**Instituto Politécnico Nacional**

**Escuela Superior de Cómputo**

**Teoría Computacional**

**Programa 01:**

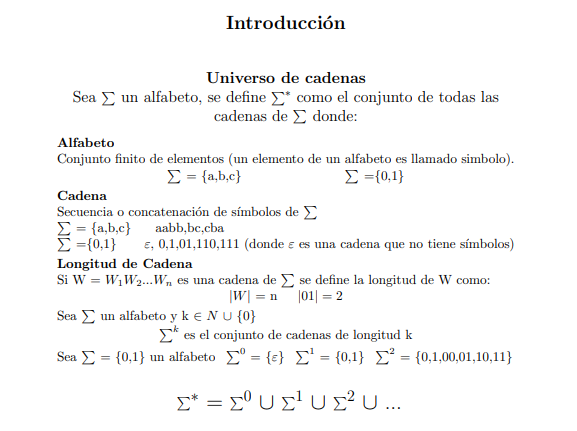
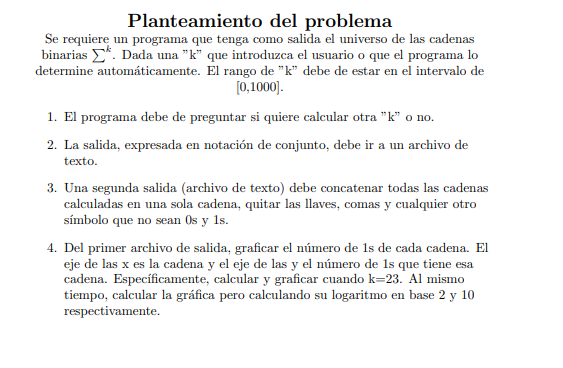
**Universo de las cadenas binarias**

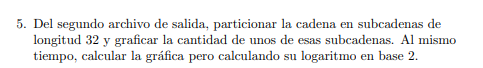
**Alumna: Solis Hernández Viridiana**

**Fecha de entrega:**

**31/10/2020**

**Juárez Martínez Genaro**





# Implementación

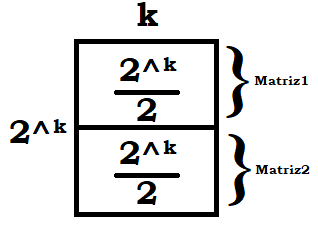
Se muestra un menú el cual tendrá como opciones:

1. Insertar valor de k: el usuario ingresa un entero k que se encuentren el intervalo [0,1000].
2. Generar valor aleatorio de k: el programa utiliza la función rand() para generar un número entero aleatorio k que se encuentre en el intervalo [0,1000].
3. Salir: finaliza el programa.

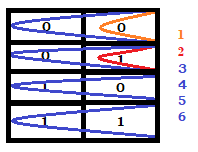
(En caso de no seleccionar una opción valida el programa volverá a solicitar una opción).

Una vez obtenido un número entero k ya sea de manera aleatoria o no, se crean dos matrices dinámicas, cada una tendrá como tamaño 2*k*/2 filas y k columnas.

Se genera la mitad de la tabla de verdad de 2*k* (por método de la división) y se guarda en matriz1 mientras la negación se guarda en la matriz2.(Figura1)

Figura1.

Se abre un primer archivo de texto donde imprimiremos \* en notación de conjunto al mismo tiempo se abre un archivo extra donde se mostrará el número de unos por cadena (el cual nos ayudará para graficar).

Figura2.

(Así es como se fueron imprimiendo las cadenas en el archivo utilizando ciclos for).

Se abre un segundo archivo de texto donde imprimiremos la concatenación de todas las cadenas del primer archivo, al mismo tiempo se particiona la cadena del segundo archivo en cadenas de longitud 32 y se abre un segundo archivo extra donde mostrara el número de unos por cadena (el cual nos ayudara para graficar).

Una vez terminado el cálculo muestra un segundo menú el cual tiene como opciones:

1. Ingresar una nueva k: lleva al menú principal.
2. Salir: finaliza el programa.

Se utilizó el software GNUplot para graficar lo solicitado.

