import java.util.Random;

import java.util.Scanner;

import java.io.IOException;

public class The8QueensPuzzle2{

    //poblacion inicial

    //fitness

    //crossover

    //mutacion  REPETIR

    public The8QueensPuzzle2(){

    }

    //----------------------------------------------------------------------

    public static void main(String[]args){

        int N=4;

        Poblacion2 poblacion = new Poblacion2(N);

        //Integer[][] cromosomas= new Integer[N][8];

        //asignar elementos de modo que no ese repitan

        //------------------------------------------------------------------

        Random rand = new Random();

        for(int i=0;i<poblacion.cromosomas.length;i++){

            poblacion.cromosomas[i].indOrden = i;

            poblacion.cromosomas[i].genes[0] = rand.nextInt(8)+1;

            for(int j=1;j<poblacion.cromosomas[0].genes.length;j++){

                int num= -1;

                //corroborar que no se repita

                int posRep = -1;

                do{

                    //System.out.println("dentro de do ,posResp="+posRep);

                    num = rand.nextInt(8)+1;

                    //busqueda en los elementos ver si se REPITE

                    int k =0;

                    for(;k<j;k++){

                        if(num == poblacion.cromosomas[i].genes[k]){

                            posRep = k;

                            break;

                        }

                    }if(k==j){

                        posRep = -1;

                    }

                }while(posRep!=-1);

                poblacion.cromosomas[i].genes[j] = num;

            }

        }

        //--------------------------------------------------------------------------

        //imprimiendo la poblacion inicial

        for(int i=0;i<poblacion.cromosomas.length;i++){

            for(int j=0;j<poblacion.cromosomas[0].genes.length;j++){

                System.out.printf("%d\t",poblacion.cromosomas[i].genes[j]);

            }

            System.out.println();

        }

       //while(acabar si fitnes de algun cromosoma es 1)

       boolean flagGlobal = false;

       int contadorGlobal =0;

       while(!flagGlobal&&(contadorGlobal<5)){

             //funcion aptitud fitness INICIO

        //------------------------------------------------------------------------

        for(int i=0;i<poblacion.cromosomas.length;i++){

            //C[i] - i para la ascendente       C[i]+i para para la descendente

            String [][] tabla = new String[8][8];

            for(int f=0;f<8;f++){

                for(int c=0;c<8;c++){

                    tabla[f][c] = "| |";

                }

            }

            for(int l=0;l<poblacion.cromosomas[i].genes.length;l++){

                tabla[poblacion.cromosomas[i].genes[l]-1][l]="|\*|";

            }

            System.out.println();

            for(int f=0;f<8;f++){

                for(int c=0;c<8;c++){

                    System.out.print(tabla[f][c]);

                }

                System.out.println();

            }

            int conflictos = 0;

            for(int j=0;j<poblacion.cromosomas[0].genes.length;j++){

                //luego del elemento

                for(int k=0;k<poblacion.cromosomas[i].genes.length;k++){

                    //si conflicto  conflictos++;

                   if(k!=j){

                        if((poblacion.cromosomas[i].genes[k]-(k+1) )==(poblacion.cromosomas[i].genes[j]-(j+1))){

                            System.out.printf("ascendente:(%d,%d)~(%d,%d)\n",poblacion.cromosomas[i].genes[k],(k+1),poblacion.cromosomas[i].genes[j],(j+1));

                            conflictos++;

                        }

                        if((poblacion.cromosomas[i].genes[k]+(k+1))==(poblacion.cromosomas[i].genes[j]+(j+1))){

                            System.out.printf("descendente:(%d,%d)~(%d,%d)\n",poblacion.cromosomas[i].genes[k],(k+1),poblacion.cromosomas[i].genes[j],(j+1));

                            conflictos++;

                        }

                   }

                }

            }

            System.out.printf("conflictos cromosoma %d :%d\n",(poblacion.cromosomas[i].indOrden+1),conflictos);

            double factor =Math.pow(10,5);

            poblacion.cromosomas[i].fitnes = Math.round(factor/(1+conflictos))/factor;

            System.out.println();

        }

        // ----fucion aptitud fitness FIN

        //-----------------------------------------------------------------------

        for(int i=0;i<poblacion.cromosomas.length;i++){

            System.out.printf("cromosoma = %d fitnes = %1.5f\n",(poblacion.cromosomas[i].indOrden+1),poblacion.cromosomas[i].fitnes);

        }

        //----SELECCION---------------------------------------------------

        //---calculo de probabilidades----

        poblacion.CalcularProbabilidades();

        for(int i=0;i<poblacion.probabilidades.length;i++){

            System.out.printf("cromosoma = %d probabilidad = %1.5f\n",(poblacion.cromosomas[i].indOrden+1),poblacion.probabilidades[i]);

        }

        poblacion.CrearRangos();

        for(int i=0;i<poblacion.cromosomas.length;i++){

            System.out.printf("[%1.5f,%1.5f)\n",poblacion.cromosomas[i].rango.lInferior,poblacion.cromosomas[i].rango.lSuperior);

        }

        //--------SELECCION---------------------generar el numero aleatorio para escoger al par que van a hacer el crossover

        Random random=new Random();

        double variableCross;

        //si variableCross esta en alguno de los intervalos

        int [] indiceElegidos = new int[2];

        int cont = 0;

        while(cont < indiceElegidos.length){

            variableCross = random.nextDouble();

            for(int i=0;i<poblacion.cromosomas.length;i++){

                if(poblacion.cromosomas[i].rango.pertenece(variableCross)){

                    if(poblacion.cromosomas[i].rango.idRango!=((i==1)?indiceElegidos[i-1]:-1)){

                        System.out.printf(" idRango: %d variableCross: %1.5f\n",poblacion.cromosomas[i].rango.idRango,variableCross);        //ind actual diferente del anterior)

                        indiceElegidos[cont] = poblacion.cromosomas[i].rango.idRango;

                        cont++;

                        break;

                    }else{

                        cont = cont ;

                        break;

                    }

                }

            }

        }

        //imprimiendo los elegidos

         System.out.println("padres");

        for(int i=0;i<indiceElegidos.length;i++){

            for(int j=0;j<poblacion.cromosomas[0].genes.length;j++){

                System.out.printf("%d\t",poblacion.cromosomas[indiceElegidos[i]].genes[j]);

            }

            System.out.printf("probabilidad: %1.5f\n",poblacion.probabilidades[indiceElegidos[i]]);

        }

        //------------------------------------CROSSOVER------------------realizar el cross

        int nHijos = 2;

        Cromosoma2[] hijos = new Cromosoma2[nHijos];

        for(int i = 0;i<hijos.length;i++){

            hijos[i] = new Cromosoma2();

            for(int j=0;j<hijos[i].genes.length;j++){

                //System.out.println(j>3?(int)Math.abs(i-1):i);

                hijos[i].genes[j] = poblacion.cromosomas[indiceElegidos[j>3?(int)Math.abs(i-1):i]].genes[j];

            }

        }

        //imprimiendo cromosomasHijos

        System.out.println("hijos cross");

        for(int i=0;i<hijos.length;i++){

            for(int j=0;j<hijos[i].genes.length;j++){

                System.out.printf("%d\t",hijos[i].genes[j]);

            }

            System.out.println();

        }

        //-----------------------MUTACION----------------------------

        //ordenar segun fitness y reemplazar los de abajo(menores)

        int indMutar ;

        int newElemFila ;

        for(int i=0;i<hijos.length;i++){

            boolean flag = false;

            int contador = 0;

            do{

                contador ++;

                indMutar = rand.nextInt(8); //columna

                newElemFila = rand.nextInt(8)+1; //elemento nuevo fila

                //System.out.println("indMutar:"+indMutar+" newElemFila: "+newElemFila);

