ESCOLA SUPERIOR DE TECNOLOGIA E GESTÃO

INSTITUTO POLITÉCNICO DA GUARDA

SIMULADOR ALGORITMO GENÉTICO

TRABALHO PRÁTICO N.º 11

Curso	Engenharia Informática
Unidade Curricular	Inteligência Artificial
Ano / Semestre	3.º / 1.º
Ano Letivo	2012/2013
Docente	Celestino Gonçalves

Responsável	**Jorge Antunes nº1009689**		
Data	26.Fevereido.2013	Refª	

Índice

Introdução

Programa desenvolvido em linguagem java, no NetBeans IDE 7.3, com o objectivo de simular um algoritmo desenvolvido nos trabalhos Práticos, com a excepção do trabalho prático nº 1 e trabalho prático nº 10, realizados ao longo do semestre.
Para a realização do mesmo foi leccionado o trabalho prático nº 9, Algoritmo genético, e simular a evolução de uma população num certo número de gerações.

Desenvolvimento

Este simulador permite ao utilizador indicar quantos indivíduos a população deverá conter, assim como o número de gerações a avaliar e a probabilidade de reprodução, como se poderá ver nos Anexos.

Depois do utilizador todos os dados pretendidos e válidos, o programa começa por gerar os valores do cromossoma de cada individuo da população. Depois da sua criação, usando a função do trabalho prático nº 9 que é a seguinte:

$$f(x) = (x-3)^2 \quad \text{com } 1 \leq x \leq 6.$$

Usando a função mencionada em cima será calculada a função de Avaliação para cada individuo, assim como a Qualidade, Probabilidade e Segmento da Roleta. Onde se irão preservar os 2 indivíduos ou 1 único individuo de melhor qualidade para selecção por elitismo e os restantes através da roleta de modo a formarem pares para se reproduzirem.

De seguida era-se proceder a recombinação dos casais, onde se irá gerir um valor aleatório entre 0 e 1 para cada casal e os seus respectivos pontos de corte, também gerados aleatoriamente, mas só se irão reproduzir os casais do qual o valor aleatório seja inferior á probabilidade introduzida pelo utilizador, passando assim à construção do operador de reprodução, onde irão ser criados os filhos de cada casal, usando os pontos de corte para tal.

Por fim será feita a mutação de 2 genes, para o caso deste simulador, que consiste em inverter o valor do gene gerado aleatoriamente, por exemplo se o gene for 0 passa a 1, se for 1 passa a 0.

Conclusão

A elaboração deste simulador foi muito interessante pois desta forma foi possível ver na prática o que foi realizado no trabalho prático nº 9 e como este algoritmo se comporta na prática, vendo uma perspectiva muito melhor e constituição do mesmo. Também foi muito útil pois com este deu para meter em prática todo o conhecimento aprendido em cadeiras anteriores em java, assim como aprender novos métodos desconhecidos para a sua elaboração.

De uma forma geral os resultados foram os pretendidos, apesar de por vezes aparecerem erros de conversão de String para Integer ou vice-versa, sem perceber bem o porque destes erros, outro grande problema que o simulador tem quando se introduzem muitas gerações, ao passar das gerações o probabilidade de o segmento são cada vez menores de modo a permitir ver o verdadeiro resultado, onde surgiu uma possível conclusão para este problema, seja devido ao erro ou devido a más conversões de String para Integer ou vice-versa.

Foi pena não ter tido mais tempo para a elaboração do mesmo, de forma a poder realizar um simulador o mais genérico possível, pois com o tempo de duração da sua elaboração só deu para permitir ao utilizador escolher o numero de gerações, numero de indivíduos da população e a probabilidade de recombinação e tendo como objectivo inicial também permitir ao utilizador a introdução de quantos pontos de corte que se iria utilizar para a recombinação ou se pretendia uma recombinação uniforme, da função matemática e quantos indivíduos seleccionados por elitismo pretendia ou usando a formula genérica do algoritmo.

Anexos

Exemplo 1:Exemplo da avaliação de uma população numa geração:

Introduza o numero de iterações que pretenda:

2

Quantos Individuos pretende na População?

