Учреждение образования «БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ  
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий

[Кафедра информационных](https://www.belstu.by/fakultety/fit/vm) систем и технологий

Специальность 1-40 01 01 Программное обеспечение информационных технологий

**Отчёт по лабораторной работе №2**

по дисциплине Информационная безопасность

Тема: Элементы теории информации. Параметры и характеристики дискретных информационных систем

Студент: Станчик М. А.

ФИТ 3 курс 4 группа

Преподаватель: Савельева М. Г.

Минск, 2025

**Лабораторная работа №1**

**Цель:** приобретение практических навыков расчета и анализа параметров и информативных характеристик дискретных ИС.

**Задачи:**

1. Закрепить теоретические знания по основам теории информации.

2. Разработать приложение для расчета и анализа параметров и информативных характеристик дискретных ИС.

3. Результаты выполнения лабораторной работы оформить в виде описания разработанного приложения, методики выполнения экспериментов с использованием приложения и результатов эксперимента.

# Теоретические сведения

Передача информации (данных) осуществляется между двумя абонентами, называемыми источником сообщения (ИcС) и получателем сообщения (ПС). Третьим элементом информационной системы является канал (среда) передачи, связывающий ИсС и ПС.

ИсС и ПС обмениваются информацией в технических системах в виде сигналов, сформированных на основе определенного алфавита. Характеристикой алфавита является его мощность, *N* – количество символов, на основе которых формируется сообщение. Например, мощность английского алфавита – 26 символов, русского – 33 символа, мощность алфавита, на основе которого функционируют и взаимодействуют между собой компьютеры, составляет 2 символа (0 и 1).

В произвольном сообщении символы алфавита могут появляться с различной вероятностью. Если длина сообщения достаточно велика, то статистический анализ этого сообщения позволит получить вероятностные характеристики данного алфавита. Очевидно, что отличные символы в произвольном сообщении (особенно при *N>*2) появляются с различной вероятностью.

Информационной характеристикой алфавита (источника сообщений на основе этого алфавита) является энтропия.

Этот термин применительно к техническим системам был введен К. Шенноном и Р. Хартли.

Энтропию алфавита *А*{} по Шеннону рассчитывают по следующей формуле:

где *P*()– вероятность *P*(ξ=)*;* – элемент алфавита,.

С физической точки зрения энтропия показывает, какое количество информации приходится в среднем на один символ алфавита. Частным случаем энтропии Шеннона является энтропия Хартли. Дополнительным условием при этом является то, что все вероятности одинаковы и постоянны для всех символов алфавита.

С учетом этого формулу энтропии Шеннона можно преобразовать к виду:

*H*​(*A*)=​*N*

Сообщение *M*, которое состоит из *n* символов, должно характеризоваться определенным количеством информации *I(M)*:

*I*(*M*)=*H*(*A*)×*n*

Здесь *Н(А)* – энтропия алфавита с соответствующим распределением вероятностей *р()*.

Двоичный канал передачи информации является дискретным – он основан на алфавите, состоящем из двух символов: 0 и 1 – *A* {0,1}.

Если вероятность ошибки в ДСК отлична от 0 (*р* > 0), переданное сообщение может содержать ошибки: *Хk* ≠ *Yk*. Количество информации в таком сообщении при его передаче по ДСК будет определяться не энтропией двоичного алфавита, а эффективной энтропией *He(A)* алфавита или пропускной способностью канала:

,

где *H(Y|X)* – условная энтропия:

*H*(*Y*|*X*) = -*p* \* *p* – *q* \* *q*

# Ход работы

Для вычисления энтропии было разработано несколько функций.

Метод readFile принимает имя файла и его путь. Он выполняет чтение файла и запись его содержимого в строку. Код представлен в листинге 2.1.

public String readFile (String filePath) {

String content = "";

try {

content = new String(Files.readAllBytes(Paths.get(filePath)), "UTF-8");

}

catch (IOException e) {

e.printStackTrace();

}

return content;

}

Листинг 2.1 – Метод для записи содержимого текстового файла в строку

Метод calculateProbabilities принимает строку, которая содержит символы. Предназначен для вычисления вероятностей появления каждого уникального символа в заданной строке. Метод возвращает отображение, где ключами являются уникальные символы из строки, а значениями - соответствующие вероятности этих символов. Вероятность символа вычисляется как отношение количества его вхождений к общему количеству символов в строке. Код метода представлен в листинге 2.2.