                    //System.out.println("dentro de if indMutar[]!=newElemFila");

                    int j;

                    for(j=0;j<hijos[i].genes.length;j++){ // buscando la no repeticion

                        if(hijos[i].genes[j]==newElemFila){

                            flag = false;

                            break;

                        }

                    }

                    if(j==hijos[0].genes.length){

                        hijos[i].genes[indMutar] = newElemFila;

                        flag = true;

                    }

            }while(!flag&&(contador<10));

        }

        //----------------------------------------------------

        System.out.println("hijos mutados");

        for(int i=0;i<hijos.length;i++){

            for(int j=0;j<hijos[i].genes.length;j++){

                System.out.printf("%d\t",hijos[i].genes[j]);

            }

            System.out.println();

        }

        //---------------------------------------------------------

        poblacion.ImprimirCromosomas();

        poblacion.OrdenarIndices();

        poblacion.ImprimirCromosomas();

        //reemplazando los 2 mas bajos

        poblacion.InsertarHijos(hijos);

        poblacion.ImprimirCromosomas();

        for(int i =0;i<poblacion.cromosomas.length;i++){

            flagGlobal =(poblacion.cromosomas[i].fitnes ==1);

        }

        contadorGlobal ++;

       }//fin While Global

    } //fin main

    //---------------------------------------------------------------------------------

}

//================================================================================

//===============================================================================

class Poblacion2{

    static Cromosoma2 [] cromosomas ;

    static double probabilidades[];

    //-----------------------------------------------------------------------------

    Poblacion2(int N){

        cromosomas = new Cromosoma2[N];

        for(int i=0;i<N;i++){

            cromosomas[i] = new Cromosoma2();//creando los cromosomas

            //cromosomas[i].genes = new int[8]; ya se hace en el constructor de Cromosoma2

        }

        probabilidades = new double[N];

    }

    //-------------------------------------------------------------------------

    public static void OrdenarIndices(){

        //reemplazar cromosomas segun fitnes

        for(int i=0;i<cromosomas.length;i++){

            for(int j=0;j<(cromosomas.length-i-1);j++){

                if(cromosomas[j].fitnes>cromosomas[j+1].fitnes){

                    Cromosoma2 temp =new Cromosoma2();

                    temp = cromosomas[j];

                    cromosomas[j] =cromosomas[j+1];

                    cromosomas[j+1] = temp;

                }

            }

        }

        for(int i=0;i<cromosomas.length;i++){

            cromosomas[i].indOrden = i;

        }

        for(int i=0;i<cromosomas.length;i++){

            cromosomas[i].rango.idRango = i;

        }

    }//---------------------------------------------------------------------------

    public static void CalcularFitnes(Cromosoma2 cromosoma){

         //C[i] - i para la ascendente       C[i]+i para para la descendente

         /\*String [][] tabla = new String[8][8];

         for(int f=0;f<8;f++){

             for(int c=0;c<8;c++){

                 tabla[f][c] = "| |";

             }

         }

         for(int l=0;l<cromosoma.genes.length;l++){

             tabla[cromosoma.genes[l]-1][l]="|\*|";

         }

         System.out.println();

         for(int f=0;f<8;f++){

             for(int c=0;c<8;c++){

                 System.out.print(tabla[f][c]);

             }

             System.out.println();

         }\*/

         int conflictos = 0;

         for(int j=0;j<cromosoma.genes.length;j++){

             //luego del elemento

             for(int k=0;k<cromosoma.genes.length;k++){

                 //si conflicto  conflictos++;

                if(k!=j){

                     if((cromosoma.genes[k]-(k+1) )==(cromosoma.genes[j]-(j+1))){

                         //System.out.printf("ascendente:(%d,%d)~(%d,%d)\n",cromosoma.genes[k],(k+1),cromosoma.genes[j],(j+1));

                         conflictos++;

                     }

                     if((cromosoma.genes[k]+(k+1))==(cromosoma.genes[j]+(j+1))){

                         //System.out.printf("descendente:(%d,%d)~(%d,%d)\n",cromosoma.genes[k],(k+1),cromosoma.genes[j],(j+1));

                         conflictos++;

                     }

                }

             }

         }

         //System.out.printf("conflictos cromosoma %d :%d\n",(cromosoma.indOrden+1),conflictos);

         double factor =Math.pow(10,5);

         cromosoma.fitnes = Math.round(factor/(1+conflictos))/factor;

         System.out.println();

    }

    //---------------------------------------------------------------------------

    public static void CalcularFitnes(int i){

         //C[i] - i para la ascendente       C[i]+i para para la descendente

         String [][] tabla = new String[8][8];

         for(int f=0;f<8;f++){

             for(int c=0;c<8;c++){

                 tabla[f][c] = "| |";

             }

         }

         for(int l=0;l<cromosomas[i].genes.length;l++){

             tabla[cromosomas[i].genes[l]-1][l]="|\*|";

         }

         System.out.println();

         for(int f=0;f<8;f++){

             for(int c=0;c<8;c++){

                 System.out.print(tabla[f][c]);

             }

             System.out.println();

         }

         int conflictos = 0;

         for(int j=0;j<cromosomas[0].genes.length;j++){

             //luego del elemento

             for(int k=0;k<cromosomas[i].genes.length;k++){

                 //si conflicto  conflictos++;

                if(k!=j){

                     if((cromosomas[i].genes[k]-(k+1) )==(cromosomas[i].genes[j]-(j+1))){

                         System.out.printf("ascendente:(%d,%d)~(%d,%d)\n",cromosomas[i].genes[k],(k+1),cromosomas[i].genes[j],(j+1));

                         conflictos++;

                     }

                     if((cromosomas[i].genes[k]+(k+1))==(cromosomas[i].genes[j]+(j+1))){

                         System.out.printf("descendente:(%d,%d)~(%d,%d)\n",cromosomas[i].genes[k],(k+1),cromosomas[i].genes[j],(j+1));

                         conflictos++;

                     }

                }

             }

         }

         System.out.printf("conflictos cromosoma %d :%d\n",(cromosomas[i].indOrden+1),conflictos);

         double factor =Math.pow(10,5);

         cromosomas[i].fitnes = Math.round(factor/(1+conflictos))/factor;

         System.out.println();

    }

    //---------------------------------------------------------------------------

    public static double FitnesTotal(){

        double fitnesTotal = 0;

        for(int i=0;i<cromosomas.length;i++){

            fitnesTotal+=cromosomas[i].fitnes;

        }

        return fitnesTotal;

    }

    //---------------------------------------------------------------------------

    public static void CalcularProbabilidades(){

        double fitnesTotal = FitnesTotal();

        for(int i=0;i<probabilidades.length;i++){

            probabilidades[i] = cromosomas[i].fitnes/fitnesTotal;

        }

    }

    //------------------------------------------------------------------

    public static void CrearRangos(){

        double acumulada = 0;

        for(int i = 0;i<cromosomas.length;i++){

            cromosomas[i].rango =new Rango2(cromosomas[i].indOrden, acumulada, acumulada +probabilidades[i]);

            acumulada +=probabilidades[i];

        }

    }

    //------------------------------------------------------------------

    public static void ImprimirCromosomas(){

        System.out.println();

        for(int i= 0;i<cromosomas.length;i++){

            for(int j=0;j<cromosomas[i].genes.length;j++){

                System.out.printf("%d\t",cromosomas[i].genes[j]);

            }

            System.out.printf("idOrden: %d ,fitnes: %1.5f  \n",cromosomas[i].indOrden,cromosomas[i].fitnes);

        }

        System.out.println();

    }

    //---------------------------------------------------------------------------------------------

    public static void InsertarHijos(Cromosoma2[] cromos){

        for(int i = 0; i<cromos.length;i++){

            for(int j=0;j<cromos[0].genes.length;j++){

                cromosomas[i].genes[j] = cromos[i].genes[j];

