**Instituto Politécnico Nacional**

**Escuela Superior de Cómputo**

**Ingeniería en Sistemas Computacionales**


# Análisis y diseño de algoritmos

## Práctica 2

## Subsecuencia común máxima con programación dinámica.

**Alumno: Sandoval Hernández Eduardo**

**Docente: Juárez Gambino Joel Omar**

**Grupo: 3CM1**

**Fecha de entrega: 07 de junio de 2021**

## Introducción

En la práctica anterior se abordó la estrategia de divide y vencerás la cual consistía en dividir un problema grande en pequeños problemas e ir resolviéndolos de manera recursiva para finalmente juntar sus soluciones y resolver el problema original. Sin embargo, para ciertos problemas esta estrategia tiene inconvenientes que se pueden presentar cuando los subproblemas dependen de otros subproblemas.

La estrategia de programación dinámica consiste en ir guardando los resultados de cada subproblema en una tabla de modo que estos puedan volver a ser utilizados para resolver los problemas posteriores.

El objetivo de esta practica es utilizar la estrategia de programación dinámica para resolver el problema de subsecuencia común máxima. El reporte incluirá el código de implementación en lenguaje java así como los resultados para 3 pruebas.

## Implementación

Para la implementación utilicé primero una clase Elemento que tiene como campos Elem de tipo entero que representará la longitud acumulada en una celda y origen de tipo String que guardará el origen del cual se tomó el carácter [sup (superior), día (diagonal) & izq (izquierdo)]. Incluye su constructor y un conjunto de métodos que servirán para: determinar si dos elementos son iguales, cuál es el mayor de dos elementos e imprimir el elemento.

```
class Elemento{
    public int Elem;
    public String origen;

    //Constructor del Elemento
    public Elemento(int e, String o){
        Elem = e;
        origen = o;
    }

    //Determina si dos elementos son iguales
    public boolean equals(Elemento x){
        return (this.Elem==x.Elem);
    }

    //Determina cuál es el mayor de dos elementos
    public int max(Elemento x){
        if(Elem>x.Elem)
           return this.Elem;
        else
           return x.Elem;
    }

    //Imprime el elemento
    public String toString(){
        return Elem + " ";
    }

}
```

La siguiente es la clase Matriz la cual tiene como campos dos enteros a y b que representan las longitudes i & j respectivamente que a su vez representan las

dimensiones de las matrices +1 donde se encuentran los dos conjuntos, por ejemplo, para la primera prueba se compararán X[ ]={1,0,0,1} & Y[ ]={0,1,0,1,0,0} por tanto los valores de a y b serán 7 y 5 (dimensión de la matriz y + 1 y dimensión de la matriz x + 1), cabe aclarar que el arreglo de mayor longitud debe ser asignado a X. El tercer campo será un arreglo bidimensional de Elementos E[ ][ ] donde se guardarán la matriz de longitudes acumuladas y la de orígenes. El 4to y 5to campo son un arreglo de enteros (R) y uno de caracteres (Rc) donde se guardarán las subsecuencias máximas a partir de la última casilla de la matriz y el uso de una u otra dependerá de si se está trabajando con enteros (como en la prueba 1 y 3) o con caracteres (prueba 2).

En la misma clase se encuentran dos constructores que a su vez serán la parte del código que resuelve el problema.

Los constructores primero asignarán los valores a a y b e inicializarán el arreglo bidimensional.

```
//Constructor de Matriz para enteros
    public Matriz(int x[], int y[], int a, int b){
        this.a = a;
        this.b = b;
        Elemento E1[][] = new Elemento[a][b];
        E = E1;
```

Posteriormente comenzará a crear la matriz con las longitudes acumuladas y los orígenes de estas. Para ello hará uso de un ciclo anidado donde recorrerá toda la matriz y hará las comprobaciones necesarias para asignar los valores siguiendo el algoritmo visto en clase. Para la primera fila y columna asigna a Elem el valor de 0 y a origen ‘000’.

```
//****Esta parte del codigo es la que crea la Matriz con las longitudes acum
uladas y los origenes****
        for(int i=0 ; i<a ; i++)
            for(int j=0 ; j<b ; j++){
                if(i==0 || j==0)
                    E[i][j] = new Elemento(0,"000");
                else if((i!=0 || j!=0) && (x[j-1]==y[i-1]))
```

```
                    E[i][j] = new Elemento(E[i-1][j-1].Elem+1,"Dia");
                else if((i!=0 || j!=0) && (x[j-1]!=y[i-1]) && (E[i][j-
1]!=E[i-1][j])){
                    String o1;
                    if(E[i][j-1].Elem>E[i-1][j].Elem){
                        o1 = "Izq";
                    }
                    else{
                        o1 = "Sup";
                    }
                    E[i][j] = new Elemento(E[i][j-1].max(E[i-1][j]),o1);
                }
                else
                    E[i][j] = new Elemento(E[i-1][j].Elem,"Sup");
            }
        //*****************************************************************
*****************************
```

