МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ

ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных Технологий

Кафедра Информационных систем и технологий

Специальность 1-40 01 05 03 Информационные системы и технологии Направление специальности 1-40 01 05 03 Информационные системы и технологии

**ОТЧЁТ К ЛАБОРАТОРНОЙ РАБОТЕ №3:**

по дисциплине «Защита и надёжность информационных систем»

Тема «Элементы теории информации. Информативность данных в различных кодировках»

Исполнитель

студентка 3 курса группы 1 Кашперко Василиса Сергеевна

(Ф.И.О.)

Руководитель работы ассистент Савельева М. Г.

(учен. степень, звание, должность, подпись, Ф.И.О.)

**Цель:** приобретение практических навыков трансформации данных и сопоставление энтропийных свойств используемых при этом алфавитов.

**Задачи:**

1. Закрепить теоретические знания по взаимной конвертации данных, представленных в кодах ASCII и base64.
2. Разработать приложение для конвертации произвольного документа в формат base64 и обратно.
3. Исследовать энтропийные характеристики используемых в конвертерах алфавитов.
4. Изучить особенности практической реализации операции XOR над данными, представленными в разных форматах.
5. Результаты выполнения лабораторной работы оформить в виде описания разработанного приложения, методики выполнения экспериментов с использованием приложения и результатов эксперимента.

**Задания к выполнению:**

1. Создать собственное приложение или воспользоваться *Base64-онлайн-кодировщиком*, с помощью которого конвертировать произвольный документ (а) на латинице (можно использовать документ из лабораторной работы № 2) в документ (б) формата base64. В качестве входных данных можно использовать указанный преподавателем вариант из списка:

• входные параметры;

• текстовый файл (\*.txt);

• документ Word (\*.doc);

• документ Word (\*.docx);

• документ PowerPoint (\*.ppt, \*.pptx);

• архив (\*.zip);

• текстовая строка;

• случайное число (от 999999);

• PDF-файл;

• архив (\*.rar);

• архив (\*.7z).

2. С помощью приложения, созданного в лабораторной работе №2, получить распределение частотных свойств алфавитов по документам (а) и (б). Вычислить энтропию Хартли и Шеннона, а также избыточность алфавитов. Объяснить полученный результат.

3. Написать функцию, которая принимает в качестве аргументов два буфера (*а* и *b*) одинакового размера и возвращает XOR (собственная фамилия (*а*) и имя (*b*); при разной длине меньшую дополнить нулями). Входные аргументы представлять: 1) в кодах ASCII; 2) в кодах base64. Что будет результатом операции *а*XOR*b*XOR*b*?

**Теоретические сведения**

**Избыточностью алфавита** называют уменьшение информационной нагрузки на один символ вследствие разной вероятности и взаимозависимости появления его символов в сообщениях.

Дополнительная избыточность обусловлена переносом сообщения из одной среды в другую или, иначе говоря, кодированием символов исходного алфавита.

*Проблемы с восьмибитной кодировкой (ASCII):*

* «обрезание восьмого бита» в системе электронной почты;
* 256 различных значений позволили уместить в одной кодовой таблице и общепринятые символы и символы кириллицы;
* большинство ПО приспособлены для семибитных кодировок => почтовый сервер при передаче письма обнулял старшие биты в каждом байте сообщения.

*Решение:*

Одним из решений проблемы стала кодировка (а точнее – алгоритм) **base64**. В PGP алгоритм base64 используется для кодирования бинарных данных.

**Кодирование base64 разработано для** представления произвольных последовательностей октетов в форме, позволяющей использовать строчные и прописные буквы.

Используется 65-символьное подмножество набора символов US-ASCII, обеспечивающее представление одним печатным символом 6 битов данных (дополнительный 65-й символ используется для обозначения функции специальной обработки).

Еще раз обратимся к процессу кодировки. Как было выше установлено, каждые 6 битов буфера, начиная с самых старших, используются как индексы строки «ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/», и ее символы, на которые указывают индексы, помещаются в выходную строку.