            }

            CalcularFitnes(cromosomas[i]);

        }

    }

}

//============================================================================

class Cromosoma2{

    int[] genes;

    int indOrden;

    double fitnes;

    double probabilidad;

    Rango2 rango;

    Cromosoma2(){

        genes = new int[8];

    }

}

//===================================================================================================

class Rango2{

    double lSuperior;

    double lInferior;

    int idRango;

    Rango2(int id, double inf, double sup){

        this.idRango = id;

        this.lInferior = inf;

        this.lSuperior = sup;

    }

    public boolean pertenece(double num){

        if((num>=lInferior) && (num<lSuperior)){

            return true;

        }

        return false;

    }

}

//===================================

FLORES VILLAR@DESKTOP-IPF07U2 MINGW64 ~/Desktop/matematica\_computacional/PC5AgoritmosGeneticos

$ javac The8QueensPuzzle.java

FLORES VILLAR@DESKTOP-IPF07U2 MINGW64 ~/Desktop/matematica\_computacional/PC5AgoritmosGeneticos

$ javac The8QueensPuzzle2.java

FLORES VILLAR@DESKTOP-IPF07U2 MINGW64 ~/Desktop/matematica\_computacional/PC5AgoritmosGeneticos

$ java The8QueensPuzzle2

1 7 3 6 8 5 2 4

6 1 7 2 8 5 4 3

4 3 7 6 8 5 2 1

4 5 3 2 7 6 1 8

|\*|| || || || || || || |

| || || || || || ||\*|| |

| || ||\*|| || || || || |

| || || || || || || ||\*|

| || || || || ||\*|| || |

| || || ||\*|| || || || |

| ||\*|| || || || || || |

| || || || ||\*|| || || |

ascendente:(3,3)~(1,1)

descendente:(2,7)~(7,2)

ascendente:(1,1)~(3,3)

descendente:(7,2)~(2,7)

conflictos cromosoma 1 :4

| ||\*|| || || || || || |

| || || ||\*|| || || || |

| || || || || || || ||\*|

| || || || || || ||\*|| |

| || || || || ||\*|| || |

|\*|| || || || || || || |

| || ||\*|| || || || || |

| || || || ||\*|| || || |

ascendente:(5,6)~(1,2)

ascendente:(1,2)~(5,6)

descendente:(4,7)~(5,6)

descendente:(3,8)~(5,6)

descendente:(5,6)~(4,7)

descendente:(3,8)~(4,7)

descendente:(5,6)~(3,8)

descendente:(4,7)~(3,8)

conflictos cromosoma 2 :8

| || || || || || || ||\*|

| || || || || || ||\*|| |

| ||\*|| || || || || || |

|\*|| || || || || || || |

| || || || || ||\*|| || |

| || || ||\*|| || || || |

| || ||\*|| || || || || |

| || || || ||\*|| || || |

descendente:(3,2)~(4,1)

ascendente:(8,5)~(4,1)

descendente:(4,1)~(3,2)

descendente:(6,4)~(7,3)

descendente:(7,3)~(6,4)

ascendente:(4,1)~(8,5)

descendente:(1,8)~(2,7)

descendente:(2,7)~(1,8)

conflictos cromosoma 3 :8

| || || || || || ||\*|| |

| || || ||\*|| || || || |

| || ||\*|| || || || || |

|\*|| || || || || || || |

| ||\*|| || || || || || |

| || || || || ||\*|| || |

| || || || ||\*|| || || |

| || || || || || || ||\*|

ascendente:(5,2)~(4,1)

ascendente:(4,1)~(5,2)

descendente:(2,4)~(3,3)

ascendente:(6,6)~(3,3)

ascendente:(8,8)~(3,3)

descendente:(3,3)~(2,4)

descendente:(6,6)~(7,5)

ascendente:(3,3)~(6,6)

descendente:(7,5)~(6,6)

ascendente:(8,8)~(6,6)

ascendente:(3,3)~(8,8)

ascendente:(6,6)~(8,8)

conflictos cromosoma 4 :12

cromosoma = 1 fitnes = 0.20000

cromosoma = 2 fitnes = 0.11111

cromosoma = 3 fitnes = 0.11111

cromosoma = 4 fitnes = 0.07692

cromosoma = 1 probabilidad = 0.40069

cromosoma = 2 probabilidad = 0.22260

cromosoma = 3 probabilidad = 0.22260

cromosoma = 4 probabilidad = 0.15411

[0.00000,0.40069)

[0.40069,0.62329)

[0.62329,0.84589)

[0.84589,1.00000)

idRango: 1 variableCross: 0.51541

idRango: 2 variableCross: 0.81084

padres

6 1 7 2 8 5 4 3 probabilidad: 0.22260

4 3 7 6 8 5 2 1 probabilidad: 0.22260

hijos cross

6 1 7 2 8 5 2 1

4 3 7 6 8 5 4 3

hijos mutados

6 1 7 2 8 5 2 3

4 3 7 6 8 1 4 3

1 7 3 6 8 5 2 4 idOrden: 0 ,fitnes: 0.20000

6 1 7 2 8 5 4 3 idOrden: 1 ,fitnes: 0.11111

4 3 7 6 8 5 2 1 idOrden: 2 ,fitnes: 0.11111

4 5 3 2 7 6 1 8 idOrden: 3 ,fitnes: 0.07692

4 5 3 2 7 6 1 8 idOrden: 0 ,fitnes: 0.07692

6 1 7 2 8 5 4 3 idOrden: 1 ,fitnes: 0.11111

4 3 7 6 8 5 2 1 idOrden: 2 ,fitnes: 0.11111

1 7 3 6 8 5 2 4 idOrden: 3 ,fitnes: 0.20000

6 1 7 2 8 5 2 3 idOrden: 0 ,fitnes: 0.14286

4 3 7 6 8 1 4 3 idOrden: 1 ,fitnes: 0.09091

4 3 7 6 8 5 2 1 idOrden: 2 ,fitnes: 0.11111

1 7 3 6 8 5 2 4 idOrden: 3 ,fitnes: 0.20000

| ||\*|| || || || || || |

| || || ||\*|| || ||\*|| |

| || || || || || || ||\*|

| || || || || || || || |

| || || || || ||\*|| || |

|\*|| || || || || || || |

| || ||\*|| || || || || |

| || || || ||\*|| || || |

ascendente:(5,6)~(1,2)

ascendente:(1,2)~(5,6)

descendente:(3,8)~(5,6)

ascendente:(3,8)~(2,7)

descendente:(5,6)~(3,8)

ascendente:(2,7)~(3,8)

conflictos cromosoma 1 :6

| || || || || ||\*|| || |

| || || || || || || || |

| ||\*|| || || || || ||\*|

|\*|| || || || || ||\*|| |

| || || || || || || || |

| || || ||\*|| || || || |

| || ||\*|| || || || || |

| || || || ||\*|| || || |

descendente:(3,2)~(4,1)

ascendente:(8,5)~(4,1)

descendente:(4,1)~(3,2)

descendente:(6,4)~(7,3)

descendente:(7,3)~(6,4)

ascendente:(4,1)~(8,5)

ascendente:(3,8)~(1,6)

descendente:(3,8)~(4,7)

ascendente:(1,6)~(3,8)

descendente:(4,7)~(3,8)

conflictos cromosoma 2 :10

| || || || || || || ||\*|

| || || || || || ||\*|| |

| ||\*|| || || || || || |

|\*|| || || || || || || |

| || || || || ||\*|| || |

| || || ||\*|| || || || |

| || ||\*|| || || || || |

| || || || ||\*|| || || |

descendente:(3,2)~(4,1)

ascendente:(8,5)~(4,1)

