Instituto Politécnico Nacional

Escuela Superior de Cómputo

Práctica 4

Códigos Huffman

Acevedo Juárez Sebastián

Introducción:

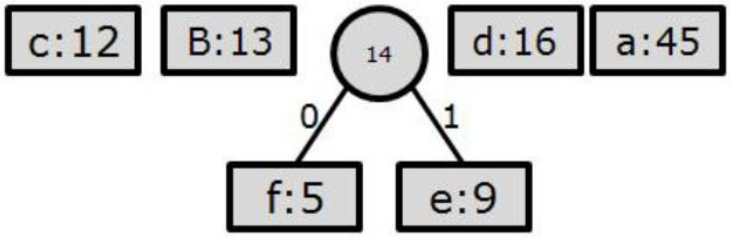
Para comprimir la información se busca representar un conjunto de símbolos (o cadena de símbolos) obtenidos a partir de un cierto alfabeto, usando el menor número de bits posible, pero preservando en todo momento la capacidad de descomprimir o decodificar la información. En general, el sistema que realiza el proceso directo lo llamamos compresor o codificador, mientras que el que reconstruye los datos originales (o una aproximación a ellos si realizamos compresión con pérdidas) lo llamamos descompresor o decodificador. El algoritmo de codificación/compresión Huffman se propuso en 1952 como una forma sencilla y optima de mapear cada símbolo de un alfabeto con un código (codeword) de longitud óptima. De esta forma, para comprimir cada símbolo de la cadena, simplemente debemos usar el código que se ha calculado mediante Huffman. Para conseguir esta asignación óptima, los símbolos se representan con códigos cuya longitud es inversamente proporcional a la probabilidad del símbolo. De esta forma, los símbolos cuya frecuencia es menor en la cadena se representan con códigos más largos, mientras que los símbolos con mayor frecuencia de apariciones representan con códigos más cortos.

Marco Teórico

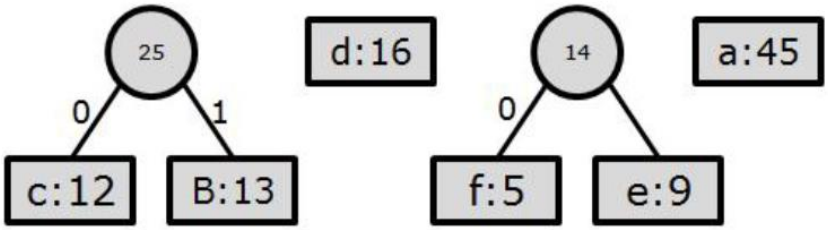
Consiste en la creación de un árbol binario en el que se etiquetan los nodos hoja con los caracteres, junto a sus frecuencias, y de forma consecutiva se van uniendo cada pareja de nodos que menos frecuencia sumen, pasando a crear un nuevo nodo intermedio etiquetado con dicha suma. Se procede a realizar esta acción hasta que no quedan nodos hoja por unir a ningún nodo superior, y se ha formado el árbol binario. Posteriormente se etiquetan las aristas que unen cada uno de los nodos con ceros y unos (hijo derecho e izquierdo, respectivamente, por ejemplo. El código resultante para cada carácter es la lectura, siguiendo la rama, desde la raíz hacia cada carácter (o viceversa) de cada una de las etiquetas delas aristas.

Representación de lo antes mencionado:

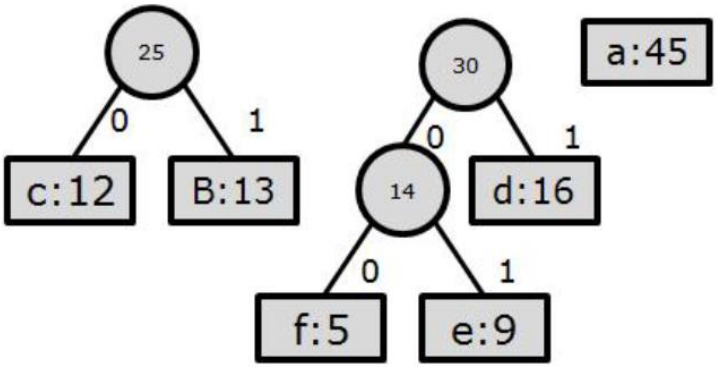
Seleccionamos los dos símbolos con menor frecuencias y los agregamos a un nodo con la suma de ambos, recalcando que el de menor frecuencia ira a la izquierda con el valor 0 en su rama.



