Министерство образования и науки РФ

ФГБОУ ВПО «Тамбовский государственный технический университет»

Кафедра « »

Лабораторная работа №1

по дисциплине «Теория информации»

«Измерение информации»

Выполнил: студент гр. −31 Д. Ю.

Проверил: Н. В.

Тамбов,

***Цели и задачи****.*

Изучить алфавитный подход к измерению информации, реализовать программу.

***Теория.***

Алфавитный подход к измерению информации

При алфавитном подходе к определению количества информации отвлекаются от содержания информации и рассматривают информационное сообщение как последовательность знаков определенной знаковой системы.

Применение алфавитного подхода удобно прежде всего при использовании технических средств работы с информацией. Алфавитный подход является объективным способом измерения информации в отличие от субъективного содержательного подхода.

**Алфавит** - все множество используемых в языке символов.

Обычно под алфавитом понимают только буквы, но поскольку в тексте могут встречаться знаки препинания, цифры, скобки, то их тоже включают в алфавит. В алфавит также следует включить и пробел, т.е. пропуск между словами.

**Мощность алфавита (N)** - полное количество символов алфавита.

Мощность русского алфавита - 33 знака, английского - 26, двоичного кода - 2 ("0" или "1"), компьютерного текста - 256.

Информационный вес символа алфавита **i** и мощность **N** алфавита связаны между собой формулой **N = 2i.**

**Информационный объем сообщения** (количество информации) представленного символами какого-либо алфавита, складывается из информационных весов составляющих его символов.

Информационный объем V сообщения равен произведению количеству K символов в сообщении на информационный вес i символа алфавита **V = K \* i**

***Решение задачи****.*



Программа состоит из блоков, каждый отвечает за свою задачу. Вводимая пользователем информация поступает на вход блока сообщений, подготавливается для расчетов путем удаления повторяющихся знаков, пробелов и другой необрабатываемой алфавитом информации.

Блок вычислений отвечает на вопросы: какова мощность, сколько символов обработано, каков объем и вес.

//info: http://urok-ikt.narod.ru/pages/cl-08\_konsp01.html

**using** System**;**

**using** System**.**Drawing**;**

**using** System**.**Linq**;**

**using** ZedGraph**;**

**namespace** Infoboard

**{**

/// <summary>

/// Храним наборы алфавитов

/// </summary>

class clAlfavit

**{**

**public** clAlfavit**(**string lng**)**

**{**

**if** **(**lng**.**Equals**(**"ru"**))**

\_flagLng **=** lng**;**

**if** **(**lng**.**Equals**(**"en"**))**

\_flagLng **=** lng**;**

**if** **(**lng**.**Equals**(**"blend"**))**

\_flagLng **=** lng**;**

**}**

**private** string \_flagLng **=** "-1"**;**

/// <summary>

/// Запрос мощьности алфавита

/// </summary>

**internal** int N**()**

**{**

**if** **(**\_flagLng**.**Equals**(**"ru"**))**

**return** 33**;**

**if** **(**\_flagLng**.**Equals**(**"en"**))**

**return** 26**;**

**if** **(**\_flagLng**.**Equals**(**"blend"**))**

**return** \_blendN**;**

**return** **-**1**;**

**}**

/// <summary>

/// Запрос алфавит-строки

/// </summary>

**internal** string StrAlf**()**

**{**

**if** **(**\_flagLng**.**Equals**(**"ru"**))**

**return** \_ru**;**

**if** **(**\_flagLng**.**Equals**(**"en"**))**

**return** \_en**;**

**return** "-1"**;**

**}**

**private** string \_ru **=** "а, б, в, г, д, е, ё, ж, з, и, й, к, л, м, н, о, п, р, с, т, у, ф, х, ц, ч, ш, щ, ъ, ы, ь, э, ю, я"**;**//33

**private** string \_en **=** "A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z"**;**//26

/// <summary>

/// принять введеную строку и выдать информацию

/// </summary>

**internal** void Analiz**(**string textin**)**

**{**

//разобрать строку на пробел, символы алфавита

string**[]** tmp **=** textin**.**Split**(**' '**);**

int lenTmp **=** tmp**.**Length**;**

//сравнить tmp с алфавитом, итогом есть К

double K **=** otborK**(**tmp**);**

//i=(Ln(N)/Ln(2))

double i **=** **(**Math**.**Log10**(this.**N**())/**Math**.**Log10**(**2.0**));**