descendente:(4,1)~(3,2)

descendente:(6,4)~(7,3)

descendente:(7,3)~(6,4)

ascendente:(4,1)~(8,5)

descendente:(1,8)~(2,7)

descendente:(2,7)~(1,8)

conflictos cromosoma 3 :8

|\*|| || || || || || || |

| || || || || || ||\*|| |

| || ||\*|| || || || || |

| || || || || || || ||\*|

| || || || || ||\*|| || |

| || || ||\*|| || || || |

| ||\*|| || || || || || |

| || || || ||\*|| || || |

ascendente:(3,3)~(1,1)

descendente:(2,7)~(7,2)

ascendente:(1,1)~(3,3)

descendente:(7,2)~(2,7)

conflictos cromosoma 4 :4

cromosoma = 1 fitnes = 0.14286

cromosoma = 2 fitnes = 0.09091

cromosoma = 3 fitnes = 0.11111

cromosoma = 4 fitnes = 0.20000

cromosoma = 1 probabilidad = 0.26219

cromosoma = 2 probabilidad = 0.16684

cromosoma = 3 probabilidad = 0.20392

cromosoma = 4 probabilidad = 0.36705

[0.00000,0.26219)

[0.26219,0.42903)

[0.42903,0.63295)

[0.63295,1.00000)

idRango: 1 variableCross: 0.38782

idRango: 2 variableCross: 0.44986

padres

4 3 7 6 8 1 4 3 probabilidad: 0.16684

4 3 7 6 8 5 2 1 probabilidad: 0.20392

hijos cross

4 3 7 6 8 5 2 1

4 3 7 6 8 1 4 3

hijos mutados

4 3 7 6 8 5 2 1

4 3 5 6 8 1 4 3

6 1 7 2 8 5 2 3 idOrden: 0 ,fitnes: 0.14286

4 3 7 6 8 1 4 3 idOrden: 1 ,fitnes: 0.09091

4 3 7 6 8 5 2 1 idOrden: 2 ,fitnes: 0.11111

1 7 3 6 8 5 2 4 idOrden: 3 ,fitnes: 0.20000

4 3 7 6 8 1 4 3 idOrden: 0 ,fitnes: 0.09091

4 3 7 6 8 5 2 1 idOrden: 1 ,fitnes: 0.11111

6 1 7 2 8 5 2 3 idOrden: 2 ,fitnes: 0.14286

1 7 3 6 8 5 2 4 idOrden: 3 ,fitnes: 0.20000

4 3 7 6 8 5 2 1 idOrden: 0 ,fitnes: 0.11111

4 3 5 6 8 1 4 3 idOrden: 1 ,fitnes: 0.09091

6 1 7 2 8 5 2 3 idOrden: 2 ,fitnes: 0.14286

1 7 3 6 8 5 2 4 idOrden: 3 ,fitnes: 0.20000

| || || || || || || ||\*|

| || || || || || ||\*|| |

| ||\*|| || || || || || |

|\*|| || || || || || || |

| || || || || ||\*|| || |

| || || ||\*|| || || || |

| || ||\*|| || || || || |

| || || || ||\*|| || || |

descendente:(3,2)~(4,1)

ascendente:(8,5)~(4,1)

descendente:(4,1)~(3,2)

descendente:(6,4)~(7,3)

descendente:(7,3)~(6,4)

ascendente:(4,1)~(8,5)

descendente:(1,8)~(2,7)

descendente:(2,7)~(1,8)

conflictos cromosoma 1 :8

| || || || || ||\*|| || |

| || || || || || || || |

| ||\*|| || || || || ||\*|

|\*|| || || || || ||\*|| |

| || ||\*|| || || || || |

| || || ||\*|| || || || |

| || || || || || || || |

| || || || ||\*|| || || |

descendente:(3,2)~(4,1)

ascendente:(8,5)~(4,1)

descendente:(4,1)~(3,2)

ascendente:(6,4)~(5,3)

ascendente:(5,3)~(6,4)

ascendente:(4,1)~(8,5)

ascendente:(3,8)~(1,6)

descendente:(3,8)~(4,7)

ascendente:(1,6)~(3,8)

descendente:(4,7)~(3,8)

conflictos cromosoma 2 :10

| ||\*|| || || || || || |

| || || ||\*|| || ||\*|| |

| || || || || || || ||\*|

| || || || || || || || |

| || || || || ||\*|| || |

|\*|| || || || || || || |

| || ||\*|| || || || || |

| || || || ||\*|| || || |

ascendente:(5,6)~(1,2)

ascendente:(1,2)~(5,6)

descendente:(3,8)~(5,6)

ascendente:(3,8)~(2,7)

descendente:(5,6)~(3,8)

ascendente:(2,7)~(3,8)

conflictos cromosoma 3 :6

|\*|| || || || || || || |

| || || || || || ||\*|| |

| || ||\*|| || || || || |

| || || || || || || ||\*|

| || || || || ||\*|| || |

| || || ||\*|| || || || |

| ||\*|| || || || || || |

| || || || ||\*|| || || |

ascendente:(3,3)~(1,1)

descendente:(2,7)~(7,2)

ascendente:(1,1)~(3,3)

descendente:(7,2)~(2,7)

conflictos cromosoma 4 :4

cromosoma = 1 fitnes = 0.11111

cromosoma = 2 fitnes = 0.09091

cromosoma = 3 fitnes = 0.14286

cromosoma = 4 fitnes = 0.20000

cromosoma = 1 probabilidad = 0.20392

cromosoma = 2 probabilidad = 0.16684

cromosoma = 3 probabilidad = 0.26219

cromosoma = 4 probabilidad = 0.36705

[0.00000,0.20392)

[0.20392,0.37076)

[0.37076,0.63295)

[0.63295,1.00000)

idRango: 3 variableCross: 0.96676

idRango: 1 variableCross: 0.22582

padres

1 7 3 6 8 5 2 4 probabilidad: 0.36705

4 3 5 6 8 1 4 3 probabilidad: 0.16684

hijos cross

1 7 3 6 8 1 4 3

4 3 5 6 8 5 2 4

hijos mutados

1 7 3 6 8 1 2 3

4 3 5 6 8 1 2 4

4 3 7 6 8 5 2 1 idOrden: 0 ,fitnes: 0.11111

4 3 5 6 8 1 4 3 idOrden: 1 ,fitnes: 0.09091

6 1 7 2 8 5 2 3 idOrden: 2 ,fitnes: 0.14286

1 7 3 6 8 5 2 4 idOrden: 3 ,fitnes: 0.20000

4 3 5 6 8 1 4 3 idOrden: 0 ,fitnes: 0.09091

4 3 7 6 8 5 2 1 idOrden: 1 ,fitnes: 0.11111

6 1 7 2 8 5 2 3 idOrden: 2 ,fitnes: 0.14286

1 7 3 6 8 5 2 4 idOrden: 3 ,fitnes: 0.20000

1 7 3 6 8 1 2 3 idOrden: 0 ,fitnes: 0.09091

4 3 5 6 8 1 2 4 idOrden: 1 ,fitnes: 0.11111

6 1 7 2 8 5 2 3 idOrden: 2 ,fitnes: 0.14286

1 7 3 6 8 5 2 4 idOrden: 3 ,fitnes: 0.20000

|\*|| || || || ||\*|| || |

| || || || || || ||\*|| |

| || ||\*|| || || || ||\*|

| || || || || || || || |

| || || || || || || || |

| || || ||\*|| || || || |

| ||\*|| || || || || || |

| || || || ||\*|| || || |

ascendente:(3,3)~(1,1)

descendente:(2,7)~(7,2)

ascendente:(1,1)~(3,3)

ascendente:(2,7)~(1,6)

ascendente:(3,8)~(1,6)