*Если кодируются только один или два байта, в результате получаются только первые два или три символа строки, а выходная строка дополняется двумя или одним символами «=». Это предотвращает добавление дополнительных битов к восстановленным данным.*

Поскольку входная информация base64 всегда включает целое число октетов, возможны лишь перечисленные ниже случаи:

• размер финального блока кодирования на входе кратен 24 битам, кодированный результат будет содержать целое число 4-символьных групп без заполнения символами «=»;

• размер финального блока кодирования на входе составляет 8 битов, выходной блок будет представлять 2 символа, дополненные последовательностью из двух символов заполнения «==»;

• размер финального блока кодирования на входе составляет 16 битов, выходной блок будет представлять 3 символа, дополненные символом заполнения «=».

**Ход работы**

**Задание 1.** Для конвертирования строки и текста в base64 разработана функция, представленная в Листинге 1.

public static string ConvertToBase64(string data)

{

var binaryArray = data.Select(x => Convert.ToString(x, 2).PadLeft(8, '0'));

var count = binaryArray.Count();

var add = count % 3 == 1 ? "==" : count % 3 == 2 ? "=" : "";

var bytes = string.Join("", binaryArray);

var bytesCount = bytes.Count();

var division = bytesCount % 6;

var myList = Enumerable.Range(0, bytesCount / 6).Select(x => bytes.Substring(x \* 6, 6)).ToList();

if (division != 0)

{

myList.Add(bytes.Substring(bytesCount / 6 \* 6, division).PadRight(6, '0'));

}

return (string.Join("", myList.Select(x => base64Table[Convert.ToByte(x, 2)])) + add);

}

Листинг 1 – Разработка функции конвертирования в base64

Входными данными являются два файла с расширением .txt. Результат перевода в Base64 представлен на рисунке 1.

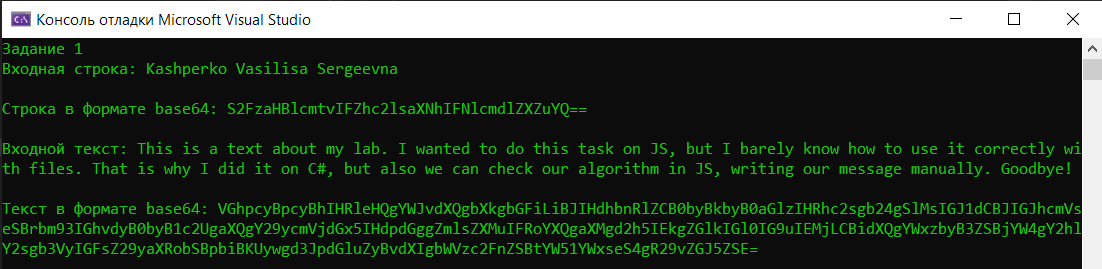


Рисунок 1 – Результат конвертирования в base64

Для проверки полученный результатов мною была разработана программа на JavaScript, представленная в Листинге 2.

const input1 ='Kashperko Vasilisa Sergeevna';

const input2 = 'This is a text about my lab. I wanted to do this task on JS, but I barely know how to use it correctly with files. That is why I did it on C#, but also we can check our algorithm in JS, writing our message manually. Goodbye!';

const buffer1 = new Buffer.from(input1);

const buffer2 = new Buffer.from(input2);

console.log(input1 + '\n');

function asciiToBin(input)

{

var result = "";

var count = 0;

for (var i = 0; i < input.length; i++)

{

var bin = input[i].toString(2);

result += Array(8 - bin.length + 1).join("0") + bin;

}

return result;

}

function check(input)

{

var count = 0;

for (var i = 0; i < input.length; i++)

{

count++;

}

return count/8;

}

const binary = asciiToBin(buffer1).toString();

console.log(binary);

console.log('Check: '+check(binary) + '\n');

const base64Converted = buffer1.toString('base64');

console.log(base64Converted);

const original = buffer1.toString('ascii');

console.log(original+ '\n');

console.log(input2 + '\n');

const binaryText = asciiToBin(buffer2).toString();

console.log(binaryText);

console.log('Check: ' + check(binaryText) + '\n');

const base64ConvertedText = buffer2.toString('base64');

console.log(base64ConvertedText);

const originalText = buffer2.toString('ascii');

console.log(originalText+ '\n');

Листинг 2 – Разработка программы конвертирования в base64 на JavaScript

Результат конвертирования в base64 представлен на рисунке 2а и рисунке 2б.