4

# Funcionamiento

## Ejecución de ”Programa01.cpp”, ”Programa01.exe” Menú Principal

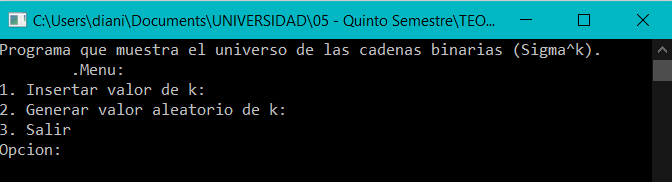


Figura3.

**PRUEBA [1]**

El usuario selecciona la Opción 1 del Menú Principal.

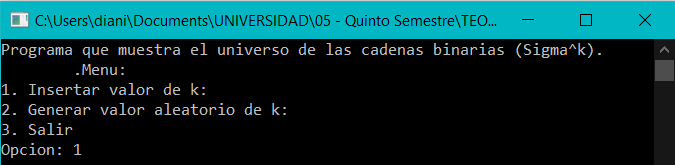


Figura4.

## El usuario ingresa un entero k = 23 el cual se encuentra en el intervalo [0,1000]



Figura5.

## Menú Secundario

Una vez finalizados todos los cálculos con k = 23 aparece el menú secundario

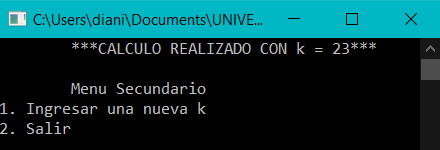
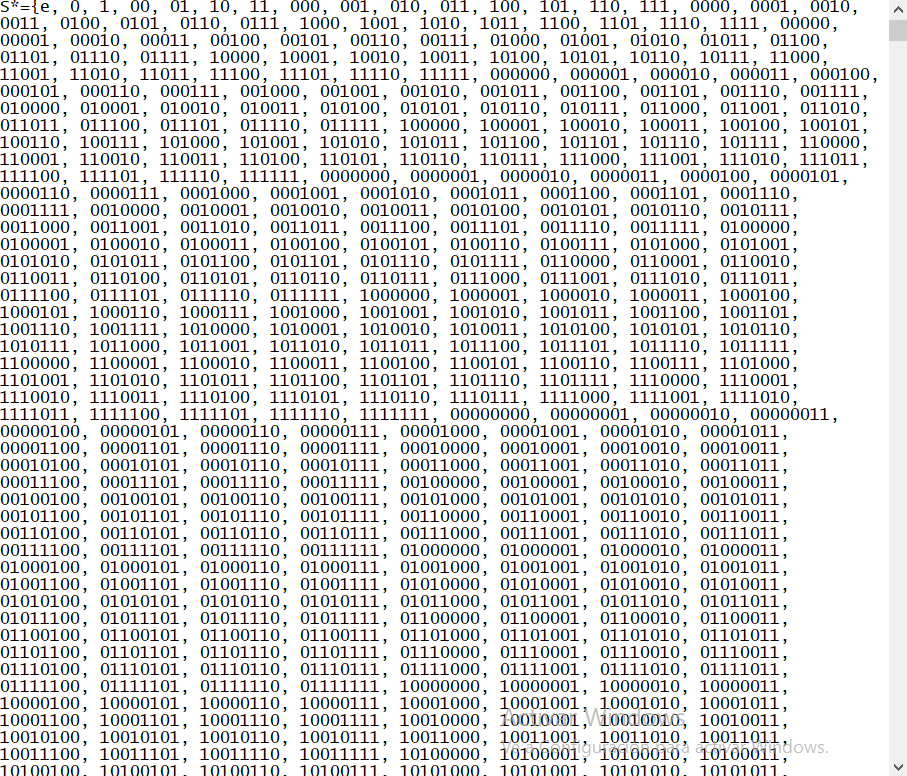


Figura6.