descendente:(7,2)~(2,7)

ascendente:(1,6)~(2,7)

ascendente:(3,8)~(2,7)

ascendente:(1,6)~(3,8)

ascendente:(2,7)~(3,8)

conflictos cromosoma 1 :10

| || || || || ||\*|| || |

| || || || || || ||\*|| |

| ||\*|| || || || || || |

|\*|| || || || || || ||\*|

| || ||\*|| || || || || |

| || || ||\*|| || || || |

| || || || || || || || |

| || || || ||\*|| || || |

descendente:(3,2)~(4,1)

ascendente:(8,5)~(4,1)

descendente:(4,1)~(3,2)

ascendente:(6,4)~(5,3)

ascendente:(5,3)~(6,4)

ascendente:(4,1)~(8,5)

ascendente:(2,7)~(1,6)

ascendente:(1,6)~(2,7)

conflictos cromosoma 2 :8

| ||\*|| || || || || || |

| || || ||\*|| || ||\*|| |

| || || || || || || ||\*|

| || || || || || || || |

| || || || || ||\*|| || |

|\*|| || || || || || || |

| || ||\*|| || || || || |

| || || || ||\*|| || || |

ascendente:(5,6)~(1,2)

ascendente:(1,2)~(5,6)

descendente:(3,8)~(5,6)

ascendente:(3,8)~(2,7)

descendente:(5,6)~(3,8)

ascendente:(2,7)~(3,8)

conflictos cromosoma 3 :6

|\*|| || || || || || || |

| || || || || || ||\*|| |

| || ||\*|| || || || || |

| || || || || || || ||\*|

| || || || || ||\*|| || |

| || || ||\*|| || || || |

| ||\*|| || || || || || |

| || || || ||\*|| || || |

ascendente:(3,3)~(1,1)

descendente:(2,7)~(7,2)

ascendente:(1,1)~(3,3)

descendente:(7,2)~(2,7)

conflictos cromosoma 4 :4

cromosoma = 1 fitnes = 0.09091

cromosoma = 2 fitnes = 0.11111

cromosoma = 3 fitnes = 0.14286

cromosoma = 4 fitnes = 0.20000

cromosoma = 1 probabilidad = 0.16684

cromosoma = 2 probabilidad = 0.20392

cromosoma = 3 probabilidad = 0.26219

cromosoma = 4 probabilidad = 0.36705

[0.00000,0.16684)

[0.16684,0.37076)

[0.37076,0.63295)

[0.63295,1.00000)

idRango: 3 variableCross: 0.85694

idRango: 3 variableCross: 0.75959

padres

1 7 3 6 8 5 2 4 probabilidad: 0.36705

1 7 3 6 8 5 2 4 probabilidad: 0.36705

hijos cross

1 7 3 6 8 5 2 4

1 7 3 6 8 5 2 4

hijos mutados

1 7 3 6 8 5 2 4

1 7 3 6 8 5 2 4

1 7 3 6 8 1 2 3 idOrden: 0 ,fitnes: 0.09091

4 3 5 6 8 1 2 4 idOrden: 1 ,fitnes: 0.11111

6 1 7 2 8 5 2 3 idOrden: 2 ,fitnes: 0.14286

1 7 3 6 8 5 2 4 idOrden: 3 ,fitnes: 0.20000

1 7 3 6 8 1 2 3 idOrden: 0 ,fitnes: 0.09091

4 3 5 6 8 1 2 4 idOrden: 1 ,fitnes: 0.11111

6 1 7 2 8 5 2 3 idOrden: 2 ,fitnes: 0.14286

1 7 3 6 8 5 2 4 idOrden: 3 ,fitnes: 0.20000

1 7 3 6 8 5 2 4 idOrden: 0 ,fitnes: 0.20000

1 7 3 6 8 5 2 4 idOrden: 1 ,fitnes: 0.20000

6 1 7 2 8 5 2 3 idOrden: 2 ,fitnes: 0.14286

1 7 3 6 8 5 2 4 idOrden: 3 ,fitnes: 0.20000

|\*|| || || || || || || |

| || || || || || ||\*|| |

| || ||\*|| || || || || |

| || || || || || || ||\*|

| || || || || ||\*|| || |

| || || ||\*|| || || || |

| ||\*|| || || || || || |

| || || || ||\*|| || || |

ascendente:(3,3)~(1,1)

descendente:(2,7)~(7,2)

ascendente:(1,1)~(3,3)

descendente:(7,2)~(2,7)

conflictos cromosoma 1 :4

|\*|| || || || || || || |

| || || || || || ||\*|| |

| || ||\*|| || || || || |

| || || || || || || ||\*|

| || || || || ||\*|| || |

| || || ||\*|| || || || |

| ||\*|| || || || || || |

| || || || ||\*|| || || |

ascendente:(3,3)~(1,1)

descendente:(2,7)~(7,2)

ascendente:(1,1)~(3,3)

descendente:(7,2)~(2,7)

conflictos cromosoma 2 :4

| ||\*|| || || || || || |

| || || ||\*|| || ||\*|| |

| || || || || || || ||\*|

| || || || || || || || |

| || || || || ||\*|| || |

|\*|| || || || || || || |

| || ||\*|| || || || || |

| || || || ||\*|| || || |

ascendente:(5,6)~(1,2)

ascendente:(1,2)~(5,6)

descendente:(3,8)~(5,6)

ascendente:(3,8)~(2,7)

descendente:(5,6)~(3,8)

ascendente:(2,7)~(3,8)

conflictos cromosoma 3 :6

|\*|| || || || || || || |

| || || || || || ||\*|| |

| || ||\*|| || || || || |

| || || || || || || ||\*|

| || || || || ||\*|| || |

| || || ||\*|| || || || |

| ||\*|| || || || || || |

| || || || ||\*|| || || |

ascendente:(3,3)~(1,1)

descendente:(2,7)~(7,2)

ascendente:(1,1)~(3,3)

descendente:(7,2)~(2,7)

conflictos cromosoma 4 :4

cromosoma = 1 fitnes = 0.20000

cromosoma = 2 fitnes = 0.20000

cromosoma = 3 fitnes = 0.14286

cromosoma = 4 fitnes = 0.20000

cromosoma = 1 probabilidad = 0.26923

cromosoma = 2 probabilidad = 0.26923

cromosoma = 3 probabilidad = 0.19231

cromosoma = 4 probabilidad = 0.26923

[0.00000,0.26923)

[0.26923,0.53846)

[0.53846,0.73077)

[0.73077,1.00000)

idRango: 1 variableCross: 0.28910

idRango: 0 variableCross: 0.22442

padres

1 7 3 6 8 5 2 4 probabilidad: 0.26923

1 7 3 6 8 5 2 4 probabilidad: 0.26923

hijos cross

1 7 3 6 8 5 2 4

1 7 3 6 8 5 2 4

hijos mutados

1 7 3 6 8 5 2 4

1 7 3 6 8 5 2 4

1 7 3 6 8 5 2 4 idOrden: 0 ,fitnes: 0.20000

1 7 3 6 8 5 2 4 idOrden: 1 ,fitnes: 0.20000

6 1 7 2 8 5 2 3 idOrden: 2 ,fitnes: 0.14286

1 7 3 6 8 5 2 4 idOrden: 3 ,fitnes: 0.20000

6 1 7 2 8 5 2 3 idOrden: 0 ,fitnes: 0.14286

1 7 3 6 8 5 2 4 idOrden: 1 ,fitnes: 0.20000

1 7 3 6 8 5 2 4 idOrden: 2 ,fitnes: 0.20000

1 7 3 6 8 5 2 4 idOrden: 3 ,fitnes: 0.20000

1 7 3 6 8 5 2 4 idOrden: 0 ,fitnes: 0.20000

1 7 3 6 8 5 2 4 idOrden: 1 ,fitnes: 0.20000

1 7 3 6 8 5 2 4 idOrden: 2 ,fitnes: 0.20000

1 7 3 6 8 5 2 4 idOrden: 3 ,fitnes: 0.20000

FLORES VILLAR@DESKTOP-IPF07U2 MINGW64 ~/Desktop/matematica\_computacional/PC5AgoritmosGeneticos