12

Introduza Probabilidade de Reprodução[50-80]:

50

*******ITERAÇÃO 1*******

Individuo Cromossoma

1 1110010000010

2 1111010010111

3 1101100101111

4 1001100010000

5 1010110110101

6 0101010000100

7 1100011001110

8 1011101111101

9 1110010100011

10 0010101010101

11	1111010001001
12	0101011000001

Individuo	Avaliação	Qualidade	Probabilidade	Segmento Roleta	
1	6.345	11.189		0.138	0.138
2	6.736	13.957		0.173	0.311
3	6.097	9.591	0.119	0.43	
4	4.574	2.477	0.03	0.459	
5	5.07	4.284	0.053	0.512	
6	2.971	0.0	0.0	0.512	
7	5.651	7.027	0.087	0.599	
8	5.404	5.779	0.071	0.669	
9	6.37	11.356	0.14	0.809	
10	1.999	1.002	0.012	0.821	
11	6.726	13.883		0.172	0.992
12	3.016	0.0	0.0	0.992	

Elitismo: 2 e 11

Roleta: Valor	Individuo
0.593	7
0.391	3

0.973 11

0.474 5

0.692 9

0.419 3

0.796 9

0.322 3

0.092 1

0.251 2

Pares	Valor Aleatório[0;1]	Ponto de Corte	Filhos
2	-	-	i'1
11	-	-	i'2
7 e 3	0.947	0,0 e 0	i'3 e 4
11 e 5	0.008	2,3 e 9	i'4 e 5
9 e 3	0.735	0,0 e 0	i'5 e 6
9 e 3	0.352	1,6 e 12	i'6 e 7
1 e 2	0.345	3,4 e 5	i'7 e 8

-------Reprodução--------

1100011001110

1101100101111

1101100101111

1100011001110

---

11 1 101000 1001

10 1 011011 0101

1111010000101

1010110111001

  1110010100011

  1101100101111

1101100101111

1110010100011

---

1 11001 010001 1

1 10110 010111 1

1101100100011

1110010101111

---

111 0 0 10000010

111 1 0 10010111

1111010010111

1110010000010

-----Recombinação-----

1101100101111

0101011000001

1101100101111

1100011001110

1111010000101

1010110111001

1101100101111

1110010100011

1101100100011

1110010101111

1111010010111

1110010000010

-----Mutação------

Genes a alterar:

-> Individuo 1 no gene 9

-> Individuo 12 no gene 11

1101100111111

0101011000001

1101100101111

1100011001110

1111010000101

1010110111001

1101100101111

1110010100011

1101100100011

1110010101111

1111010010111

1110010000110

*******ITERAÇÃO 2*******

Individuo	Cromossoma
1	1101100111111
2	0101011000001
3	1101100101111
4	1100011001110
5	1111010000101
6	1010110111001
7	1101100101111
8	1110010100011
9	1101100100011
10	1110010101111
11	1111010010111
12	1110010000110

Individuo	Avaliação	Qualidade	Probabilidade	Segmento Roleta
1	6.109	9.665	0.05	0.05

2	3.016	0.0	0.0	0.05
3	6.097	9.591	0.049	0.099
4	5.651	7.027	0.036	0.135
5	6.723	13.86	0.072	0.207
6	5.073	4.297	0.022	0.228
7	6.097	9.591	0.049	0.277
8	6.37	11.356	0.059	0.336
9	6.088	9.535	0.049	0.385
10	6.378	11.41	0.059	0.444
11	6.736	13.957	0.072	0.516
12	6.348	11.209	0.058	0.574

Elitismo: 0 e 0

Roleta: Valor Individuo

0.351 9

0.099 1

0.288 8

0.414 10

0.288 8

0.37 9

0.23 7

0.683 1

0.914 1

0.566 12

Pares	Valor Aleatório[0;1]	Ponto de Corte	Filhos
0	-	-	i'1
0	-	-	i'2
9 e 1	0.865	0,0 e 0	i'3 e 4
8 e 10	0.389	3,10 e 11	i'4 e 5
8 e 9	0.904	0,0 e 0	i'5 e 6
7 e 1	0.253	2,6 e 9	i'6 e 7
1 e 12	0.464	1,9 e 12	i'7 e 8

