Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Пермский национальный исследовательский

политехнический университет»

(ПНИПУ)

Кафедра вычислительной математики и механики

Отчёт

по дисциплине: «Технологии обработки информации»

по теме: «Арифметическое кодирование и декодирование»

Выполнил:

Студент группы ИСТ-19-1бзу

Соболь Евгений Валерьевич

Проверил:

доцент кафедры ВМиМ

Сабатулина Татьяна Леонидовна

Пермь

2021 г.

**Листинг программы с комментариями по каждой функции:**

from collections import Counter  
from math import copysign, fabs, floor, isfinite, modf, log, ceil  
from ast import literal\_eval  
  
  
def beautiful\_output(d: dict, step: int):  
 *"""Функция для вывода результатов, 1-3, 5 шагов"""* if step == 1:  
 print("-----ПОДСЧЁТ СИМВОЛОВ-----")  
 for i in d:  
 print(f"Символ {i} встречается {d[i]}")  
 print()  
 if step == 2:  
 print("-----ПОДСЧЁТ ЧАСТОТЫ ПОЯВЛЕНИЯ-----")  
 for i in d:  
 print(f"Символ {i} c частотой {d[i]}")  
 print()  
 if step == 3:  
 print("-----СОРТИРОВКА ПО УБЫВАНИЮ-----")  
 k = 1  
 for i in d:  
 print(f"{k} элемент - символ {i} c частотой {d[i]}")  
 k += 1  
 print()  
 if step == 4:  
 print("-----НАЙДЁМ ТОЧКИ НА ОТРЕЗКЕ ОТ 0 ДО 1-----")  
 k = 1  
 for i in d:  
 print(f"{k} '- {d[i]}")  
 k += 1  
 print()  
 if step == 5:  
 print("-----НАЙДЁМ ОТРЕЗКИ И ДЛИНУ-----")  
 for i in d:  
 print(f"{i} - [{d[i][1]}: {d[i][0]}], длина: {d[i][0] - d[i][1]}")  
 print()  
  
  
def float\_to\_bin(f: float) -> str:  
 *"""Функция преобразования вещественного числа из 10 с.ч в 2 с.ч"""* if not isfinite(f):  
 return repr(f) # inf nan  
 sign = '-' \* (copysign(1.0, f) < 0)  
 frac, fint = modf(fabs(f)) # split on fractional, integer parts  
 n, d = frac.as\_integer\_ratio() # frac = numerator / denominator  
 assert d & (d - 1) == 0 # Модульный тест  
 return f'{sign}{floor(fint):b}.{n:0{d.bit\_length() - 1}b}'  
  
  
def bin\_to\_float(binary):  
 *"""Функция преобразования вещественного числа из 2 с.ч в 10 с.ч"""* m = [list(tuple(i)) for i in binary.split(".")]  
 result = 0  
 for i in range(len(m)):  
 if i == 0:  
 for j in range(len(m[i])):  
 m[i][j] = int(m[i][j]) \* 2 \*\* (len(m[i]) - j - 1)  
 result += sum(m[i])  
 if i == 1:  
 for j in range(len(m[i])):  
 m[i][j] = int(m[i][j]) / 2 \*\* (j + 1)  
 result += sum(m[i])  
 return result  
  
  
def sorted\_dict(dict\_original: dict) -> dict:  
 *"""Функция сортировки словаря по ключу"""* sort\_dict = dict(sorted(dict\_original.items(), key=lambda x: x[0]))  
 beautiful\_output(sort\_dict, 3)  
 return sort\_dict  
  
  
def new\_value(lo: float, ho: float, rh: float, rl: float) -> tuple:  
 *"""Функция расчёта границ"""* new\_low\_value = lo + (ho - lo) \* rl  
 new\_high\_value = lo + (ho - lo) \* rh  
 print(f"СДВИНУЛИСЬ ГРАНИЦЫ [{new\_low\_value}:{new\_high\_value}]")  
 return new\_low\_value, new\_high\_value  
  
  
def generic\_dict(line: str) -> dict:  
 *"""Создаём словарь c вероятностями по символу"""* final\_dict = dict(Counter(line))  
 beautiful\_output(final\_dict, 1)  
 for i in final\_dict.keys():  
 final\_dict[i] = final\_dict[i] / len(line)  
 beautiful\_output(final\_dict, 2)  
 return final\_dict  
  
  
def create\_section(d: dict) -> dict:  
 *"""Функция для нахождения точек на отрезке от 0 до 1"""* s = 0  
 for i in d.keys():  
 s += d[i]  
 d[i] = s  
 beautiful\_output(d, 4)  
 return d  
  