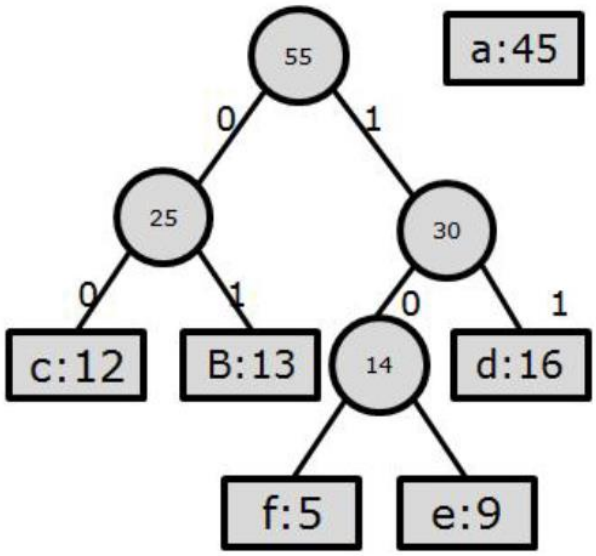
Repetimos el proceso de seleccionar los dos menores sumando sus cantidades.



Observemos que, ahora los dos menores incluyen a un nodo que hemos creado así que usamos este junto con el segundo menor y se crea una nueva rama y un nuevo nivel.



Seguimos con este proceso hasta terminar con los nodos restantes y obtener el árbol completo



Implementación:

Para poder implementar este problema, realizamos los siguientes pasos:

1. Desplegar menú con los archivos disponibles a codificar.
2. Ejecutar función madre Huffman() del archivo seleccionado
    a. Obtener frecuencia de los caracteres.
    b. Crear un árbol binario en base a las frecuencias.
    c. Crear un código en base al árbol previamente creado.
    d. Codificar el archivo seleccionado.
    e. Decodificar la compresión y crear un nuevo archivo con el mensaje decodificado.

Al usar una imagen, los pasos tienen algún cambio, debido a que primero se tiene que convertir la imagen a un mapeado, para así poder escribirlo en un txt y realizar los pasos mencionados.

La imagen es representada mediante una matriz que representan los pixeles de una imagen. Al trabajar con una foto en escala de grises, cada celda de la matriz representa la intensidad de color. Este mapa se imprime en un nuevo archivo, al cual se le aplica Huffman.

Al ejecutar la función Huffman(), el archivo de entrada es un archivo ya existente. Sin embargo, el archivo de salida no existe si no se ha ejecutado previamente el programa. En este archivo, se encuentra el mensaje decodificado.

Ejecución:

Para “ADN1.txt”

```
Seleccione una opción
1 .- ADN1.txt
2 .- ADN2.txt
3 .- ADN3.txt
4 .- Texto.txt
1

========================================== Codificacion ==========================================

000100001100101101010010101011100101010101111101011001011011011011011011011001100101111111110101000111101010001011101101101010110
110101010110111100111111101111010101010001011111010111001100010011011100110100010011010011111000011011110100111100001000001111000000
001101001101011001011011000110101001011001110110101111011111011110101010011101111000101100101010011110101001110101011111101010101100100
100011110011001111000011011001111111001111101111010100111011111010011110000111101110100010101010101001101000010010111011101110101111
111100010011000010110101011011010011110000011010011010011011110000110001011010011001011001011011010000010000000000101100100010101010010
110000101101100100011101000011101010000010100001001100010110110000110001101010101010101010101010010111010101111110000001111100011111
001111101110100111000011111100110100110101100111111111000101010110110100011011111100100111110110100010111000011011000110001110111010
010101100110111010010101101010111011001110101111101111110101110101111100001001010111110101110000011011011101101101101111010001110
111100110001100111111010101011011001011010111111010011100111100000011011011100010001100001101010010101110101011010111001101110
011111011010101110011100010100011101100100101010101100101101110100111000011010101010000001101001110110011110111001111000110001010
111110111011100100010011100001001110100000001111111110110000110010011000000011101010000000011001100001110100000100100010000100011110
110011111010101111110010000000000011101011111010101101010101011010010000011100101010011000000110110111110011111000110101111001100
111111111101010010101111110000111010101010101011111011100011111100100111000111000110000011001000000011111110000011010011001111010111
111101001101111111111100110101110110000111010010110100000111100001111001111100111011101011010110110100110101011001010100000011010011
011001010011010110001011011001100101111101000101001111000011101001010000111110100101011010010101011011111001100101000000011010010011111
100010100101100110010101011001000100010101011010010000010110010011011001101011110110011000011001011010000110010100011011101001011001
011011101010010101010110100101011010101001011011111010011010100010001010101011010000011101111110010010110100000111011111001001011100001101011011000110010
101010010100001010011001010100001101010011000101001100110100110101100110101010000000111100101111001100011010011011011110100110010110110
011101101100000000101110111111110111000001011010111111000110100011001110101011000101100011010101010001101100110101011000110010110100
101110111010101010010111011001100111011001111010010011001101010100101111000111000011010101100101100010011101010110000010100010011010110100100010
000101101010101001100101100110111010101011100110100011111111110000110001101010000001100011111111011101100100101110111010011001011101100100111101
011010101000101011101100110011011111000100110001001001011100110010101001001111010000111010011011101111101111011011010101011111101010
101001011011110100110100010001010101011010000011101111110010010111000001101010101010101010101010101010101010010101010100000000000000
110011001111111111010101010101101001010101110101010010111111010011011111010001000101010101110101110011010001111111111000011000101100101
101101111100111111100010001011110110101001111111000100110000010000101101101011001011011011111001111110001000101111011010100111111100
010011000000100001011011
```

```
========================================== Arbol Binario ==========================================
Nota:
' ' = espacio
\n = salto de linea
{} = Nodo
(a,23) = (letra,frecuencia)
[0] = Camino izquiera
[1] = Camino derecha

 [TOTAL] { 1687 }
         [0] { 757 }
                 [0] ( A , 349 )
                 [1] ( C , 408 )
         [1] { 930 }
                 [0] ( T , 461 )
                 [1] ( G , 469 )

========================================== CASO ADN1.txt  ==========================================

Archivo de entrada = ADN1.txt ||  Archivo de salida = newADN1.txt

Bits totales sin comprimir: 13496
Bits totales después de comprimir: 3374
Porcentaje total de compresión = 25.0 %
Tiempo de compresión y decompresión:  2.4442 milisegundos
```

Simbolo	Frecuencia	Codigo
A	349	00
C	408	01
G	469	11
T	461	10

Para “ADN2.txt”

```
Seleccione una opción
1 .- ADN1.txt
2 .- ADN2.txt
3 .- ADN3.txt
4 .- Texto.txt
2

========================================== Codificacion ===========================================

01110101100100101111010000100011111111110101111001111001011001011001011001011001100111110101010111101111010111101111110110010111111100
10111111110010100110101000101011111111011111101011110011001110110111001101101000100101101101001011011101011011010010100010110100101010
01101101101111001111001001110000001111001110110000101000101000000010011000101001111100111111011010111101100010111110101011111111001101
00011010011001101001011011001110101001101000101011110110101110101101101001011010111011011111111111111011011010100111110111011101111110
10100111011001010010111111001011011010010111000011000011001010010110011111000011001111001111001011010101000101010111110011011111110100
10010100101100110111101101011000000001010000011101100111110010010101110101100100101111010000100011111111110101111001111001011001011001100
10110010110011001111101010101111011110101111011111101100101111110010111111100101001101010001010111111110111111010111110011001110110111
10011011010001001011011010010110111010110110100101000101101001010101101010101101101111001111001001110000001111001110110000101000101000000010
01100010100111110011111101101011110110001011111010101111111001101000110100110011010010110110011101010011010001010111101101011101011101
10100101101011101101111111111111101101101010011111011101101111101010011101100101001011111001011011010010111000011000011001010010110
01111100001100111100111100101101010100010101011111001101111111010010010100101100110111101101011000000000101000000111011001111100100101101
01110000000000000000000000000111100000010101001010110100001110101100100101111010000100011111011001011001011001100111110101010111110111
10101110111111011001011111001011111110010100110101000101011111111011111101011111001100111011011101010111011010010111011010110100101011001
00101111010000100011001111001011001011001011001011001100111110101010111101111010111101111110110010111111001011111110010100110101000010
10111111110111111010111110011001110110111001101101000100101101101001010101101001010101101101101111001111001001110110000101000101000000
10011000101001111100111111011010111101100010111110101011111111001101000110100110011010010110110011101010011010001010111101101011101001
10110101001111101110111011111010100111011001010010111111001011011010010111110000110011110011110010110101010001010101111100110111111101
00101011010110101000110011110101011100000000011100101000001100000001000100110000110110100001000011110011001111111100000001010000011101
1001111100100101
```

```
============================================ Arbol Binario ===========================================
Nota:
' ' = espacio
\n = salto de linea
{} = Nodo
(a,23) = (letra,frecuencia)
[0] = Camino izquiera
[1] = Camino derecha

 [TOTAL] { 1214 }
         [0] { 474 }
                 [0] ( T , 223 )
                 [1] ( A , 251 )
         [1] { 740 }
                 [0] ( G , 343 )
                 [1] ( C , 397 )

=========================================== CASO ADN2.txt  ===========================================

Archivo de entrada = ADN2.txt ||  Archivo de salida = newADN2.txt

Bits totales sin comprimir: 9712
Bits totales después de comprimir: 2428
Porcentaje total de compresión = 25.0 %
Tiempo de compresión y decompresión:  2.0505 milisegundos
```

