Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

**Элементы теории информации. Информативность данных**

**в различных кодировках**

Студент: Рубашек А. А.

ФИТ 3 курс 5 группа

Преподаватель:

Савельева М. Г.

1. **Цель работы**

Создать приложение для расчета и анализа параметров и информативных характеристик дискретных ИС, с помощью которого:

а) конвертировать произвольный документ (а) на латинице (можно использовать документ из лабораторной работы № 1) в документ (б) формата base64.

б) получить распределение частотных свойств алфавитов по документам (а) и (б). Вычислить энтропию Хартли и Шеннона, а также избыточность алфавитов. Объяснить полученный результат.

в) 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.

**2 Теоретические сведения**

Энтропия максимальна при равномерном появлении букв на любом месте сообщения. Для характеристики источника сообщений с различным алфавитом представляет интерес сравнение фактической энтропии источника с максимально возможной. В этом смысле введено понятие избыточности источника сообщений, или избыточности алфавита.

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

В наиболее общем виде избыточность алфавита *R* можно оценить отношением энтропии по Хартли и по Шеннону:

При выполнении предыдущей работы мы убедились, что формально одно и то же сообщение, но представленное на основе алфавита русского (белорусского, английского или иного) языка – с одной стороны, и представленное в кодах ASCII – с другой, будут характеризоваться различным количеством содержащейся в них информации. Эта дополнительная избыточность обусловлена переносом сообщения из одной среды в другую или, иначе говоря, кодированием символов исходного алфавита.

Уже созданное к тому времени и работающее программное обеспечение зачастую было приспособлено для семибитных кодировок, что приводило, например, к тому, что почтовый сервер при передаче письма обнулял старшие биты в каждом байте сообщения. 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).

Процесс кодирования представляет группу из 24 последовательных битов в форме строки из 4 символов. Обработка выполняется слева направо, а 24-битная исходная группа образуется конкатенацией трех 8-битных групп (байтов). Данные 24 бита после Элементы теории информации. Информативность данных в различных кодировках 29 этого трактуются как 4 сцепленных группы по 6 битов, каждая из которых транслируется в один символ алфавита base64.

Еще раз обратимся к процессу кодировки. Как было выше установлено, каждые 6 битов буфера, начиная с самых старших, используются как индексы строки «ABCDEFGHIJKLMNOPQRSTUV WXYZabcdefghijklmnopqrstuvwxyz0123456789+/», 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.

Процесс повторяется над оставшимися входными данными. Такая обработка выполняется в тех случаях, когда последняя группа входных данных содержит меньше 24 битов. Кодируемое значение всегда завершается полным квантом кодирования.

Если на входе доступно менее 24 битов, входная группа дополняется (справа) нулями до формирования целого числа 6-битных групп. Заполнение в конце данных осуществляется как раз с использованием символа «=». 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 «==»;

1. **Ход работы**

Для выполнения первого задания был выбран язык Python. Это приложение читает текстовый файл input.txt. Читает содержимое и конвертирует данный текст на латинице в формат base64. Сохраняет его в другом файле output\_base64. Код предоставлен на рисунке 3.1

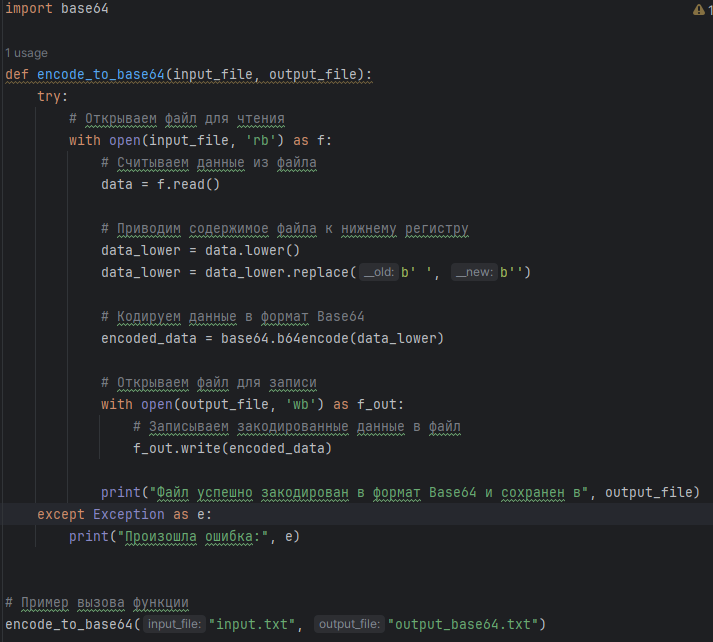


Рисунок 3.1 – Код приложения

После успешной обработки данных, приложение из лабораторной 1 считывает оба файла и начинает: получать распределение частотных свойств алфавитов, вычислять энтропию Хартли и Шеннона. Код считывания данных в приложении для первой ла3бораторной работы находится на рисунке 3.2