  
def computation\_result(line: str, d: dict) -> float:  
 *"""Функция отвечающая за арифметическое кодирование"""* lo = 0  
 ho = 1  
 print("-----АРИФМЕТИЧЕСКОЕ КОДИРОВАНИЕ-----")  
 for i in list(line):  
 lo, ho = new\_value(lo, ho, d[i][0], d[i][1])  
 print("G=", ho - lo)  
 return (lo + ho) / 2, ho - lo  
  
  
def encoding(line: str) -> tuple:  
 *"""Функция кодирования"""* result = generic\_dict(line)  
 result = sorted\_dict(result)  
 section = create\_section(result)  
 k = 0  
 for i in section.keys():  
 section[i] = [section[i]]  
 section[i].append(k)  
 k = section[i][0]  
 beautiful\_output(section, 5)  
 result, n = computation\_result(line, section)  
 n = ceil(-log(n, 2) + 1)  
 print(f"F+G/2={result}, ^-logG + 1^={n}")  
 return float\_to\_bin(result)[2:n+2], section  
  
  
def decoding(enc\_value: float, length: int, d: dict) -> str:  
 *"""Функция декодирования"""* print("-----ДЕКОДИРОВАНИЕ-----")  
 enc\_value = bin\_to\_float("0." + enc\_value)  
 result = ""  
 k = 1  
 for \_ in range(length):  
 for i in d.keys():  
 if (enc\_value >= d[i][1]) and (enc\_value <= d[i][0]):  
 print(f"Итерация ({k}) - {enc\_value} > {d[i][1]} и {enc\_value} < {d[i][0]}")  
 k += 1  
 result += i  
 enc\_value = (enc\_value - d[i][1]) / (d[i][0] - d[i][1])  
 print("Значение:", enc\_value)  
 print("Расшифровано:", result)  
 print("-------------------------")  
 break  
 return result  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 print("Задание № 4 Арифметическое кодирование")  
 source\_line = input("Введите строку: ")  
 encoding\_value, dictionary = encoding(source\_line)  
 print("--------------------РЕЗУЛЬТАТ КОДИРОВАНИЯ--------------------")  
 print("Закодированное значение: ", encoding\_value, "\n")  
 length\_line = len(source\_line)  
 decoding\_value = decoding(encoding\_value, length\_line, dictionary)  
 print("--------------------РЕЗУЛЬТАТ ДЕКОДИРОВАНИЯ--------------------")  
 print("Расшифрованное значение: ", decoding\_value, "\n")

**Пример работы программы № 1:**

Пример 2.8.1 из учебника Кудряшова Б.Д стр. 81

Тестовое значение для ввода: bcbabbbbcc

**Результат кодирования:**

10001010101010

**Результат декодирования:**

bcbabbbbcc

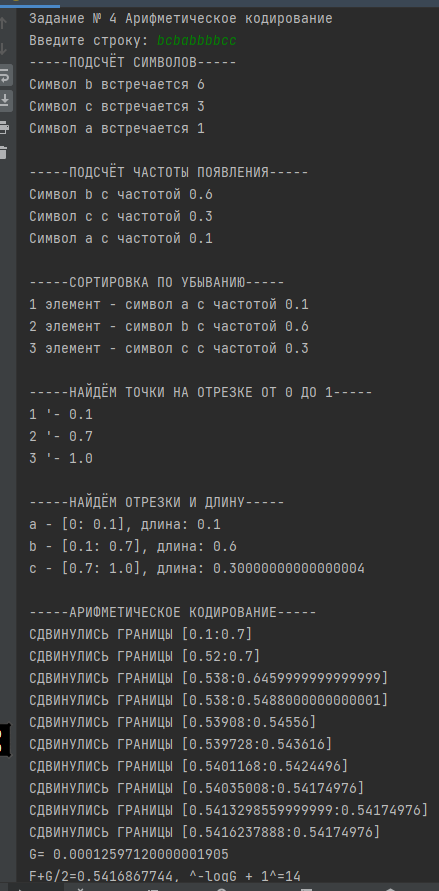


Рисунок 1 – Скриншот программы (кодирование)



Рисунок 2 – Скриншот программы (кодирование)

