

```
1 #include <iostream>
2 #include <string>
3 #include <list>
4 #include <vector>
5
6 using namespace std;
7
8 template <typename T> using Par_INT_T = pair<int,T>;
9
10 template <typename T>
11 class vdisperso
12 {
13 private:
14     list<pair<int, T>> coefs;
15     int n;
16     T nulo;
17
18 public:
19     vdisperso(const vector<T> &v,T nulo=T());
20     void asignar_coeficiente(int i, const T &x);
21     vector<T> convertir() const;
22     void mostrarVectorDis() const;
23     void cambiarNulo(const T &nuevo_nulo);
24 };
25
26 template <typename T>
27 void vdisperso<T>::mostrarVectorDis() const{
28     for(typename list<pair<int,T>>::const_iterator
29     it=coefs.cbegin();it!=coefs.cend();it++)
30     {
31         cout << "POSICION:" << (*it).first << "->" << (*it).second << endl;
32     }
33 }
34
35 template <typename T>
36 vdisperso<T>::vdisperso(const vector<T> &v, T nulo)
37 {
38     int i;
39     this->nulo=nulo;
40     pair<int,T> aux;
41     this->n=v.size();
42     for(i=0;i<v.size();i++)
43     {
44         if(v[i]!=nulo)
45         {
46             aux.first=i;
47             aux.second=v[i];
48             this->coefs.push_back(aux);
49         }
50     }
51 }
52
53 template <typename T>
54 void vdisperso<T>::asignar_coeficiente(int i, const T &x)
55 {
56     typename list<pair<int, T>>::iterator it;
57     it=this->coefs.begin();
58     while(it!=this->coefs.end() && (*it).first!=i)
59     {
```

```
60         it++;
61     }
62     if(it!=this->coefs.end())
63     {
64         this->coefs.erase(it);
65     }
66     pair<int, T> aux;
67     aux.first=i;
68     aux.second=x;
69     this->coefs.push_back(aux);
70
71 }
72
73 template <typename T>
74 vector<T> vdisperso<T>::convertir() const
75 {
76     vector<T> ret;
77     typename list<pair<int,T>>::const_iterator it;
78     it=this->coefs.begin();
79     for(int i=0;i<this->n;i++)
80     {
81         if(i==(*it).first)
82         {
83             ret.push_back((*it).second);
84             it++;
85         }else
86         {
87             ret.push_back(nulo);
88         }
89     }
90     return ret;
91 }
92
93 template <typename T>
94 void vdisperso<T>::cambiarNulo(const T &nuevo_nulo)
95 {
96     this->nulo=nuevo_nulo;
97 }
98
99 template <typename T>
100 void mostrar_vector(const vector<T> & v)
101 {
102     for(typename vector<T>::const_iterator it=v.cbegin();it!=v.cend();it++)
103     {
104         cout << (*it) << " ";
105     }
106     cout << endl;
107 }
108
109 int main()
110 {
111     vector<int> aux(10,1);
112     aux[0]=104353;
113     aux[2]=3;
114     aux[4]=9;
115     aux[7]=99;
116     aux[9]=81;
117     vdisperso<int> prueba(aux,1);
118     prueba.mostrarVectorDis();
119     aux=prueba.convertir();
```

```
120     mostrar_vector(aux);
121
122     prueba.cambiarNulo(8);
123     prueba.mostrarVectorDis();
124     aux=prueba.convertir();
125     mostrar_vector(aux);
126 }
```