$ javac The8QueensPuzzle2.java

FLORES VILLAR@DESKTOP-IPF07U2 MINGW64 ~/Desktop/matematica\_computacional/PC5AgoritmosGeneticos

$ java The8QueensPuzzle2

6 7 3 5 2 4 8 1

3 7 8 2 1 4 5 6

1 5 3 8 7 2 4 6

1 6 7 2 4 8 3 5

| || || || || || || ||\*|

| || || || ||\*|| || || |

| || ||\*|| || || || || |

| || || || || ||\*|| || |

| || || ||\*|| || || || |

|\*|| || || || || || || |

| ||\*|| || || || || || |

| || || || || || ||\*|| |

ascendente:(7,2)~(6,1)

descendente:(2,5)~(6,1)

ascendente:(6,1)~(7,2)

descendente:(5,4)~(7,2)

descendente:(1,8)~(7,2)

descendente:(7,2)~(5,4)

ascendente:(8,7)~(5,4)

descendente:(1,8)~(5,4)

descendente:(6,1)~(2,5)

ascendente:(5,4)~(8,7)

descendente:(7,2)~(1,8)

descendente:(5,4)~(1,8)

conflictos cromosoma 1 :12

| || || || ||\*|| || || |

| || || ||\*|| || || || |

|\*|| || || || || || || |

| || || || || ||\*|| || |

| || || || || || ||\*|| |

| || || || || || || ||\*|

| ||\*|| || || || || || |

| || ||\*|| || || || || |

ascendente:(8,3)~(7,2)

ascendente:(7,2)~(8,3)

descendente:(1,5)~(2,4)

ascendente:(4,6)~(2,4)

ascendente:(5,7)~(2,4)

ascendente:(6,8)~(2,4)

descendente:(2,4)~(1,5)

ascendente:(2,4)~(4,6)

ascendente:(5,7)~(4,6)

ascendente:(6,8)~(4,6)

ascendente:(2,4)~(5,7)

ascendente:(4,6)~(5,7)

ascendente:(6,8)~(5,7)

ascendente:(2,4)~(6,8)

ascendente:(4,6)~(6,8)

ascendente:(5,7)~(6,8)

conflictos cromosoma 2 :16

|\*|| || || || || || || |

| || || || || ||\*|| || |

| || ||\*|| || || || || |

| || || || || || ||\*|| |

| ||\*|| || || || || || |

| || || || || || || ||\*|

| || || || ||\*|| || || |

| || || ||\*|| || || || |

ascendente:(3,3)~(1,1)

ascendente:(1,1)~(3,3)

descendente:(7,5)~(8,4)

descendente:(8,4)~(7,5)

conflictos cromosoma 3 :4

|\*|| || || || || || || |

| || || ||\*|| || || || |

| || || || || || ||\*|| |

| || || || ||\*|| || || |

| || || || || || || ||\*|

| ||\*|| || || || || || |

| || ||\*|| || || || || |

| || || || || ||\*|| || |

ascendente:(7,3)~(6,2)

ascendente:(6,2)~(7,3)

descendente:(3,7)~(7,3)

descendente:(7,3)~(3,7)

conflictos cromosoma 4 :4

cromosoma = 1 fitnes = 0.07692

cromosoma = 2 fitnes = 0.05882

cromosoma = 3 fitnes = 0.20000

cromosoma = 4 fitnes = 0.20000

cromosoma = 1 probabilidad = 0.14358

cromosoma = 2 probabilidad = 0.10979

cromosoma = 3 probabilidad = 0.37332

cromosoma = 4 probabilidad = 0.37332

[0.00000,0.14358)

[0.14358,0.25337)

[0.25337,0.62668)

[0.62668,1.00000)

idRango: 2 variableCross: 0.62499

idRango: 0 variableCross: 0.12272

padres

1 5 3 8 7 2 4 6 probabilidad: 0.37332

6 7 3 5 2 4 8 1 probabilidad: 0.14358

hijos cross

1 5 3 8 2 4 8 1

6 7 3 5 7 2 4 6

hijos mutados

1 5 3 8 2 4 8 6

6 8 3 5 7 2 4 6

6 7 3 5 2 4 8 1 idOrden: 0 ,fitnes: 0.07692

3 7 8 2 1 4 5 6 idOrden: 1 ,fitnes: 0.05882

1 5 3 8 7 2 4 6 idOrden: 2 ,fitnes: 0.20000

1 6 7 2 4 8 3 5 idOrden: 3 ,fitnes: 0.20000

3 7 8 2 1 4 5 6 idOrden: 0 ,fitnes: 0.05882

6 7 3 5 2 4 8 1 idOrden: 1 ,fitnes: 0.07692

1 5 3 8 7 2 4 6 idOrden: 2 ,fitnes: 0.20000

1 6 7 2 4 8 3 5 idOrden: 3 ,fitnes: 0.20000

1 5 3 8 2 4 8 6 idOrden: 0 ,fitnes: 0.14286

6 8 3 5 7 2 4 6 idOrden: 1 ,fitnes: 1.00000

1 5 3 8 7 2 4 6 idOrden: 2 ,fitnes: 0.20000

1 6 7 2 4 8 3 5 idOrden: 3 ,fitnes: 0.20000

|\*|| || || || || || || |

| || || || ||\*|| || || |

| || ||\*|| || || || || |

| || || || || ||\*|| || |

| ||\*|| || || || || || |

| || || || || || || ||\*|

| || || || || || || || |

| || || ||\*|| || ||\*|| |

ascendente:(3,3)~(1,1)

descendente:(2,5)~(5,2)

ascendente:(1,1)~(3,3)

descendente:(5,2)~(2,5)

ascendente:(6,8)~(4,6)

ascendente:(4,6)~(6,8)

conflictos cromosoma 1 :6

| || || || || || || || |

| || || || || ||\*|| || |

| || ||\*|| || || || || |

| || || || || || ||\*|| |

| || || ||\*|| || || || |

|\*|| || || || || || ||\*|

| || || || ||\*|| || || |

| ||\*|| || || || || || |

conflictos cromosoma 2 :0

|\*|| || || || || || || |

| || || || || ||\*|| || |

| || ||\*|| || || || || |

| || || || || || ||\*|| |

| ||\*|| || || || || || |

| || || || || || || ||\*|

| || || || ||\*|| || || |

| || || ||\*|| || || || |

ascendente:(3,3)~(1,1)

ascendente:(1,1)~(3,3)

descendente:(7,5)~(8,4)

descendente:(8,4)~(7,5)

conflictos cromosoma 3 :4

|\*|| || || || || || || |

| || || ||\*|| || || || |

| || || || || || ||\*|| |

| || || || ||\*|| || || |

| || || || || || || ||\*|

| ||\*|| || || || || || |

| || ||\*|| || || || || |

| || || || || ||\*|| || |

ascendente:(7,3)~(6,2)

ascendente:(6,2)~(7,3)

descendente:(3,7)~(7,3)

descendente:(7,3)~(3,7)

conflictos cromosoma 4 :4

cromosoma = 1 fitnes = 0.14286

cromosoma = 2 fitnes = 1.00000

cromosoma = 3 fitnes = 0.20000

cromosoma = 4 fitnes = 0.20000

cromosoma = 1 probabilidad = 0.09259

cromosoma = 2 probabilidad = 0.64815

cromosoma = 3 probabilidad = 0.12963

cromosoma = 4 probabilidad = 0.12963

[0.00000,0.09259)