Lo siguiente es crear el arreglo (R) donde se guarda la subsecuencia común máxima desde la última casilla de la matriz, la dimensión del arreglo es determinada gracias al valor de Elem en esta última casilla. Comienza en las coordenadas de la última casilla y compara si los valores de los arreglos con los datos coinciden, si es así agrega el valor que se encuentra en esa posición -1 del arreglo X y le resta 1 a una variable auxiliar que sirve para ir guardando los datos en determinada posición del arreglo, posteriormente cambia los valores de i & j dependiendo del origen que tenga la casilla actual, en caso de que no coincidan los elementos de los arreglos simplemente cambia los valores de i & j. Todo esto lo hace en un ciclo while el cual terminará en el momento en que la coordenada i llegue a 0.

```
//****Esta parte crea el arreglo donde se guarda la subsecuencia mayor a par
tir de la ultima casilla****
        int i = a-1, j = b-1, aux = E[i][j].Elem-1, R1[] = new int[aux+1];
        R = R1;
        while(i>0){
            if(y[i-1]==x[j-1]){
                R[aux] = x[j-1];
                aux--;
                if(E[i][j].origen.equals("Sup"))
                    i--;
                else if(E[i][j].origen.equals("Dia")){
```

```
                    i--;
                    j--;
                }
                else
                    j--;
            }
            else{
                if(E[i][j].origen.equals("Sup"))
                    i--;
                else if(E[i][j].origen.equals("Dia")){
                    i--;
                    j--;
                }
                else
                    j--;
            }
        }
        //*****************************************************************
*****************************
```

El constructor para caracteres funciona exactamente de la misma forma que el anterior solo que en lugar de recibir dos arreglos de enteros (X & Y) recibe dos arreglos de caracteres y el arreglo donde se guarda la subsecuencia máxima es Rc en lugar de R, además de pequeños cambios en las comparaciones para caracteres.

```
//Constructor de Matriz para caracteres
    public Matriz(char x[], char y[], int a, int b){
        this.a = a;
        this.b = b;
        Elemento E1[][] = new Elemento[a][b];
        E = E1;
        //****Esta parte del codigo es la que crea la Matriz con las longitu
des acumuladas y lo origenes****
        for(int i=0 ; i<a ; i++)
            for(int j=0 ; j<b ; j++){
                if(i==0 || j==0)
                    E[i][j] = new Elemento(0,"000");
                else if((i!=0 || j!=0) && (Character.compare(x[j-1],y[i-
1])==0))//(x[j-1]==y[i-1])
                    E[i][j] = new Elemento(E[i-1][j-1].Elem+1,"Dia");
                else if((i!=0 || j!=0) && (Character.compare(x[j-1],y[i-
1])!=0) && (E[i][j-1]!=E[i-1][j])){
                    int a1, b1;
```

```
                    String o1;
                    if(E[i][j-1].Elem>E[i-1][j].Elem){
                        a1 = i;
                        b1 = j-1;
                        o1 = "Izq";
                    }
                    else{
                        a1 = i-1;
                        b1 = j;
                        o1 = "Sup";
                    }
                    E[i][j] = new Elemento(E[i][j-1].max(E[i-1][j]),o1);
                }
                else
                    E[i][j] = new Elemento(E[i-1][j].Elem,"Sup");
            }
        //*****************************************************************
*****************************
        //****Esta parte crea el arreglo donde se guarda la subsecuencia may
or a partir de la ultima casilla****
        int i = a-1, j = b-1, aux = E[i][j].Elem-1;
        char R2[] = new char[aux+1];
        Rc = R2;
        while(i>0){
            if(Character.compare(x[j-1],y[i-1])==0){
                Rc[aux] = x[j-1];
                aux--;
                if(E[i][j].origen.equals("Sup"))
                    i--;
                else if(E[i][j].origen.equals("Dia")){
                    i--;
                    j--;
                }
                else
                    j--;
            }
            else{
                if(E[i][j].origen.equals("Sup"))
                    i--;
                else if(E[i][j].origen.equals("Dia")){
                    i--;
                    j--;
                }
                else
                    j--;
```

```
            }
        }
        //*****************************************************************
*****************************
    }
```

