```
16
17 public class Case extends Fragment {
18
19     /*
20     candidats = new boolean[9][9][9];
21
22     Matrice "public" déclarée et mise-à-jour ailleurs.
23     Cette matrice contient l'état des candidats de chacune des 81 cases de la grille.
24     */
25
26     public static Case[] case_de = new Case[81];
27     public int numéro, ligne, colonne, bloc;
28
29     public Case() {
30
31     }
32
33     public Case(int _numéro) {
34
35         numéro = _numéro;                    // numéro = {0,...,80}.
36         ligne = _numéro / 9;                 // ligne = {0,...,8}.
37         colonne = _numéro % 9;               // colonne = {0,...,8}.
38         bloc = colonne/3 + 3 * (ligne/3);    // bloc = {0,...,8}.
39
40     }
41
42     public static Case la_case(int numéro) {
43         return case_de[numéro];
44     }
45
46     public boolean contient_le(int candidat) {  // candidat = {1,...,9}.
47         return candidats[this.ligne][this.colonne][candidat - 1];
48     }
49
50     public static final ArrayList<Integer> liste_des_cachés = new ArrayList<>();
51     public static final ArrayList<Integer> liste_des_isolés = new ArrayList<>();
52
53     /*
54     liste_des_cases_vides = ArrayList<Case>;
55
56     Liste "public" déclarée et mise-à-jour ailleurs.
57     */
58
59     public static void Recherche_des_Célibataires() {
60
61         for (int candidat : de_1_à_9) {
62             for (Case kase : liste_des_cases_vides) {
63                 if (kase.contient_le(candidat)) {
64
65                     if (Célibataire_isolé_trouvé_dans_la_case(kase, candidat)) {
66
67                         /* Pour l'affichage sur la grille des célibataires isolés. */
68                         liste_des_isolés.add(candidat);
69                         liste_des_isolés.add(kase.numéro);
70
71                         continue;
72                     }
73
74                     if (Célibataire_caché_trouvé_dans_la_case(kase, candidat)) {
75
76                         /* Pour l'affichage sur la grille des célibataires cachés. */
77                         liste_des_cachés.add(candidat);
78                         liste_des_cachés.add(kase.numéro);
79
80                     }
81                 }
82             }
83         }
84     }
85
```

```java

    static boolean Célibataire_isolé_trouvé_dans_la_case(Case kase, int candidat0) {

        ArrayList<Integer> liste = new ArrayList<>();

        for (int candidat : de_1_à_9) {
            if (kase.contient_le(candidat)) liste.add(candidat);
        }
        return liste.size() == 1 && liste.get(0) == candidat0;
    }

    static boolean Célibataire_caché_trouvé_dans_la_case(Case kase, int candidat) {

        /* Si le candidat est seul dans le bloc */
        if (Compte_du_candidat_dans_le_bloc(kase.bloc, candidat) == 1) return true;

        /* Si le candidat est seul dans la colonne */
        if (Compte_du_candidat_dans_la_colonne(kase.colonne, candidat) == 1) return true;

        /* Si le candidat est seul dans la ligne */
        return Compte_du_candidat_dans_la_ligne(kase.ligne, candidat) == 1;
    }

    static int Compte_du_candidat_dans_le_bloc(int bloc, int candidat) {

        int compteur = 0;

        for (int case_de_bloc : de_1_à_9) {
            Case case_de_grille = la_case(3 * (bloc % 3) + (case_de_bloc - 1) % 3
                    + 9 * (3 * (bloc / 3) + (case_de_bloc - 1) / 3));
            if (case_de_grille.contient_le(candidat)) compteur++;
        }
        return compteur;
    }


    static int Compte_du_candidat_dans_la_colonne(int colonne, int candidat) {

        int compteur = 0;

        for (int ligne : de_1_à_9) {
            if (la_case(colonne + 9 * (ligne - 1)).contient_le(candidat)) compteur++;
        }
        return compteur;
    }


    static int Compte_du_candidat_dans_la_ligne(int ligne, int candidat) {

        int compteur = 0;

        for (int colonne : de_1_à_9) {
            if (la_case(colonne - 1 + 9 * ligne).contient_le(candidat)) compteur++;
        }
        return compteur;
    }

/* ----------------------------------------------------------------------------*/

```