CMPT365 – Project
Student name: TRUNGLAM NGUYEN
Student ID: 301326848

Abstract

Algorithms in multimedia are often complicated and not easy for learners to understand. Instead of just explaining algorithm implementations by examples and code, we should have some more intuitive way to illustrate how such algorithms work step by step. Therefore, the aim of this project is to build an intuitive web UI to simulate each step of a lossless compression algorithm, namely LZW.

Introduction

Data compression is always an interesting and challenging problem in multimedia and there have been various research to improve data compression technology over the past decades. There are two major categories of compression, namely lossless and lossy. One of the most common lossless compression technique is Lempel-Ziv-Welch (LZW) algorithm. LZW was developed as an improved version of LZ78 algorithm created by Lempel and Ziv in 1978. LZW is surprisingly straightforward and easy to implement but it is not quite intuitive for learners. This is also the reason for me to build a web UI that can help me and others understand this algorithm by intuitive visualization.

Solution

1. Algorithm

I implemented LZW algorithm from scratch as a function that takes a string as input and produces a result that contains the encoded string and information of each step. The core implementation is as straightforward as the pseudocode below:

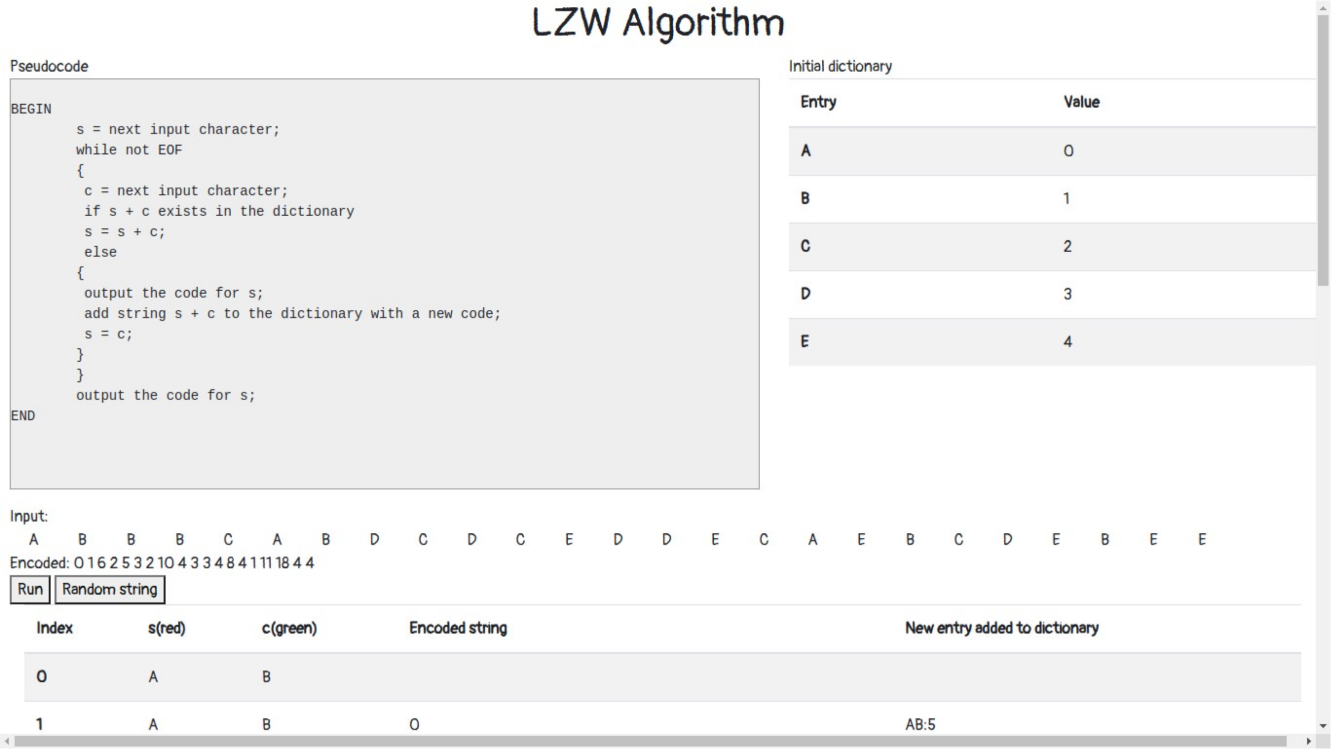
```
BEGIN
    s = next input character;
    while not EOF
    {
        c = next input character;
        if s + c exists in the dictionary
            s = s + c;
            store information of s
        else
        {
            output the code for s;
            add string s + c to the dictionary with a new code
            s = c;
            store information of s, s+c with a new code and newly encoded string
        }
    }
    store information of s
    output the code for s;
END
```

2. User Interface

Since this is a web project, I utilized HTML, CSS and JavaScript to implement all the UI and animation. The UI has 3 main parts: header, algorithm setup and algorithm execution. Algorithm setup displays the pseudocode and the initial dictionary used for encoding string. Algorithm execution section shows the input string, encoded string and all the information of each step for LZW algorithm. For each step of LZW algorithm, each symbol is colored green and red to illustrate the state of algorithm and all the information is stored in table.

