```java
package quarto;

public class Plateau {

    public static final int NOMBRE_COLONNES = 4;
    public static final int NOMBRE_LIGNES = 4;
    private Case[][] cases;

    public Plateau() {
        this.cases = new Case[NOMBRE_LIGNES][NOMBRE_COLONNES];
        for (int i = 0; i < NOMBRE_LIGNES; i++) {
            for (int j = 0; j < NOMBRE_COLONNES; j++) {
                this.cases[i][j] = new Case();
            }
        }
    }

    public Case[][] getCases() {
        return this.cases;
    }

    public int nombreDeLignes() {
        return NOMBRE_LIGNES;
    }

    public int nombreDeColonnes() {
        return NOMBRE_COLONNES;
    }

    public boolean estVide() {
        for (int i = 0; i < NOMBRE_LIGNES; i++) {
            for (int j = 0; j < NOMBRE_COLONNES; j++) {
                if (!this.cases[i][j].estVide()) {
                    return false;
                }
            }
        }
        return true;
    }

    public boolean ontEnCommunUneCaractériqtique(Pièce... pièces) {
        int somme[] = {0, 0, 0, 0};
        for (int i = 0; i < pièces.length; i++) {
```

```
44              if (pièces[i].estClaire()) somme[0] += 1;
45              if (pièces[i].estHaute()) somme[1] += 1;
46              if (pièces[i].estRonde()) somme[2] += 1;
47              if (pièces[i].estPleine()) somme[3] += 1;
48          }
49          return (somme[0] == 0 || somme[0] == 4 ||
50                  somme[1] == 0 || somme[1] == 4 ||
51                  somme[2] == 0 || somme[2] == 4 ||
52                  somme[3] == 0 || somme[3] == 4);
53      }
54
55      public boolean ligneComplèteEtVictorieuse(int ligne) {
56          for (int colonne = 0; colonne < NOMBRE_COLONNES; colonne++) {
57              if (this.getCases()[ligne][colonne].estVide())
58                  return false;
59          }
60          return ontEnCommunUneCaractériqtique(this.getCases()[ligne][0].getPièce(),
61                  this.getCases()[ligne][1].getPièce(),
62                  this.getCases()[ligne][2].getPièce(),
63                  this.getCases()[ligne][3].getPièce());
64      }
65
66      public boolean colonneComplèteEtVictorieuse(int colonne) {
67          for (int ligne = 0; ligne < NOMBRE_LIGNES; ligne++) {
68              if (this.getCases()[ligne][colonne].estVide())
69                  return false;
70          }
71          return ontEnCommunUneCaractériqtique(this.getCases()[0][colonne].getPièce(),
72                  this.getCases()[1][colonne].getPièce(),
73                  this.getCases()[2][colonne].getPièce(),
74                  this.getCases()[3][colonne].getPièce());
75      }
76
77      public boolean premièreDiagonaleComplèteEtVictorieuse() {
78          for (int ligne = 0; ligne < NOMBRE_LIGNES; ligne++) {
79              if (this.getCases()[ligne][ligne].estVide())
80                  return false;
81          }
82          return ontEnCommunUneCaractériqtique(
83                  this.getCases()[0][0].getPièce(),
84                  this.getCases()[1][1].getPièce(),
85                  this.getCases()[2][2].getPièce(),
86                  this.getCases()[3][3].getPièce());
```

```
87      }
88
89      public boolean deuxièmeDiagonaleComplèteEtVictorieuse() {
90          for (int ligne = 0; ligne < NOMBRE_LIGNES; ligne++) {
91              if (this.getCases()[ligne][3 - ligne].estVide())
92                  return false;
93          }
94          return ontEnCommunUneCaractériqtique(
95                  this.getCases()[0][3].getPièce(),
96                  this.getCases()[1][2].getPièce(),
97                  this.getCases()[2][1].getPièce(),
98                  this.getCases()[3][0].getPièce());
99      }
100
101 }
102
103 public class Quarto {
104
105     private Plateau plateau;
106     private Set<Pièce> piècesRestantesAJouer;
107
108     public Quarto() {
109         this.plateau = new Plateau();
110         this.piècesRestantesAJouer = new HashSet<>();
111         this.piècesRestantesAJouer.add(new Pièce(false, false, false, false));
112         this.piècesRestantesAJouer.add(new Pièce(false, false, false, true));
113         this.piècesRestantesAJouer.add(new Pièce(false, false, true, false));
114         this.piècesRestantesAJouer.add(new Pièce(false, false, true, true));
115         this.piècesRestantesAJouer.add(new Pièce(false, true, false, false));
116         this.piècesRestantesAJouer.add(new Pièce(false, true, false, true));
117         this.piècesRestantesAJouer.add(new Pièce(false, true, true, false));
118         this.piècesRestantesAJouer.add(new Pièce(false, true, true, true));
119         this.piècesRestantesAJouer.add(new Pièce(true, false, false, false));
120         this.piècesRestantesAJouer.add(new Pièce(true, false, false, true));
121         this.piècesRestantesAJouer.add(new Pièce(true, false, true, false));
122         this.piècesRestantesAJouer.add(new Pièce(true, false, true, true));
123         this.piècesRestantesAJouer.add(new Pièce(true, true, false, false));
124         this.piècesRestantesAJouer.add(new Pièce(true, true, false, true));
125         this.piècesRestantesAJouer.add(new Pièce(true, true, true, false));
126         this.piècesRestantesAJouer.add(new Pièce(true, true, true, true));
127     }
128
129
```

```
130
131     public boolean estVide() {
132         return this.plateau.estVide();
133     }
134
135     public int nombreDePiècesRestantAJouer() {
136         return this.piècesRestantesAJouer.size();
137     }
138
139     public boolean choisirUnePieceNonJouée(Pièce pièce) {
140         if (this.piècesRestantesAJouer.contains(pièce)) {
141             this.piècesRestantesAJouer.remove(pièce);
142             return true;
143         }
144         return false;
145     }
146
147     public boolean emplacementLibre(int ligne, int colonne) {
148         return this.plateau.getCases()[ligne][colonne].estVide();
149     }
150
151     public void poserPièce(Pièce pièce, int ligne, int colonne) {
152         if (!this.emplacementLibre(ligne, colonne)) {
153             throw new RuntimeException("Emplacement non Libre !");
154         }
155         this.plateau.getCases()[ligne][colonne].poserPièce(pièce);
156
157     }
158
159     public boolean coupVictorieux(int ligne, int colonne) {
160         return this.plateau.ligneComplèteEtVictorieuse(ligne) ||
161                 this.plateau.colonneComplèteEtVictorieuse(colonne) ||
162                 (ligne == colonne && this.plateau.premièreDiagonaleComplèteEtVictorieuse()) ||
163                 (ligne + colonne == 3 && this.plateau.deuxièmeDiagonaleComplèteEtVictorieuse());
164     }
165
166 }
167
```