

Федеральное государственное бюджетное образовательное учреждение высшего
образования
«Сибирский государственный университет телекоммуникаций и информатики»
(СибГУТИ)

Кафедра прикладной математики и кибернетики

Лабораторная работа № 2
по дисциплине «Теория Информации»

Выполнил:
студент группы ИП-712
Алексеев Степан
Владимирович
ФИО студента

Работу проверил:
доцент кафедры ПМИК Мачикина Е.П.
ФИО преподавателя

Новосибирск 2021 г.

Оглавление

ЗАДАНИЕ	2
Решение	3
Анализ	3
Скриншоты	4
Листинг кода	4

ЗАДАНИЕ

Теория информации

Практическая работа №2

Вычисление энтропии Шеннона

Цель работы: Экспериментальное изучение свойств энтропии Шеннона для текстов на естественном языке.

Язык программирования: C, C++, C#, Python

Результат: программа, тестовые примеры, отчет.

Задание:

1. Выбрать художественный текст на русском (английском) языке. Объем файла в формате txt более 10 Кб. Для алфавита текста предполагается, что строчные и заглавные символы не отличаются, знаки препинания опущены, к алфавиту добавлен пробел, для русских текстов буквы «е» и «ё», «ь» и «ъ» совпадают.
2. Составить программу, определяющую несколько оценок энтропии данного текстового файла. Оценки энтропии необходимо вычислить по формуле Шеннона двумя способами, т.е. используя частоты отдельных символов и используя частоты пар символов. По желанию можно продолжить процесс вычисления оценок с использованием частот троек, четверок символов и т.д.
3. После тестирования программы необходимо заполнить таблицу для отчета и проанализировать полученные результаты. Сравнить полученные результаты с результатами работы 1.

Название текста	Максимально возможное значение энтропии	Оценка энтропии (одиночные символы)	Оценка энтропии (частоты пар символов)

4. Оформить отчет, загрузить отчет и файл с исходным кодом в электронную среду. Отчет обязательно должен содержать заполненную таблицу и анализ полученных результатов. По желанию в отчет можно включить описание программной реализации. В отчет не нужно включать содержимое этого файла.

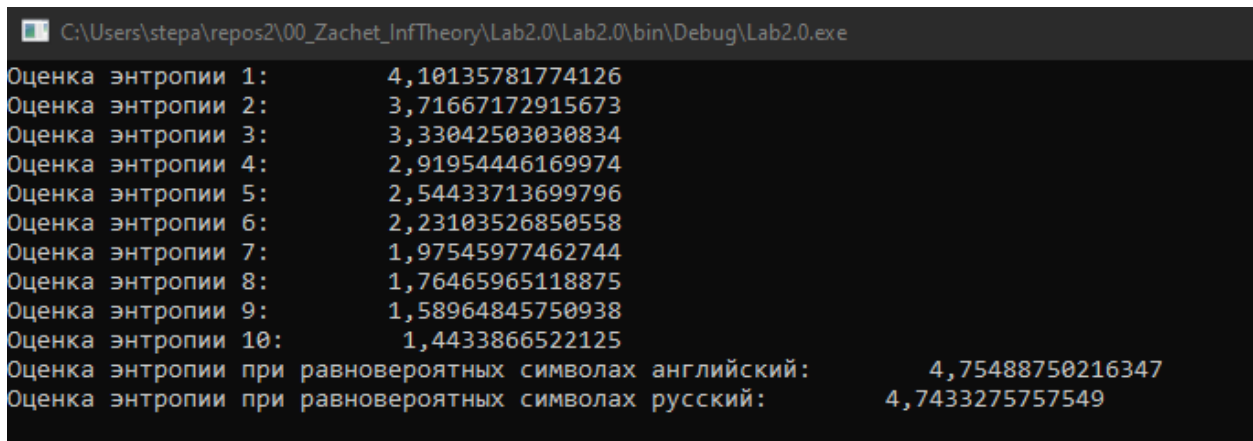
Решение

Название текста	Максимально возможное значение энтропии	Оценка энтропии (одиночные символы)	Оценка энтропии (частоты пар символов)
Ден Симмонс Гиперион	4,754887	4,10135781774126	3,71667172915673