InfoiMsg **=** Math**.**Round**(**i**,** 2**);**

//V=K\*i

//K - колич символов в сообщении

double V **=** K **\*** i**;**// Информационный объем V сообщения

InfoVMsg **=** Math**.**Round**(**V**,** 2**);**

**}**

**private** string \_blend**;**

**private** int \_blendN **=** **-**1**;**

/// <summary>

/// Свой алфавитЪ разбанковать

/// </summary>

**internal** void Razbor**(**string textinAlf**)**

**{**

//textinAlf = textinAlf.Remove((textinAlf.LastIndexOf(' ')), 1);

textinAlf **=** textinAlf**.**Replace**(**" "**,** ""**);**

var Massiv **=** **new** string**[**textinAlf**.**Length **+** 1**];**

int i **=** 0**,** j **=** 0**;**

**foreach** **(**char ts **in** textinAlf**)**

**{**

Massiv**[**j**]** **=** ts**+**""**;**

j**++;**

**for** **(**int e **=** 0**;** e **<** i**;** e**++)**

**if** **(**ts**.**ToString**().**CompareTo**(**Massiv**[**e**])** **==** 0**)**

Massiv**[**e**]** **=** **null;**//Занулим все совпадения

i**++;**

**}**

textinAlf **=** Massiv**.**Where**(**t **=>** **!**string**.**IsNullOrEmpty**(**t**)).**Aggregate**(**""**,** **(**current**,** t**)** **=>** current **+** t**);**

\_blend **=** textinAlf**;**

\_blendN **=** textinAlf**.**Length**;**

**}**

/// <summary>

/// V

/// </summary>

**internal** double InfoVMsg **{** get**;** **private** set**;** **}**

/// <summary>

/// i

/// </summary>

**internal** double InfoiMsg **{** get**;** **private** set**;** **}**

/// <summary>

/// Сравнить строку и алф

/// </summary>

**private** double otborK**(**string**[]** tmp**)**

**{**

int errSmbl **=** 0**,** iterSymb **=** 0**;**

**switch** **(**\_flagLng**)**

**{**

**case** "ru"**:**

string**[]** rutmp **=** \_ru**.**Split**(**','**,** ' '**);**

//слово

**for** **(**int i **=** 0**;** i **<** tmp**.**Length**;** i**++)**

**{**//символ слова

**for** **(**int j **=** 0**;** j **<** tmp**[**i**].**Length**;** j**++)**

**{**

iterSymb**++;**//количество символов в фразе

char ch **=** tmp**[**i**][**j**];**

**foreach** **(**var sRu **in** rutmp**)**

**{**

**if** **(**sRu **!=** "" **||** ch **!=** ' '**)**

**if** **(**ch**.**ToString**().**Equals**(**sRu**,** StringComparison**.**CurrentCultureIgnoreCase**))**

errSmbl**++;**

**}**

**}**

**}**

**break;**

**case** "en"**:**

string**[]** entmp **=** \_en**.**Split**(**','**,** ' '**);**

//слово

**for** **(**int i **=** 0**;** i **<** tmp**.**Length**;** i**++)**

**{**//символ слова

**for** **(**int j **=** 0**;** j **<** tmp**[**i**].**Length**;** j**++)**

**{**

iterSymb**++;**//количество символов в фразе

char ch **=** tmp**[**i**][**j**];**

**foreach** **(**var sEn **in** entmp**)**

**{**

**if** **(**sEn **!=** "" **||** ch **!=** ' '**)**

**if** **(**ch**.**ToString**().**Equals**(**sEn**,** StringComparison**.**CurrentCultureIgnoreCase**))**

errSmbl**++;**

**}**

**}**

**}**

**break;**

**case** "blend"**:**

//слово

**for** **(**int i **=** 0**;** i **<** tmp**.**Length**;** i**++)**

**{**//символ слова

**for** **(**int j **=** 0**;** j **<** tmp**[**i**].**Length**;** j**++)**

**{**

iterSymb**++;**//количество символов в фразе

char ch **=** tmp**[**i**][**j**];**

**foreach** **(**var sEn **in** \_blend**)**

**{**

**if** **(**sEn **!=** ' ' **||** ch **!=** ' '**)**

**if** **(**ch **==** sEn**)**

errSmbl**++;**

**}**

**}**

**}**

**break;**

**}**

//колич\_Нераспознаных = Колич\_символов - Колич\_Распознан

double result **=** **(**iterSymb **-** errSmbl**);**

//процент неопознаных символов:X//проц опознаных:Y

var symb **=** **new** PointD**((**result**\***100.0**)/**iterSymb**,** 100 **-** **((**result**\***100.0**)/**iterSymb**));**

PieGraffik **=** symb**;**

**return** **(**iterSymb**-**result**);**

**}**

/// <summary>

/// Числа для графики распознания символов

/// </summary>

**internal** PointD PieGraffik**{**get**;** **private** set**;** **}**

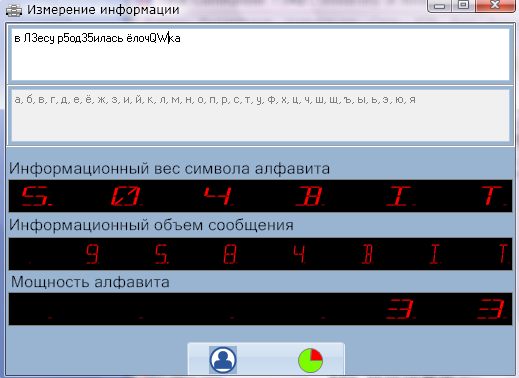
**}**

**}**

***Выбор языка программирования***

Язык C#, самый легкий и больше всего изученный мною.

***Тестирование программы***

Итого: N(ru) = 33, i = 5.04, V = 95.8;

Проверим результат, посчитав:

K – количество опознаных символов в сообщении.