Институт информационных технологий

Кафедра: Математическое и программное обеспечение ЭВМ

Дисциплина: Теория Информации

ЛАБОРАТОРНАЯ РАБОТА № 4

Тема: Исследование свойств форматов сжатия графических и текстовых данных

Выполнил:

студент гр. 1ПИб-02-2оп-23

Кринкин Олег Алексеевич

Проверил:

доцент, к.т.н. Ганичева Оксана Георгиевна

ЗАДАНИЕ

**Задание 1**

1. Откройте графический редактор Paint. Загрузите в него многоцветный рисунок (например, С:\Windows\Облака.bmp).
2. Определите размер рисунка в пикселах. Оцените теоретический размер рисунка в 24-разрядной палитре (3 байта на точку) по формуле:

***S= M·N·3*** , где

S - размер файла с рисунком (байт)

M - ширина рисунка (точек)

N - высота рисунка (точек).

4. Сохраните рисунок под именем Рисунок\_1, назначив тип файла : 24-разрядный рисунок (.bmp). (в заранее созданную папку с именем Мои\_Рисунки)

5. Повторно сохраните рисунок, с именем Рисунок\_2 , назначив тип файла .gif. (в заранее созданную папку с именем Мои\_Рисунки)

При сохранении произойдет потеря определенной части графической информации.

6. Восстановите рисунок, загрузив его из ранее сохраненного файла Рисунок\_2.bmp, и вновь сохраните его под именем Рисунок\_3, назначив тип файла .jpeg. (в заранее созданную папку с именем Мои\_Рисунки)

7. Запустите программу Проводник. Откройте папку Мои\_исунки в режиме Таблица. Определите размеры файлов Рисунок\_1.bmp, Рисунок\_2.gif Рисунок\_3.jpeg. Определите коэффициенты сжатия файлов К, взяв отношения размеров файлов к теоретической величине, полученной расчетным путем в п.3 Результаты занесите в таблицу:

|  |  |  |
| --- | --- | --- |
| **Формат файла** | **Размер файла (Кбайт)** | **Степень сжатия (%)** |
| 24-разрядный рисунок .bmp |  |  |
| .gif |  |  |
| .jpeg |  |  |

8. Сделайте вывод о степени сжатия данных в разных форматах

9. В графическом редакторе Paint дайте команду создания нового документа. Убедитесь в том, что полотно имеет размер 640х480. Если это не так, измените его размер.

10. В качестве инструмента выберите Кисть. Задайте максимальный размер кисти. Поочередно используя 8-10 разных красок, грубо закрасьте полотно.

11. Сохраните рисунок под именем Test\_1 в формате 24-разрядный рисунок .bmp.

12. Сохраните рисунок под именем Test\_2 в формате .gif.

13. Восстановите рисунок из файла Test\_1.bmp

14. Сохраните рисунок под именем Test\_3 в формате .jpeg.

15. С помощью программы проводник определите размеры сохраненных файлов и заполните таблицу.

|  |  |  |
| --- | --- | --- |
| **Формат файла** | **Размер файла (Кбайт)** | **Степень сжатия (%)** |
| 24-разрядный рисунок .bmp |  |  |
| .gif |  |  |
| .jpeg |  |  |

16. Сделайте вывод о степени сжатия файлов разных форматах.

Какой формат графических данных из рассмотренных в работе наилучшим образом подходит для передачи цветного фотографического материала по каналам электронных сетей?

Какой формат графических файлов данных целесообразно использовать для передачи черно-белого фотографического материала по каналам связи?

Какой формат наиболее подходит для передачи рисунков, имеющих малое количество цветовых оттенков (до 256).

**Задание 2.** Выполнить сжатие информации методом RLE

Выполнить вручную кодирование сообщения методом RLE. В качестве исходной фразы взять текст из табл. 1. С помощью таблицы CP-1251 перевести символы заданной фразы в десятичные числа, а затем десятичные числа перевести в двоичные. Выполнить сжатие информации, вычислить контрольные суммы и коэффициент сжатия.

Табл. 1.

|  |  |  |  |
| --- | --- | --- | --- |
| Вар | Текст | Вар | Текст |
| 1 | Кредитка 2235555666122 | 17 | Ккккктттттттто тттттам? |
| 2 | Паспорт 25700000333215 | 18 | Длинношеее животное |
| 3 | ИНН 78888255555488856 | 19 | Урааааааааааааа в атаку |
| 4 | Пароль 177775556666612 | 20 | Долг 3255566667444444 |
| 5 | Пароль abcWWWWZZZq | 21 | Телефон 8904222211111 |
| 6 | Автомобиль 78999994441 | 22 | Ауууууууу заблудились |
| 7 | Алло это 4565555544488 | 23 | Свидетельство 22263333 |
| 8 | Удостоверение 265444111 | 24 | Возраст 1000000000 лет |
| 9 | Счет 95122244445333333 | 25 | Заработали 522211112 |
| 10 | Касса 1478885555233333 | 26 | До дембеля 60440000 с |
| 11 | Прошло 11100002 секунд | 27 | Кредитка 235556999922 |
| 12 | Пролетели 82223333352 м | 28 | ИНН 8825577777488856 |
| 13 | Вес 1597555553333331 кг | 29 | Шифр 159222666644444 |
| 14 | Цена 2598888666611 коп | 30 | Улов 98544477778555 кг |
| 15 | Мощность 3574444555 Вт | 31 | Пароль RRWQQQQ6666 |
| 16 | Выиграл 10000555 рублей | 32 | Пароль 778SSЫЫzzzzN |

