МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МОЭВМ

ОТЧЕТ

по лабораторной работе №5
по дисциплине «Алгоритмы и структуры данных»
Тема: Кодирование и декодирование

Студент гр. 9303 _____ Низовцов Р. С.

Преподаватель _____ Филатов А. Ю.

Санкт-Петербург
2020

Цель работы.

Написание алгоритма статической кодировки по методу Шеннона — Фано.

Задание.

Вариант 1

Кодирование: Шеннона — Фано

Основные теоретические положения.

Алгоритм Шеннона — Фано — один из первых алгоритмов сжатия, который впервые сформулировали американские учёные Клод Шеннон и Роберт Фано. Данный метод сжатия имеет большое сходство с алгоритмом Хаффмана, который появился на несколько лет позже и является логическим продолжением алгоритма Шеннона. Алгоритм использует коды переменной длины: часто встречающийся символ кодируется кодом меньшей длины, редко встречающийся — кодом большей длины. Коды Шеннона — Фано — беспрефиксные, то есть никакое кодовое слово не является префиксом любого другого. Это свойство позволяет однозначно декодировать любую последовательность кодовых слов.

Описание алгоритма.

Основные этапы:

- Символы первичного алфавита m_1 выписывают по убыванию вероятностей.
- Символы полученного алфавита делят на две части, суммарные вероятности символов которых максимально близки друг другу.
- В префиксном коде для первой части алфавита присваивается двоичная цифра «0», второй части — «1».

- Полученные части рекурсивно делятся, и их частям назначаются соответствующие двоичные цифры в префиксном коде.

Когда размер подалфавита становится равен нулю или единице, то дальнейшего удлинения префиксного кода для соответствующих ему символов первичного алфавита не происходит, таким образом, алгоритм присваивает различным символам префиксные коды разной длины. На шаге деления алфавита существует неоднозначность, так как разность суммарных вероятностей $p_0 - p_1$ может быть одинакова для двух вариантов разделения (учитывая, что все символы первичного алфавита имеют вероятность больше нуля).

Выполнение работы.

Для выполнения программы были реализованы функции stringAnalys, stringSort, searchTree, stringCoder.

В функции void stringAnalys(...) производится первичный анализ введенной строки, вычесляются уникальный символы и количество их вхождений в исходную строку. Потом эта функция вызывает stringSort, searchTree, stringCoder по очереди. Итогом выполнения функции будет кодирования строки.

В функции void stringSort(…) производится сортировка символов по убыванию количества вхождений символа в строку.

В функции void searchTree(…) производится обработка данных и присваивание каждому символу его уникальный шифр-код.

В функции void stringCoder(…) производится кодирование исходной строки на основе созданной статической кодировки.

Исходный код программы представлен в приложении А. Результаты тестирования включены в приложение Б.

Выводы.

Ознакомился с алгоритмом Шеннона - Фано, изучил его особенности и реализовал программу, решающую поставленную задачу с помощью данного алгоритма.

Была реализована программа, включающая в себя функцию stringAnalys для статической кодировки введенной строки. Программа выполняет запись результата в файл, а также производит проверку и обработку полученных данных.