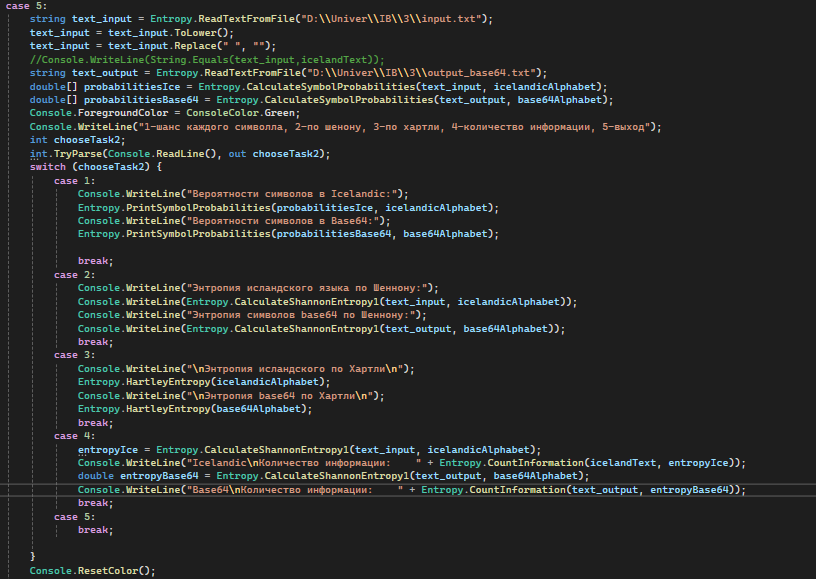


Рисунок 3.2 – Дополнительный код для приложения из лабораторной работы №1

В результате получаем такие результаты расчёта энтропии исландского и base64 алфавитов для исходных текстов, которые представлены на рисунках 3.3 и 3.4

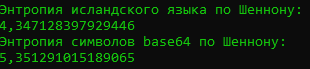


Рисунок 3.3 – Энтропия Шеннона



Рисунок 3.4 – Энтропия Хартли

Частота появления каждого символа в каждом из алфавитов предоставлена на рисунках 3.5 и 3.6