Анализ

Видим, что во-первых энтропия одиночных символов больше, чем у текста из первой лабораторной работы. Это объясняется большим размером алфавита. Во-вторых, энтропия уменьшается при увеличении размеров блоков. Это объясняется тем, что в естественных языках символы зависят от контекста (от предыдущих символов), поэтому неопределённость (энтропия) уменьшается с каждым новым символом в блоке. Средняя энтропия для европейских языков около 2 бит. Это много (много синтаксического сахара).

Скриншоты



```
C:\Users\stepa\repos2\00_Zachet_InfTheory\Lab2.0\bin\Debug\Lab2.0.exe
Оценка энтропии 1: 4,10135781774126
Оценка энтропии 2: 3,71667172915673
Оценка энтропии 3: 3,33042503030834
Оценка энтропии 4: 2,91954446169974
Оценка энтропии 5: 2,54433713699796
Оценка энтропии 6: 2,23103526850558
Оценка энтропии 7: 1,97545977462744
Оценка энтропии 8: 1,76465965118875
Оценка энтропии 9: 1,58964845750938
Оценка энтропии 10: 1,4433866522125
Оценка энтропии при равновероятных символах английский: 4,75488750216347
Оценка энтропии при равновероятных символах русский: 4,7433275757549
```

Листинг кода

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.IO;
using System.Text.RegularExpressions;

namespace Lab2._0
{
    class Program
    {
        static int numberOfChars = 0;
        static Dictionary<string, double> dicti1 = new Dictionary<string, double>();
        static Dictionary<string, double> dicti2 = new Dictionary<string, double>();
        static Dictionary<string, double> dicti3 = new Dictionary<string, double>();
        static Dictionary<string, double> dictiEvenEng = new Dictionary<string, double>();
        static Dictionary<string, double> dictiEvenRus = new Dictionary<string, double>();
        static int numberOfLettersInABlock = 1;

        static void Main(string[] args)
        {
            convertFileAndCountChars("C:/Users/stepa/repos2/00_Zachet_InfTheory/Lab2.0/F1.txt");

            countProbabilitiesBasedOnRealFrequencyInFile("C:/Users/stepa/repos2/00_Zachet_InfTheory/
Lab2.0/F1_Converted.txt", dicti1, numberOfLettersInABlock);
            Console.WriteLine("Оценка энтропии 1: " + ShannonFormulaForEntropy(dicti1,
numberOfLettersInABlock));
        }
    }
}
```

```

        numberOfLettersInABlock = 2;

countProbabilitiesBasedOnRealFrequencyInFile("C:/Users/stepa/repos2/00_Zachet_InfTheory/
Lab2.0/F1_Converted.txt", dicti2, numberOfLettersInABlock);
    Console.WriteLine("Оценка энтропии 2:    " + ShannonFormulaForEntropy(dicti2,
numberOfLettersInABlock));

        numberOfLettersInABlock = 3;

countProbabilitiesBasedOnRealFrequencyInFile("C:/Users/stepa/repos2/00_Zachet_InfTheory/
Lab2.0/F1_Converted.txt", dicti3, numberOfLettersInABlock);
    Console.WriteLine("Оценка энтропии 3:    " + ShannonFormulaForEntropy(dicti3,
numberOfLettersInABlock));

        numberOfLettersInABlock = 4;
        dicti3 = new Dictionary<string, double>();

countProbabilitiesBasedOnRealFrequencyInFile("C:/Users/stepa/repos2/00_Zachet_InfTheory/
Lab2.0/F1_Converted.txt", dicti3, numberOfLettersInABlock);
    Console.WriteLine("Оценка энтропии 4:    " + ShannonFormulaForEntropy(dicti3,
numberOfLettersInABlock));