**Задание 3.** Выполнить сжатие информации методом Шеннона-Фано

Используя фразу из табл. 1, построить код Шеннона-Фано и определить коэффициент и степень сжатия этим методом

**Задание 4**. Выполнить сжатие информации методом Хаффмана

Используя фразу из табл. 1, построить кодовое дерево и определить коэффициент и степень сжатия методом Хаффмана

**Задание 5.** Сделать общий вывод о степени сжатия исходного текста (фраза из табл.1) каждым методом.

ХОД РАБОТЫ

Задание 1.

1. Из интернета загружено изображение (рис. 1), которое затем открыто в программе MS Paint.



Рис. 1: Рабочее изображение

1. По формуле (1.1) определён теоретический размер рисунка, который составил .

где, – размер файла с рисунком (байт), – ширина рисунка (точки), – высота рисунка (точки).

1. Изображение сохранено в формате .bmp (точечный рисунок), затем в формате .gif, а после в формате .jpg (рис. 2).

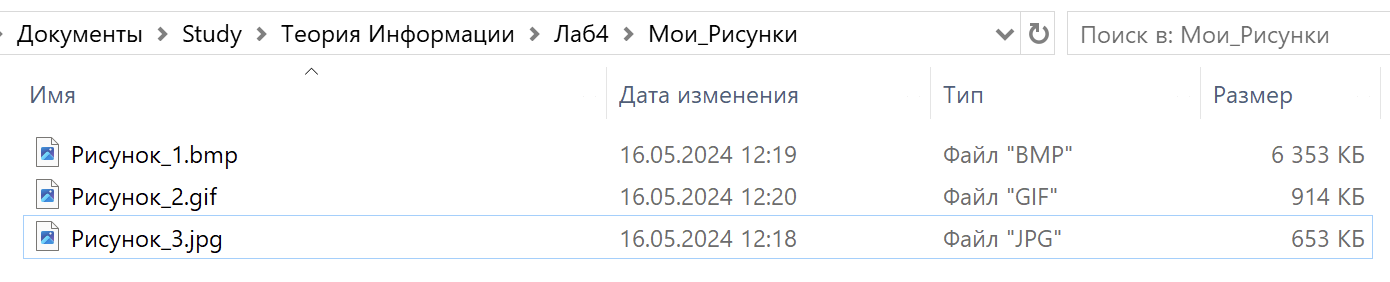


Рис. 2: Сохранённые изображения

1. В меню «Свойства» для каждого файла найден его занимаемый размер (рис. 3). При этом был учёт только пункт «Размер», а не пункт «На диске».

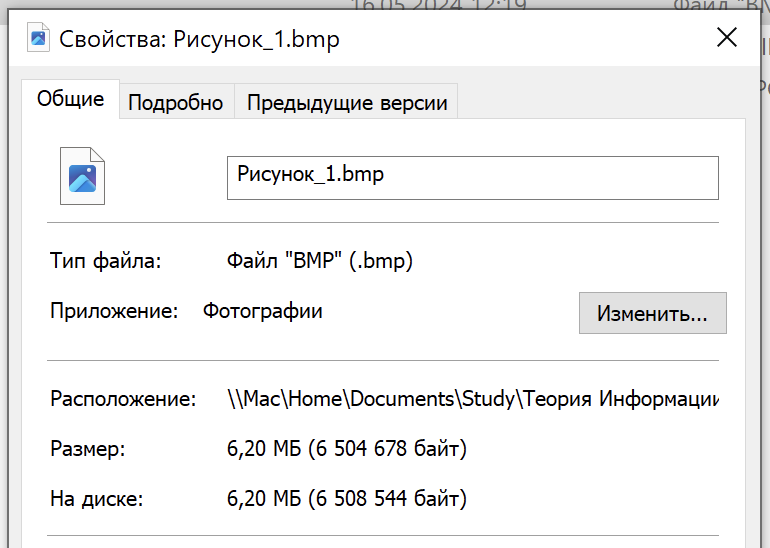


Рис. 3: Свойства файла

1. Данные о теоретическом и действительном размере файлов были выписаны в таблицу (рис. 4).