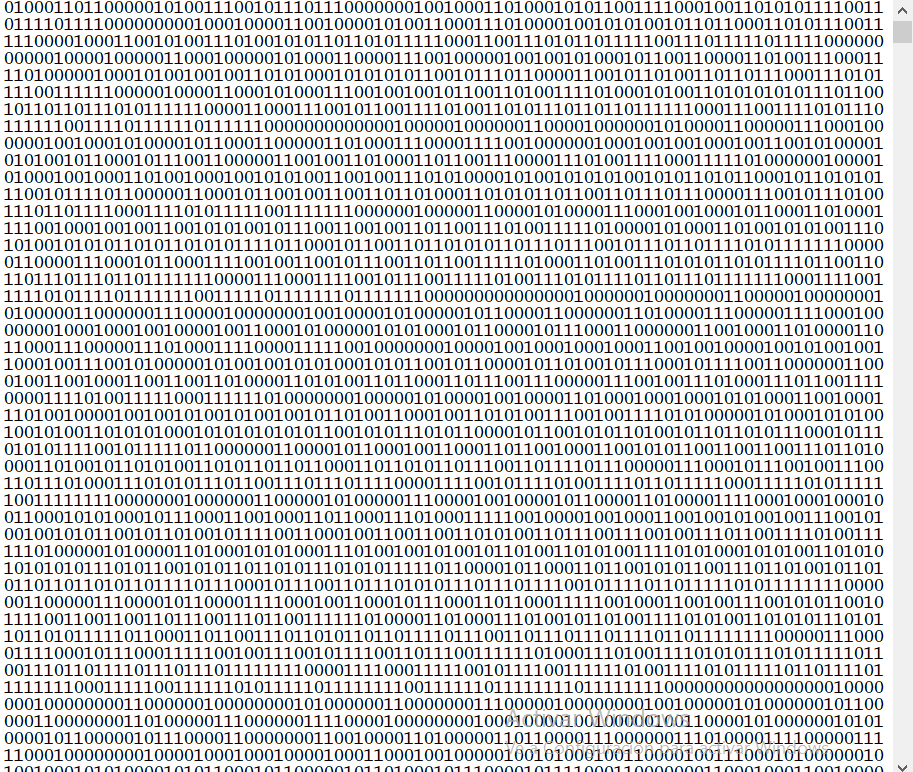
**Archivo Salida 1: ’conjunto universo.txt’.**

Muestra  en notación de conjunto.

Figura7.

**Archivo Salida 2: ’cadena universo.txt’.**

Concatena todas las cadenas de en una sola cadena

Figura8.

Para realizar las gráficas se requirió de la salida de dos archivos de texto extra ’unos cadena.txt’ y ’unos conjunto.txt’. **Archivo Extra de Salida 1:’unos conjunto.txt’.**

Número de 1s de cada cadena que pertenece a 

Figura9.

**Archivo Extra de Salida 2: “unoscadena.txt”.**

Número de 1s de subcadenas de 32 símbolos de cadena concatenada 

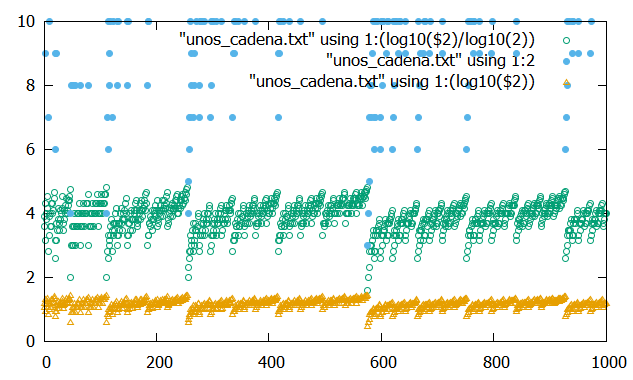
Figura10.

**Grafica ’unos cadena.txt’:**

(AZUL) Número de 1s de cada cadena.

(VERDE) Logaritmo base 2 de número de 1s de cada cadena.

(AMARILLO) Logaritmo base 10 de número de 1s de cada cadena.

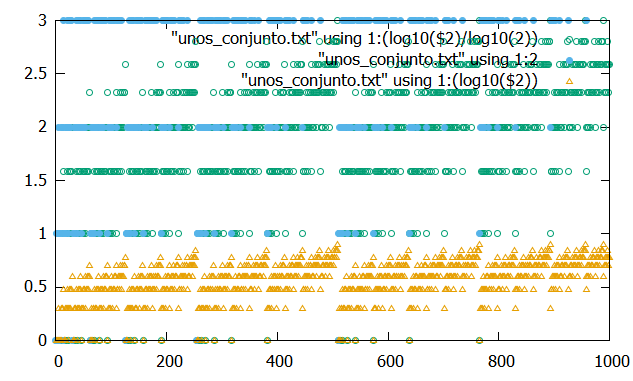
****

**Grafica ’unos conjunto.txt’:**

(COLOR) Número de 1s de cada cadena.