La clase Matriz además incluye cuatro métodos los cuales imprime la matriz de longitudes acumuladas, imprime la matriz de orígenes, imprime el arreglo donde se guarda la subsecuencia máxima de números enteros o imprime el arreglo donde se guarda la subsecuencia máxima de caracteres dependiendo el caso.

```
//Este metodo imprime la Matriz de longitudes
    public void impMatriz(){
        System.out.println("---------------------------------------");
        for(int x=0 ; x < E.length ; x++){
            for(int y=0 ; y < E[x].length ; y++)
                System.out.print(" |" + E[x][y]+ "| ");
            System.out.println("\n");
        }
        System.out.println("---------------------------------------");
    }

    //Este metodo imprime la matriz de origenes
    public void impOrigenes(){
        System.out.println("---------------------------------------");
        for(int x=0 ; x < E.length ; x++){
            for(int y=0 ; y < E[x].length ; y++)
                System.out.print(" |" + E[x][y].origen+ "| ");
            System.out.println("\n");
        }
        System.out.println("---------------------------------------");
    }

    //Este metodo imprime el arreglo donde se guarda la subseciencia maxima
de numeros enteros
    public void impResultadoInt(){
        System.out.println("---------------------------------------");
        System.out.println("El resultado es:" + Arrays.toString(R));
        System.out.println("---------------------------------------");
    }

    //Este metodo imprime el arreglo donde se guarda la subseciencia maxima
de caracteres
```

```
    public void impResultadoChar(){
        System.out.println("----------------------------------------");
        System.out.println("El resultado es:" + Arrays.toString(Rc));
        System.out.println("----------------------------------------");
    }

}
```

Finalmente se tiene la clase Subsecuencia la cual sirve para ejecutar todo el programa. Para realizar las pruebas, por ejemplo la número 1, se crean los arreglos int X1[ ] = {1,0,0,1} & int Y1[ ] = {0,1,0,1,0,0} teniendo las dimensiones de 4 y 6 respectivamente, por lo que a = 7 y b = 5, posteriormente se crea la Matriz M1 = new Matriz(X1,Y1,7,5). Finalmente se utilizan los métodos impMatriz(), impOrigenes() & impResultadoInt() para mostrar la matriz de longitudes acumuladas, la de orígenes y la subsecuencia común máxima. Se repite el mismo procedimiento para las demás pruebas.

```
public class Subsecuencia {
    public static void main(String[] args){
        System.out.println("*****Prueba 1*****");
        int x1[] = {1,0,0,1};
        int y1[] = {0,1,0,1,0,0};
        Matriz M1 = new Matriz(x1,y1,7,5);
        System.out.println("Matriz de longitudes");
        M1.impMatriz();
        System.out.println("Matriz de origenes");
        M1.impOrigenes();
        M1.impResultadoInt();

        System.out.println("*****Prueba 2*****");
        char x2[] = {'a','b','c','d','e','f','g','h','i','j'};
        char y2[] = {'e','c','d','g','i'};
        Matriz M2 = new Matriz(x2,y2,6,11);
        System.out.println("Matriz de longitudes");
        M2.impMatriz();
        System.out.println("Matriz de origenes");
        M2.impOrigenes();
        M2.impResultadoChar();

        System.out.println("*****Prueba 3*****");
        int x[] = {1,1,0,1,0,1,1,0};
```

```
        int y[] = {0,1,0,1,1,0};
        Matriz M = new Matriz(x,y,7,9);
        System.out.println("Matriz de longitudes");
        M.impMatriz();
        System.out.println("Matriz de origenes");
        M.impOrigenes();
        M.impResultadoInt();

    }
}
```