Simbolo	Frecuencia	Codigo
A	251	01
C	397	11
G	343	10
T	223	00

Para “ADN3.txt”

```
Seleccione una opción
1 .- ADN1.txt
2 .- ADN2.txt
3 .- ADN3.txt
4 .- Texto.txt
3

=========================================== Codificacion ===========================================

01110101100100101111010000100011111111111010111100111100101100101100101100101100110011111010101011110111101011110111111011001011111100
10111111110010100110101000101011111111011111101011111001100111011011100110110100010010110110100101101110101101101001010001011010010101
01101101101111001111001001110000000111100111011000010100010100000000100110001010011111001111110110101111011000101111101010111111111001101
00011010011001101001011011001110101001101000101011110110101110101101101001011010111011011111111111111101101101010011111011101110111110
10100111011001010010111111001011011010010111000011000011001010010110011111000011001111001111001011010101000101010111110011011111110100
10010100101100110111101101011000000000101000001110110011111001001010111010110010010111101000010001111111111101011110011110010110010110
01011001011001100111110101010111101111010111101111110110010111111001011111111001010011010100010101111111101111110101111100110011101101 1
10011011010001001011011010010110111010110110100101000101101001010101101101101111001111001001110000000111100111011000010100010100000000010
01100010100111110011111101101011110110001011111010101111111110011010001101001100110100101101100111010100110100010101111011010111010110 1
10100101101011101101111111111111111011011010100111110111011101111101010011101100101001011111100101101101001011100001100001100101001011 0
01111110000110011110011110010110101010001010101111100110111111101001001010010110011011110110101100000000101000001110110011111001001011 0
01110000000000000000000000000001111000000101001000001110110011111001001 01
```

```
============================================ Arbol Binario ============================================
Nota:
' ' = espacio
\n = salto de linea
{} = Nodo
(a,23) = (letra,frecuencia)
[0] = Camino izquiera
[1] = Camino derecha

 [TOTAL] { 773 }
         [0] { 306 }
                 [0] ( T , 141 )
                 [1] ( A , 165 )
         [1] { 467 }
                 [0] ( G , 215 )
                 [1] ( C , 252 )

=========================================== CASO ADN3.txt  ===========================================

Archivo de entrada = ADN3.txt ||  Archivo de salida = newADN3.txt

Bits totales sin comprimir: 6184
Bits totales después de comprimir: 1546
Porcentaje total de compresión = 25.0 %
Tiempo de compresión y decompresión:  1.4894 milisegundos
```

