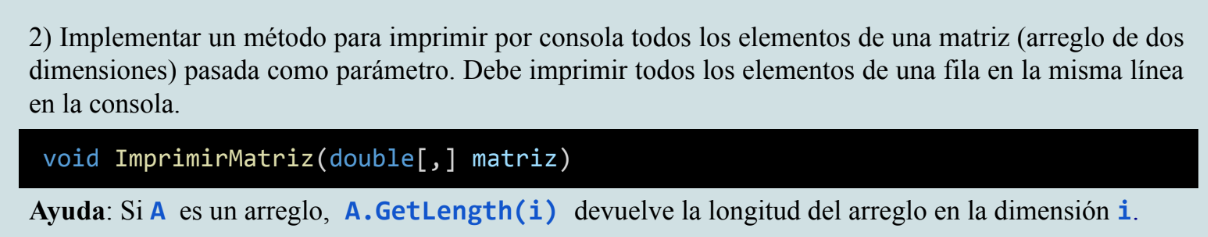
PRACTICA 3 .NET



double [,] matriz = new double [4,4];

for (int i=0;i<16;i++){

    matriz[i/4,i%4] =i +0.5;

}

ImprimirMatriz(matriz);

void ImprimirMatriz(double[,] matriz){

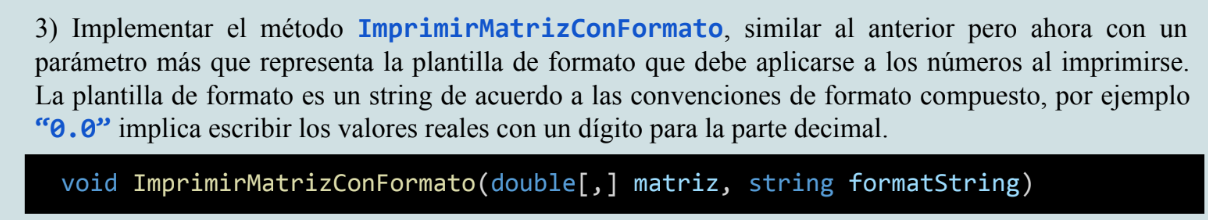
    for (int i=0;i<16;i++){

        Console.Write(matriz[i/4,i%4]+ "|| ");

        Console.Write((i%4)==3 ? "\n": " ");  //para imprimir por fila//

    }

}



Console.WriteLine("INGRESE EL FORMATO DE LA MATRIZ");

String? st = Console.ReadLine();

double [,] matriz = new double [4,4];

for (int i=0;i<16;i++){

    matriz[i/4,i%4] = i;

}

ImprimirMatrizConFormato(matriz,st);

void ImprimirMatrizConFormato (in double[,] matriz,String? Format){

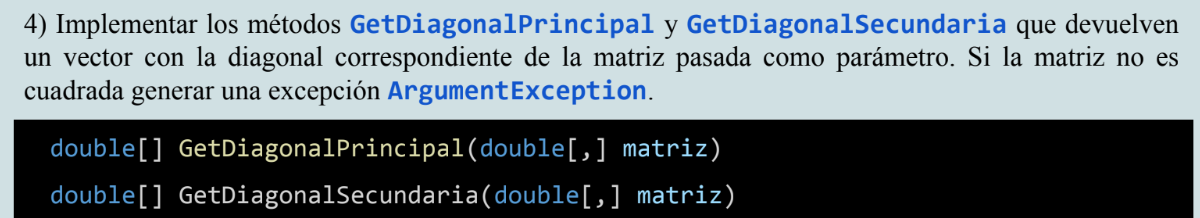
    for (int i=0;i<16;i++){

        Console.Write(matriz[i/4,i%4].ToString(Format));

        Console.Write((i%4)==3 ? "\n": " ");

    }

}



Console.WriteLine("INGRESE EL FORMATO DE LA MATRIZ");

String? st = Console.ReadLine();

double [,] matriz = new double [4,4];

for (int i=0;i<16;i++){

    matriz[i/4,i%4] = i+0.5;

}

double [ ]v = GetDiagonalPrincipal(matriz);

foreach (double i in v)

Console.WriteLine(i);

Console.WriteLine("------------------------");

Console.WriteLine("-------------Vector invertido-------------");

double [ ]v2 = GetDiagonalSecundaria(matriz);

foreach (double i in v2)

Console.WriteLine(i);

double [] GetDiagonalPrincipal (in double [,] matriz){

    if (matriz.GetLength(0)!= matriz.GetLength(1)){

        throw new ArgumentException ("LA MATRIZ NO ES CUADRADA");

    }

    double []DiagonalPrincipal = new double[4];

    {

        for (int i=0;i<4;i++){

        DiagonalPrincipal[i]=matriz[i,i];