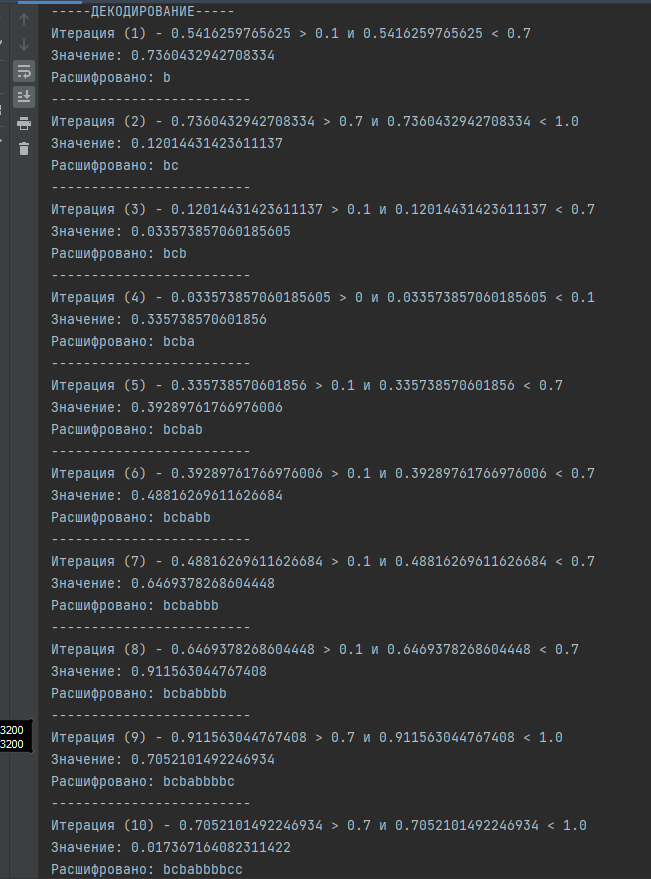


Рисунок 3 – Скриншот программы (декодирование)

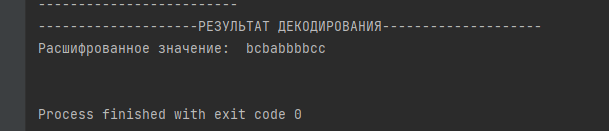


Рисунок 4 – Скриншот программы (декодирование) результат

**Пример работы программы № 2:**

Тестовое значение для ввода: Hello world!1

**Результат кодирования:**

0100001110010001011100100110010111001110011

**Результат декодирования:**

Hello world!1

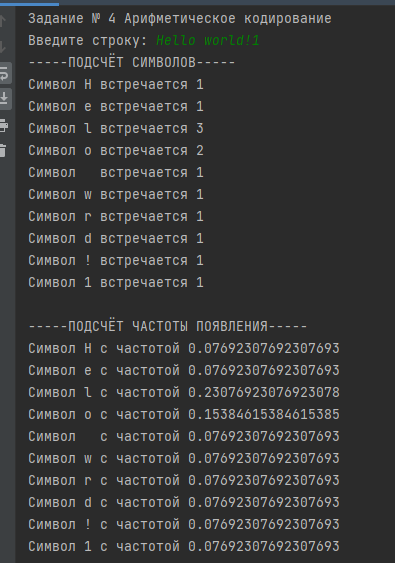


Рисунок – Скриншот программы кодирования (1)

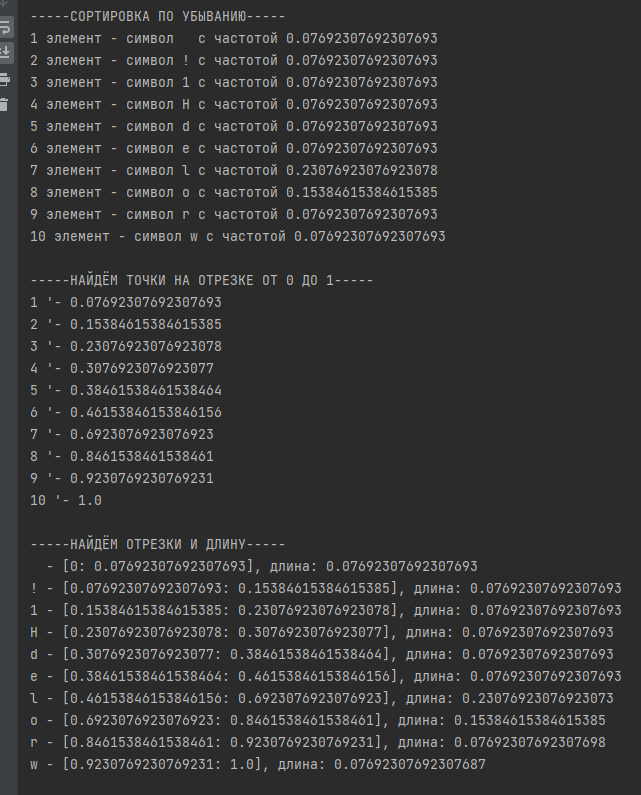


Рисунок - Скриншот программы кодирования (2)