Simbolo	Frecuencia	Codigo
A	165	01
C	252	11
G	215	10
T	141	00

Para “Texto.txt”

```
========================================== Arbol Binario ==========================================
Nota:
' ' = espacio
\n = salto de linea
{} = Nodo
(a,23) = (letra,frecuencia)
[0] = Camino izquiera
[1] = Camino derecha

 [TOTAL] { 1887 }
         [0] { 792 }
                 [0] { 353 }
                         [0] { 165 }
                                 [0] { 79 }
                                         [0] ( p , 39 )
                                         [1] { 40 }
                                                 [0] ( y , 19 )
                                                 [1] { 21 }
                                                         [0] { 9 }
                                                                 [0] { 4 }
                                                                         [0] { 2 }
                                                                                 [0] ( V , 1 )
                                                                                 [1] ( j , 1 )
                                                                         [1] ( W , 2 )
                                                                 [1] { 5 }
                                                                         [0] ( q , 2 )
                                                                         [1] ( ( , 3 )
                                                         [1] { 12 }
                                                                 [0] { 6 }
                                                                         [0] ( ) , 3 )
                                                                         [1] ( - , 3 )
                                                                 [1] { 6 }
                                                                         [0] { 3 }
                                                                                 [0] ( z , 1 )
                                                                                 [1] { 2 }
                                                                                         [0] ( 3 , 1 )
                                                                                         [1] ( B , 1 )
                                                                         [1] ( T , 3 )
                                 [1] { 86 }
                                         [0] ( g , 42 )
                                         [1] ( f , 44 )
                         [1] ( e , 188 )
                 [1] { 439 }
                         [0] { 206 }
                                 [0] { 100 }
                                         [0] { 49 }
                                                 [0] ( b , 24 )
                                                 [1] { 25 }
                                                         [0] { 12 }
                                                                 [0] ( A , 6 )
                                                                 [1] ( N , 6 )
                                                         [1] ( . , 13 )
                                         [1] ( h , 51 )
                                 [1] ( r , 106 )
                         [1] { 233 }
                                 [0] { 113 }
                                         [0] ( l , 54 )
                                         [1] { 59 }
                                                 [0] { 27 }
                                                         [0] ( v , 13 )
                                                         [1] { 14 }
                                                                 [0] { 6 }
                                                                         [0] ( k , 3 )
                                                                         [1] ( x , 3 )
                                                                 [1] { 8 }
                                                                         [0] ( \n , 4 )
                                                                         [1] { 4 }
                                                                                 [0] { 2 }
                                                                                         [0] ( E , 1 )
                                                                                         [1] ( I , 1 )
                                                                                 [1] ( H , 2 )
                                                 [1] ( u , 32 )
                                 [1] ( t , 120 )
   [1] { 1095 }
                 [0] { 517 }
                         [0] { 253 }
                                 [0] ( a , 124 )
                                 [1] ( i , 129 )
                         [1] { 264 }
                                 [0] ( n , 129 )
                                 [1] { 135 }
                                         [0] ( d , 66 )
                                         [1] { 69 }
                                                 [0] ( m , 34 )
                                                 [1] { 35 }
                                                         [0] { 16 }
                                                                 [0] ( D , 8 )
                                                                 [1] ( w , 8 )
                                                         [1] ( , , 19 )
                 [1] { 578 }
                         [0] { 285 }
                                 [0] ( o , 139 )
                                 [1] { 146 }
                                         [0] ( c , 69 )
                                         [1] ( s , 77 )
                         [1] ( ' ' , 293 )
```