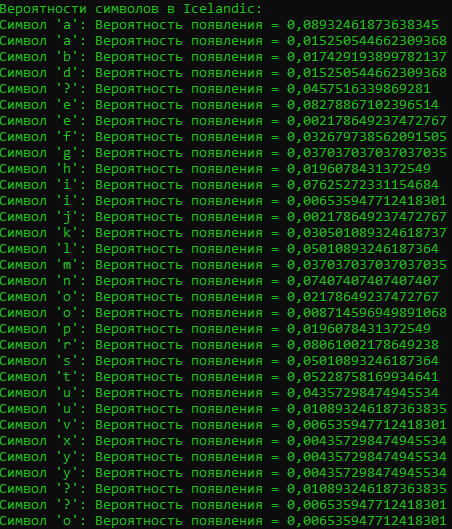


Рисунок 3.5 – Частота появления для Исландского

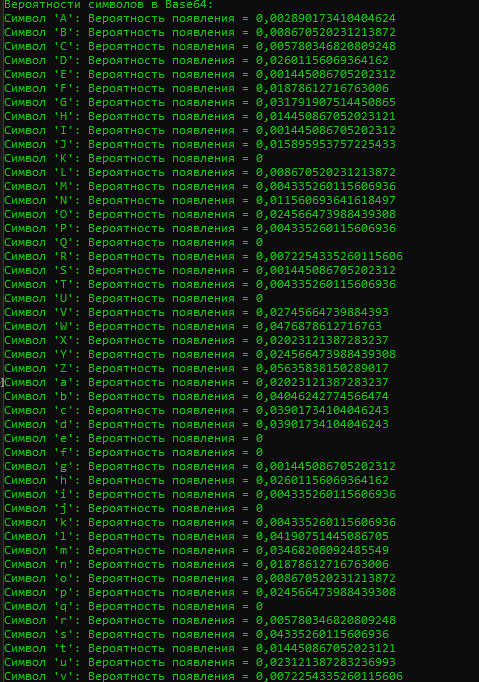


Рисунок 3.6 – Частота появления для Base64

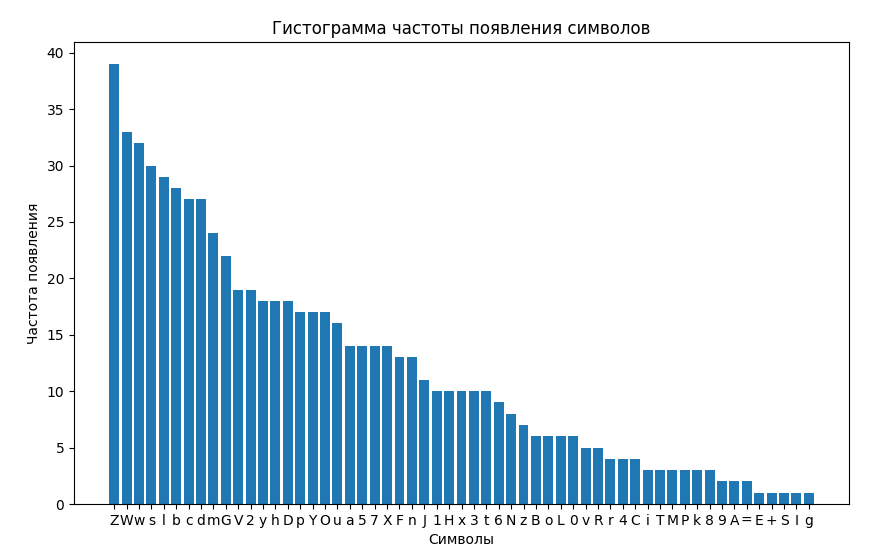


Рисунок 3.7 – Гистограмма частоты появления символов Base64

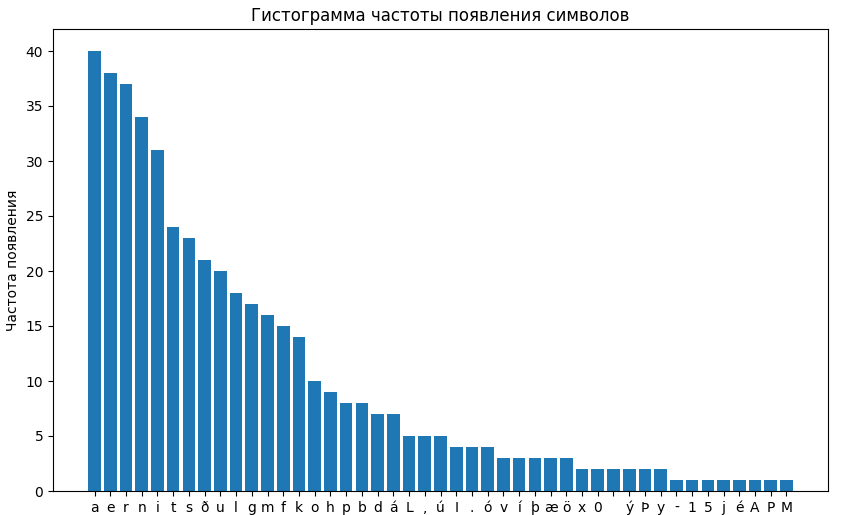


Рисунок 3.8 – Гистограмма частоты появления символов Исландского

****

Рисунок 3.7 – Количество информации в тексте



Рисунок 3.8 – Избыточность алфавитов

Так же была разработана функция xor\_buffers принимающая два текстовых параметра(имя и фамилия) которая добавляет «нули» при необходимости, а также применяет XOR к нашим параметрам. Функция приведена на рисунке 3.9

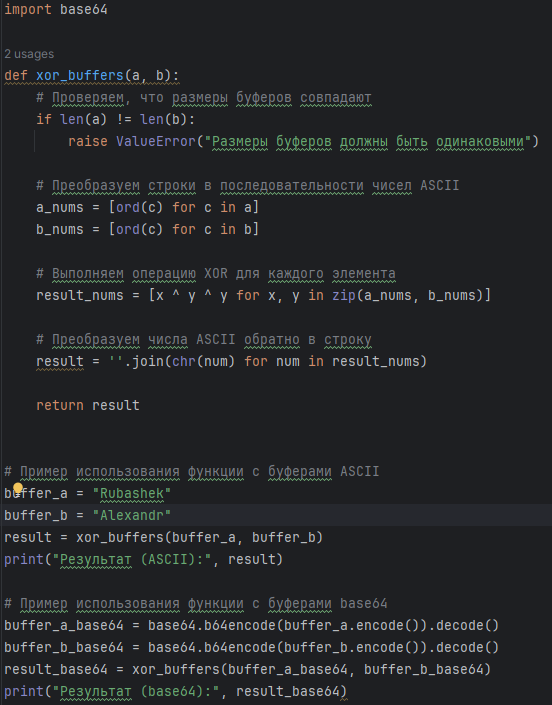


Рисунок 3.9 – Функция для расчёта *a*XOR*b*XOR*b*

В результате получаем следующий вывод:



Рисунок 3.10 – Результат работы функции xor\_buffers

**Вывод**

В ходе лабораторной работы были приобретены практические навыки работы с кодированием и обработкой данных различных форматов. В частности, было разработано приложение, позволяющее преобразовывать произвольные документы на латинице в формат Base64.

Далее, в рамках второй части лабораторной работы, был произведен анализ частотных свойств алфавитов в документах и вычислена их энтропия по методам Хартли и Шеннона. Полученные результаты дали представление о разнообразии символов в алфавитах и степени их "предсказуемости". Кроме того, вычисленная избыточность алфавитов позволила оценить, насколько информативен каждый из алфавитов в рамках рассматриваемого контекста.

Наконец, третья часть лабораторной работы включала разработку функции для выполнения операции XOR между двумя буферами данных одинакового размера. Эта операция может быть применена как к данным в кодах ASCII, так и к данным в кодах Base64, предоставляя возможность для защиты и обработки информации в различных форматах.