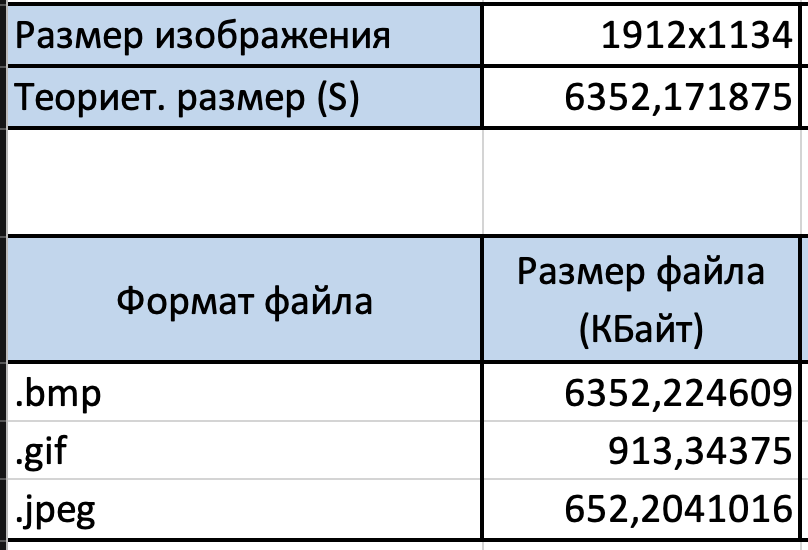


Рис. 4: Выписанные размеры

1. По формуле (1.2) вычислена степень сжатия для каждого из форматов файлов, а данные выписаны в таблицу (рис. 5).

где, – степень сжатия (%), – теоретический объём файла (байт), – объём сжатого файла (байт).

Для Excel эта формула выглядит следующим образом (1.3):

где, $C$3/1024 – теоретический объём (КБайт), B6 – объём сжатого файла (КБайт).

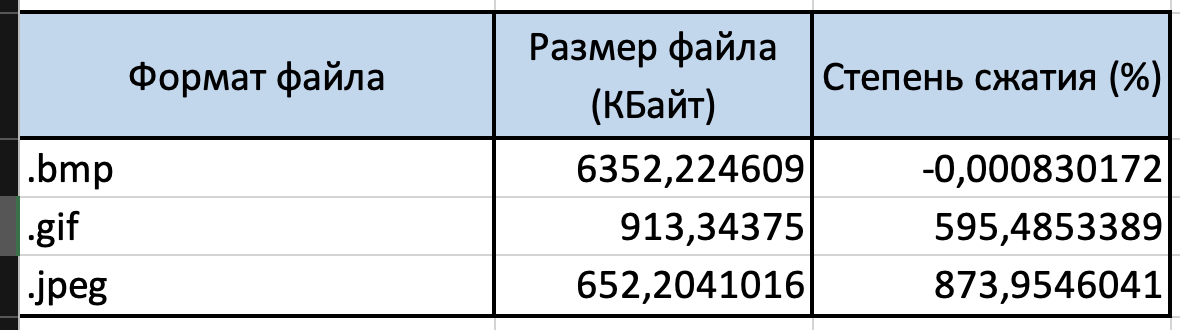


Рис. 5: Вычисленные степени сжатия

1. Из полученной таблицы видно, что самым экономичным форматом хранения изображений в данном случае является jpeg, степень сжатия которого относительно теоретического размера составила (874%).
2. Затем в MS Paint был создан новый документ с размером 640x480 точек. При помощи инструмента «Кисть» на созданном полотне были нарисованы 8 полос разного цвета (рис. 6).



Рис. 6: Рисунок из цветных полос

1. После того как линии были нарисованы, изображение поочерёдно было сохранено в форматах bmp, gif, jpeg (рис. 7). После чего размеры изображений также были рассчитаны и выписаны в таблицу (рис. 8).