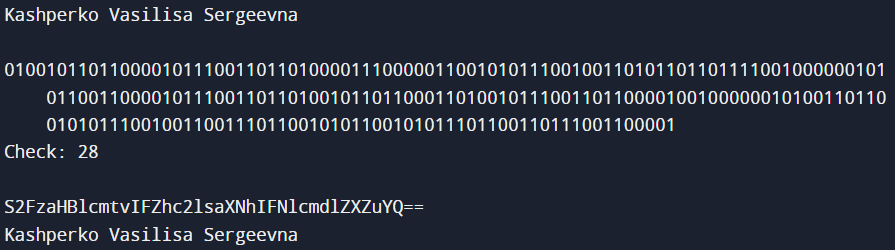


Рисунок 2а – Результат перевода в base64

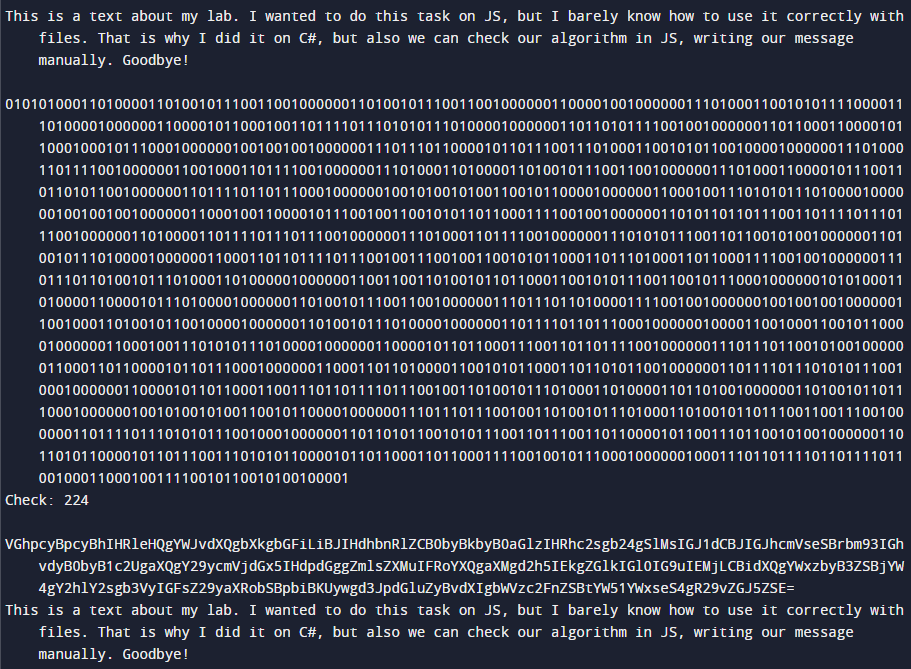


Рисунок 2б – Результат перевода в base64

В ходе проверки результаты остались идентичны, следовательно, разработанная нами ранее функция на языке C# работает корректно.

**Задание 2.** Используя функции подчёта энтропии по Шеннону и Хартли, рассчитываем соответствующие энтропии, а также избыточность алфавитов, которая рассчитывается по формуле 1.



Результат выполнения программы приведен на рисунке 2.

