**Московский государственный технический**

**университет им. Н.Э. Баумана**

Факультет «Информатика и системы управления»

Кафедра ИУ5 «Системы обработки информации и управления»

Отчет по домашнему заданию №3

**«Разархивация данных»**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Выполнил:** | |  | **Принял:** | |
| ФИО: | \_Цыпышев Т. А.\_\_\_\_\_ |  | ФИО: | \_Семёнов Д. В.\_\_\_\_\_\_ |
| Группа: | \_ИУ5-11Б\_\_\_\_\_\_\_\_\_\_ |  | Должность: | \_Преподаватель\_\_\_\_\_ |
| Дата: | \_20.01.2023\_\_\_\_\_\_\_\_ |  | Дата: | \_20.01.2023\_\_\_\_\_\_\_\_\_ |
| Подпись: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |  | Подпись: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Москва, 2022 г.

**Описание алгоритма**

**Алгоритм:**

Пользователь вводит путь к оригинальному файлу, он передаётся в функцию Decompress2. В ней мы узнаём длину файла и начинаем считывать байты из закодированного текста. Сдвигаем побитно буфер на 8 и добавляем byte. Пока длина буфер меньше type мы берём первые type бит из буфера. Если эти биты равны 0, то это разделитель и мы увеличиваем type на 1 и берём следующий бит из буфера, пока он не будет равен 1. Если биты не равны 0, то добавляем их в массив с расшифрованными числами.

Как только мы дойдём до конца файла мы вызываем функцию Decompress1. В ней создаётся словарь из 1-255 символов из таблицы ASCI. Потом начинаем перебирать элементы из массива, полученного из функции Decompress2 и присваиваем tmp значение out[(\*index) - 1]. Потом если закодированное слово уже есть в словаре, то мы берем значение из словаря, в противном случае из последнего вывода. Потом, если такого значения в словаре нет, то добавляем. Далее мы сохраняем значение из словаря для последующих итераций.

После 2 декодирования мы просим пользователя ввести путь и сохраняем полученный результат в файл с помощью функции Output.

**Описание функций**

1. **int main()** – здесь происходит вся магия, отсюда вызываются все функции
2. **int Find(int in[size\_dictionary][size\_word], int last\_position, int str[size\_word], int size)**

**Описание:**

Эта функция ищет первый массив str в массиве in.

**Переменные:**

* **int in[size\_dictionary][size\_word] –** двумерный массив, в котором нужно найти
* **int last\_position –** индекс первого незаписанного элемента в массиве **in**
* **int str[size\_word] –** массив, который нужно найти
* **int size –** индекс первого незаписанного элемента в массиве **str**

1. **int CreateMask(int a, int b)**

**Описание:**

Данная фи-я создаёт битовую маску, которая в последующем будет применяться, для получения конкретной последовательности битов.

**Переменные:**

* **int a –** правая граница (с конца)
* **int b –** левая граница (с начала)

1. **void Output(char filename[], int \*\*dec)**

**Описание:**

Данная фи-я сохраняет в файл раскодированный результат

**Переменные:**

* **char filename[]** – массив, хранящий путь к файлу
* **int \*\*dec** – массив с декодированными символами

1. **int \*Decompress2(char filename[], int \*size\_dec)**

**Описание:**

Данная фи-я считывает из закодированного файла символы и раскодирует их 1 методом

**Переменные:**

* **char filename[]** – массив, хранящий путь к закодированному файлу
* **int \*size\_dec** – длина массива с раскодированными элементами

1. **int \*\*Decompress1(int \*in, int size\_in, int \*index)**

**Описание:**

Ф-ия раскодирует 2 методом раскодированный 1 методом текст.