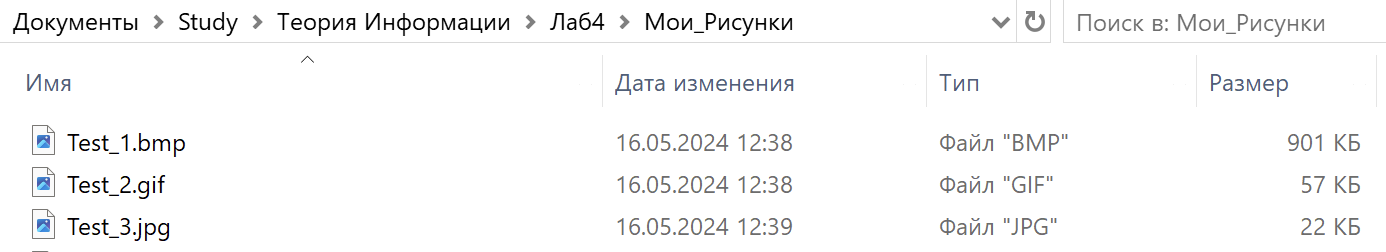


Рис. 7: Сохранённые изображения

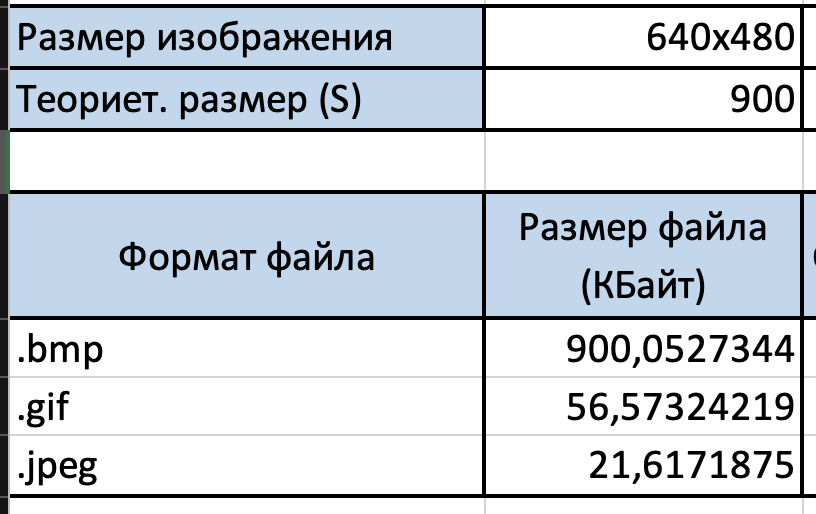


Рис. 8: Таблица с размерами

1. По формуле (1.2) рассчитана степень сжатия для каждого из файлов, после чего результаты были занесены в таблицу (рис. 9).

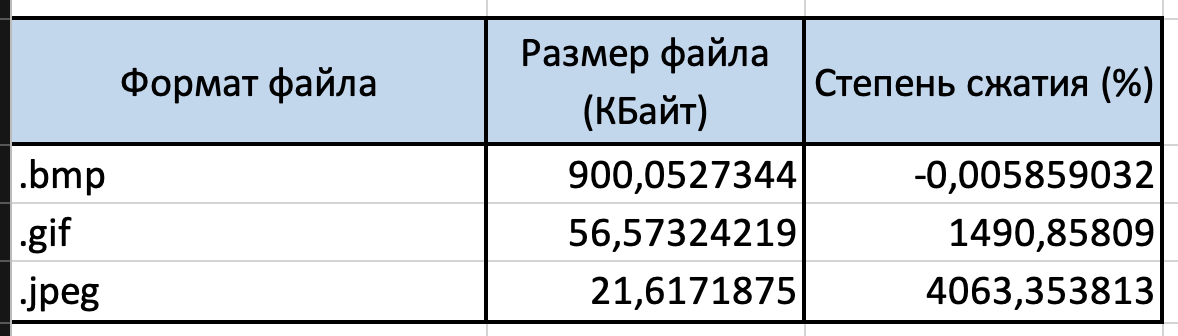


Рис. 9: Степени сжатия форматов

1. По результатам из таблицы видно, что самая высокая степень сжатия снова оказалась присуще формату .jpg (4063%).

Задание 2.

В качестве исходной использована фраза «Мощность 3574444555 Вт» (вариант 15).

1. Для каждого символа по таблице CP1251 найден его код в десятичном, а затем и в двоичном представлении при помощи Excel функции (рис. 10).