    }

    }

    return DiagonalPrincipal;

}

double [] GetDiagonalSecundaria(in double [,] matriz){

    if (matriz.GetLength(0)!= matriz.GetLength(1)){

        throw new ArgumentException ("LA MATRIZ NO ES CUADRADA");

    }

    double []DiagonalSecundaria = new double[4];

    {

        int j=3;

        for (int i=0;i<4;i++){

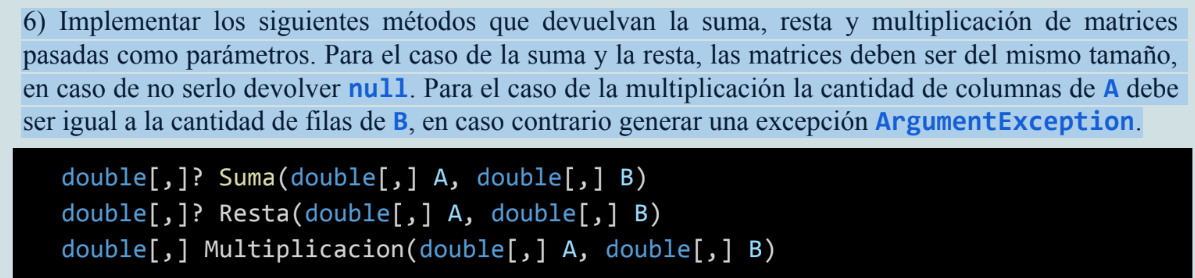
        DiagonalSecundaria[i]=matriz[j--,i];

    }

    }

    return DiagonalSecundaria;

}



double [,] m = new double [,] {{1,2},{4,5}};

double [,] m2 = new double [,] {{1,2,3},{4,5,6}};

double[,] sumatoria;

sumatoria = Multiplicacion(m,m2);

if (sumatoria != null){

for (int i=0;i<sumatoria.GetLength(0);i++){

    for (int j=0;j<sumatoria.GetLength(1);j++)

        Console.WriteLine(sumatoria[i,j]);

     }

}

else

    Console.WriteLine("SUMATORIA ES NULL");

double[,]? Suma(double[,] A, double[,] B){

   double [,]? sumas = null;

    if (sonIguales(A,B)){ //si las filas y columnas son iguales

        sumas = new double [A.GetLength(0),A.GetLength(1)] ;

        int Columnas = A.GetLength(0);

       for (int i=0;i<sumas.GetLength(0);i++){

    for (int j=0;j<sumas.GetLength(1);j++)

        sumas[i,j]  = A[i,j] + B[i,j];

    }

    }

    return sumas;

}

double[,]? Resta(double[,] A, double[,] B){

    double [,] ? Resta = null;

    if (sonIguales(A,B)){

        Resta = new double [A.GetLength(0),A.GetLength(1)] ;

          int Columnas = A.GetLength(0);

        for (int i=0;i<Columnas;i++)

            Resta [i/Columnas,i%Columnas] = A [i/Columnas,i%Columnas] - B[i/Columnas,i%Columnas];

    }

    return Resta;

}

 Boolean sonIguales(double[,]A,double[,]B){

    return A.GetLength(0)== B.GetLength(0) && (A.GetLength(1)==B.GetLength(1));

    }

    double[,] Multiplicacion(double[,] A, double[,] B){

      {

        if (A.GetLength(1)! = B.GetLength(0))

            throw new ArgumentException("El número de columnas de A debe coincidir con el número de filas de B.");

        double[,] Resultado = new double[A.GetLength(0), B.GetLength(1)];

        for (int i = 0; i < A.GetLength(0); i++)

        {

            for (int j = 0; j < B.GetLength(1); j++)

            {

                Resultado[i, j] = 0;

                for (int k = 0; k < A.GetLength(1); k++)

                {

                    Resultado[i, j] += A[i, k] \* B[k, j];

                }

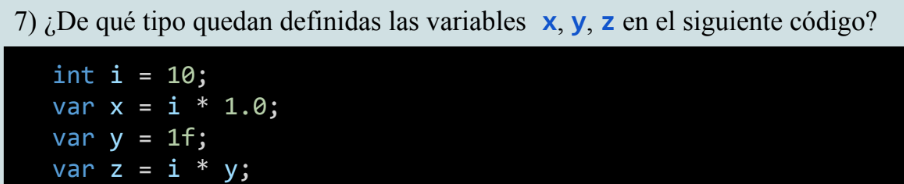
            }

        }

        return Resultado;