-------Reprodução--------

1101100100011

1101100111111

1101100111111

1101100100011

---

111 0010100 0 11

111 0010101 1 11

1110010101011

1110010100111

1110010100011

1101100100011

1101100100011

1110010100011

---

11 0110 010 1111

11 0110 011 1111

1101100101111

1101100111111

---

1 10110011 111 1

1 11001000 011 0

1110010001110

1101100110111

-----Recombinação-----

1101100111111

1101100111111

1101100111111

1101100100011

1110010101011

1110010100111

1101100100011

1110010100011

1101100101111

1101100111111

1110010001110

1101100110111

-----Mutação------

Genes a alterar:

-> Individuo 6 no gene 4

-> Individuo 4 no gene 13

1101100111111

1101100111111

1101100111111

1101100100010

1110010101011

1111010100111

1101100100011

1110010100011

1101100101111

1101100111111

1110010001110

1101100110111

****

********Avaliação*******

Geração Melhor Qualidade Media

1	13.957	6.712
2	13.957	16.003

Exemplo 2: Pelo facto do output do simulador ser muito extenso só para uma população será apresentado um exemplo abaixo só com a tabela de avaliação da população em 60 gerações:

Introduza o numero de iterações que pretenda:

60

Quantos Individuos pretende na População?

20

Introduza Probabilidade de Reprodução[50-80]:

75

*******ITERAÇÃO 1*******

Individuo	Cromossoma
1	1110001001100
2	0011111001000
3	0010110010110
4	1100100001010
5	0110110110111
6	1101011110010
7	0101101101101
8	1000110011100
9	0001001100110
10	1110100010111

11	0000000001111
12	1011011010100
13	1100110010110
14	0111001001001
15	1101111101111
16	0000011100110
17	0101100100101
18	1101011101100
19	1000010111000
20	0001110100101

****

********Avaliação*******

Geração	Melhor Qualidade	Media
1	11.937	4.434
2	11.937	11.185
3	11.937	18.645
4	11.937	23.332
5	11.937	27.501
6	11.937	29.788
7	11.937	31.42
8	11.937	33.052
9	11.937	34.662
10	11.937	36.265
11	11.937	37.878

12	11.937	40.922
13	11.937	44.027
14	11.937	47.218
15	11.937	50.7
16	11.937	54.099
17	11.937	57.204
18	11.937	60.305
19	11.937	63.302
20	11.937	66.437
21	11.937	69.542
22	11.937	72.641
23	11.937	75.638
24	11.937	79.132
25	11.937	89.548
26	11.937	100.18
27	11.937	110.813
28	11.937	121.454
29	11.937	132.15
30	11.937	142.819
31	11.937	153.527
32	11.937	163.856
33	11.937	174.565
34	11.937	185.046
35	11.937	195.753
36	11.937	206.105

37	11.937	216.801
38	11.937	227.501
39	11.937	238.208
40	11.937	248.679
41	12.124	259.6
42	12.787	271.63
43	12.787	282.154
44	12.787	294.262
45	12.787	306.385
46	14.876	318.583
47	14.876	329.986
48	14.876	340.616
49	14.876	351.537
50	14.876	362.458
51	14.876	373.095
52	14.876	383.94
53	14.876	394.322
54	14.876	394.408
55	14.876	394.495
56	14.876	395.12
57	14.876	395.744
58	14.876	395.831
59	14.876	395.916
60	14.876	396.003