**Переменные:**

**int \*in** – массив с раскодированными символами

**int size\_in** – длина входящего массива

**int \*index** – длина выходящего массива

**Описание переменных**

**#define size\_dictionary 100000 –** максимальное кол-во слов в словаре

**#define size\_out 1000000 –** максимальная длина массива с сжатыми данными

**#define size\_word 100 –** максимальная длина слов в словаре

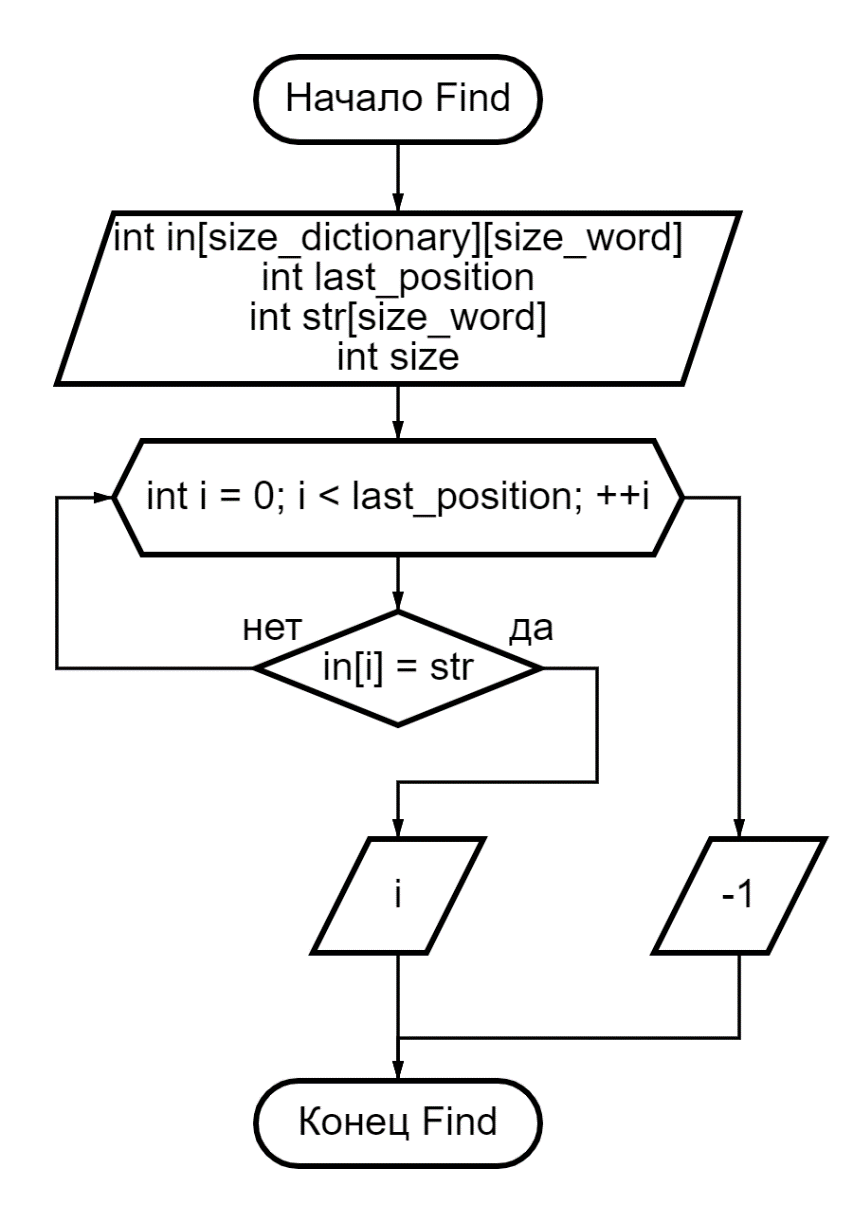
**char filename[100] –** переменная, в которой храниться путь до файлов