## Resultados

Prueba 1

```
*****Prueba 1*****
Matriz de longitudes
---------------------------------------
 |0 |  |0 |  |0 |  |0 |  |0 |

 |0 |  |0 |  |1 |  |1 |  |1 |

 |0 |  |1 |  |1 |  |1 |  |2 |

 |0 |  |1 |  |2 |  |2 |  |2 |

 |0 |  |1 |  |2 |  |2 |  |3 |

 |0 |  |1 |  |2 |  |3 |  |3 |

 |0 |  |1 |  |2 |  |3 |  |3 |

---------------------------------------
Matriz de origenes
---------------------------------------
 |000|  |000|  |000|  |000|  |000|

 |000|  |Sup|  |Dia|  |Dia|  |Izq|

 |000|  |Dia|  |Sup|  |Sup|  |Dia|

 |000|  |Sup|  |Dia|  |Dia|  |Sup|

 |000|  |Dia|  |Sup|  |Sup|  |Dia|

 |000|  |Sup|  |Dia|  |Dia|  |Sup|

 |000|  |Sup|  |Dia|  |Dia|  |Sup|

---------------------------------------
---------------------------------------
El resultado es:[0, 0, 1]
---------------------------------------
```

Prueba 2

```
*****Prueba 2*****
Matriz de longitudes
---------------------------------------
 |0 |  |0 |  |0 |  |0 |  |0 |  |0 |  |0 |  |0 |  |0 |  |0 |  |0 |

 |0 |  |0 |  |0 |  |0 |  |0 |  |1 |  |1 |  |1 |  |1 |  |1 |  |1 |

 |0 |  |0 |  |0 |  |1 |  |1 |  |1 |  |1 |  |1 |  |1 |  |1 |  |1 |

 |0 |  |0 |  |0 |  |1 |  |2 |  |2 |  |2 |  |2 |  |2 |  |2 |  |2 |

 |0 |  |0 |  |0 |  |1 |  |2 |  |2 |  |2 |  |3 |  |3 |  |3 |  |3 |

 |0 |  |0 |  |0 |  |1 |  |2 |  |2 |  |2 |  |3 |  |3 |  |4 |  |4 |

---------------------------------------
Matriz de origenes
---------------------------------------
 |000|  |000|  |000|  |000|  |000|  |000|  |000|  |000|  |000|  |000|  |000|  |000|

 |000|  |Sup|  |Sup|  |Sup|  |Sup|  |Dia|  |Izq|  |Izq|  |Izq|  |Izq|  |Izq|  |Izq|

 |000|  |Sup|  |Sup|  |Dia|  |Izq|  |Sup|  |Sup|  |Sup|  |Sup|  |Sup|  |Sup|  |Sup|

 |000|  |Sup|  |Sup|  |Sup|  |Dia|  |Izq|  |Izq|  |Izq|  |Izq|  |Izq|  |Izq|  |Izq|

 |000|  |Sup|  |Sup|  |Sup|  |Sup|  |Sup|  |Sup|  |Dia|  |Izq|  |Izq|  |Izq|  |Izq|

 |000|  |Sup|  |Sup|  |Sup|  |Sup|  |Sup|  |Sup|  |Sup|  |Sup|  |Dia|  |Izq|  |Izq|

---------------------------------------
---------------------------------------
El resultado es:[c, d, g, i]
---------------------------------------
*****Prueba 3*****
Matriz de longitudes
---------------------------------------
```

Prueba 3

```
*****Prueba 3*****
Matriz de longitudes
----------------------------------------
 |0 |  |0 |  |0 |  |0 |  |0 |  |0 |  |0 |  |0 |  |0 |

 |0 |  |0 |  |0 |  |1 |  |1 |  |1 |  |1 |  |1 |  |1 |

 |0 |  |1 |  |1 |  |1 |  |2 |  |2 |  |2 |  |2 |  |2 |

 |0 |  |1 |  |1 |  |2 |  |2 |  |3 |  |3 |  |3 |  |3 |

 |0 |  |1 |  |2 |  |2 |  |3 |  |3 |  |4 |  |4 |  |4 |

 |0 |  |1 |  |2 |  |2 |  |3 |  |3 |  |4 |  |5 |  |5 |

 |0 |  |1 |  |2 |  |3 |  |3 |  |4 |  |4 |  |5 |  |6 |

----------------------------------------
Matriz de origenes
----------------------------------------
 |000|  |000|  |000|  |000|  |000|  |000|  |000|  |000|  |000|  |000|

 |000|  |Sup|  |Sup|  |Dia|  |Izq|  |Dia|  |Izq|  |Izq|  |Dia|  |Dia|

 |000|  |Dia|  |Dia|  |Sup|  |Dia|  |Izq|  |Dia|  |Dia|  |Izq|  |Dia|

 |000|  |Sup|  |Sup|  |Dia|  |Sup|  |Dia|  |Izq|  |Izq|  |Dia|  |Dia|

 |000|  |Dia|  |Dia|  |Sup|  |Dia|  |Sup|  |Dia|  |Dia|  |Izq|  |Dia|

 |000|  |Dia|  |Dia|  |Sup|  |Dia|  |Sup|  |Dia|  |Dia|  |Izq|  |Dia|

 |000|  |Sup|  |Sup|  |Dia|  |Sup|  |Dia|  |Sup|  |Sup|  |Dia|  |Dia|

----------------------------------------
----------------------------------------
El resultado es:[0, 1, 0, 1, 1, 0]
----------------------------------------
```