Código:

```
package simulador;

import java.text.ParseException;
import java.util.Scanner;

/**
 *
 * @author Jorge
 */
public class Simulador {

    /**
     * @param args the command line arguments
     */
    private static double recombinacao = 0.75;
    private static int rec; //probabilidade de reprodução introduzida pelo utilizador
    private static int pcorte1 = 0;
    private static int pcorte2 = 0;
    private static int pcorte3 = 0;
    private static String individuo1; //cromossoma do individuo 1 da mutação
    private static String individuo2; //cromossoma do individuo 2 da mutação
    private static int gene1 = 0; //mutação
    private static int gene2 = 0; //mutação
    private static int ind1 = 0; //individuo 1 da mutação
    private static int ind2 = 0; //individuo 2 da mutação
    private static double soma = 0;
    private static double media = 0;
    private static double melhor1 = 0;
    private static double melhor2 = 0;
    private static int x = 8192; //2^13 ->numero máximo para a cadeira binário
    private static Integer[] ind; //guarda o valor decimal corresponde ao cromossoma de
cada ind.
    private static String[] cromossoma; //guarda o cromossoma de cada ind.
    private static double[] avaliacao; //guarada a função avaliação de cada ind.
    private static double[] qualidade; //guarda a qualidade de cada ind.
    private static double[] mediaQual; //guarda a melhor qualidade dos ind. de cada
iteração
    private static double[] melhorInd; //guarda o valor do melhor ind. de todas as
iterações
    private static double[] probab; //guarada o valor da probabilidade de cada ind.
    private static double[] segRoleta; //guarda o valor do segmento da roleta de cada
ind.
    private static int[] indElitismo; //guarda o num dos ind. escolhidos por elitismo
    private static double[] roleta; //guarda o valor gerado pela roleta
    private static int[] indRoleta; //guarda o num dos ind. escolhdos por roleta
    private static double[] recomAleatoria; //quarda os valores aleatórios das
recombinações de cada casal
    private static String[] cromossomaFilho; //guarda o cromossoma de cada filho
    private static int[][] pontosCorte;
    private static int cont;
```

```
    private static int[][] mutacao;//guarda os ind. e os genes onde vai existir mutação
    private static boolean vf; //para o ciclo while
    private static String[] spai; //guardar os valores do pai dos pontos de corte
    private static String[] smae; //guardar os valores da mae dos pontos de corte
    private static String pai; //guardar o pai
    private static String mae; //guarda a mae
    private static String filho1; //guarda o 1º filho
    private static String filho2; //guarda o 2º filho
    private static int comp; //comprimento do cromossoma
    private static StringBuilder sb;
    private static char g; //gene a comutar
    private static int iteracao = 1;
    private static int numInd = 0; // numero de individuos escolhido pelo utilizador
    private static int numPares = 0; //mumero de pares resultante do num de ind
escolhidos
    private static int numRoleta = 0; // numero de individuos a gerar por roleta
    private static int numElitismo = 0; //numero de individuos por elitismo
    private static int nPares = 0;//numero de pares que não vão reproduzir


    public static void main(String[] args) throws ParseException {
        Scanner in = new Scanner(System.in);
        //pedir ao utilizador o numero de iterações
        vf = true;
        while(vf == true){
            System.out.println("Introduza o numero de iterações que pretenda:");
            String it = in.nextLine();
            if(it.matches("[0-9]+")){
                vf = false;
                iteracao = Integer.parseInt(it);
            }else{
                System.out.println("numero invalido");
            }
        }
         //pedir ao utilizador o numero de individuos
        vf = true;
        while (vf == true) {
            System.out.println("Quantos Individuos pretende na População?");
            String it = in.nextLine();
            if (it.matches("[0-9]+")) {
                vf = false;
                numInd = Integer.parseInt(it);
            } else {
                System.out.println("numero invalido");
            }
        }


//        //pedir ao utilizador o numero de elementos escolhidos por elitismo
//        vf = true;
//        while (vf == true) {