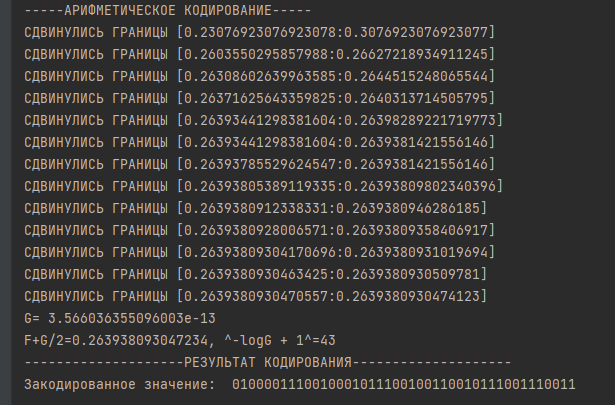


Рисунок - Скриншот программы кодирования (3)

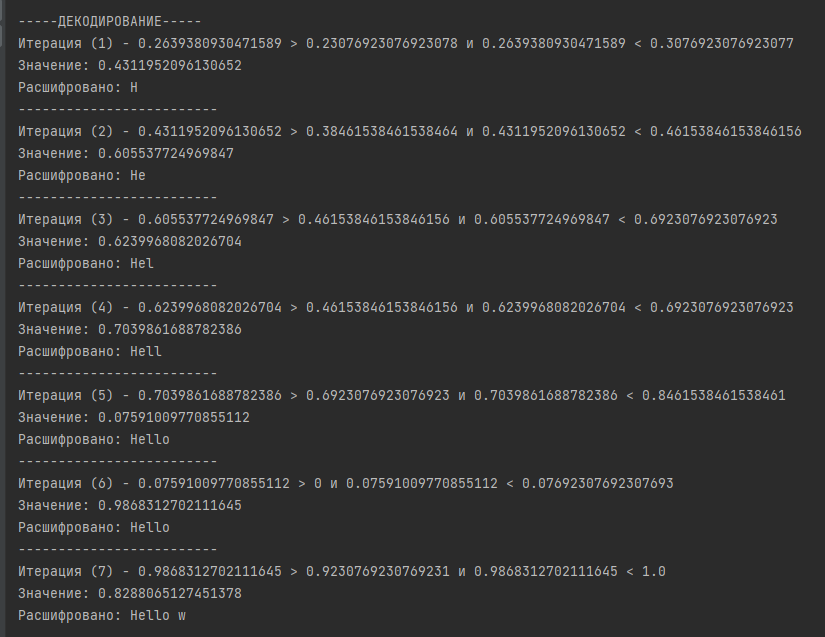


Рисунок - Скриншот программы кодирования (4)

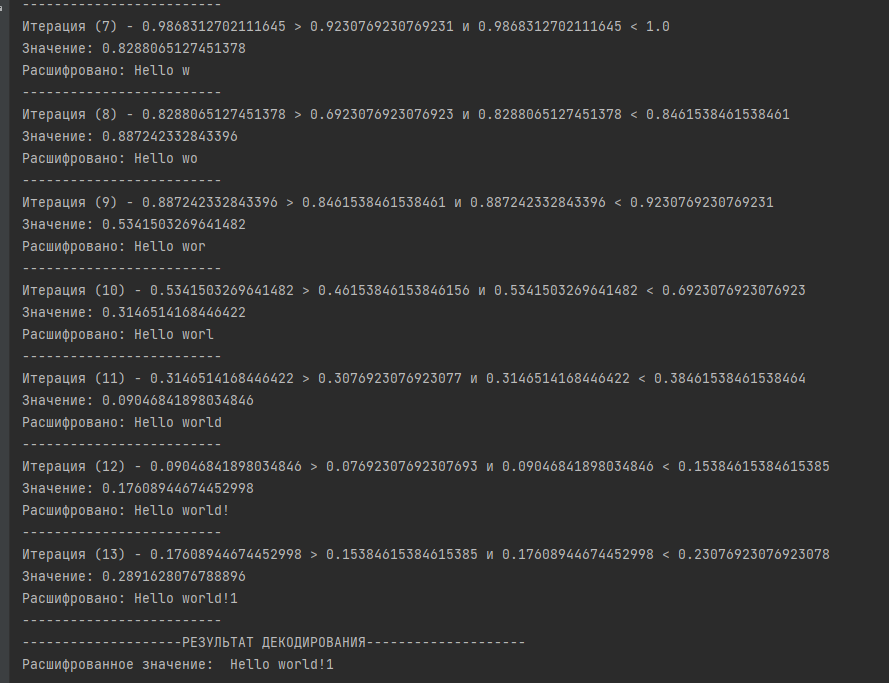


Рисунок - Скриншот программы кодирования (5)