**int size\_dec** – переменная, хранящая длину раскодированного 1 методом массива

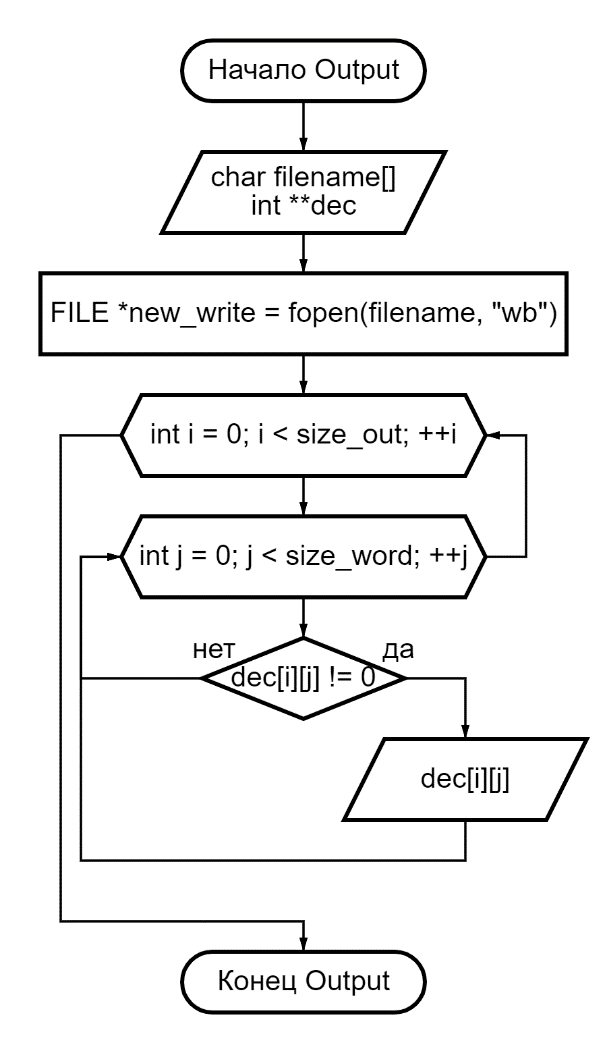
**int size\_dec1** – переменная, хранящая длину раскодированного 2 методом массива

