WriteLine("\nРезультат через обработку массива символов:");
    Console.WriteLine(result);
}

```

```

static string ReplaceDigitsWithWordsArray(string input)
{
    char[] chars = input.ToCharArray();
    StringBuilder result = new StringBuilder();

    foreach (char c in chars)
    {
        switch (c)
        {
            case '1': result.Append("one"); break;
            case '2': result.Append("two"); break;
            case '3': result.Append("three"); break;
            case '4': result.Append("four"); break;
            case '5': result.Append("five"); break;
            default: result.Append(c); break;
        }
    }

    return result.ToString();
}

```

```

01C22940008C1059CEC2A054B1
Введите текст:
У меня было 2 яблока и 4 груши

Результат через обработку массива символов:
У меня было two яблока и four груши
PS C:\Users\Mikhail\Documents\Repos\University
-Homework\Base_Programmin_1\ЛР4>

```

2 способ:

```

// Использование методов String и StringBuilder
static void Main(string[] args)
{
    Console.WriteLine("Введите текст:");
    string inputText = Console.ReadLine();

    string result = ReplaceDigitsWithWordsStringBuilder(inputText);
    Console.WriteLine("\nРезультат через методы String/StringBuilder:");
    Console.WriteLine(result);
}

static string ReplaceDigitsWithWordsStringBuilder(string input)
{
    StringBuilder result = new StringBuilder(input);

    result.Replace("1", "one");
    result.Replace("2", "two");
    result.Replace("3", "three");
    result.Replace("4", "four");
    result.Replace("5", "five");

    return result.ToString();
}

```

```

C:\Users\Mikhail\Documents\Repos\University> .\Base_Programmin_1\ЛР4.exe
97a7b34496ad1d1890b0f1076a'
Введите текст:
Было 2 собаки и 1 игрушка

Результат через методы String/StringBuilder:
Было two собаки и one игрушка
PS C:\Users\Mikhail\Documents\Repos\University
-Homework\Base_Programmin_1\ЛР4>

```

3. Написать регулярные выражения для поиска подстроки в строке по правилу или шаблону, в соответствии с вариантом.

Найти в тексте все логические выражения – подстроки вида «x && y», «x & y», где x и y – любые слова. Количество пробелов может быть также любым.

```
static void Main(string[] args)
{
    Console.WriteLine("Введите текст:");
    string inputText = Console.ReadLine();

    // Регулярное выражение
    string pattern = @"\"b\\w+\\s*&{1,2}\\s*\\w+\\b";

    // Поиск совпадений
    MatchCollection matches = Regex.Matches(inputText, pattern);

    Console.WriteLine("\nНайденные логические выражения:");
    foreach (Match match in matches)
    {
        Console.WriteLine(match.Value);
    }
}
```

7336d24bbeaea115d1176bb806'
Введите текст:
яблоко & груша, кофе && термос

Найденные логические выражения:
яблоко & груша
кофе && термос
PS C:\Users\Mikhail\Documents\Repos\Universi
-Homework\Base_Programmin_1\LP4>

Контрольные вопросы

Лабораторная работа №4
Курс "Основы программирования"

Тема: Работа со строками в C#. Знакомство с регулярными выражениями.

Цель: научиться работать с типом string на языке C#, а также со строкой как с массивом символов.

Контрольные вопросы

1. Как строка предст. как объект класса String из простран. имен System.

Кравченко Михаил

Они явл. неизм. послед. символов Unicode;
Задаваемым строковым литералом ("привет")
или интерполированным строком (§ "Привет,
{имя}").

2. Особенности:

- строка неизменяема;
- поддержка unicode;
- методы для чувств и нечувств. к регистру сравнения.

Они методы:

- Length;
- Substring();
- IndexOf();
- Replace();
- ToLower(), ToUpper();
- Split();
- Join();
- Trim() - удалит пробелы.

Курсовая работа МДП-4

2

3. `Split()` - делит строку на массив подстрок по заданным разделителям.

а, б, с: `Split("abc", ",");`

`Join()`: Объединяет массив строк с заданным разделителем:

`string.Join("-", new[] { "a", "b" });`

4. Особенности сравнения строк:

- чувств. к регистру по умолчанию

- методы: `Equals()`, `Compare()`, `StringComparer`.

`OrdinalIgnoreCase`.

- сравнение может учитывать культуру,

с помощью `String.Compare()`.

5. AST: абстрактное дерево, представляющее

структуру программы, исп. для анализа и

оптимизации.

Дерево разбора: более детальное дерево, показ.

правила и структуру кода.

AST обходит его.

Курсовая работа 19.11.19

6. Пример ГBNF - выражений:

$\langle \text{выражение} \rangle ::= \langle \text{термин} \rangle \mid \langle \text{термин} \rangle "+"$
 $\langle \text{выражение} \rangle;$

$\langle \text{термин} \rangle ::= \langle \text{число} \rangle \mid "(" \langle \text{выражение} \rangle$
 $") ";$

$\langle \text{число} \rangle ::= "0" \mid "1" \mid \dots \mid "9";$

7. Регулярные выражения (Регекс): шаблоны
для поиска или проверки текста.

Применение: проверка форматов (например,
email), разбор и анализ текста.

8. Баз. элементы регулярных
выражений:

- "." - любой символ;
- "x", "+", "?", "{n}" - квантификаторы (повторы).
- "[" - набор символов;
- "\d, \w" - цифра и слово;
- "^, \$" - начало и конец строки;
- "(a|b)" - альтернатива.

Курсово

Минин

11.07.14

4

2. Функции - Массивы (x, +): захватывают максимум символов, но могут вернуть ~~макс.~~
- Массивы (x?, +?): захватывают ~~макс.~~ минимум символов;
- Собств. (x+, ++): захватывают макс. и не возвр.

Вывод: в результате выполнения данной лабораторной работы мы научились работать с типом string на языке C#, а также со строкой как с массивом символов, научились писать простейшие регулярн. выражения для поиска подстроки в строке.

Красно Мичам 19.11.19-4

5

Вывод: в результате выполнения данной лабораторной работы мы научились работать с типом string на языке C#, а также со строкой как с массивом символов, научились писать простейшие регулярные выражения для поиска подстроки в строке.