INSTITUTO POLITÉCNICO NACIONAL

ESCUELA SUPERIOR DE CÓMPUTO

UNIDAD DE APRENDIZAJE

TEORÍA COMPUTACIONAL

TAREA

PRÁCTICA 1

ALUMNO

BARRERA ESTRELLA EMANUEL

PROFESOR

JUÁREZ MARTÍNEZ GENARO

GRUPO

2CM1

Introducción

Una cadena es una secuencia finita de caracteres de símbolos elegidos de un alfabeto. Por ejemplo, 01101 es una cadena del alfabeto binario$\sum = \{0, 1\}$.

La cadena vacía es la cadena con cero ocurrencias con los símbolos. La cadena está denotada por ε y es una cadena que puede ser elegida de cualquier alfabeto.

Si $\sum$ es un alfabeto, podemos expresar el conjunto de todas las cadenas de cierta longitud del alfabeto dado usando una notación exponenecial. Definimos $\sum^k$ como el conjunto de cadenas de longitud k, cada una de ellas con símbolos dentro de $\sum$.

Independientemente de que $\sum$sea,$\sum^0 = \{\varepsilon\}$, ya que, ε es la única cadena cuya longitud es cero.

Entonces si $\sum = \{0, 1\}$, entonces $\sum^1 = \{0, 1\}$, $\sum^2 = \{00, 01, 10, 11\}$,

$\sum^1 = \{000, 001, 010, 011, 100, 101, 110, 111\}$ y así sucesivamente.

El conjunto de todas las cadenas sobre el alfabeto $\sum$es convencionalmente denotado $\sum^*$. Diciéndolo de otra forma:

$$\sum{}^* = \sum{}^0 \cup \sum{}^1 \cup \sum{}^2 \cup \ldots$$

A veces, se puede excluir la cadena vacía del resto del conjunto de las cadenas. El conjunto de las cadenas no vacías del alfabeto $\sum$ es denotado$\sum^+$. Así, dos equivalencias apropiadas serían:

$$\sum{}^+ = \sum{}^1 \cup \sum{}^2 \cup \sum{}^3 \cup \ldots$$

$$\sum{}^* = \sum{}^0 \cup \sum{}^+$$

Desarrollo

La práctica que se presenta a continuación es de un programa que pide un número entero k tal que, $0 \leq$ k ≤ 100, para aproximar al universo $\sum^*$, de forma que $\sum^k$ es el sigma más grande conjunto de cadenas de longitud k unión con los demás conjuntos de menor dimensión a k

$$\sum{}^+ = \sum{}^1 \cup \sum{}^2 \cup \sum{}^3 \cup \ldots \sum{}^k$$

Si consideramos que k se aproxima a un número muy grande, se puede decir que $\sum^k = \sum^*$ si $k \to \infty$.

Implementando este programa en lenguaje C, se tiene el siguiente código:

```
void numeroABinario (char **cadena, long n, int bits);
int main ()
{
  int n, i, l, opc;
  long o;
  char *cadena, *s_opc, *s_n;
  FILE *archivo;
  do
    {
      printf ("Generador de universo Sigma hasta n\n");
      printf ("1)Generar universo de manera manual\n");
      printf ("2)Generar universo de manera aleatoria\n");
      printf ("3)Salir\n");
      scan (&s_opc);
    }
  while (!(cadenaEsNumero (&s_opc)));
  opc = cadenaANumero (&s_opc);
  if (opc >= 1 && opc <= 2)
    {
```

La opción de generar el universo a partir de un número que ingrese el usuario, trae consigo todo el contenido del conjunto total de los valores, cuando se tiene un número específico, éste quitará su forma compleja de tener que buscar el valor adecuado.

```
if (opc == 1)
  {
    printf ("Ingrese la n de Sigma:\n");
    do
      {
        scan (&s_n);
      }
    while (!(cadenaEsNumero (&s_n)));
    n = cadenaANumero (&s_n);
  }
if (opc == 2)
```

En el siguiente fragmento se genera un Random que brinda un entero al azar con el contenido de la longitud máxima de cadena en el universo.

```
if (opc == 2)
  {
    srand (time (NULL));
  //numero aleatorio entre 1 y 100
    n = rand () % (101);
    printf ("Universo hecho aleatoriamente hasta %d\n",n);
  }
```