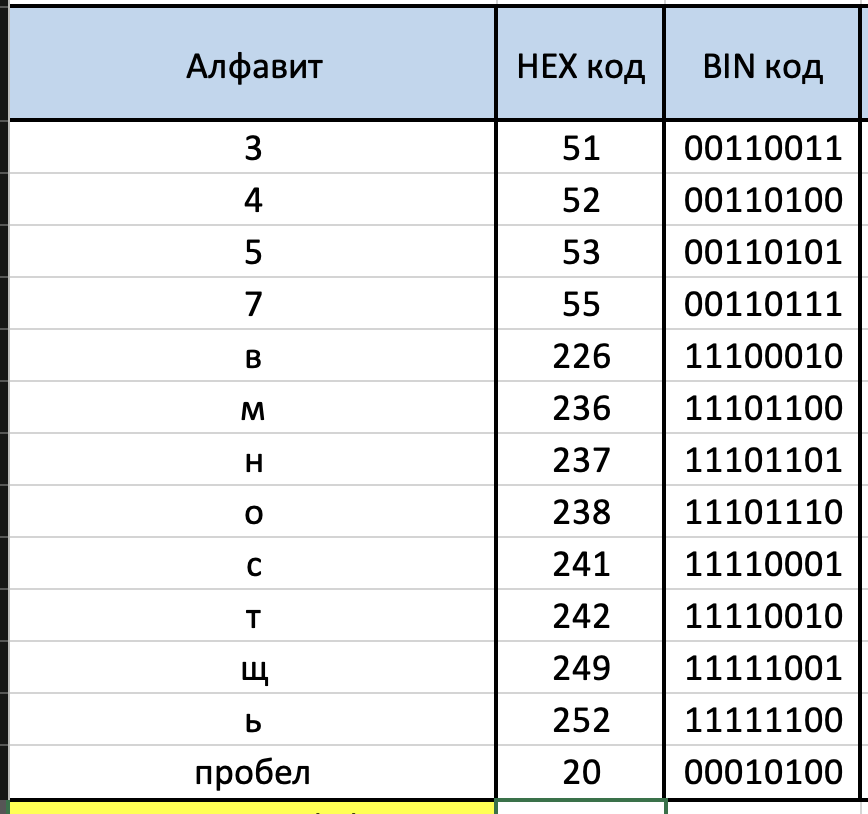


Рис. 10: Коды символов согласно CP1251

1. После чего из полученных двоичных кодов составлено исходное сообщение, которое по итогу оказалось длиной 22 байта (176 бит).
2. Согласно методу RLE составлено следующее сообщение:

(12, мощность 357), (4, 4), (3, 5), (3, Вт)

Которое затем представлено в двоичном виде:

* Число 12 прибавлено к 128 и переведено в двоичную систему счисления;
* Затем записана последовательность неповторяющихся символов при помощи их двоичных кодов;
* Число 4 записано в двоичной форме, а затем за ним записан код символа «4»;
* Затем записано число 3 в двоичной системе счисления, а за ним код символа «5» в той же системе;
* Записано число 131 (128+3) в двоичном виде, а за ним коды неповторяющихся символов.

1. Полученное таким способом сжатое сообщение имеет длину 20 байт (160 бит).
2. По формуле (1.3) рассчитана степень сжатия информации, которая составила 10%.

Задание 3.

1. В задании использованы тот же текст те же коды символов, что и в предыдущем, которые были выписаны в таблицу.
2. При помощи Excel формулы (3.1) для каждого символа найдено его количество в тексте, после чего по формуле (3.2) найдена вероятность его появления.

где, $A$4 – исходное сообщение, A8 – искомый символ.

где, – количество символа в тексе, – длина исходного сообщения.

1. Вероятности сгруппированы в две группы с как можно близким по значению результатом суммы, после чего в каждая из групп снова разбивалась надвое по этому же правилу. Разбиение продолжалось до тех пор, пока в группе не оставалось по одному элементу. При этом для верхних групп присваивался зелёный цвет, а для нижних – синий (рис. 11).

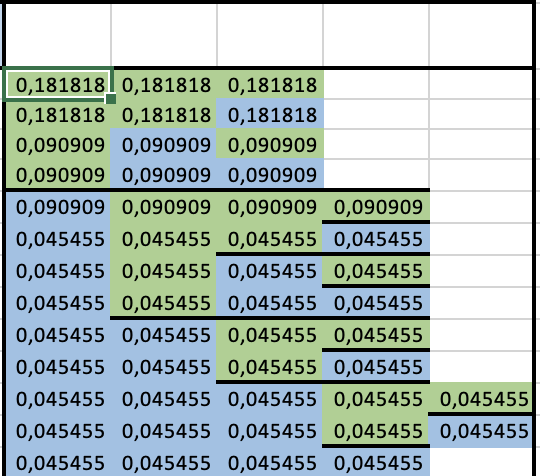


Рис. 11: Группировка вероятностей

1. По созданной таблице построены новые коды для каждого из символов (рис. 12). Принималось что зелёный цвет – единица, а синий – ноль.



Рис. 12: Полученные из таблицы коды

1. Из полученных кодов построено новое сообщение, которое имеет длину 76 бит (вместо 176).
2. По формуле (1.3) рассчитана степень сжатия, которая составила 132%.

Задание 4.

1. Вероятности, найденные на шаге 2 предыдущего способа, были также отсортированы по убыванию. Затем последние две вероятности были сложены и записаны в столбец справа, а ячейке с суммой присвоен цвет. Оставшиеся вероятности переписаны в него же без изменений, после чего столбец был отсортирован по убыванию вероятностей. Процедура повторялась пока сумма вероятностей в ячейке не станет равна единице (рис. 13).