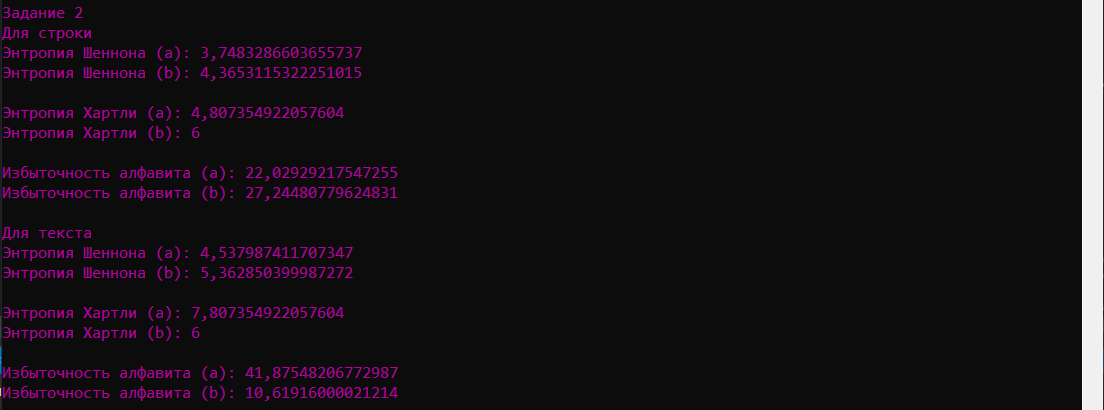


Рисунок 2 – Результат перевода в base64

**Задание 3.** Далее необходимо написать функцию, которая принимает два аргумента (а и b, где а – фамилия, а b – имя) и возвращает XOR. Пример разработанной функции приведен в Листинге 3, результат выполнения на рисунке 3. Бинарный конвертер и конвертер в base64 приведены в Листинге 2.

function XORFun(string1, string2)

{

var resultXOR = "";

for (var i = 0; i < string1.length; i++)

{

if (string1[i] == string2[i])

{

resultXOR += "0";

}

else

{

resultXOR += "1";

}

}

return resultXOR;

}

const input1 ='Kashperko';

const buffer1 = new Buffer.from(input1);

const binary1 = asciiToBin(buffer1).toString();

const input2 ='Vasiliska';

const buffer2 = new Buffer.from(input2);

const binary2 = asciiToBin(buffer2).toString();

console.log(binary1 + '\n');

console.log(binary2 + '\n');

var XORResultBin1 = XORFun(binary1,binary2);

console.log('XOR имени и фамилии ASCII binary: ' + XORResultBin1 + '\n');

console.log(check(XORResultBin1) + '\n');

const base64Converted1 = buffer1.toString('base64');

console.log(base64Converted1);

const base64Converted2 = buffer2.toString('base64');

console.log(base64Converted2 + '\n');

var base64Bin1 = asciiToBin(base64Converted1).toString();

var base64Bin2 = asciiToBin(base64Converted2).toString();

var XORResultBin2 = XORFun(base64Bin1,base64Bin2);

console.log('XOR имени и фамилии base64 binary: ' + XORResultBin2 + '\n');

var XORStarASCIIBin = XORFun(XORResultBin1, binary2);

var XORStarBse64Bin = XORFun(XORResultBin2, base64Bin2);

Листинг 3 – Реализация программы с выполнением оператора XOR на JavaScript

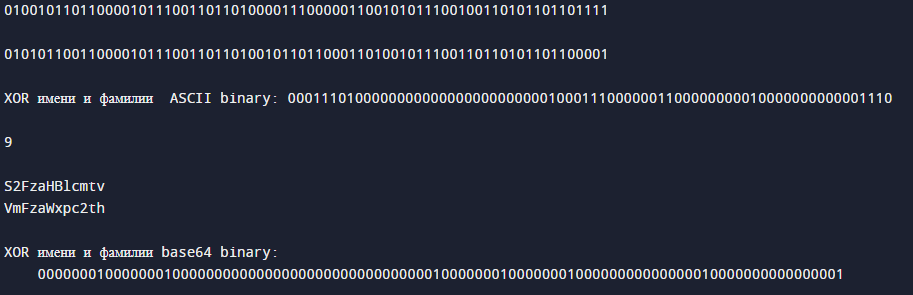


Рисунок 3 – Результат выполнения функции XOR

***Задание 3\*.*** Что будет результатом операции aXORbXORb?

Блок кода программы выполнения и результат выполнения операции приведены на рисунках 4 и 5 соответственно.



Рисунок 4 – Блок кода программы выполнения операции

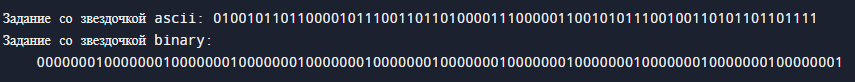


Рисунок 5 – Результат выполнения операции aXORbXORb

**Вывод:** в ходе лабораторной работы был изучен алгоритм и особенности кодирования base64. Base64 — стандарт кодирования двоичных данных при помощи только 64 символов ASCII. А также приобретены навыки по переводу текстов на основе естественного языка в base64 и использование оператора XOR.