        numberOfLettersInABlock = 5;
        dicti3 = new Dictionary<string, double>();

countProbabilitiesBasedOnRealFrequencyInFile("C:/Users/stepa/repos2/00_Zachet_InfTheory/
Lab2.0/F1_Converted.txt", dicti3, numberOfLettersInABlock);
    Console.WriteLine("Оценка энтропии 5:    " + ShannonFormulaForEntropy(dicti3,
numberOfLettersInABlock));

        numberOfLettersInABlock = 6;
        dicti3 = new Dictionary<string, double>();

countProbabilitiesBasedOnRealFrequencyInFile("C:/Users/stepa/repos2/00_Zachet_InfTheory/
Lab2.0/F1_Converted.txt", dicti3, numberOfLettersInABlock);
    Console.WriteLine("Оценка энтропии 6:    " + ShannonFormulaForEntropy(dicti3,
numberOfLettersInABlock));

        numberOfLettersInABlock = 7;
        dicti3 = new Dictionary<string, double>();

countProbabilitiesBasedOnRealFrequencyInFile("C:/Users/stepa/repos2/00_Zachet_InfTheory/
Lab2.0/F1_Converted.txt", dicti3, numberOfLettersInABlock);
    Console.WriteLine("Оценка энтропии 7:    " + ShannonFormulaForEntropy(dicti3,
numberOfLettersInABlock));

        numberOfLettersInABlock = 8;

```

```

dicti3 = new Dictionary<string, double>();

countProbabilitiesBasedOnRealFrequencyInFile("C:/Users/stepa/repos2/00_Zachet_InfTheory/
Lab2.0/F1_Converted.txt", dicti3, numberOfLettersInABlock);
    Console.WriteLine("Оценка энтропии 8:    " + ShannonFormulaForEntropy(dicti3,
numberOfLettersInABlock));

    numberOfLettersInABlock = 9;
    dicti3 = new Dictionary<string, double>();

countProbabilitiesBasedOnRealFrequencyInFile("C:/Users/stepa/repos2/00_Zachet_InfTheory/
Lab2.0/F1_Converted.txt", dicti3, numberOfLettersInABlock);
    Console.WriteLine("Оценка энтропии 9:    " + ShannonFormulaForEntropy(dicti3,
numberOfLettersInABlock));

    numberOfLettersInABlock = 10;
    dicti3 = new Dictionary<string, double>();

countProbabilitiesBasedOnRealFrequencyInFile("C:/Users/stepa/repos2/00_Zachet_InfTheory/
Lab2.0/F1_Converted.txt", dicti3, numberOfLettersInABlock);
    Console.WriteLine("Оценка энтропии 10:   " + ShannonFormulaForEntropy(dicti3,
numberOfLettersInABlock));

    numberOfLettersInABlock = 11;
    dicti3 = new Dictionary<string, double>();

countProbabilitiesBasedOnRealFrequencyInFile("C:/Users/stepa/repos2/00_Zachet_InfTheory/
Lab2.0/F1_Converted.txt", dicti3, numberOfLettersInABlock);
    Console.WriteLine("Оценка энтропии 11:   " + ShannonFormulaForEntropy(dicti3,
numberOfLettersInABlock));

    numberOfLettersInABlock = 12;
    dicti3 = new Dictionary<string, double>();

countProbabilitiesBasedOnRealFrequencyInFile("C:/Users/stepa/repos2/00_Zachet_InfTheory/
Lab2.0/F1_Converted.txt", dicti3, numberOfLettersInABlock);
    Console.WriteLine("Оценка энтропии 12:   " + ShannonFormulaForEntropy(dicti3,
numberOfLettersInABlock));

    numberOfLettersInABlock = 13;
    dicti3 = new Dictionary<string, double>();

countProbabilitiesBasedOnRealFrequencyInFile("C:/Users/stepa/repos2/00_Zachet_InfTheory/
Lab2.0/F1_Converted.txt", dicti3, numberOfLettersInABlock);
    Console.WriteLine("Оценка энтропии 13:   " + ShannonFormulaForEntropy(dicti3,
numberOfLettersInABlock));

```