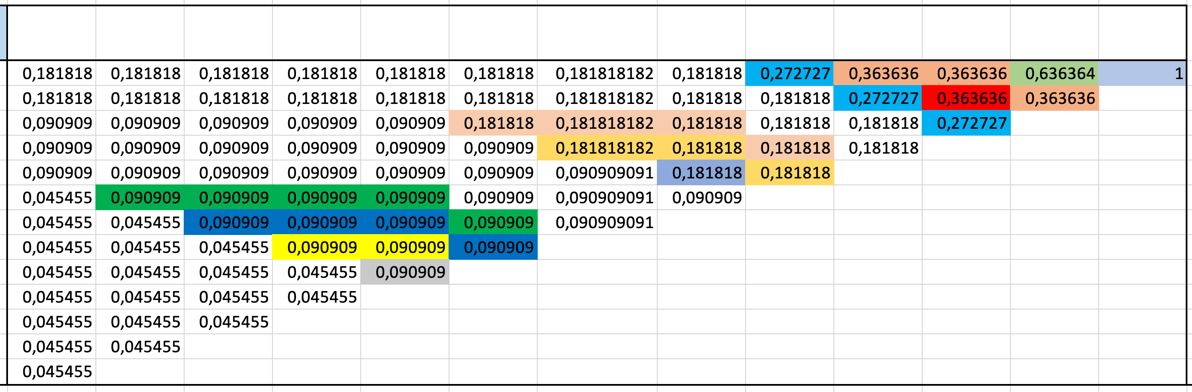


Рис. 13: Разбиение вероятностей по Хаффману

1. После по построенному разбиению было составлено дерево (рис. 14), по которому были определены новые коды символов (рис. 15).

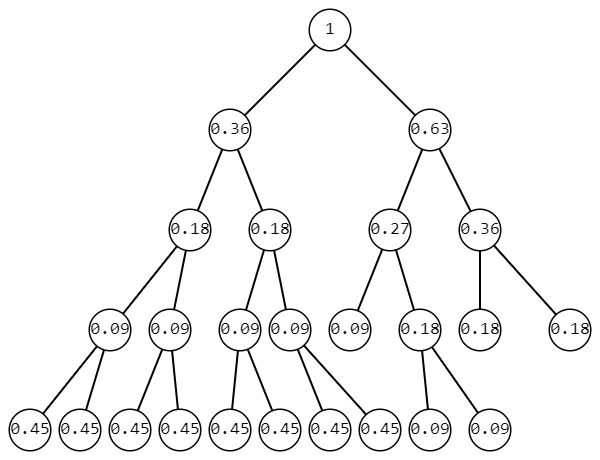


Рис. 14: Дерево Хаффмана



Рис. 15: Полученные коды для символов

1. Из полученных кодов составлено исходное сообщение, которое теперь стало занимать 78 бит вместо 176.
2. По формуле (1.3) вычислена степень сжатия, которая составила 126%.

Задание 5.

Из двух использованных методов степень сжатия оказалась выше при использовании метода кодирования Шеннона-Фано, но при этом незначительно. В предыдущих лабораторных работах было выяснено, что в отличие от метода кодирования Хаффмана этот метод не является однозначным, а это значит, что в другой раз результаты могут сильно различаться.

РЕЗУЛЬТАТ ВЫПОЛНЕНИЯ РАБОТЫ

Ниже представленные построенные во время лабораторной работы таблицы (табл. 1-5):