**Código completo del programa**

```java
package ada_sandovalhernandez.practica2;

import java.util.Arrays;

class Elemento{
    public int Elem;
    public String origen;

    //Constructor del Elemento
```

```
    public Elemento(int e, String o){
        Elem = e;
        origen = o;
    }

    //Determina si dos elementos son iguales
    public boolean equals(Elemento x){
        return (this.Elem==x.Elem);
    }

    //Determina cuál es el mayor de dos elementos
    public int max(Elemento x){
        if(Elem>x.Elem)
           return this.Elem;
        else
           return x.Elem;
    }

    //Imprime el elemento
    public String toString(){
        return Elem + " ";
    }

}

class Matriz{
    public int a;
    public int b;
    public Elemento E[][];
    public int R[];
    public char Rc[];

    //Constructor de Matriz para enteros
    public Matriz(int x[], int y[], int a, int b){
        this.a = a;
        this.b = b;
        Elemento E1[][] = new Elemento[a][b];
        E = E1;
        //****Esta parte del codigo es la que crea la Matriz con las longitu
des acumuladas y los origenes****
        for(int i=0 ; i<a ; i++)
            for(int j=0 ; j<b ; j++){
                if(i==0 || j==0)
                    E[i][j] = new Elemento(0,"000");
                else if((i!=0 || j!=0) && (x[j-1]==y[i-1]))
```

```
                    E[i][j] = new Elemento(E[i-1][j-1].Elem+1,"Dia");
                else if((i!=0 || j!=0) && (x[j-1]!=y[i-1]) && (E[i][j-
1]!=E[i-1][j])){
                    String o1;
                    if(E[i][j-1].Elem>E[i-1][j].Elem){
                        o1 = "Izq";
                    }
                    else{
                        o1 = "Sup";
                    }
                    E[i][j] = new Elemento(E[i][j-1].max(E[i-1][j]),o1);
                }
                else
                    E[i][j] = new Elemento(E[i-1][j].Elem,"Sup");
            }
        //*****************************************************************
*****************************
        //****Esta parte crea el arreglo donde se guarda la subsecuencia may
or a partir de la ultima casilla****
        int i = a-1, j = b-1, aux = E[i][j].Elem-1, R1[] = new int[aux+1];
        R = R1;
        while(i>0){
            if(y[i-1]==x[j-1]){
                R[aux] = x[j-1];
                aux--;
                if(E[i][j].origen.equals("Sup"))
                    i--;
                else if(E[i][j].origen.equals("Dia")){
                    i--;
                    j--;
                }
                else
                    j--;
            }
            else{
                if(E[i][j].origen.equals("Sup"))
                    i--;
                else if(E[i][j].origen.equals("Dia")){
                    i--;
                    j--;
                }
                else
                    j--;
            }
        }
```

```
        //*****************************************************************
*****************************
    }

    //Constructor de Matriz para caracteres
    public Matriz(char x[], char y[], int a, int b){
        this.a = a;
        this.b = b;
        Elemento E1[][] = new Elemento[a][b];
        E = E1;
        //****Esta parte del codigo es la que crea la Matriz con las longitu
des acumuladas y lo origenes****
        for(int i=0 ; i<a ; i++)
            for(int j=0 ; j<b ; j++){
                if(i==0 || j==0)
                    E[i][j] = new Elemento(0,"000");
                else if((i!=0 || j!=0) && (Character.compare(x[j-1],y[i-
1])==0))//(x[j-1]==y[i-1])
                    E[i][j] = new Elemento(E[i-1][j-1].Elem+1,"Dia");
                else if((i!=0 || j!=0) && (Character.compare(x[j-1],y[i-
1])!=0) && (E[i][j-1]!=E[i-1][j])){
                    int a1, b1;
                    String o1;
                    if(E[i][j-1].Elem>E[i-1][j].Elem){
                        a1 = i;
                        b1 = j-1;
                        o1 = "Izq";
                    }
                    else{
                        a1 = i-1;
                        b1 = j;