========== CASO Texto.txt ==========

Archivo de entrada = Texto.txt || Archivo de salida = newTexto.txt

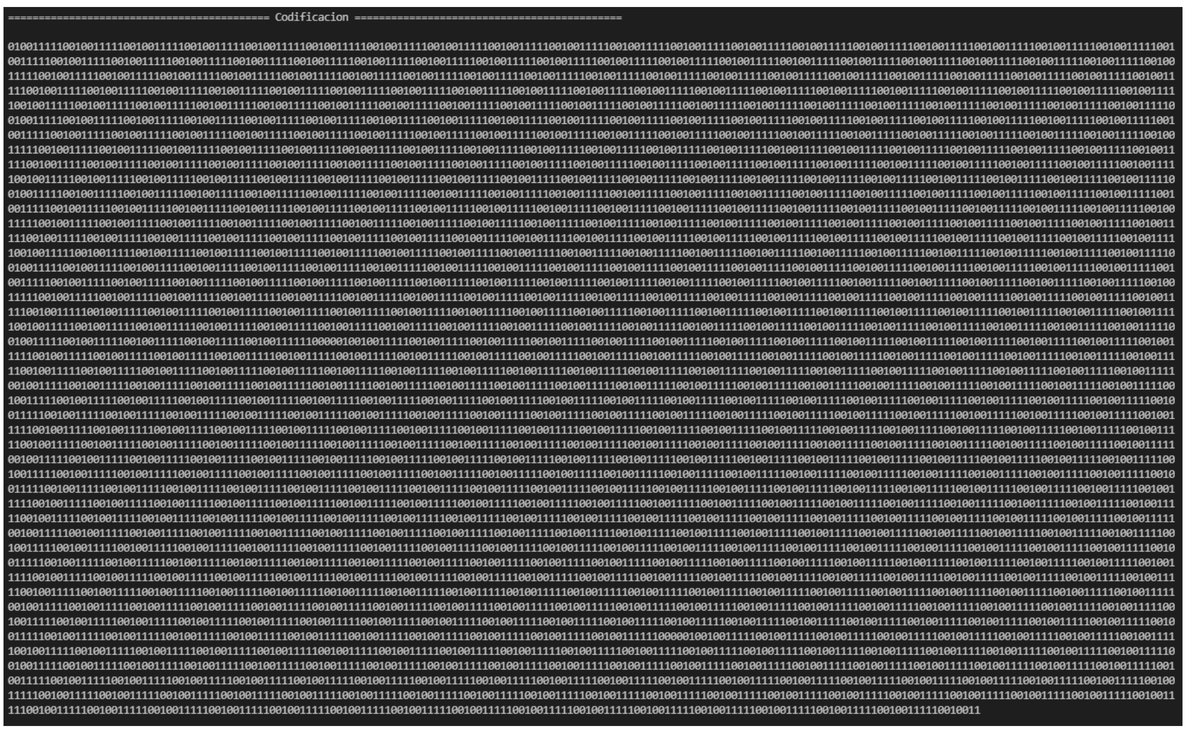
Bits totales sin comprimir: 15096
Bits totales después de comprimir: 8162
Porcentaje total de compresión = 54.07 %
Tiempo de compresión y decompresión: 3.9375 milisegundos

Simbolo	Frecuencia	Codigo
' '	293	111
(	3	000011011
)	3	000011100
,	19	1011111
-	3	000011101
.	13	0100011
3	1	00001111010
A	6	01000100
B	1	00001111011
D	8	10111100
E	1	01101011100
H	2	0110101111
I	1	01101011101
N	6	01000101
T	3	000011111
V	1	0000110000
W	2	000011001
\n	4	011010110
a	124	1000
b	24	010000
c	69	11010

d	66	10110
e	188	001
f	44	00011
g	42	00010
h	51	01001
i	129	1001
j	1	0000110001
k	3	011010100
l	54	01100
m	34	101110
n	129	1010
o	139	1100
p	39	00000
q	2	000011010
r	106	0101
s	77	11011
t	120	0111
u	32	011011
v	13	0110100
w	8	10111101
x	3	011010101
y	19	000010
z	1	0000111100