```

        numberOfLettersInABlock = 14;
        dicti3 = new Dictionary<string, double>();

        countProbabilitiesBasedOnRealFrequencyInFile("C:/Users/stepa/repos2/00_Zachet_InfTheory/
        Lab2.0/F1_Converted.txt", dicti3, numberOfLettersInABlock);
        Console.WriteLine("Оценка энтропии 14: " + ShannonFormulaForEntropy(dicti3,
        numberOfLettersInABlock));

```

//Для подсчёта максимально возможной энтропии, видимо, нужно взять тот же алфавит и сделать символы равновероятными...

```

        numberOfLettersInABlock = 1;
        numberOfChars = 28746;
        dictiEvenEng.Add("a", (double)1 / (double)27);
        dictiEvenEng.Add("b", (double)1 / (double)27);
        dictiEvenEng.Add("c", (double)1 / (double)27);
        dictiEvenEng.Add("d", (double)1 / (double)27);
        dictiEvenEng.Add("e", (double)1 / (double)27);
        dictiEvenEng.Add("f", (double)1 / (double)27);
        dictiEvenEng.Add("g", (double)1 / (double)27);
        dictiEvenEng.Add("h", (double)1 / (double)27);
        dictiEvenEng.Add("i", (double)1 / (double)27);
        dictiEvenEng.Add("j", (double)1 / (double)27);
        dictiEvenEng.Add("k", (double)1 / (double)27);
        dictiEvenEng.Add("l", (double)1 / (double)27);
        dictiEvenEng.Add("m", (double)1 / (double)27);
        dictiEvenEng.Add("n", (double)1 / (double)27);
        dictiEvenEng.Add("o", (double)1 / (double)27);
        dictiEvenEng.Add("p", (double)1 / (double)27);
        dictiEvenEng.Add("q", (double)1 / (double)27);
        dictiEvenEng.Add("r", (double)1 / (double)27);
        dictiEvenEng.Add("s", (double)1 / (double)27);
        dictiEvenEng.Add("t", (double)1 / (double)27);
        dictiEvenEng.Add("u", (double)1 / (double)27);
        dictiEvenEng.Add("v", (double)1 / (double)27);
        dictiEvenEng.Add("w", (double)1 / (double)27);
        dictiEvenEng.Add("x", (double)1 / (double)27);
        dictiEvenEng.Add("y", (double)1 / (double)27);
        dictiEvenEng.Add("z", (double)1 / (double)27);
        dictiEvenEng.Add(" ", (double)1 / (double)27);
        Console.WriteLine("Оценка энтропии при равновероятных символах английский:
        " + ShannonFormulaForEntropy(dictiEvenEng, numberOfLettersInABlock));

        dictiEvenRus.Add("a", (double)1 / (double)30);
        dictiEvenRus.Add("б", (double)1 / (double)30);
        dictiEvenRus.Add("в", (double)1 / (double)30);
        dictiEvenRus.Add("г", (double)1 / (double)30);

```