    }

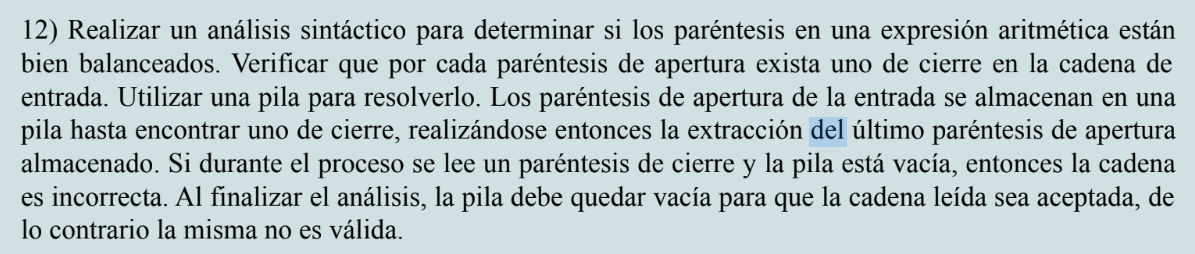
}



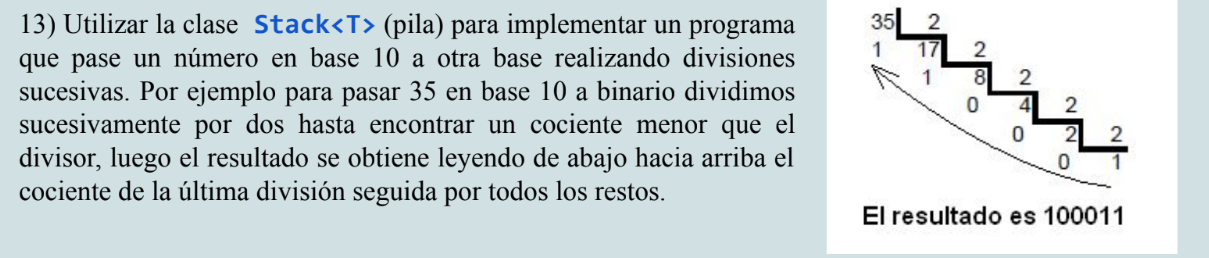
Int i = 10 tipo integer;

Var x=i \* 1.0 Integer

Otras dos las toma como float.



ESTA EN MI CELU XD

Console.WriteLine("INGRESE UN NUMERO");

int n =int.Parse(Console.ReadLine()); //lo casteo

Stack<int>pila =new Stack<int>(); //genero la pila

while (n>0){

    pila.Push(n%2);

    n=n/2;

}

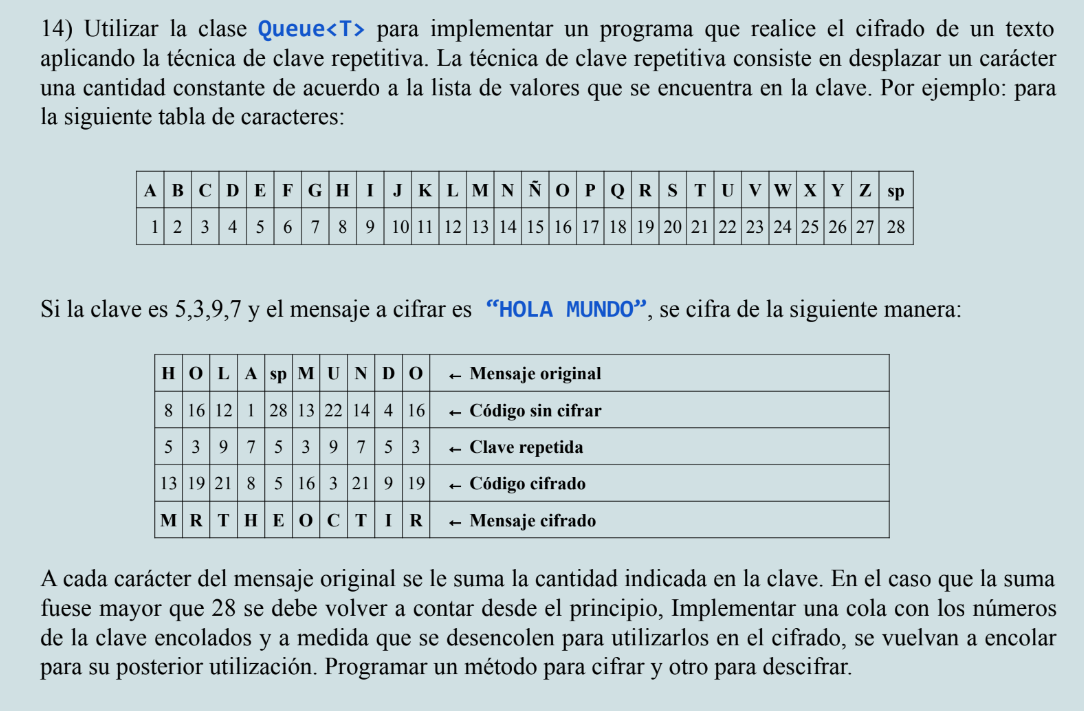
int binario=0;

while (pila.Count != 0 ){

    binario =pila.Pop()+binario\*10 ;