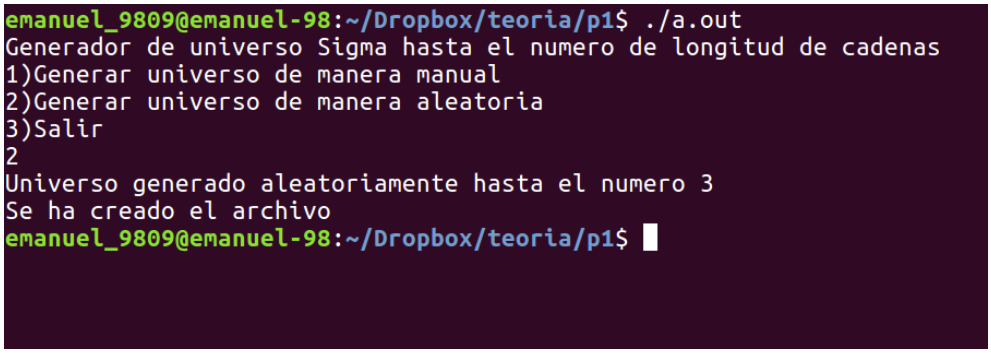
En este fragmento se escribe un fichero para exportar el universo en un formato adecuado, básico y muy simple para escribir en ficheros

```
        system ("rm sigma.txt");
        system ("echo > sigma.txt");
        archivo = fopen ("sigma.txt", "w");
        if (i >= 0)
          {
            fprintf (archivo, "{%c", 163);
            for (i = 0; i < n; i++)
              {
                for (o = 1, l = 2; o < i + 1; o++)
                  l *= 2;
                for (o = 0; o < l; o++)
                  {
                    fprintf (archivo, ",");
                    numeroABinario (&cadena, o, i + 1);
                    fprintf (archivo, "%s", cadena);
                  }
              }
            fprintf (archivo, "}\n");
            printf ("Se ha creado el archivo\n");
          }
        fclose (archivo);
      }
    return 0;
}
```

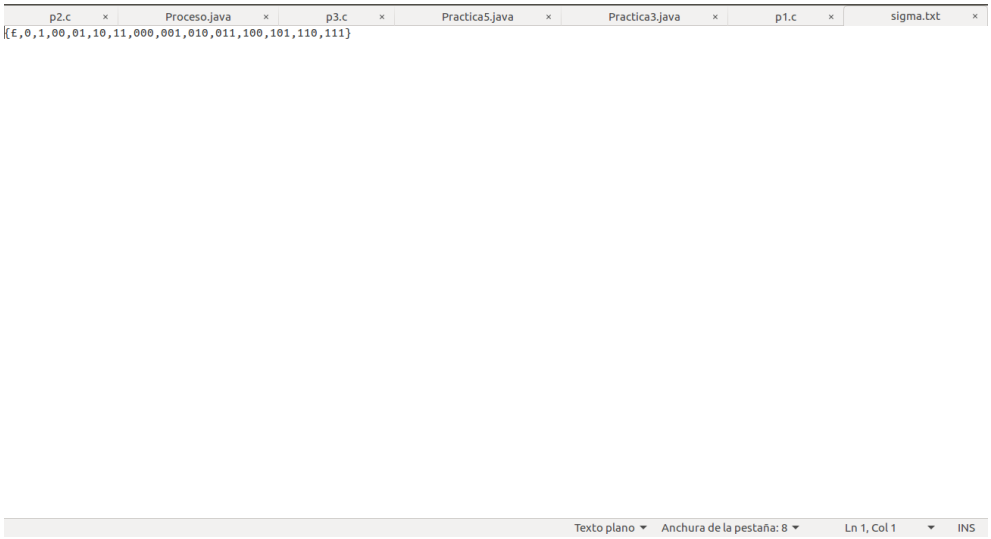
Finalmente, se tiene el método principal que ejecuta la conversión de cada número a una cadena de 'n' bits, que se acomoda en los espacios requeridos, con relleno de bits para cumplir siempre la longitud de la cadena y que al hacerlo de manera automática, genera todos los números hasta 2^n.

```
int cadenaEsNumero (char **cadena)
{
  int numero = 1;
  int i = 0;
  while ((*cadena)[i] != '\0')
    {
      if (!((*cadena)[i] >= 48 && (*cadena)[i] <= 57))
        {
          numero = 0;
          break;
        }
      i++;
    }
  if (i < 1)
    {
      numero = 0;
    }
  return numero;
```

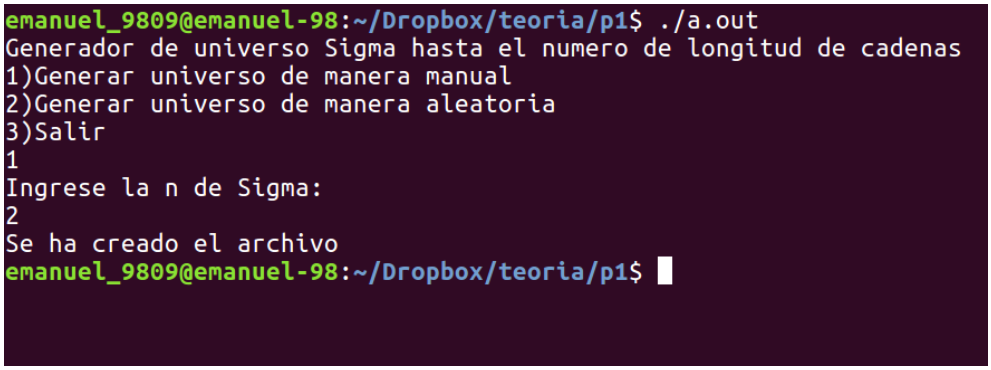
La ejecución sería:



Y después se genera en un archivo txt todo el conjunto de cadenas de longitud $i = 0, 1, 2, \ldots, k$ donde $0 \leq i \leq k$:



Para determinar un universo de manera manual, ingrresamos los valores en la linea de comando:



Donde su salida en el archivo de texto es: