Exercice 1 : **4 pts**

```
const int M=100;

void deleteOcurence(int tab[], int k){

   for(int i=0 ; i<M;i++)   0,5 (pt)
     cin>> tab[i] ; 0,5 pt
   for(int i = 0; i < M; i++)   0,5 pt
   {
      if (tab[i]==k) { 0,5 pt

         for(int j = i; j < M-1; j++) 1 pt
         {
            tab[j] = tab[j+1];   1 pt
         }
      }
   }
}
```

Exercice 2 : **3 pts**

```
typedef struct ev   0,25 pt
{
   int an,m,j,h;
}event;

const int N=100;
class Agenda     //déclaration  de la classe  0,25 pt
{
private:
   event events[N];  0,25 pt
   int index;
public:
   Agenda();
   int SearchEvent(event ev);
   bool equal(event e1, event e2);
   bool addEvent(event e);
   bool deleteEvent(event e);
};

Agenda::Agenda(){
   index = -1;
}

bool Agenda::equal(event e1, event e2){   0,5 pt

   if (e1.an==e2.an && e1.m==e2.m && e1.j==e2.j && e1.h==e2.h) {
      return true;
   }

   return false;
}
```

```
int Agenda::SearchEvent(event ev){  0,5 pt

    for(int i = 0 ; i <= index && index < N; i++)
    {
        if (equal(ev, events[i])) {
            return i;
        }
    }
    return -1;
}

bool Agenda::addEvent(event e){ 0,5 pt

    if (SearchEvent(e)>-1) {
        return false;
    }

    index++;

    if (index <N) {
        events[index] = e;
        return true;
    }
    return false;
}

bool Agenda::deleteEvent(event e){  0,75 pt

    int ix= SearchEvent(e);
    if (ix==-1) {
        return false;
     }

     for(int i = ix ; i <=index && index <N; i++)
     {
         events[i] = events[i+1];
     }
     index --;

    return true;
}
```