# Practico 4 – Ejercicio 2

a) Plantee los pseudocódigos de los algoritmos compresor y descompresor según el método LZW

b) Codifique el siguiente mensaje: “s1 s2 s3 s1 s2 s4 s1 s2 s3 s2 s2 s4 s1 s3 s4 s1 s2 s2 s4 s1"

c) Decodifique el mensaje comprimido en b) y compare con el original

## Compresor LZW

## Algoritmo

|  |  |  |  |
| --- | --- | --- | --- |
| Entrada | Salida | Diccionario | |
|  |  | 1 | S1 |
|  |  | 2 | S2 |
|  |  | 3 | S3 |
|  |  | 4 | S4 |
| S1 | 1 | 5 | S1-S2 |
| S2 | 2 | 6 | S2-S3 |
| S3 | 3 | 7 | S3-S1 |
| S1-S2 | 5 | 8 | S1-S2-S4 |
| S4 | 4 | 9 | S4-S1 |
| S1-S2 | 5 | 10 | S1-S2-S3 |
| S3 | 3 | 11 | S3-S2 |
| S2 | 2 | 12 | S2-S2 |
| S2 | 2 | 13 | S2-S4 |
| S4-S1 | 9 | 14 | S4-S1-S3 |
| S3 | 3 | 15 | S3-S4 |
| S4-S1 | 9 | 16 | S4-S1-S2 |
| S2-S2 | 12 | 17 | S2-S2-S4 |
| S3-S1 | 9 |  |  |

### Pseudocódigo

//Metodo para comprimir un mensaje

**public** **static** String comprimir(Vector<String> mensaje)

{

//Inicializo el diccionario

*initDiccionarioCod*(mensaje);

//Obtengo la cantidad de codigos que tengo en el diccionario para poder seguir

//creando a partir de ese indice

**int** j = *diccionarioCod*.size();

//Variables auxiliares

String salida = "";

String s = "";

String sant = "";

String ssig = "";

//Obtengo el primer simbolo del mensaje

s = *GetSimbolo*(mensaje);

**while**(s != **null**)

{

//Incremento en uno el indice del proximo codigo.

j++;

//Si el diccionario contiene el simbolo, trato de agregar otro.

**while**(*diccionarioCod*.containsKey(s))

{

//Guardo el simbolo actual en anterior

sant = s;

//Obtengo el siguiente simbolo

ssig = *GetSimbolo*(mensaje);

//Si no terminó el mensaje

**if**(ssig != **null**)

{

//Mi simbolo nuevo es el anterior y el siguiente.

s = sant + "-" + ssig;

}

**else**

{

//Si es fin del mensaje, no agrego mas simbolos

s = **null**;

}

}

//Busco la codificacion del simbolo y la agrego a la salida

salida += *diccionarioCod*.get(sant) + ",";

//Si no es el fin del mensaje

**if**(s != **null**)

{

//Agrego un nuevo simbolo a mi diccionario.

*diccionarioCod*.put(s, String.*valueOf*(j));

//el simbolo se vuelve el siguiente.

s = ssig;

}

}

//Retorno el mensaje codificado

**return** salida;

}

## Algoritmo de Descompresión

### Algoritmo

|  |  |  |  |
| --- | --- | --- | --- |
| Entrada | Salida | Diccionario | |
|  |  | 1 | S1 |
|  |  | 2 | S2 |
|  |  | 3 | S3 |
|  |  | 4 | S4 |
| 1 | S1 |  | - |
| 2 | S2 | 5 | S1-S2 |
| 3 | S3 | 6 | S2-S3 |
| 5 | S1-S2 | 7 | S3-S1~~-S2~~ |
| 4 | S4 | 8 | S1-S2-S4 |
| 5 | S1-S2 | 9 | S4-S1 |
| 3 | S3 | 10 | S1-S2-S3 |
| 2 | S2 | 11 | S3-S2 |
| 2 | S2 | 12 | S2-S2 |
| 9 | S4-S1 | 13 | S2-S4 |
| 3 | S3 | 14 | S4-S1-S3 |
| 9 | S4-S1 | 15 | S3-S4 |
| 12 | S2-S2 | 16 | S4-S1-S2 |
| 9 | S4-S1 | 17 | S2-S2-S4 |

### Pseudocódigo

//Metodo para descomprimir un mensaje

**public** **static** String descomprimir(Vector<String> mensaje)

{

//Obtengo el tamaño del diccionario con los simbolos originales.

**int** j = *diccionarioDeco*.size();

//Variables auxiliares

String salida = "";

String s = "";

String sant = **null**;

//Obtengo el primer codigo.

String c = *GetCodigo*(mensaje);

//Mientras que no sea el fin del mensaje

**while**(c != **null**)

{

//Obtengo el simbolo que se corresponde con el codigo

s = *diccionarioDeco*.get(c);

//Si no es el primer simbolo

**if**(sant != **null**)

{

//Incremento en la clave del proximo codigo a agregar.

j++;

//Si el simbolo no estaba codificado

**if**(s==**null**)

{

//lo codifico con la cadena anterior + el primer simbolo de la cadena anterior.

sant += "-" +sant.substring(0,sant.indexOf('-'));

//El simbolo pasa a ser el simbolo anterior.

s = sant;

}

**else**

{

//Si el simbolo es compuesto

**if**(s.contains("-"))

{

//A la cadena anterior le agrego el primer simbolo

sant += s.substring(0,s.indexOf("-"));

}

**else**

{

//Si no es compuesto, agrego a la cadena anterior, el simbolo actual.

sant += "-" + s;

}

}

//Agrego el nuevo codigo, con el nuevo simbolo en el diccionario.

*diccionarioDeco*.put(String.*valueOf*(j), sant);

}

//Guardo el simbolo actual en el anterior

sant = s;

//Agrego el simbolo descomprimido a la salida

salida += s + ",";

//Busco el siguiente codigo.

c = *GetCodigo*(mensaje);

}

//retorno la cadena descomprimida

**return** salida;

}