There are two buttons: Run and Generate string. When “Run” button is clicked, it will trigger LZW algorithm and visualization. During this, both buttons are disabled so that nothing will interrupt the process. “Generate string” is used to randomly generate a new input string that can have length from 1 to 50 and only contains characters from “A” to “E”. The reason for this character limitation is that a larger range can cause the initial dictionary become too large, therefore, the page will not be user-friendly hard to navigate.

Sample Results



LZW Algorithm

Pseudocode

```
BEGIN
        s = next input character;
        while not EOF
        {
         c = next input character;
         if s + c exists in the dictionary
         s = s + c;
         else
        {
         output the code for s;
         add string s + c to the dictionary with a new code;
         s = c;
        }
        }
        output the code for s;
END
```

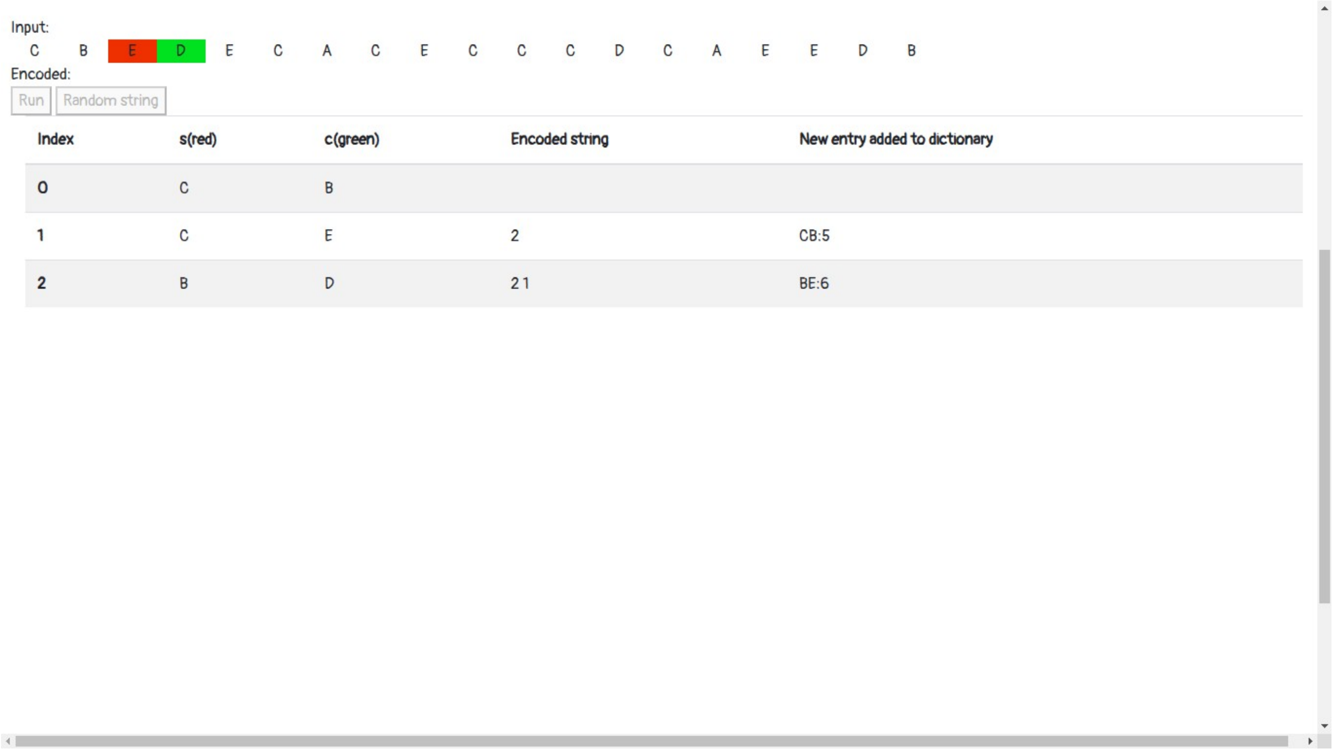
Initial dictionary

Entry	Value
A	0
B	1
C	2
D	3
E	4

Input:
A B B B C A B D C D C E D D E C A E B C D E B E E

Encoded: 0 1 6 2 5 3 2 10 4 3 3 4 8 4 1 11 18 4 4

Index	s(red)	c(green)	Encoded string	New entry added to dictionary
0	A	B		
1	A	B	0	AB:5

