

```

1 #include <iostream>
2 #include <string>
3 #include <vector>
4 #include <algorithm>
5 #include <sstream>
6 #include <map>
7
8 using namespace std;
9
10 // Possible solution:
11 // A map with a character key and vector of int as elements.
12
13 // NOTE: The datasets identifiers (e.g. a, b) can be ANY character (it can be
14 // programmed to limit the character range allowed, but I'm not doing it here
15 // to keep it short).
16
17 // To be completely robust, possible errors in the input stream should be
18 // checked because situations like the following will happen (because of
19 // std::cin):
20 // "a 3y4" will cause this program to read 'a' as a key, put 3 in the dataset
21 // 'a', then read 'y' as another key and, and add 4 to the 'y' dataset...
22
23 // Also note that a map is ordered with respect to the keys AUTOMATICALLY
24 // so dataset 'a' will be printed before 'b', and so on.
25
26
27 bool createSets(const string &, map< char, vector<int> > &);
28 void display(const vector<int> &);
29
30
31
32 int main() {
33     std::string input;
34     std::map<char, std::vector<int> > Dict;
35     std::getline(cin, input);
36
37     if (createSets(input, Dict)) {
38         // Scan through keys in the map
39         // - Order the elements in the vector values
40         // - Display
41         for (auto it = Dict.begin(); it != Dict.end(); it++) {
42             sort(it->second.begin(), it->second.end());
43             display(it->second);
44         }
45     }
46     else
47         std::cout << "Error" << std::endl;
48     return 0;
49 }
50
51
52
53
54 ////////////////////////////////////////
55
56
57
58

```

```
59
60 // No changes compared to assignment 5-1
61
62 void display(const std::vector<int> & V) {
63     for (auto n : V)
64         std::cout << n << " ";
65 }
66
67
68
69
70
71
72
73
74 // Assume the stream has the correct format so that the
75 // stringstream can be read alternatively to get keys and values
76
77 bool createSets(const string & input, map<char , vector<int>> & Dict) {
78     int n;
79     char c;
80
81     std::stringstream stream(input);
82
83     while (stream >> c) {
84         stream >> n;
85         Dict[c].push_back(n);
86     }
87     return true;
88 }
```