//            System.out.println("Quantos Individuos quer selecionar por Elitismo?");
//            String it = in.nextLine();
```

```
//          if (it.matches("[0-9]+")) {
//             vf = false;
//             numElitismo = Integer.parseInt(it);
//          } else {
//             System.out.println("numero invalido");
//          }
//       }

      //calcular quantos individuos vão ser gerados pela roleta e quantos pares vão
formar
      numRoleta = numInd - 2;

      double par = (double) numRoleta / 2 + 0.5;

      if(numRoleta % 2 == 0){
         numPares = (int) par;
      }else{
         numPares = (int) par +1;
      }

       //pedir ao utilizador a Probabilidade de reprodução
      vf=true;
      while(vf == true){
         System.out.println("Introduza Probabilidade de Reprodução[50-80]:");
         String it = in.nextLine();
         if(it.matches("[0-9]+")){

            rec = Integer.parseInt(it);
            if((rec >= 50)&&(rec <= 80)){
               vf = false;
            }else{
               System.out.println("O numero não se encontra dentro do limite
pretendido");
            }
         }else{
            System.out.println("numero invalido");
         }
      }

      recombinacao = rec /100.0;


//*************************************************************************
**********************************************
      mediaQual = new double[100];
      melhorInd = new double[100];



      ind = new Integer[numInd]; //criar um vector com n possiçoes
      cromossoma = new String[numInd];
```

```
        //gerar os n indivduos aleatorios decimais e convertelos em binário e colocalos nos
vectores correspondentes
        for (int i = 0; i < numInd; i++) {
            ind[i] = (int) (1 + Math.random() * x);
            cromossoma[i] = Integer.toString(ind[i], 2);
        }


//        //mostrar valores do cromossoma em decimal
//        for (int i =0; i <10;i++){
//            System.out.println(ind[i]);
//
//        }

        //acrescentar 0 a esquerda
        for (int i = 0; i < numInd; i++) {
            comp = cromossoma[i].length();

            for (int a = 0; a < (13 - comp); a++) {
                cromossoma[i] = "0" + cromossoma[i];
            }

        }
        for(int j=0;j<iteracao;j++){
        System.out.println("\n\n*******ITERAÇÃO "+(j+1)+"*******");
        //Escrever tabela com os cromossodas de cada individuo
        System.out.println("Individuo \t Cromossoma");
        for (int i = 0; i < numInd; i++) {
            System.out.println("     " + (i + 1) + "  \t" + cromossoma[i]);

        }




//
//Função de avaliação

        avaliacao = new double[numInd];
        for (int i = 0; i < numInd; i++) {
//            DecimalFormat df = (DecimalFormat) DecimalFormat.getInstance();
            double aval = (1 + ((ind[i] * 5) / ((Math.pow(2, 13)) - 1)));
//            //formatar o numero com 3 casas decimais numa String
//            String av = String.format("%.3f", aval);
//            //Converter a String para Double e guardar
//            avaliacao[i] = (Double) df.parse(av);

            aval = ((int)(aval*1000)/1000.0);
            avaliacao[i]=aval;
        }


//qualidade
```

```
        qualidade = new double[numInd];
        for (int i = 0; i < numInd; i++) {
//          DecimalFormat df = (DecimalFormat) DecimalFormat.getInstance();
            double qualid = (Double) (Math.pow(avaliacao[i] - 3, 2));
//            //formatar o numero com 3 casas decimais numa String
//            String av = String.format("%.3f", qualid);
//            //Converter a String para Double e guardar
//            qualidade[i] = (Double) df.parse(av);
            qualid = ((int)(qualid*1000)/1000.0);
            qualidade[i]=qualid;


        }




        //probabilidade
          soma =0;

        for (int i = 0; i < numInd; i++) {
            soma += qualidade[i];
        }

        media = soma / numInd;
        media = ((int)(media*1000)/1000.0);
        mediaQual[j]=media;

        probab = new double[numInd];
        for (int i = 0; i < numInd; i++) {
//          DecimalFormat df = (DecimalFormat) DecimalFormat.getInstance();
            double prob = qualidade[i] / soma;
//            //formatar o numero com 3 casas decimais numa String
//            String av = String.format("%.3f", prob);
//            //Converter a String para Double e guardar
//            probab[i] = (Double) df.parse(av);