[0.09259,0.74074)

[0.74074,0.87037)

[0.87037,1.00000)

idRango: 1 variableCross: 0.69452

idRango: 2 variableCross: 0.76046

padres

6 8 3 5 7 2 4 6 probabilidad: 0.64815

1 5 3 8 7 2 4 6 probabilidad: 0.12963

hijos cross

6 8 3 5 7 2 4 6

1 5 3 8 7 2 4 6

hijos mutados

6 8 3 5 7 2 4 6

1 5 3 8 7 2 4 6

1 5 3 8 2 4 8 6 idOrden: 0 ,fitnes: 0.14286

6 8 3 5 7 2 4 6 idOrden: 1 ,fitnes: 1.00000

1 5 3 8 7 2 4 6 idOrden: 2 ,fitnes: 0.20000

1 6 7 2 4 8 3 5 idOrden: 3 ,fitnes: 0.20000

1 5 3 8 2 4 8 6 idOrden: 0 ,fitnes: 0.14286

1 5 3 8 7 2 4 6 idOrden: 1 ,fitnes: 0.20000

1 6 7 2 4 8 3 5 idOrden: 2 ,fitnes: 0.20000

6 8 3 5 7 2 4 6 idOrden: 3 ,fitnes: 1.00000

6 8 3 5 7 2 4 6 idOrden: 0 ,fitnes: 1.00000

1 5 3 8 7 2 4 6 idOrden: 1 ,fitnes: 0.20000

1 6 7 2 4 8 3 5 idOrden: 2 ,fitnes: 0.20000

6 8 3 5 7 2 4 6 idOrden: 3 ,fitnes: 1.00000

import java.util.Random;

import java.util.Scanner;

import java.io.IOException;

public class The8QueensPuzzle{

    //poblacion inicial

    //fitness

    //crossover

    //mutacion  REPETIR

    public The8QueensPuzzle(){

    }

    //----------------------------------------------------------------------

    public static void main(String[]args){

        int N=4;

        Poblacion poblacion = new Poblacion(N);

        //Integer[][] cromosomas= new Integer[N][8];

        //asignar elementos de modo que no ese repitan

        //------------------------------------------------------------------

        Random rand = new Random();

        for(int i=0;i<poblacion.cromosomas.length;i++){

            poblacion.cromosomas[i].indOrden = i;

            poblacion.cromosomas[i].genes[0] = rand.nextInt(8)+1;

            for(int j=1;j<poblacion.cromosomas[0].genes.length;j++){

                int num= -1;

                //corroborar que no se repita

                int posRep = -1;

                do{

                    //System.out.println("dentro de do ,posResp="+posRep);

                    num = rand.nextInt(8)+1;

                    //busqueda en los elementos ver si se REPITE

                    int k =0;

                    for(;k<j;k++){

                        if(num == poblacion.cromosomas[i].genes[k]){

                            posRep = k;

                            break;

                        }

                    }if(k==j){

                        posRep = -1;

                    }

                }while(posRep!=-1);

                poblacion.cromosomas[i].genes[j] = num;

            }

        }

        //--------------------------------------------------------------------------

        //imprimiendo la poblacion inicial

        for(int i=0;i<poblacion.cromosomas.length;i++){

            for(int j=0;j<poblacion.cromosomas[0].genes.length;j++){

                System.out.printf("%d\t",poblacion.cromosomas[i].genes[j]);

            }

            System.out.println();

        }

       //----------------WHILE-------------------(acabar si fitnes de algun cromosoma es 1)

       boolean flagGlobal = false;

       int contadorGlobal =0;

       while(!flagGlobal&&(contadorGlobal<5)){

             //funcion aptitud fitness INICIO

        //------------------------------------------------------------------------

        for(int i =0;i<poblacion.cromosomas.length;i++){

            poblacion.CalcularFitness(i);

        }

        // ----fucion aptitud fitness FIN

        //-----------------------------------------------------------------------

        for(int i=0;i<poblacion.cromosomas.length;i++){

            System.out.printf("cromosoma = %d fitnes = %1.5f\n",(poblacion.cromosomas[i].indOrden+1),poblacion.cromosomas[i].fitness);

        }

        //----SELECCION---------------------------------------------------

        //---calculo de probabilidades----

        poblacion.CalcularProbabilidades();

        for(int i=0;i<poblacion.probabilidades.length;i++){

            System.out.printf("cromosoma = %d probabilidad = %1.5f\n",(poblacion.cromosomas[i].indOrden+1),poblacion.probabilidades[i]);

        }

        poblacion.CrearRangos();

        for(int i=0;i<poblacion.cromosomas.length;i++){

            System.out.printf("[%1.5f,%1.5f)\n",poblacion.cromosomas[i].rango.lInferior,poblacion.cromosomas[i].rango.lSuperior);

        }

        //--------SELECCION---------------------generar el numero aleatorio para escoger al par que van a hacer el crossover

        Random random=new Random();

        double variableCross;

        //si variableCross esta en alguno de los intervalos

        int [] indiceElegidos = new int[2];

        int cont = 0;

        while(cont < indiceElegidos.length){

            variableCross = random.nextDouble();

            for(int i=0;i<poblacion.cromosomas.length;i++){

                if(poblacion.cromosomas[i].rango.pertenece(variableCross)){

                    if(poblacion.cromosomas[i].rango.idRango!=((i==1)?indiceElegidos[i-1]:-1)){

                        System.out.printf(" idRango: %d variableCross: %1.5f\n",poblacion.cromosomas[i].rango.idRango,variableCross);        //ind actual diferente del anterior)

                        indiceElegidos[cont] = poblacion.cromosomas[i].rango.idRango;

                        cont++;

                        break;

                    }else{

                        cont = cont ;

                        break;

                    }

                }

            }

        }

        //imprimiendo los elegidos

         System.out.println("padres");

        for(int i=0;i<indiceElegidos.length;i++){

            for(int j=0;j<poblacion.cromosomas[0].genes.length;j++){

                System.out.printf("%d\t",poblacion.cromosomas[indiceElegidos[i]].genes[j]);

            }

            System.out.printf("probabilidad: %1.5f\n",poblacion.probabilidades[indiceElegidos[i]]);

        }

        //------------------------------------CROSSOVER------------------realizar el cross

        int nHijos = 2;

        Cromosoma[] hijos = new Cromosoma[nHijos];

        for(int i = 0;i<hijos.length;i++){

            hijos[i] = new Cromosoma();

            for(int j=0;j<hijos[i].genes.length;j++){

                //System.out.println(j>3?(int)Math.abs(i-1):i);

                hijos[i].genes[j] = poblacion.cromosomas[indiceElegidos[j>3?(int)Math.abs(i-1):i]].genes[j];

            }

        }

        //imprimiendo cromosomasHijos

        System.out.println("hijos cross");

        for(int i=0;i<hijos.length;i++){

            for(int j=0;j<hijos[i].genes.length;j++){

                System.out.printf("%d\t",hijos[i].genes[j]);

            }

            System.out.println();

        }

        //-----------------------MUTACION----------------------------

        //ordenar segun fitness y reemplazar los de abajo(menores)

        int indMutar ;

        int newElemFila ;

        for(int i=0;i<hijos.length;i++){

            boolean flag = false;

            int contador = 0;

            do{

                contador ++;

                indMutar = rand.nextInt(8); //columna

                newElemFila = rand.nextInt(8)+1; //elemento nuevo fila

                //System.out.println("indMutar:"+indMutar+" newElemFila: "+newElemFila);

                    //System.out.println("dentro de if indMutar[]!=newElemFila");

                    int j;

                    for(j=0;j<hijos[i].genes.length;j++){ // buscando la no repeticion

                        if(hijos[i].genes[j]==newElemFila){

                            flag = false;