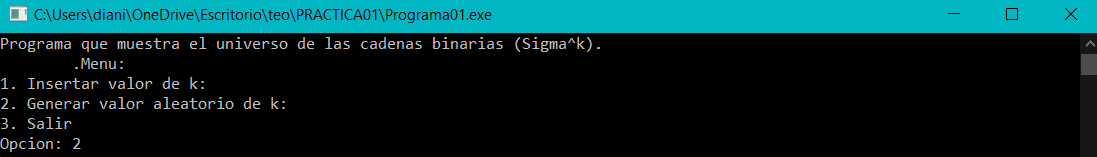
(COLOR) Logaritmo base 2 de número de 1s de cada cadena.

(COLOR) Logaritmo base 10 de número de 1s de cada cadena.



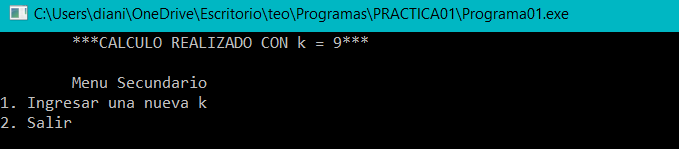
**PRUEBA [2]**

**El programa genera un número entero aleatorio k = que se encuentra en el intervalo [0,1000]**

 Figura13

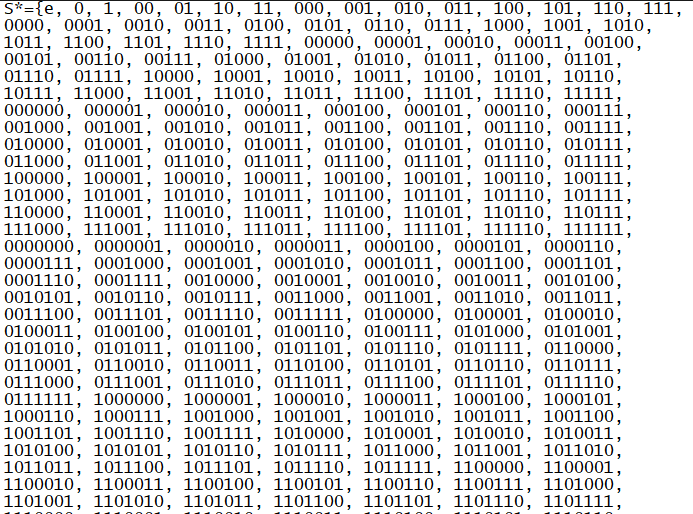
## Menú Secundario

Una vez finalizados todos los cálculos con k = 9 aparece el menú secundario

****Figura14

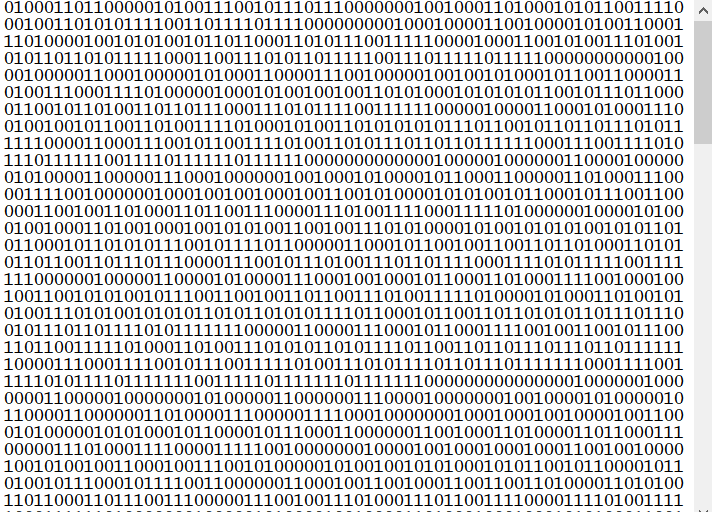
**Archivo Salida 1: ’conjunto universo.txt’.**

Muestra  en notación de conjunto.

 Figura15

**Archivo Salida 2: ’cadena universo.txt’.**

Concatena todas las cadenas de en una sola cadena

 Figura16

Para realizar las gráficas se requirió de la salida de dos archivos de texto extra ’unos cadena.txt’ y ’unos conjunto.txt’. **Archivo Extra de Salida 1:’unos conjunto.txt’.**

Número de 1s de cada cadena que pertenece a 

Figura17

**Archivo Extra de Salida 2: “unoscadena.txt”.**

Número de 1s de subcadenas de 32 símbolos de cadena concatenada 

Figura18.