            prob =((int) (prob*1000)/1000.0);
            probab[i] = prob;
        }


//Segmento roleta

        segRoleta = new double[numInd];
        segRoleta[0] = probab[0];
        for (int i = 1; i < numInd; i++) {
//          DecimalFormat df = (DecimalFormat) DecimalFormat.getInstance();
            double rol = segRoleta[i - 1] + probab[i];
//            //formatar o numero com 3 casas decimais numa String
//            String av = String.format("%.3f", rol);
//            //Converter a String para Double e guardar
//            segRoleta[i] = (Double) df.parse(av);
```

```
            rol = ((int)(rol*1000)/1000.0);
            segRoleta[i]=rol;
        }


        //Escrever tabela com os individuos,avaliação,Qualidade,Probabilidade e Segmento
Roleta
        System.out.println("\n\n");
        System.out.println("Individuo  Avaliação  Qualidade  Probabilidade  Segmento
Roleta");
        for (int i = 0; i < numInd; i++) {
            System.out.println("    " + (i + 1) + "\t\t" + avaliacao[i] + "  \t" + qualidade[i]
+ "\t\t" + probab[i] + "\t\t" + segRoleta[i]);

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


//selecção
        //escolher os 2 melhores ind. para elitismo
        indElitismo = new int[2];
        for (int i = 0; i < numInd; i++) {

            if (qualidade[i] > melhor1) {
                melhor1 = qualidade[i];
                indElitismo[0] = i + 1;
            }
        }
        for (int i = 0; i < numInd; i++) {
            if ((qualidade[i] > melhor2) && (qualidade[i] < melhor1)) {
                melhor2 = qualidade[i];
                indElitismo[1] = i + 1;
            }
        }
        melhorInd[j]=melhor1;

        //escolher os restantes 8 ind. por roleta

        indRoleta = new int[numRoleta];
        roleta = new double[numRoleta];
        for (int i = 0; i < numRoleta; i++) {
//            DecimalFormat df = (DecimalFormat) DecimalFormat.getInstance();
            double rol = (double) (Math.random());
//            //formatar o numero com 3 casas decimais numa String
//            String av = String.format("%.3f", rol);
//            //Converter a String para Double e guardar
//            roleta[i] = (Double) df.parse(av);
            rol = ((int)(rol*1000)/1000.0);
            roleta[i]=rol;


        }
        //procular o individuo que corresponde ao valor gerado pela roleta
```

```
        for (int i = 0; i < numRoleta; i++) {
            for (int a = 1; a < numInd; a++) {
                if ((roleta[i] < segRoleta[a]) && (roleta[i] > segRoleta[a - 1])) {
                    indRoleta[i] = a + 1;
                }
            }
        }
        for (int i = 0; i < numRoleta; i++) {
            if (indRoleta[i] == 0) {
                indRoleta[i] = 1;
            }
        }
        //tabela de Seleção
        System.out.println("Elitismo: " + indElitismo[0] + " e " + indElitismo[1]);
        System.out.println("Roleta: Valor   Individuo");
        for (int i = 0; i < numRoleta; i++) {
            System.out.println("\t" + roleta[i] + "\t" + indRoleta[i]);
        }



//Recombinação

        recomAleatoria = new double[numPares];

        double pr; //pr=pares da recombinação a não reproduzir

        pr = (double)((rec * numPares) / 100.0);
        nPares = (int) pr;

        //garantir que pelo menos um casal não reproduz
            vf = true;
            while (vf == true) {
                for (int i = 0; i < numPares; i++) {
//                  DecimalFormat df = (DecimalFormat) DecimalFormat.getInstance();
                     double val = (Math.random());
//                  //formatar o numero com 3 casas decimais numa String
//                  String av = String.format("%.2f", val);
//                  //Converter a String para Double e guardar
//                  recomAleatoria[i] = (Double) df.parse(av);

                     val = ((int) (val * 1000) / 1000.0);
                     recomAleatoria[i] = val;

                }
                cont = 0;

                    for (int i = 0; i < numPares; i++) {
                        if (recomAleatoria[i] >= recombinacao) {
                            cont++;
                        }
                    }
                    if (cont == nPares) {
```

```
            vf = false;
        }

    }

//pontos de corte
pontosCorte = new int[numPares][3];//cada coluna corresponde a cada casal
for (int i = 0; i < numPares; i++) {
    //garantir que não existem 2 pontos de corte iguais