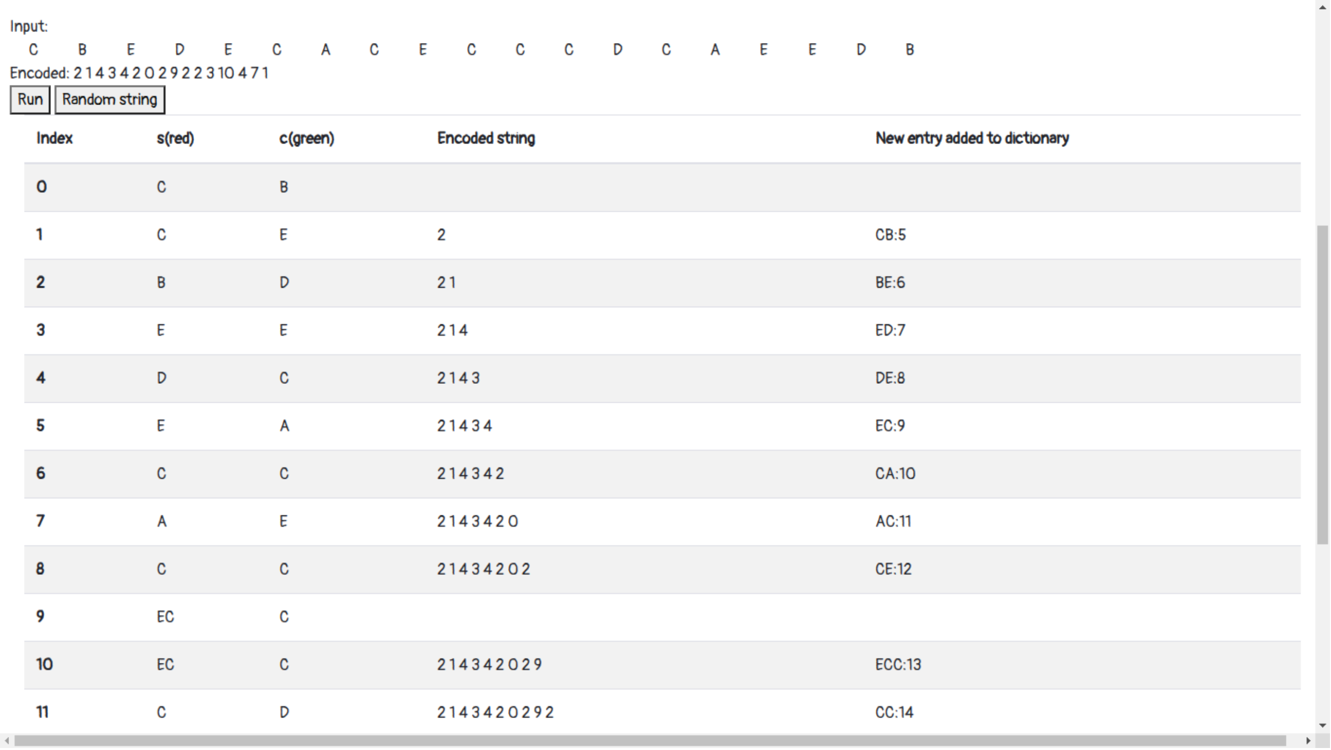


Input:

C B 🟥E 🟩D E C A C E C C C D C A E E D B

Encoded:

Index	s(red)	c(green)	Encoded string	New entry added to dictionary
0	C	B		
1	C	E	2	CB:5
2	B	D	2 1	BE:6



Input:

C B E D E C A C E C C C D C A E E D B

Encoded: 2 1 4 3 4 2 0 2 9 2 2 3 10 4 7 1

Index	s(red)	c(green)	Encoded string	New entry added to dictionary
0	C	B		
1	C	E	2	CB:5
2	B	D	2 1	BE:6
3	E	E	2 1 4	ED:7
4	D	C	2 1 4 3	DE:8
5	E	A	2 1 4 3 4	EC:9
6	C	C	2 1 4 3 4 2	CA:10
7	A	E	2 1 4 3 4 2 0	AC:11
8	C	C	2 1 4 3 4 2 0 2	CE:12
9	EC	C		
10	EC	C	2 1 4 3 4 2 0 2 9	ECC:13
11	C	D	2 1 4 3 4 2 0 2 9 2	CC:14