

```
1  #include <iostream>
2  #include <string.h>
3  #include <unordered_set>
4
5  using namespace std;
6
7
8  // This time repeated values in the bag ARE IMPORTANT.
9  // A multiset allows storing repeated values.
10 // Unordered multiset is faster accessing values than both
11 // multiset and vector.
12
13 // The code is almost unchanged compared to the previous assignment.
14
15 // By this time in the course I'm not supposed to know templates but I.
16 // used them anyway to work both with int and double numbers because the code
17 // is exactly the same in both cases.
18
19
20 template <typename T>
21 void funWithBags(unordered_multiset<T>);
22
23 int main() {
24     string select;
25     cout << "Choose int or double: ";
26     cin >> select;
27
28     // Compiler doesn't like to have the function call outside.
29     // Potentially this if-else may not execute anything and
30     // Bag wouldn't be declared before the function call.
31     if (select == "int") {
32         unordered_multiset<int> Bag;           // Assignment 4-2
33         funWithBags(Bag);
34     }
35     else if (select == "double") {
36         unordered_multiset<double> Bag;       // Assignment 4-3
37         funWithBags(Bag);
38     }
39     else
40         cout << "Error: Nothing valid selected";
41     return 0;
42 }
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
```

```
59 template <typename T>
60 void funWithBags(unordered_multiset<T> Bag) {
61     string command;
62     T x;
63
64     cin >> command;
65
66     while (command != "quit") {
67         if (command == "add") {
68             cin >> x;
69             Bag.insert(x);
70         }
71         else if (command == "del") {
72             cin >> x;
73             // The only difference compared to the previous
74             // assignment is here (read below) *:
75             if (!Bag.empty())
76                 Bag.erase(Bag.find(x));
77         }
78         else if (command == "qry") {
79             cin >> x;
80             if (Bag.empty())
81                 cout << "F";
82             else if (Bag.find(x) == Bag.end())
83                 cout << "F";
84             else
85                 cout << "T";
86         }
87         else {
88             cout << "error!" << endl;
89             return;
90         }
91         cin >> command;
92     }
93 }
94
95 // * Using the method 'erase' with a value like in the
96 // previous assignment for EVERY occurrences will be removed.
97 // I have to use the method 'find' to get the position of one
98 // element and pass the position to 'erase'.
99
100 // I can overlook the fact that there are iterators involved
101 // (at this point in the course they weren't introduced).
102
103 // Actually exactly the same code can be used for all point in the
104 // assignment but for the first point, it would be superfluous to
105 // give an iterator to the method erase.
106
107 // A vector now may be used more easily because of the repeated values
108 // but vectors are still slower.
109
110 // In the course, an automatic online testing system was used to hand in
111 // the assignments and this source code couldn't pass because the choice
112 // between the type of elements was not expected by the test...
```