    vf = true;
    while (vf == true) {

        for (int y = 0; y < 3; y++) {
            pontosCorte[i][y] = (int) (1 + Math.random() * 12);

        }
        cont = 0;

        if (pontosCorte[i][2] == pontosCorte[i][1]) {
            cont++;
        } else if (pontosCorte[i][2] == pontosCorte[i][0]) {
            cont++;
        } else if (pontosCorte[i][1] == pontosCorte[i][0]) {
            cont++;
        }


        if (cont == 0) {
            vf = false;
        }
    }
    if (recomAleatoria[i] >= recombinacao) {
        for (int y = 0; y < 3; y++) {
            pontosCorte[i][y] = 0;
            vf = false;
        }
    }
}



//ordenar pontos de corte
for (int i = 0; i < numPares; i++) {
    for (int y = 0; y < 3; y++) {
        for (int a = y + 1; a < 3; a++) {

            if (pontosCorte[i][y] > pontosCorte[i][a]) {
                int aux = pontosCorte[i][y];
                pontosCorte[i][y] = pontosCorte[i][a];
                pontosCorte[i][a] = aux;

            }
        }
```

```
            }
        }
        //escrever tabela da recombinação c/ respetivos pontos de corte
        System.out.println("Pares   Valor Aleatório[0;1]   Ponto de Corte   Filhos");
        for (int i = 0; i < 2; i++) {
            System.out.println(" " + indElitismo[i] + "           " + "-" + "                " +
"-" + "            " + "i'" + (i+1));
        }
        for(int i=0; i<numPares;i++){

            System.out.println(" "+indRoleta[(i*2)]+" e "+indRoleta[(i*2+1)]+"
"+recomAleatoria[i]+"              "+pontosCorte[i][0]+","+pontosCorte[i][1]+" e
"+pontosCorte[i][2]+"         i'"+(i+3)+" e "+(i+4));
        }
            System.out.println("\n\n");
            System.out.println("-------Reprodução--------\n");
//          //---tabela de recombinação----
//          for(int i=0;i<4;i++){
//              System.out.println("");
//              for(int y=0;y<3;y++){
//                  System.out.println(pontosCorte[i][y]);
//              }
//          }
//
//
//          System.out.println("---");
//          for(int i=0;i<4;i++){
//              for(int y=0;y<3;y++){
//                  System.out.println(pontosCorte[i][y]);
//              }
//              System.out.println("--");
//          }
//          //--fim----



//operador de recombinação
        //filhos por elitismo
        cromossomaFilho = new String[numInd];
        for (int i = 0; i < 2; i++) {
            for (int y = 0; y < numInd; y++) {
                if (indElitismo[i] == y) {
                    cromossomaFilho[i] = cromossoma[y];
                }
            }
        }

        //filhos por roleta
        //se numero de individuos por rolete for par
            if (numRoleta % 2 == 0) {
                for (int i = 0; i < numPares; i++) {

                    int p = indRoleta[(i * 2)];
```

```
        int m = indRoleta[(i * 2 + 1)];

        pai = cromossoma[p - 1];
        mae = cromossoma[m - 1];

        pcorte1 = pontosCorte[i][0];
        pcorte2 = pontosCorte[i][1];
        pcorte3 = pontosCorte[i][2];
        //adicionar um espaço nos pontos de corte
        //pai
        sb = new StringBuilder(pai);
        sb.insert(pai.length() - (13 - pcorte1), ' ');
        sb.insert(pai.length() - (12 - pcorte2), ' ');
        sb.insert(pai.length() - (11 - pcorte3), ' ');
        pai = sb.toString();
        System.out.println(pai);
        //mae
        sb = new StringBuilder(mae);
        sb.insert(mae.length() - (13 - pcorte1), ' ');
        sb.insert(mae.length() - (12 - pcorte2), ' ');
        sb.insert(mae.length() - (11 - pcorte3), ' ');
        mae = sb.toString();
        System.out.println(mae);
        //separar a string em varias pelos pontos de corte
        spai = new String[4];
        smae = new String[4];

        spai = pai.split(" ");
        smae = mae.split(" ");

        filho1 = spai[0] + smae[1] + spai[2] + smae[3];
        filho2 = smae[0] + spai[1] + smae[2] + spai[3];
        cromossomaFilho[i * 2 + 2] = filho1;
        cromossomaFilho[i * 2 + 3] = filho2;
        System.out.println(filho1);
        System.out.println(filho2);
        System.out.println("---");
    }
}else{ // se for impar
    for (int i = 1; i < numPares; i++) {

        int p = indRoleta[(i * 2 - 1)];
        int m = indRoleta[(i * 2)];
        System.out.println("p:"+p+"|m:"+m);
        pai = cromossoma[p - 1];
        mae = cromossoma[m - 1];