                        o1 = "Sup";
                    }
                    E[i][j] = new Elemento(E[i][j-1].max(E[i-1][j]),o1);
                }
                else
                    E[i][j] = new Elemento(E[i-1][j].Elem,"Sup");
            }
        //*****************************************************************
*****************************
        //****Esta parte crea el arreglo donde se guarda la subsecuencia may
or a partir de la ultima casilla****
        int i = a-1, j = b-1, aux = E[i][j].Elem-1;
        char R2[] = new char[aux+1];
        Rc = R2;
```

```
        while(i>0){
            if(Character.compare(x[j-1],y[i-1])==0){
                Rc[aux] = x[j-1];
                aux--;
                if(E[i][j].origen.equals("Sup"))
                    i--;
                else if(E[i][j].origen.equals("Dia")){
                    i--;
                    j--;
                }
                else
                    j--;
            }
            else{
                if(E[i][j].origen.equals("Sup"))
                    i--;
                else if(E[i][j].origen.equals("Dia")){
                    i--;
                    j--;
                }
                else
                    j--;
            }
        }
        //**********************************************************************
*****************************
    }

    //Este metodo imprime la Matriz de longitudes
    public void impMatriz(){
        System.out.println("----------------------------------------");
        for(int x=0 ; x < E.length ; x++){
            for(int y=0 ; y < E[x].length ; y++)
                System.out.print(" |" + E[x][y]+ "| ");
            System.out.println("\n");
        }
        System.out.println("----------------------------------------");
    }

    //Este metodo imprime la matriz de origenes
    public void impOrigenes(){
        System.out.println("----------------------------------------");
        for(int x=0 ; x < E.length ; x++){
            for(int y=0 ; y < E[x].length ; y++)
                System.out.print(" |" + E[x][y].origen+ "| ");
```

```
            System.out.println("\n");
        }
        System.out.println("----------------------------------------");
    }

    //Este metodo imprime el arreglo donde se guarda la subseciencia maxima
de numeros enteros
    public void impResultadoInt(){
        System.out.println("----------------------------------------");
        System.out.println("El resultado es:" + Arrays.toString(R));
        System.out.println("----------------------------------------");
    }

    //Este metodo imprime el arreglo donde se guarda la subseciencia maxima
de caracteres
    public void impResultadoChar(){
        System.out.println("----------------------------------------");
        System.out.println("El resultado es:" + Arrays.toString(Rc));
        System.out.println("----------------------------------------");
    }

}

public class Subsecuencia {
    public static void main(String[] args){
        System.out.println("*****Prueba 1*****");
        int x1[] = {1,0,0,1};
        int y1[] = {0,1,0,1,0,0};
        Matriz M1 = new Matriz(x1,y1,7,5);
        System.out.println("Matriz de longitudes");
        M1.impMatriz();
        System.out.println("Matriz de origenes");
        M1.impOrigenes();
        M1.impResultadoInt();

        System.out.println("*****Prueba 2*****");
        char x2[] = {'a','b','c','d','e','f','g','h','i','j'};
        char y2[] = {'e','c','d','g','i'};
        Matriz M2 = new Matriz(x2,y2,6,11);
        System.out.println("Matriz de longitudes");
        M2.impMatriz();
        System.out.println("Matriz de origenes");
        M2.impOrigenes();
        M2.impResultadoChar();
```

```
        System.out.println("*****Prueba 3*****");
        int x[] = {1,1,0,1,0,1,1,0};
        int y[] = {0,1,0,1,1,0};
        Matriz M = new Matriz(x,y,7,9);
        System.out.println("Matriz de longitudes");
        M.impMatriz();
        System.out.println("Matriz de origenes");
        M.impOrigenes();
        M.impResultadoInt();

    }
}
```