```java
// sequencement
public class RecuitSeq {
    /*******************************************************/
    /* Parametres du recuit */
    private static final int nbTransitions = 2000;
    private static final double alpha = 0.995;
    private static final boolean minimisation = true;
    /*******************************************************/
    /* dimension du probleme */
    private static final int DIMENSION = 100;

     // principe d'acceptation
    private boolean accept(double yi, double yj, double T, boolean min) {
        boolean isAccepted = false;
        double dE = yj-yi;
        double proba = Math.exp(-Math.abs(dE)/T);
        double tirage = Math.random();

        if(min) {
            if(dE<0) isAccepted = true;
            else if (tirage <= proba) isAccepted=true;//acceptation frequente si T grand car alors proba proche de 1
        } else {// opt en maximisation
            if(dE>0) isAccepted = true;
            else if (tirage <= proba) isAccepted=true;
        }
        return isAccepted;
    }
    // ************************************
    // Heat Up
    // ************************************
    public double heatUpLoop() { // HeatUp heat = new HeatUp();
        int acceptCount = 0;
        double yi = 0, yj;
        double T = 0.01, tauxAccept = 0.0;
        EtatSeq xi = new EtatSeq(DIMENSION);

        do {
            acceptCount = 0;
            for (int i = 0; i < nbTransitions; i++) {

                // generation d'un point de l'espace d'etat
                xi.initAleatEtat();
                yi = xi.calculCritere();

                // generation d'un voisin
                xi.genererVoisin();
                yj = xi.calculCritere();

                if (accept(yi, yj, T, minimisation))
                    acceptCount++;
                tauxAccept = (double) acceptCount / (double) nbTransitions;
            }
            T = T * 1.1;
            System.out.println("T= " + T + " tauxAccept= " + tauxAccept + " currentCost= " + yi);
        } while (tauxAccept < 0.8);
        return T;
    }
    // ****************************************
    // COOLING
    // ****************************************
```

```
61      public EtatSeq coolingLoop(double Tinit) { // HeatUp heat = new HeatUp();
62          double yi = 0.0, yj = 0.0;
63          double T = Tinit;
64          EtatSeq xi = new EtatSeq(DIMENSION);
65
66          xi.initAleatEtat();
67          yi = xi.calculCritere();
68          do {
69              for (int i = 0; i < nbTransitions; i++) {
70                  xi.genererVoisin();
71                  yj = xi.calculCritere();
72                  if (accept(yi, yj, T, minimisation)) {
73                      yi = yj;
74                  } else {
75                      xi.comeBack();
76                  }
77              }
78              T = T * alpha;
79              System.out.println("T= " + T + " valeur critere " + yi);
80          } while (T > 0.0001 * Tinit);
81          xi.afficherEtat();
82          //Data.afficherAvions();
83          return xi;
84      }
85      // ****************************************
86      // MAIN
87      // ****************************************
88      public static void main(String args[]) {
89          double temperature;
90          RecuitSeq monRecuit = new RecuitSeq();
91          // generation des donnees
92          System.out.println("**********************Generation des donnee
    ***************");
93          Data.genererAvions(DIMENSION);
94
95          System.out.println("**************************Chauffage ***************");
96          temperature = monRecuit.heatUpLoop();
97          System.out.println("======================Refroidissement ============");
98          monRecuit.coolingLoop(temperature);
99      }// end main
100 }// End class HeatUp2
101
```