        pcorte1 = pontosCorte[i][0];
        pcorte2 = pontosCorte[i][1];
        pcorte3 = pontosCorte[i][2];
        //adicionar um espaço nos pontos de corte
        //pai
        sb = new StringBuilder(pai);
```

```
                sb.insert(pai.length() - (13 - pcorte1), ' ');
                sb.insert(pai.length() - (12 - pcorte2), ' ');
                sb.insert(pai.length() - (11 - pcorte3), ' ');
                pai = sb.toString();
                System.out.println(pai);
                //mae
                sb = new StringBuilder(mae);
                sb.insert(mae.length() - (13 - pcorte1), ' ');
                sb.insert(mae.length() - (12 - pcorte2), ' ');
                sb.insert(mae.length() - (11 - pcorte3), ' ');
                mae = sb.toString();
                System.out.println(mae);
                //separar a string em varias pelos pontos de corte
                spai = new String[4];
                smae = new String[4];

                spai = pai.split(" ");
                smae = mae.split(" ");
                cromossomaFilho[3] = cromossoma[indRoleta[0]];
                filho1 = spai[0] + smae[1] + spai[2] + smae[3];
                filho2 = smae[0] + spai[1] + smae[2] + spai[3];
                cromossomaFilho[i * 2 + 3] = filho1;
                cromossomaFilho[i * 2 + 4] = filho2;
                System.out.println(filho1);
                System.out.println(filho2);
                System.out.println("---");
            }
        }

        //tabela da recombinação
        System.out.println("\n-----Recombinação-----");
        for (int i = 0; i < numInd; i++) {
            System.out.println(cromossomaFilho[i]);
        }

//mutação

        mutacao = new int[2][2];//1ª linha para os individuos e 2ªlinha para os genes
        //gerar individuos
        for (int i = 0; i < 2; i++) {
            mutacao[0][i] = (int) (1 + Math.random() * numInd);
        }
        //gerar genes
        for (int i = 0; i < 2; i++) {
            mutacao[1][i] = (int) (1 + Math.random() * 13);
        }


//aplicar mutação no gene escolhido

        ind1 = mutacao[0][0];
        ind2 = mutacao[0][1];
        gene1 = mutacao[1][0];
```

```
        gene2 = mutacao[1][1];

        for (int i = 0; i < numInd; i++) {
            if(i==(ind1-1)){
                individuo1=cromossomaFilho[i];
            }if (i==(ind2-1)){
                individuo2=cromossomaFilho[i];
            }
        }

        //1º gene
        sb = new StringBuilder(individuo1);
        g = sb.charAt(gene1-1);
        if(g=='1'){
            g = '0';
        }else{
            g = '1';
        }
        sb.deleteCharAt(gene1-1);
        sb.insert(gene1-1, g);
        individuo1 = sb.toString();
        cromossomaFilho[ind1-1] = individuo1;

        //2º gene
        sb = new StringBuilder(individuo2);
        g = sb.charAt(gene2-1);
        if(g=='1'){
            g = '0';
        }else{
            g = '1';
        }
        sb.deleteCharAt(gene2-1);
        sb.insert(gene2-1, g);
        individuo2 = sb.toString();
        cromossomaFilho[ind2-1] = individuo2;

        //tabela com a mutação
        System.out.println("\n-----Mutação------");
            System.out.println("\n Genes a alterar:");
            System.out.println("-> Individuo "+ind1+" no gene "+gene1);
            System.out.println("-> Individuo "+ind2+" no gene "+gene2+"\n");
        for(int i=0;i<numInd;i++){
            System.out.println(cromossomaFilho[i]);
        }


////restaurar vetores do ind. e do cromossoma
//        ind = new Integer[10]; //criar um vector com 10 possiçoes
//        cromossoma = new String[10];
//
```

```
        //passar os cromossomas do vetor dos filhos para o vetor cromossoma
        for(int i=0;i<numInd;i++){
            cromossoma[i] = cromossomaFilho[i];
        }
        //converter os cromossomas em decimal e guardar o valor correspontente no
vetor ind
        for(int i=0;i<numInd;i++){
            ind[i]=Integer.parseInt(cromossoma[i], 2);
        }
       } //fim da iteração

       //escrever tabela de avaliação

System.out.println("*****************************************************");
        System.out.println("********Avaliação*******");
        System.out.println("Geração    Melhor Qualidade    Media");
        for (int j=0;j<iteracao;j++){
            System.out.println("    "+(j+1)+"\t    "+melhorInd[j]+"\t\t "+mediaQual[j]);
        }


    }

}
```