```

dictiEvenRus.Add("д", (double)1 / (double)30);
//dictiEvenRus.Add("е", (double)1 / (double)30);// По заданию буквы е, ё, ь, ъ нужно
считать одной буквой
dictiEvenRus.Add("ё", (double)1 / (double)30);
dictiEvenRus.Add("ж", (double)1 / (double)30);
dictiEvenRus.Add("з", (double)1 / (double)30);
dictiEvenRus.Add("и", (double)1 / (double)30);
dictiEvenRus.Add("к", (double)1 / (double)30);
dictiEvenRus.Add("л", (double)1 / (double)30);
dictiEvenRus.Add("м", (double)1 / (double)30);
dictiEvenRus.Add("н", (double)1 / (double)30);
dictiEvenRus.Add("о", (double)1 / (double)30);
dictiEvenRus.Add("п", (double)1 / (double)30);
dictiEvenRus.Add("р", (double)1 / (double)30);
dictiEvenRus.Add("с", (double)1 / (double)30);
dictiEvenRus.Add("т", (double)1 / (double)30);
dictiEvenRus.Add("у", (double)1 / (double)30);
dictiEvenRus.Add("ф", (double)1 / (double)30);
dictiEvenRus.Add("х", (double)1 / (double)30);
dictiEvenRus.Add("ц", (double)1 / (double)30);
dictiEvenRus.Add("ч", (double)1 / (double)30);
dictiEvenRus.Add("ш", (double)1 / (double)30);
dictiEvenRus.Add("щ", (double)1 / (double)30);
//dictiEvenRus.Add("ъ", (double)1 / (double)30);
dictiEvenRus.Add("ы", (double)1 / (double)30);
//dictiEvenRus.Add("ь", (double)1 / (double)30);
dictiEvenRus.Add("э", (double)1 / (double)30);
dictiEvenRus.Add("ю", (double)1 / (double)30);
dictiEvenRus.Add("я", (double)1 / (double)30);
Console.WriteLine("Оценка энтропии при равновероятных символах русский: " +
ShannonFormulaForEntropy(dictiEvenRus, numberOfLettersInABlock));

Console.ReadKey();
}
static string convertFileAndCountChars(string path)
{
//To lower case; get rid of punctuation; add whitespace as a character; для русских
текстов буквы «е» и «ё», «ь» и «ъ» совпадают.
string newPath =
@"C:/Users/stepa/repos2/00_Zachet_InfTheory/Lab2.0/F1_Converted.txt";
string str;
using (StreamReader sr = File.OpenText(path))
{
str = sr.ReadToEnd();
}

str = str.Replace("е", "ё");
str = str.Replace("ь", "ъ");

```



```

str = str.Replace("«", "");
str = str.Replace("№", "");
str = str.Replace("-", "");
//tr = str.Replace("\t", "");
str = str.ToLower();
using (StreamWriter sw = File.CreateText(newPath))
{
    sw.Write(str);
    numberOfChars = str.Length;
}
return newPath;
}

static double ShannonFormulaForEntropy(Dictionary<string, double> dict, int
numberOfLettersInABlock)
{
    //Количество информации, которое мы получаем, достигает максимального
    значения, если события равновероятны... Здесь, видимо,
    //сравниваются значения, полученные применением формулы Хартли...
    //Формула Шеннона позволяет высчитать среднее кол-во информации,
    передаваемое любым сообщением(блоком символов).
    double sum = 0;
    foreach (var item in dict)
    {
        sum += item.Value * Math.Log(1 / item.Value, 2);
    }
    return sum / numberOfLettersInABlock;
}

static void countProbabilitiesBasedOnRealFrequencyInFile(string path, Dictionary<string,
double> dict, int numberOfLettersInABlock)
{
    string str;
    using (StreamReader sr = File.OpenText(path))
    {
        str = sr.ReadToEnd();
    }
    char[] str_chars = str.ToCharArray();
    for (int i = 0; i < numberOfChars - numberOfLettersInABlock; i++)
    {
        string block = str_chars[i].ToString();
        for (int j = 1; j < numberOfLettersInABlock; j++)
        {
            block += str_chars[i + j].ToString();
        }
        if (dict.ContainsKey(block))
        {
            dict[block] += ((double)1 / ((double)numberOfChars));///
            (double)numberOfLettersInABlock);
        }
    }
}

```

```

        else
            dict.Add(block, ((double)1 / ((double)numberOfChars)); // /
            (double)numberOfLettersInABlock));
        } // up to here all occurrences of blocks are counted and frequencies (counted
        probabilities) are counted.
        // Time to use Shannon's formula
    }
}
}}}

```