ПРИЛОЖЕНИЕ А

ИСХОДНЫЙ КОД ПРОГРАММЫ

Название файла: main.cpp

```
#include <iostream>
#include <fstream>
#include <algorithm>
#include <cmath>
#include <string>

using namespace std;

void stringSort(string &symbols, int *freq){
    int n = symbols.length();
    int temp = 0;
    char ctemp;
    for (int k = 0; k < n; k++){
        for (int j = k % 2; j + 1 < n; j += 2){
            if (freq[j] < freq[j + 1]){
                temp = freq[j];
                freq[j] = freq[j + 1];
                freq[j + 1] = temp;
                ctemp = symbols[j];
                symbols[j] = symbols[j + 1];
                symbols[j + 1] = ctemp;
            }
        }
    }
}

void searchTree (string &symbols, int *freq, char lastCodeElem, string
&code, int start, int end, string* &cipher)
{
    double totalSum = 0;
    int i, currSum = 0;
    string currCode = "";
    if(lastCodeElem != '\0')
        currCode = code + lastCodeElem;
    else
        currCode = code;

    if (start==end)
    {
        cipher[start] = currCode;
        if(symbols[start] == ' ')
            cout << "space (" << freq[start] << ") --- " << currCode <<
endl;
        else
            cout << symbols[start] << " (" << freq[start] << ") --- " <<
currCode << endl;
        return;
    }
    for (i=start;i<=end;i++)
        totalSum+=freq[i];
    totalSum/=2;
```

```
    i=start+1;
    currSum +=freq[start];
    while (fabs(totalSum-(currSum+freq[i])) < fabs(totalSum-currSum) &&
(i<end))
    {
        currSum+=freq[i];
        i++;
    }
    searchTree(symbols, freq, '0', currCode , start, i-1, cipher);
    searchTree(symbols, freq, '1', currCode , i, end, cipher);
}

void stringCoder(string &line, string* &cipher, string &symbols, ofstream
&outFile){
    size_t pos;
    int size = line.length();
    cout << "Coded message: ";
    for(int i = 0; i < size; i++){
        pos = symbols.find(line[i]);
        cout << cipher[pos];
        outFile << cipher[pos];
    }
    cout << endl << endl;
    outFile << endl;
}

void stringAnalys(string &line, string &symbols, ofstream &outFile){
    int size = 0;
    int start_size = line.length();
    size_t pos;
    int* freq = new int[size];
    for(int i = 0; i < start_size; i++){
        pos = symbols.find(line[i]);
        if(pos == string::npos){
            symbols+=line[i];
            int *freq1 = new int[++size];
            for(int j = 0; j < size - 1; j++)
                freq1[j] = freq[j];
            freq1[size-1] = 1;
            delete[] freq;
            freq = freq1;
        }
        else{
            freq[pos]++;
        }
    }
    string* cipher = new string[size];
    cout << "Unique symbols:" << endl << symbols << endl;
    string code = "";
    stringSort(symbols, freq);
    searchTree(symbols, freq, '\0', code, 0, size - 1, cipher);
    stringCoder(line, cipher, symbols, outFile);
}

int main() {
    string file_name;
    cout << "Enter the name of an input file: " << endl;
    cin >> file_name;
```

```
    ifstream inFile(file_name);
    if (!inFile.is_open()){
        cout << "Permission denied or wrong path" << endl;
        return 0;
    }
    cout << "Result was saved in result.txt" << endl << endl;
    ofstream outFile("result.txt");
    string line;

    while(getline(inFile, line)){
        transform(line.begin(), line.end(), line.begin(), ::tolower);
        cout << "Input line:" << endl << line << endl;
        outFile << line << endl;
        string symbols;
        stringAnalys(line, symbols, outFile);
    }
    cout << "End of work" << endl;
    inFile.close();
    outFile.close();
    return 0;
}
```

ПРИЛОЖЕНИЕ Б

ТЕСТИРОВАНИЕ

Таблица Б.1 — Примеры тестовых случаев

№ п/п	Входные данные	Выходные данные	Комментарий
1.	I love the subject of Algorithms and data structures	10110000101111000111 00010000001110001000 00110100111101011101 10101101000100010001 11100000011110111111 10110001010101100011 10011111001100000111 11111111011000110110 11100101110000110001 10101001110100011001 10100100110	Программа работает корректно
2.	I can't live without it..	00010100110101011010 11101101011000011010 11011010111000001111 10111110111110110100 001110001000	Программа работает корректно
3.	Sometimes I dream about him, especially my term paper, mmm	01100111001010101101 00000101001100001000 00011101010010101010 00100010101110110111 10111101100001111001 00000111000000100110 11110101011010100010 10110111101111100000 00111100000110100111 10111100101100100001 11110011110011111111 100000001001001	Программа работает корректно