}

Console.WriteLine(binario);



Codificar(st,clave);

void Codificar(String st,int [] clave){

    Queue <int> q = new Queue <int>(clave);

    foreach (Char Carac in st){

        int aux;

        if (Carac =='Ñ')

            aux=15;

        else

            if(Carac==' ')

                aux= 28;

            else

                if (Carac <='N')

                    aux=Carac-'A'+1;

                else

                    aux=Carac-'A'+2;

        int queue = q.Dequeue();

        q.Enqueue(queue);

        aux+=queue;

        if (aux>28)

            aux=aux-28;

        // debo pasar el aux a chad

        Console.Write(ConvertirNumero(aux));

        //hasta ahi esta codificado en numeros

        //para imprimirlo

    }

}

Char ConvertirNumero(int aux){

if (aux==15) return 'Ñ';

if (aux==28) return ' ';

if (aux <=14)

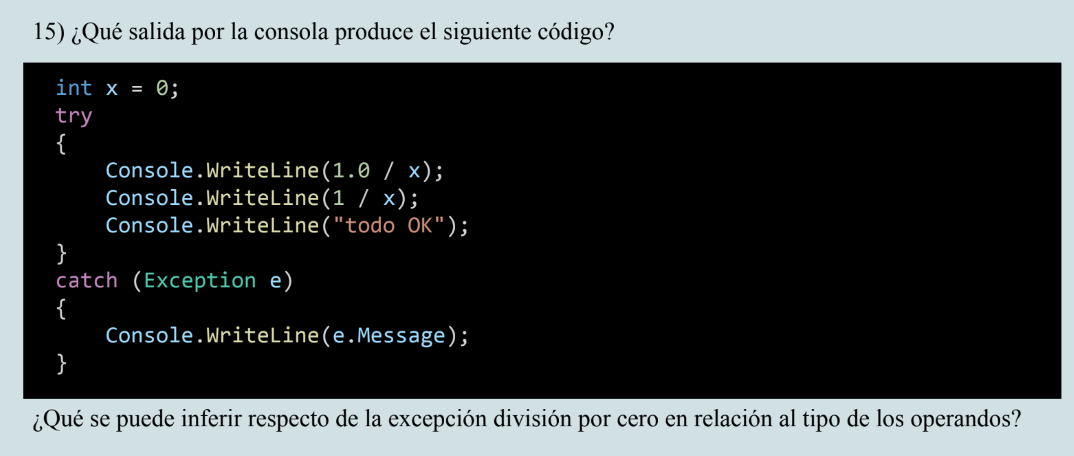
     aux=aux+'A'-1;

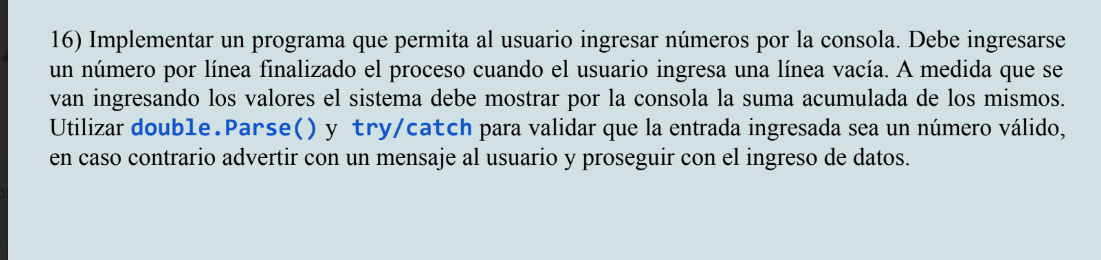
else

    aux=aux+'A'-2;

return (char)aux;

}





double suma = 0;

String num ;

while (true){

   Console.WriteLine("INGRESE NUMERO");

   num = Console.ReadLine();

   if (String.IsNullOrEmpty(num)){

        break;

   }

   //si no es vacio convierto el numero

   try{

        double aux = double.Parse(num);

        suma +=aux;

        Console.WriteLine("SUMA ACUMULADA :" + suma);

   }

   catch (FormatException)

   {

    Console.WriteLine("INGRESE UN NUMERO VALIDO");

   }

}

Bloque finally en Metodo1

Método 1 propagó una excepción no tratada

Método 2 propagó una excepción no tratada

Excepción InvalidCast en Metodo3

Método 3 propagó una excepción