Таблица 1

Сжатие изображений

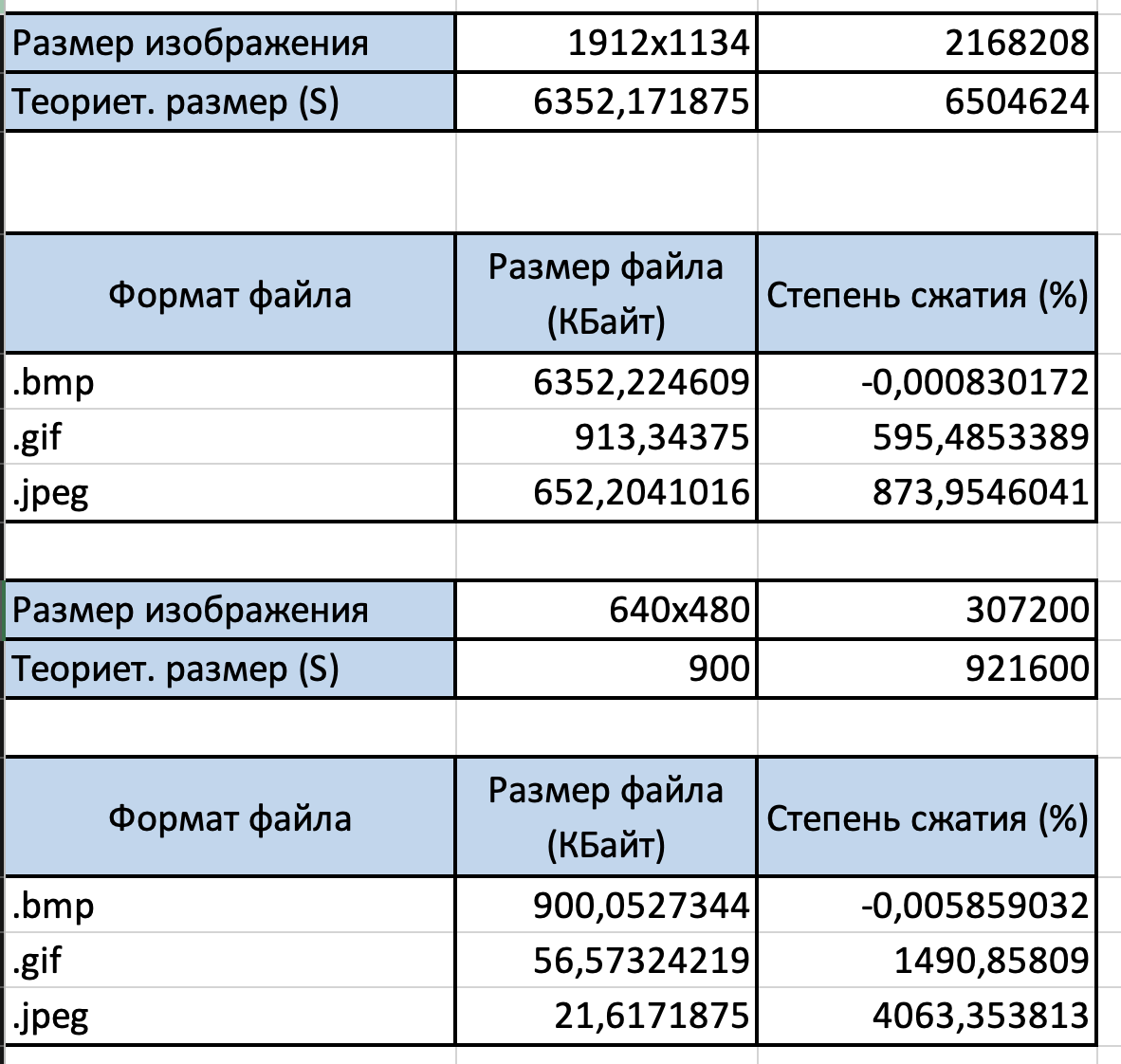


Таблица 2

Метод RLE

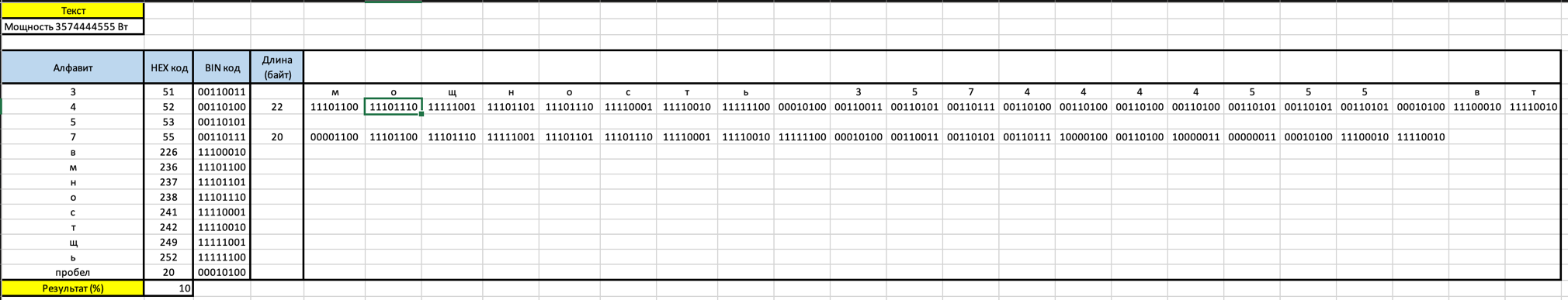


Таблица 3

Метод Шеннона-Фано

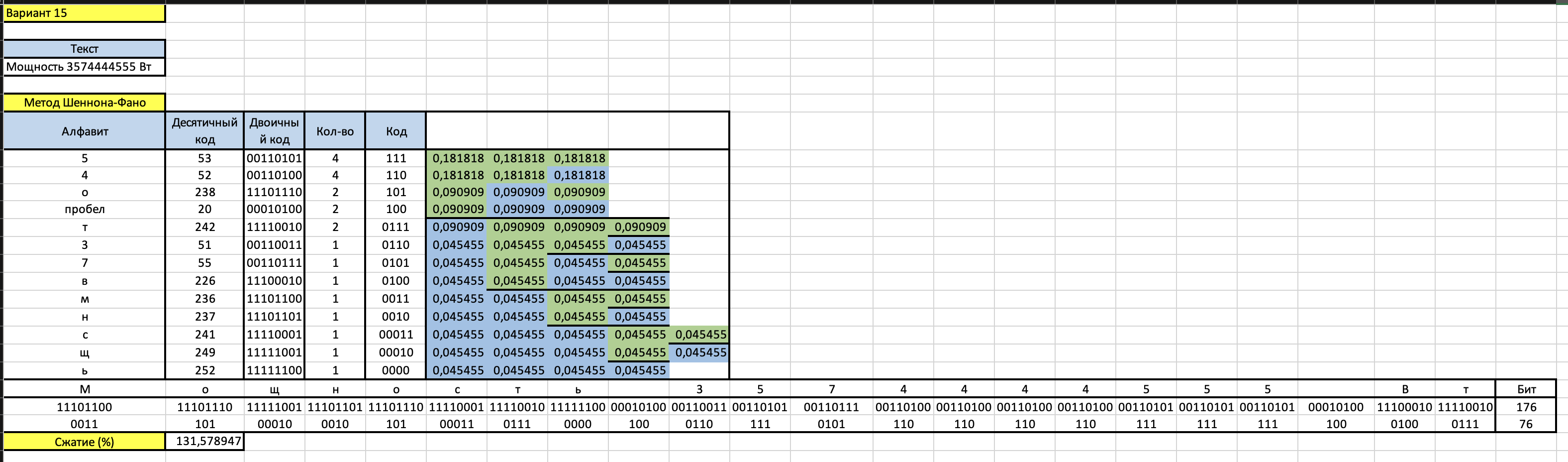
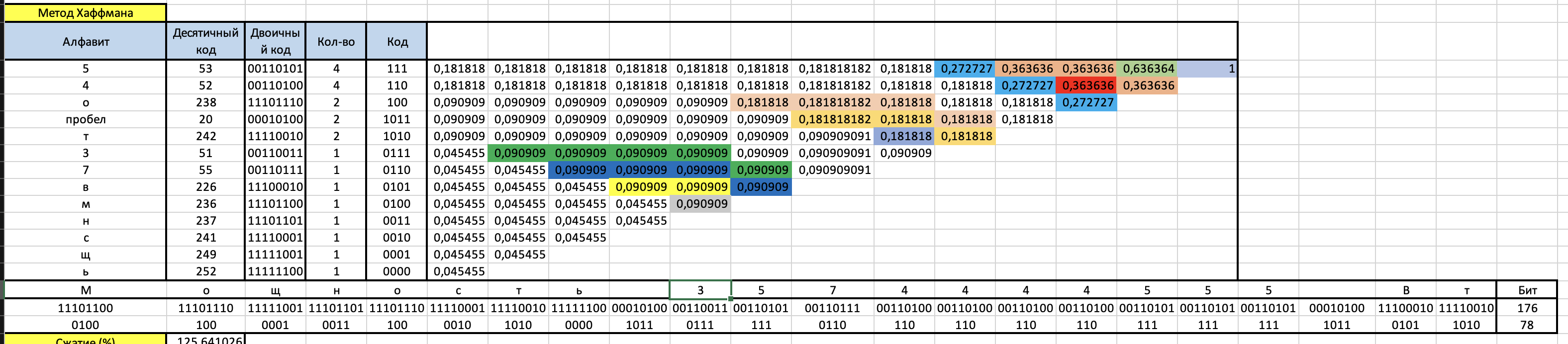


Таблица 4

Метод Хаффмана



ЗАКЛЮЧЕНИЕ

В ходе лабораторной работы было выполнено сжатие изображений путём сохранения их в различных форматах, а также сжатие текстового сообщения методом RLE, методом упаковки в эффективное кодирование (метод Шеннона-Фано и Хаффмана).

В ходе работы с изображениями было выяснено что самым экономичным форматом хранения изображений среди .bmp, .gif, .jpg оказался последний. При этом это оказалось применимо, как и многоцветным, так и к изображениям с малой палитрой цветов.

При многоцветном изображении степень сжатия составила 874%, тогда как у других этот показатель был сильно меньше:

где, - bmp, - gif, - jpg.

Отрицательная степень сжатия .bmp – вероятно, наличие у файла заголовка.

Для ограниченной палитры цветов степень сжатия составила 4063%, что сильно превысило показатели других форматов:

где, - bmp, - gif, - jpg.

По полученным результатам можно судить, что формат jpg наиболее благоприятен для передачи по каналам связи, нежели другие.

При кодировании текстового сообщения самая низкая степень сжатия была с использованием алгоритма RLE – всего 10%, тогда как при использовании метода упаковки степень сжатия более чем 10 раз превышает RLE (132% и 126%). Метод RLE оказался таким неэффективным только лишь потому, что в выбранном сообщении было довольно мало идущих подряд символов.

Методы эффективного кодирования по результатам отличаются не сильно, но из-за того, что метод Шеннона-Фано не является однозначным, поэтому результат может сильно отличаться от полученного, о чём был сделан вывод выше.