```
import java.util.*;

public class TableDeCorrespondanceVect<TCle extends Comparable<TCle>, TVal>
  implements TableDeCorrespondance<TCle, TVal> {

  private static final int TAILLE_MAX = 50;
  private ArrayList<Couple<TCle, TVal>> table;
  private int position;
  private int capacite;

  public TableDeCorrespondanceVect(int taille) {
    table = new ArrayList<Couple<TCle, TVal>> (taille);
    position = 0;
    capacite = taille;
  }

  private int nbElements() {
    return table.size();
  }

  private int capacite() {
    return capacite;
  }

  public void debut() {
    position = 0;
  }

  public boolean enFin() {
    return position == nbElements();
  }

  private boolean estALAvantDernier() {
    return position == nbElements() − 1;
  }

  public boolean auDebut() {
    return position == 0;
  }

  public TCle cleCourante() {
    if (enFin())
      throw new RuntimeException("Fin de la table");
   return (table.get(position)).getCle();
  }

  public TVal valeurCourante(){
    if (enFin())
      throw new RuntimeException("Fin de la table");
    return (table.get(position)).getVal();
  }

  public void avancer() {
    if (enFin())
      throw new RuntimeException("Fin de la table");
    position++;
  }

  public void reculer() {
    if (auDebut())
      throw new RuntimeException("Debut de la table");
    position−−;
```



```
  }

  public boolean present(TCle cle) {
    int a;

    debut();
    while(!enFin()) {
      a = cleCourante().compareTo(cle);
      if (a == 0)
        return true;
      if (a > 0)
        return false;
      avancer();
    }
    return false;
  }

  public TVal valeur(TCle cle) {
    if(present(cle))
      return valeurCourante();
    else
      throw new RuntimeException("Cle absente");
  }

  public void ajouter(TCle cle, TVal val) {

    if(nbElements() == capacite())
      return;

    debut();

    while(!enFin() && cleCourante().compareTo(cle) < 0)
      avancer();
    if(!enFin() && cleCourante().compareTo(cle) == 0)
      return;

    table.add(position, new Couple<TCle, TVal>(cle, val));
  }

  public void modifier(TCle cle, TVal val) {
    if(present(cle))
      (table.get(position)).setVal(val);
  }

  public void retirer(TCle cle) {
    if(present(cle))
      table.remove(position);
  }
}
```