```
1 #include "event.h"
 2
 3
    /* CONSTRUCTEURS */
 4
 5
     event::event(){
 6
 7
      label="";
      occured=false;
 8
 9
      nb_occurence=0;
10
       date.clear();
11
12
13
     // Constructeur évolué, créé l'event à partir des données
    event::event(string nom){
14
15
      label=nom;
16
      occured=false;
17
      nb_occurence=0;
18
       date.clear();
19
20
21
    /* OPERATEURS */
22
   event event::operator = (const event &e) {
23
24
      label=e.label;
25
      occured=e.occured;
26
      nb_occurence=e.nb_occurence;
27
      date=date;
28
      return *this;
29
30
     /* ASSESSEURS */
31
32
     // Assesseur, ressort le nom de l'event
33
     string event::get_label(){
34
      return label;
35
36
37
     // Assesseur, ressort l'état de l'event (occured ou non)
38
39
     bool event::get_occured(){
40
      return occured;
41
42
43
      // Assesseur, ressort le nombre d'occurence
44
     int event::get_nb_occurence(){
45
      return nb_occurence;
46
47
     // Assesseur, ressort le tableau des heures d'occurence
48
49
     std::vector<int> event::get_h_occur(){
50
      return date;
51
52
53
     // Assesseur, ressort l'heure de la dernière occurence
54
     int event::get_last_h_occur(){
      return date.back();
55
56
57
     // Mutateur, change le label de l'élément
58
59
     void event::set_label(string name){
60
      label = name;
61
      }
62
63
     \ensuremath{//} Mutateur, change la valeur du nombre de validation de la chronique
64
     void event::set_occurence(int nb){
65
      nb_occurence = nb;
66
```

```
67
 68
      // Mutateur, change la valeur du booleen occured
     void event::set_occured(bool occur){
 69
 70
        occured = occur;
 71
 72
 73
      \ensuremath{//} Mutateur, ajoute une valeur dans les heures d'apparition de l'event
     void event::set_h_event(int heure){
 74
        date.push_back(heure);
 75
 76
 77
        // Reinitialisateur, remet à 0 le tableau des heures de validation
 78
 79
     void event::reset_h_event(){
 80
       date.clear();
81
     }
82
83
      // Reinitialisateur, réinitialise tous les attributs
84
    void event::reinit_event_all(){
 85
       label="";
 86
       occured=false;
 87
       nb_occurence=0;
 88
       date.clear();
 89
     }
 90
      /* METHODES */
 91
 92
 93
     // Affiche certains éléments dans la console
 94
    void event::afficher(){
 95
       cout<<"nom = "<<label<<endl;</pre>
96
       cout<<"event occured = "<<occured<<endl;</pre>
97
       if (occured) {
            cout<<"nombre d'occurence = "<<nb_occurence<<endl;</pre>
98
            cout<<"derniere validation = "<<date.back()<<endl;</pre>
99
100
       }
101
102
103
      // Affiche les heures d'occurence de l'event dans le terminal
      void event::afficher_heures_occur(){
104
105
       vector<int>::iterator it; // iterateur
        for (it = date.begin(); it != date.end(); it++)
106
107
108
            cout<<"occurence à "<<(*it)<<endl;</pre>
109
         }
110
111
112
       // Affiche le nombre de fois où l'event est arrivé
113
      void event::afficher_nb_occurence(){
114
       cout<<"nombre d'occurence = "<<nb_occurence<<endl;</pre>
115
```