# Conclusión

Durante la práctica 1 se obtuvo el conocimiento de cómo obtener las cadenas del alfabeto .

Además de como utilizar las tablas de verdad imprimir el universo de cadenas binarias de 

Durante el desarrollo de la práctica se presentaron dificultades en cuanto a la memoria que utilizaba el programa al momento de trasladarse a archivos. Utilice el software ”EmEditor” el cual permite abrir archivos pesados pero los datos de k = 23 fueron más grandes de lo soportado. Por inducción comprobamos que si funciona para k = n funciona para k = n+1.

# Anexo

Programa01.cpp

|  |
| --- |
| #include<iostream>  #include<time.h> //seed de numero random  #include<stdlib.h>  #include<math.h> //Funcion pow  #include<stdio.h>  #include<fstream> //ARCHIVOS  using namespace std;  //Variables Globales  int k;  int \*\*parte1;  int \*\*parte2;  long int filas;  //Prototipos de funciones  void MenuPrincipal();  int Numero\_Aleatorio();  void Binario(int);  void Conjunto();  void Cadena();  void SubMenu();  void LiberarMemoria();  //Funcion principal  int main(){  cout<<"Programa que muestra el universo de las cadenas binarias (Sigma^k)."<<endl;  MenuPrincipal();  LiberarMemoria();  return 0;  }  //Menú  void MenuPrincipal(){  int opcion=0;  do{  cout<<"\t.Menu: "<<endl;  cout<<"1. Insertar valor de k: "<<endl;  cout<<"2. Generar valor aleatorio de k: "<<endl;  cout<<"3. Salir "<<endl;  cout<<"Opcion: ";  cin>>opcion;  system("cls");  switch(opcion){  case 1:  cout<<"Digite el valor de k : ";  cin>>k;  //cout<<"2^"<<k<<" = "<<pow(2,k)<<endl;  Binario(k);  Conjunto();  Cadena();  system("cls");  cout<<"\t\*\*\*CALCULO REALIZADO CON k = "<<k<<"\*\*\*\n\n";  LiberarMemoria();  SubMenu();  break;  case 2:  k = Numero\_Aleatorio();  //cout<<"\nk = "<<k<<endl;  //cout<<"2^"<<k<<" = "<<pow(2,k)<<endl;  Binario(k);  Conjunto();  Cadena();  system("cls");  cout<<"\t\*\*\*CALCULO REALIZADO CON k = "<<k<<"\*\*\*\n\n";  LiberarMemoria();  SubMenu();  break;  }  if(opcion > 3){  cout<<"\nOpcion invalida. "<<endl;  system("cls");  }  }while(opcion > 3);  }  //Genera un numero aleatorio entre 1 - 1000  int Numero\_Aleatorio(){  srand((unsigned)time(NULL));  return (rand()%10)+1;  }  void Binario(int n){    int resultado\_div;  filas = (pow(2,n))/2;  //Reservando memoria para filas y columnas de la 1a mitad de las combinaciones  parte1 = new int\*[filas];  for(int i = 0; i <filas; i++){  parte1[i] = new int[n];  }  //Reservando memoria para filas y columnas de la 2da mitad de las combinaciones  parte2 = new int\*[filas];  for(int i = 0; i <filas; i++){  parte2[i] = new int[n];  }  //Utilizar metodo de la division para conseguir combinaciones  for(int i=0; i<filas; i++)  {  resultado\_div=i;  for(int j=n-1; j>=0; j--)  {  if(resultado\_div%2==0)  {  parte1[i][j]=0;  parte2[i][j]=1;  }  else{  parte1[i][j]=1;  parte2[i][j]=0;  }  resultado\_div = resultado\_div/2;  }  }  }  //Imprimir en un archivo de texto el universo en forma de conjunto  void Conjunto(){  int potencia,z,unos,contador=0;  //Imprimir archivo  ofstream conjunto;  ofstream unosConjunto;  conjunto.open("conjunto\_universo.txt",ios::out); //Abrir archivo  unosConjunto.open("unos\_conjunto.txt",ios::out);    if(conjunto.fail())  {  cout<<"ERROR