public Map<Character, Double> calculateProbabilities(String text) {

Map<Character, Integer> frequencyMap = new HashMap<>();

int totalCount = 0;

for (char c : text.toCharArray()) {

frequencyMap.put(c, frequencyMap.getOrDefault(c, 0) + 1);

totalCount++;

}

Map<Character, Double> probabilities = new HashMap<>();

for (Map.Entry<Character, Integer> entry : frequencyMap.entrySet()) {

char character = entry.getKey();

int frequency = entry.getValue();

double probability = (double) frequency / totalCount;

probabilities.put(character, probability);

}

return probabilities;

}

Листинг 2.2 – Метод для вычисления вероятностей появления каждого уникального символа в строке

Метод calculateEntropy нужен для вычисления энтропии текста. Метод принимает отображение, где ключами являются уникальные символы из строки, а значениями - соответствующие вероятности этих символов. Этотпараметр должен быть получен с помощью метода calculateProbabilities. Код метода представлен в листинге 2.3.

public double calculateEntropy(Map<Character, Double> charProbabilities) {

double entropy = 0.0;

for (double charProbability : charProbabilities.values()) {

if (charProbability > 0) {

entropy -= charProbability \* (Math.log(charProbability) / Math.log(2));

}

}

return entropy;

}

Листинг 2.3 – Метод для вычисления вероятностей появления каждого уникального символа в строке

Метод saveFrequenciesToExcel предназначен для сохранения частот символов в Excel файл. Принимает отображение, где ключами являются уникальные символы, а значениями - соответствующие частоты этих символов, и имя файла (включая путь), в который будет сохранён Excel файл с частотами символов. Код метода представлен в листинге 2.4.

public void saveFrequenciesToExcel(Map<Character, Double> frequencies, String filename) {

Workbook workbook = new XSSFWorkbook();

Sheet sheet = workbook.createSheet("Frequencies");

Row headerRow = sheet.createRow(0);

headerRow.createCell(0).setCellValue("Символ");

headerRow.createCell(1).setCellValue("Частота");

int rowNum = 1;

for (Map.Entry<Character, Double> entry : frequencies.entrySet()) {

Row row = sheet.createRow(rowNum++);

row.createCell(0).setCellValue(entry.getKey().toString());

row.createCell(1).setCellValue(entry.getValue());

}

try (FileOutputStream fileOut = new FileOutputStream(filename)) {

workbook.write(fileOut);

} catch (Exception e) {

e.printStackTrace();

}

System.out.println("Частота символов сохранена в " + filename);

}

Листинг 2.4 – Метод для сохранения частот символов в Excel файл

Метод createDataset предназначен для создания набора данных, который будет использоваться для построения гистограммы с помощью библиотеки JFreeChart. Принимает отображение, которое вычисляется с помощью метода calculateProbabilities. Метод возвращает заполненный объект DefaultCategoryDataset. Код метода представлен в листинге 2.5.

private CategoryDataset createDataset(Map<Character, Double> frequencies) {

DefaultCategoryDataset dataset = new DefaultCategoryDataset();

for (Map.Entry<Character, Double> entry : frequencies.entrySet()) {

dataset.addValue(entry.getValue(), "Частота", entry.getKey().toString());

}

return dataset;

}

Листинг 2.5 – Метод для создания набора данных

Метод saveHistogramToExcel предназначен для создания и сохранения гистограммы частот символов в Excel файл. Метод принимает отображение и путь к файлу Excel, ничего не возвращает. Код метода представлен в листинге 2.6.

public void saveHistogramToExcel(Map<Character, Double> frequencies, String excelFilePath) {

CategoryDataset dataset = createDataset(frequencies);

JFreeChart histogram = ChartFactory.createBarChart(

"Частоты символов",

"Символ",

"Частота",

dataset

);

String imagePath = "histogram.png";

try {

ChartUtils.saveChartAsPNG(new File(imagePath), histogram, 800, 600);

System.out.println("Гистограмма сохранена в " + imagePath);

} catch (IOException e) {

e.printStackTrace();

}

insertImageToExcel(imagePath, excelFilePath);

}

Листинг 2.6 – Метод для создания и сохранения гистограммы частот символов в Excel файл

На основе данных, полученных при работе данных методов, были построены гистограммы частот алфавитов. Гистограмма частот появления символов сербского алфавита представлена на рисунке 2.1.

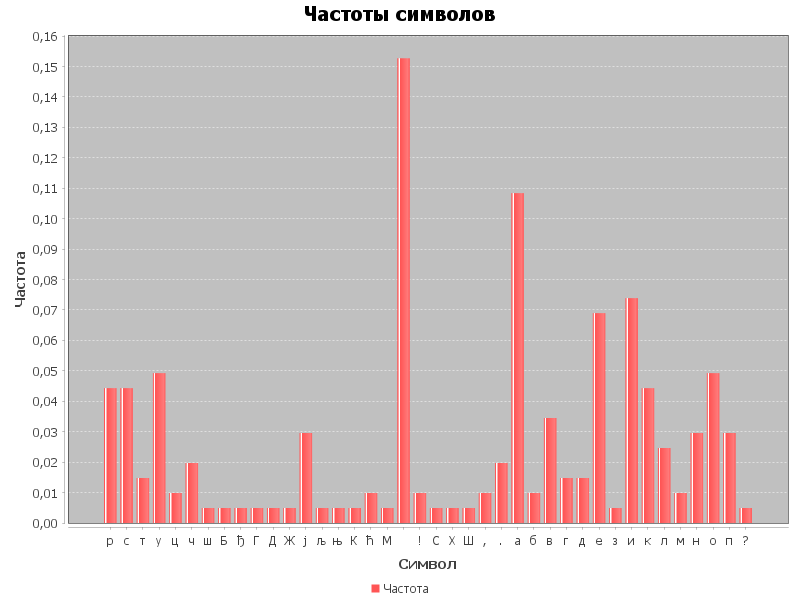


Рисунок 2.1 – Гистограмма частот для сербского алфавита

Гистограмма частот появления символов испанского алфавита представлена на рисунке 2.2.

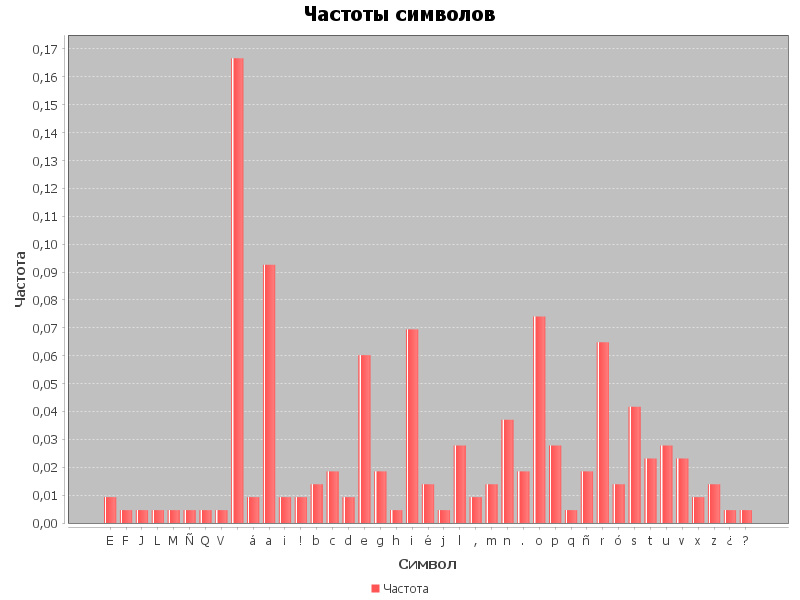


Рисунок 2.2 – Гистограмма частот для испанского алфавита

Результат работы метода для вычисления энтропии представлен на рисунке 2.3.

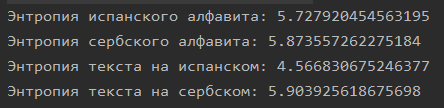


Рисунок 2.3 – Энтропия литовского и македонского алфавитов

Далее, нужно было подсчитать энтропию бинарного алфавита. Для этого, символы текста были представлены в бинарных кодах. Для этого, были разработаны методы countZerosAndOnes и calculateBinaryEntropy. Первый метод принимает строку символов и возвращает количество нулей и единиц в строке. Второй метод принимает результат работы первого метода и высчитывает бинарную энтропию. Код представлен в листинге 2.7.

public int[] countZerosAndOnes(String text) {

int countZero = 0;

int countOne = 0;

for (char c : text.toCharArray()) {

if (c == '0') {

countZero++;

} else if (c == '1') {

countOne++;

}

}

return new int[]{countZero, countOne};}

public double calculateBinaryEntropy(int countZero, int countOne) {

int totalCount = countZero + countOne;

if (totalCount == 0) {

return 0.0;

}

double p0 = (double) countZero / totalCount;

double p1 = (double) countOne / totalCount;

if (p0 == 0 || p1 == 0) {

return 0.0;

}

double entropy = - (p0 \* (Math.log(p0) / Math.log(2)) + p1 \* (Math.log(p1) / Math.log(2)));

return entropy;

}

Листинг 2.7 – Реализация методов countZerosAndOnes и calculateBinaryEntropy

Результат вычисления энтропии бинарного алфавита представлен на рисунке 2.4.



Рисунок 2.4 – Энтропия бинарного алфавита

В следующем задании было нужно подсчитать количество информации в сообщении, состоящем из собственных фамилии, имени и отчества (на основе исходного алфавита и в кодах ASCII). С помощью вычисленной энтропии и длины сообщения высчитывается информация сообщения. Результат представлен на рисунке 2.5.

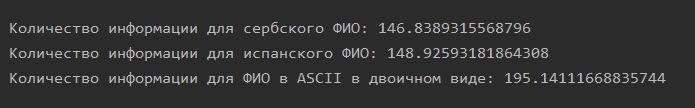


Рисунок 2.5 – Количество информации в сообщениях

Можно заметить, что количество информации в бинарной строке значительно превышает количество информации в строках на сербском и на испанском языках. Данное отличие можно объяснить избыточностью бинарного алфавита.

В последнем задании, требовалось найти количество информации при условии, что вероятность ошибочной передачи единичного бита сообщения составляет: 0,1; 0,5; 1,0. В таком случае, количество информации в таком сообщении при его передаче по ДСК будет определяться эффективной энтропией. Для определения эффективной энтропии был разработан метод getEffectiveEntropy, который принимает строку текста и вероятность ошибочной передачи единичного бита. Также был реализован метод isBinaryText для определения является ли текст бинарным, чтобы правильно отобразить количество информации при вероятности ошибки, равной 1. Еще был написан метод getInformationAmount для оценки информативности текста. Реализация методов представлена в листинге 2.8.

public double getEffectiveEntropy(String text, double p) {

double q = 1 - p;

if (isBinaryText(text) && (p == 0 || q == 0))

return 1;

if (!isBinaryText(text) && p == 1)

return 0;

return 1 - (-p \* (Math.log(p) / Math.log(2)) - q \* (Math.log(q) / Math.log(2)));

}

public double getInformationAmount(double entropy, String str, double effEntropy) {

double informationAmount = entropy \* str.length() \* effEntropy;

return Math.round(informationAmount \* 1000.0) / 1000.0;

}

public boolean isBinaryText(String text) {

for (char c : text.toCharArray()) {

if (c != '0' && c != '1' && c != ' ') {

return false;

}

}

return true;

}

Листинг 2.8 – Методы для определения информативности текста с учетом вероятности возникновения ошибки единичного бита сообщения

Результат работы метода представлен на рисунке 2.6.

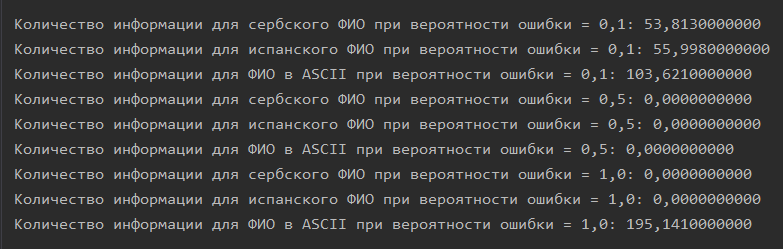


Рисунок 2.6 – Количество информации, если есть вероятность ошибки

Когда вероятность ошибочной передачи единичного бита равна 1, то количество информации будет примерно таким же, как и при вероятности ошибки равной 0, поскольку все биты исходного сообщения инвертированы, а значит при применении операции сложения по модулю 2 сообщение может быть восстановлено.

**Вывод:** в ходе выполнения данной лабораторной работы была определена энтропия литовского, македонского и бинарного алфавитов, а также было определено количество информации в сообщениях на основе этих алфавитов. Кроме того, было определено количество информации при условии, что есть вероятность ошибочной передачи единичного бита.