**Схема алгоритма**

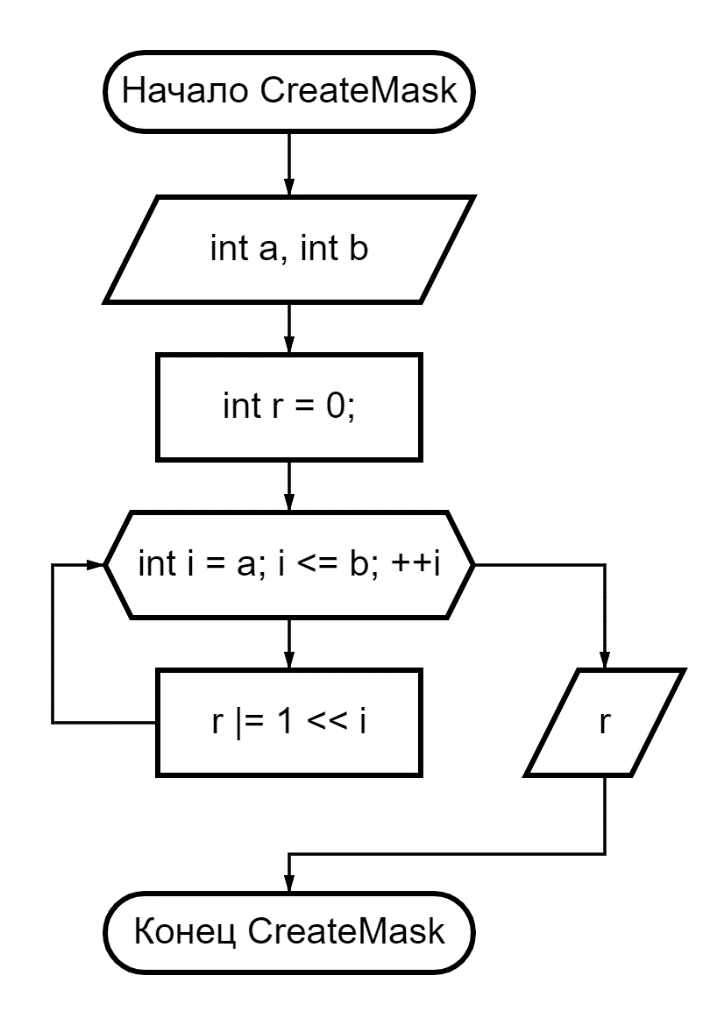
Find

****

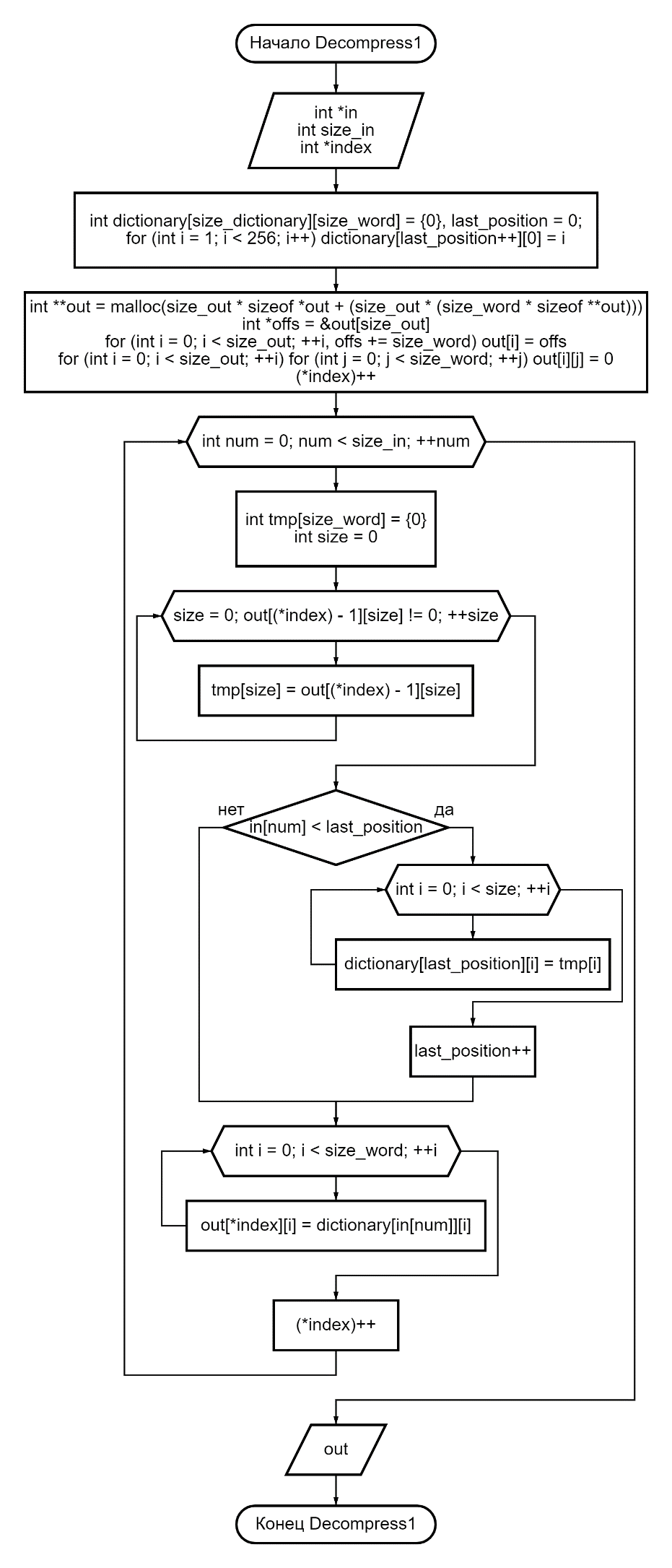
Output



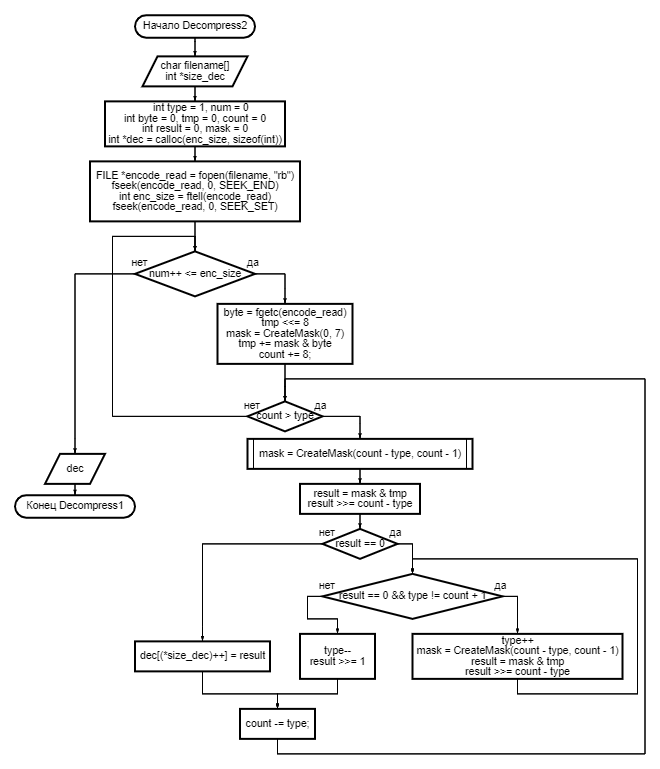
CreateMask



Decompress 1



Decompress 2



Main