                            break;

                        }

                    }

                    if(j==hijos[0].genes.length){

                        hijos[i].genes[indMutar] = newElemFila;

                        flag = true;

                    }

            }while(!flag&&(contador<10));

        }

        //----------------------------------------------------

        System.out.println("hijos mutados");

        for(int i=0;i<hijos.length;i++){

            for(int j=0;j<hijos[i].genes.length;j++){

                System.out.printf("%d\t",hijos[i].genes[j]);

            }

            System.out.println();

        }

        //---------------------------------------------------------

        poblacion.ImprimirCromosomas();

        poblacion.OrdenarIndices();

        poblacion.ImprimirCromosomas();

        //reemplazando los 2 mas bajos

        poblacion.InsertarHijos(hijos);

        poblacion.ImprimirCromosomas();

        for(int i =0;i<poblacion.cromosomas.length;i++){

            flagGlobal =(poblacion.cromosomas[i].fitness ==1);

        }

        contadorGlobal ++;

       }//fin While Global

    } //fin main

    //---------------------------------------------------------------------------------

}

//===============================================================================

class Poblacion{

    static Cromosoma [] cromosomas ;

    static double probabilidades[];

    //-----------------------------------------------------------------------------

    Poblacion(int N){

        cromosomas = new Cromosoma[N];

        for(int i=0;i<N;i++){

            cromosomas[i] = new Cromosoma();//creando los cromosomas

            //cromosomas[i].genes = new int[8]; ya se hace en el constructor de Cromosoma2

        }

        probabilidades = new double[N];

    }

    //-------------------------------------------------------------------------

    public static void OrdenarIndices(){

        //reemplazar cromosomas segun fitnes

        for(int i=0;i<cromosomas.length;i++){

            for(int j=0;j<(cromosomas.length-i-1);j++){

                if(cromosomas[j].fitness>cromosomas[j+1].fitness){

                    Cromosoma temp =new Cromosoma();

                    temp = cromosomas[j];

                    cromosomas[j] =cromosomas[j+1];

                    cromosomas[j+1] = temp;

                }

            }

        }

        for(int i=0;i<cromosomas.length;i++){

            cromosomas[i].indOrden = i;

        }

        for(int i=0;i<cromosomas.length;i++){

            cromosomas[i].rango.idRango = i;

        }

    }//---------------------------------------------------------------------------

    public static void CalcularFitness(Cromosoma cromosoma){

         //C[i] - i para la ascendente       C[i]+i para para la descendente

         int conflictos = 0;

         for(int j=0;j<cromosoma.genes.length;j++){

             //luego del elemento

             for(int k=0;k<cromosoma.genes.length;k++){

                 //si conflicto  conflictos++;

                if(k!=j){

                     if((cromosoma.genes[k]-(k+1) )==(cromosoma.genes[j]-(j+1))){

                         //System.out.printf("ascendente:(%d,%d)~(%d,%d)\n",cromosoma.genes[k],(k+1),cromosoma.genes[j],(j+1));

                         conflictos++;

                     }

                     if((cromosoma.genes[k]+(k+1))==(cromosoma.genes[j]+(j+1))){

                         //System.out.printf("descendente:(%d,%d)~(%d,%d)\n",cromosoma.genes[k],(k+1),cromosoma.genes[j],(j+1));

                         conflictos++;

                     }

                }

             }

         }

         //System.out.printf("conflictos cromosoma %d :%d\n",(cromosoma.indOrden+1),conflictos);

         double factor =Math.pow(10,5);

         cromosoma.fitness = Math.round(factor/(1+conflictos))/factor;

         System.out.println();

    }

    //---------------------------------------------------------------------------

    public static void CalcularFitness(int i){

         //C[i] - i para la ascendente       C[i]+i para para la descendente

         String [][] tabla = new String[8][8];

         for(int f=0;f<8;f++){

             for(int c=0;c<8;c++){

                 tabla[f][c] = "| |";

             }

         }

         for(int l=0;l<cromosomas[i].genes.length;l++){

             tabla[cromosomas[i].genes[l]-1][l]="|\*|";

         }

         System.out.println();

         for(int f=0;f<8;f++){

             for(int c=0;c<8;c++){

                 System.out.print(tabla[f][c]);

             }

             System.out.println();

         }

         int conflictos = 0;

         for(int j=0;j<cromosomas[0].genes.length;j++){

             //luego del elemento

             for(int k=0;k<cromosomas[i].genes.length;k++){

                 //si conflicto  conflictos++;

                if(k!=j){

                     if((cromosomas[i].genes[k]-(k+1) )==(cromosomas[i].genes[j]-(j+1))){

                         System.out.printf("ascendente:(%d,%d)~(%d,%d)\n",cromosomas[i].genes[k],(k+1),cromosomas[i].genes[j],(j+1));

                         conflictos++;

                     }

                     if((cromosomas[i].genes[k]+(k+1))==(cromosomas[i].genes[j]+(j+1))){

                         System.out.printf("descendente:(%d,%d)~(%d,%d)\n",cromosomas[i].genes[k],(k+1),cromosomas[i].genes[j],(j+1));

                         conflictos++;

                     }

                }

             }

         }

         System.out.printf("conflictos cromosoma %d :%d\n",(cromosomas[i].indOrden+1),conflictos);

         double factor =Math.pow(10,5);

         cromosomas[i].fitness = Math.round(factor/(1+conflictos))/factor;

         System.out.println();

    }

    //---------------------------------------------------------------------------

    public static double FitnessTotal(){

        double fitnessTotal = 0;

        for(int i=0;i<cromosomas.length;i++){

            fitnessTotal+=cromosomas[i].fitness;

        }

        return fitnessTotal;

    }

    //---------------------------------------------------------------------------

    public static void CalcularProbabilidades(){

        double fitnesTotal = FitnessTotal();

        for(int i=0;i<probabilidades.length;i++){

            probabilidades[i] = cromosomas[i].fitness/fitnesTotal;

        }

    }

    //------------------------------------------------------------------

    public static void CrearRangos(){

        double acumulada = 0;

        for(int i = 0;i<cromosomas.length;i++){

            cromosomas[i].rango =new Rango(cromosomas[i].indOrden, acumulada, acumulada +probabilidades[i]);

            acumulada +=probabilidades[i];

        }

    }

    //------------------------------------------------------------------

    public static void ImprimirCromosomas(){

        System.out.println();

        for(int i= 0;i<cromosomas.length;i++){

            for(int j=0;j<cromosomas[i].genes.length;j++){

                System.out.printf("%d\t",cromosomas[i].genes[j]);

            }

            System.out.printf("idOrden: %d ,fitnes: %1.5f  \n",cromosomas[i].indOrden,cromosomas[i].fitness);

        }

        System.out.println();

    }

    //---------------------------------------------------------------------------------------------

    public static void InsertarHijos(Cromosoma[] cromos){

        for(int i = 0; i<cromos.length;i++){

            for(int j=0;j<cromos[0].genes.length;j++){

                cromosomas[i].genes[j] = cromos[i].genes[j];

            }

            CalcularFitness(cromosomas[i]);

        }

    }

}

//============================================================================

class Cromosoma{

    int[] genes;

    int indOrden;

    double fitness;

    double probabilidad;

    Rango rango;

    Cromosoma(){

        genes = new int[8];

    }

}

//===================================================================================================

class Rango{

    double lSuperior;

    double lInferior;

    int idRango;

    Rango(int id, double inf, double sup){

        this.idRango = id;

        this.lInferior = inf;

        this.lSuperior = sup;

    }

    public boolean pertenece(double num){

        if((num>=lInferior) && (num<lSuperior)){

            return true;

        }

        return false;

    }

}