Para “Imagen.PNG”

```
Seleccione una opción
1 .- ADN1.txt
2 .- ADN2.txt
3 .- ADN3.txt
4 .- Texto.txt
5 .- Imagen.png
5
C:\Users\user\AppData\Local\Programs\Python\Python39\lib\site-packages\PIL\Image.py:975: UserWarning: Palette images wit
h Transparency expressed in bytes should be converted to RGBA images
  warnings.warn(
[[255 255 255 ... 255 255 255]
 [255 255 255 ... 255 255 255]
 [255 255 255 ... 255 255 255]
 ...
 [255 255 255 ... 255 255 255]
 [255 255 255 ... 255 255 255]
 [255 255 255 ... 255 255 255]]
```



```
========================================== Arbol Binario ==========================================
Nota:
' ' = espacio
\n = salto de linea
{} = Nodo
(a,23) = (letra,frecuencia)
[0] = Camino izquiera
[1] = Camino derecha

 [Inicio] { 619705 }
         [0] ( ' ' , 302253 )
         [1] { 317452 }
                 [0] { 132700 }
                         [0] { 26084 }
                                 [0] { 11093 }
                                         [0] { 4753 }
                                                 [0] { 2107 }
                                                         [0] ( 4 , 1020 )
                                                         [1] { 1087 }
                                                                 [0] ( \n , 389 )
                                                                 [1] ( 1 , 698 )
                                                 [1] ( 8 , 2646 )
                                         [1] { 6340 }
                                                 [0] ( 6 , 2898 )
                                                 [1] { 3442 }
                                                         [0] { 1647 }
                                                                 [0] ( 0 , 712 )
                                                                 [1] ( 3 , 935 )
                                                         [1] ( 7 , 1795 )
                                 [1] ( 2 , 14991 )
                         [1] ( 9 , 106616 )
                 [1] ( 5 , 184752 )
```

```
========================================== CASO p1cadena.txt  ==========================================

Archivo de entrada = p1cadena.txt ||  Archivo de salida = newp1cadena.txt

Bits totales sin comprimir: 4957640
Bits totales después de comprimir: 1126410
Porcentaje total de compresión = 22.72 %
Tiempo de compresión y decompresión:  682.2523 milisegundos
```

Simbolo	Frecuencia	Codigo
' '	302253	0
0	712	10001100
1	698	10000011
2	14991	1001
3	935	10001101
4	1020	1000000
5	184752	11
6	2898	100010
7	1795	1000111
8	2646	100001
9	106616	101
\n	389	10000010

Tablas de Resultados

Texto.txt

Simbolo	Frecuencia	Codigo
' '	293	111
(	3	000011011
)	3	000011100
,	19	1011111
-	3	000011101
.	13	0100011
3	1	00001111010
A	6	01000100
B	1	00001111011
D	8	10111100
E	1	01101011100
H	2	0110101111
I	1	01101011101
N	6	01000101
T	3	000011111
V	1	0000110000
W	2	000011001
\n	4	011010110
a	124	1000
b	24	010000
c	69	11010
d	66	10110
e	188	001
f	44	00011
g	42	00010
h	51	01001
i	129	1001
j	1	0000110001
k	3	011010100
l	54	01100
m	34	101110
n	129	1010
o	139	1100
p	39	00000
q	2	000011010
r	106	0101
s	77	11011
t	120	0111
u	32	011011
v	13	0110100
w	8	10111101
x	3	011010101
y	19	000010
z	1	0000111100

ADN1.txt

Simbolo	Frecuencia	Codigo
A	349	00
C	408	01
G	469	11
T	461	10

ADN2.txt

Simbolo	Frecuencia	Codigo
A	251	01
C	397	11
G	343	10
T	223	00

ADN3.txt

Simbolo	Frecuencia	Codigo
A	165	01
C	252	11
G	215	10
T	141	00

Imagen.PNG

Simbolo	Frecuencia	Codigo
' '	302253	0
0	712	10001100
1	698	10000011
2	14991	1001
3	935	10001101
4	1020	1000000
5	184752	11
6	2898	100010
7	1795	1000111
8	2646	100001
9	106616	101
\n	389	10000010

Conclusión

En la práctica se llevó a cabo un método de codificación (compresión y descompresión) para el cual se formuló una serie de algoritmos recursivos y por demás para poder tener una compresión satisfactoria, nos percatamos que es una codificación sin pérdida de datos. Teniendo un porcentaje de compresión promedio del 30%.La asignación de claves para cada carácter depende de la cantidad de veces que se repita cada carácter en el texto mientras se repita más veces el carácter, tendrá una clave más pequeña mientras menos se repita el carácter tendrá una clave más grande. Independientemente del archivo a comprimir, lo que determina que tan compacto sea el archivo resultante, será el número de veces que se repita los caracteres y símbolos que estén en el mismo. Con esto es evidente que una compresión sumamente eficiente es cuando la gran mayoría de los caracteres se repiten. De forma particular tomando en cuenta el lenguaje que se utilizó (Python) nos apoyamos en algunas librerías que facilitaron el análisis de datos, así como su acomodo e impresión de estos. Observamos que de igual forma se pudieron comprimir imágenes PNG con el mismo principio a coste de operaciones mayor.