al abrir archivo"<<endl;  exit(1);  }  if(unosConjunto.fail())  {  cout<<"ERROR al abrir archivo"<<endl;  exit(1);  }    conjunto<<"S\*={e, ";  unos = 0;  contador=0;  for(z=1;z<k;z++){  potencia = pow(2,z);  for(int i=0; i<potencia; i++)  {  for(int j=0; j<z; j++)  {  conjunto<<parte1[i][j+k-z];  if(parte1[i][j+k-z] == 1)  {  unos += 1;  }  }  unosConjunto<<contador<<" "<<unos<<endl;  contador +=1 ;  unos = 0;  conjunto<<", ";  }  }    for(int i=0; i<filas; i++)  {  for(int j=0; j<k; j++)  {  conjunto<<parte1[i][j];  if(parte1[i][j] == 1)  {  unos += 1;  }  }  unosConjunto<<contador<<" "<<unos<<endl;  contador +=1 ;  unos = 0;  conjunto<<", ";  }  unos=0;  for(int i=filas-1; i>=0; i--)  {  for(int j=0; j<k; j++)  {  conjunto<<parte2[i][j];  if(parte2[i][j] == 1)  {  unos += 1;  }  }  if(i>0)  {  conjunto<<", ";  }  unosConjunto<<contador<<" "<<unos<<endl;  contador +=1 ;  unos = 0;    }  conjunto<<" }";    //Cerrar archivo  conjunto.close();  unosConjunto.close();  }  //Imprimir en un archivo de texto el universo en forma de cadena  void Cadena(){  int potencia,z,longitud,unos,cadenas;  //Imprimir archivo  ofstream cadena;  ofstream unosCadena;  cadena.open("cadena\_universo.txt",ios::out); //Abrir archivo  unosCadena.open("unos\_cadena.txt",ios::out);  if(cadena.fail())  {  cout<<"ERROR al abrir archivo"<<endl;  exit(1);  }  if(unosCadena.fail())  {  cout<<"ERROR al abrir archivo"<<endl;  exit(1);  }  longitud = 0;  cadenas = 0;  unos = 0;  for(z=1;z<k;z++){  potencia = pow(2,z);  for(int i=0; i<potencia; i++)  {  for(int j=0; j<z; j++)  {    if(longitud == 32){    longitud=0;  unosCadena<<cadenas<<" "<<unos<<endl;  cadenas += 1;  unos=0;  }  if(parte1[i][j+k-z] == 1)  {  unos += 1;  //cout<<"\nunos: "<<unos;  }  cadena<<parte1[i][j+k-z];  longitud += 1;  //cout<<"\nlong: "<<longitud;  }  if(longitud == 0)  {  //cout<<"\ncadenas: "<<cadenas;  unosCadena<<cadenas<<" "<<unos<<endl;  cadenas += 1;  }  }  }  for(int i=0; i<filas; i++)  {    for(int j=0; j<k; j++)  {    if(longitud == 32){    longitud=0;  unosCadena<<cadenas<<" "<<unos<<endl;  cadenas += 1;  unos=0;  }  if(parte1[i][j] == 1)  {  unos += 1;  //cout<<"\nunos: "<<unos;  }  cadena<<parte1[i][j];  longitud += 1;  //cout<<"\nlong: "<<longitud;  }  }    for(int i=filas-1; i>=0; i--)  {    for(int j=0; j<k; j++)  {    if(longitud == 32){    longitud=0;  unosCadena<<cadenas<<" "<<unos<<endl;  cadenas += 1;  unos=0;  }  if(parte2[i][j] == 1)  {  unos += 1;  // cout<<"\nunos: "<<unos;  }  cadena<<parte2[i][j];  longitud += 1;  //cout<<"\nlong: "<<longitud;  }  if(i == 0)  {    //cout<<"\ncadenas: "<<cadenas;  unosCadena<<cadenas<<" "<<unos<<endl;  cadenas += 1;  }  }  //Cerrar archivo  cadena.close();  unosCadena.close();    }  //Submenu  void SubMenu(){  int opcion=0;  do{  cout<<"\tMenu Secundario "<<endl;  cout<<"1. Ingresar una nueva k "<<endl;  cout<<"2. Salir "<<endl;  cin>>opcion;  switch(opcion){  case 1:  system("cls");  MenuPrincipal();  break;  case 2:  exit(1);  break;  }  if(opcion > 2){  cout<<"\nOpcion invalida. "<<endl;  system("pause");  }  }while(opcion > 2);  }  //Liberar memoria  void LiberarMemoria(){    for(int i=0; i<k; i++)  {  free(parte1[i]);  free(parte2[i]);  }  free(parte1);  free(parte2);  } |