```cpp
#include <iostream>
#include <string>
#include <vector>
#include <algorithm>
#include <sstream>
#include <map>

using namespace std;

// Possible solution:
// A map with a character key and vector of int as elements.

// NOTE: The datasets identifiers (e.g. a, b) can be ANY character (it can be
// programmed to limit the character range allowed, but I'm not doing it here
// to keep it short).

// To be completely robust, possible errors in the input stream should be
// checked because situations like the following will happen (because of
// std::cin):
// "a 3y4" will cause this program to read 'a' as a key, put 3 in the dataset
// 'a', then read 'y' as another key and, and add 4 to the 'y' dataset...

// Also note that a map is ordered with respect to the keys AUTOMATICALLY
// so dataset 'a' will be printed before 'b', and so on.


bool createSets(const string &, map< char, vector<int> > &);
void display(const vector<int> &);



int main() {
    std::string input;
    std::map<char, std::vector<int> > Dict;
    std::getline(cin, input);

    if (createSets(input, Dict)) {
        // Scan through keys in the map
        // - Order the elements in the vector values
        // - Display
        for (auto it = Dict.begin() ; it != Dict.end() ; it++) {
            sort(it->second.begin(), it->second.end());
            display(it->second);
        }
    }
    else
        std::cout << "Error" << std::endl;
    return 0;
}




//////////////////////////////////////////////////////////////////////




```

```

// No changes compared to assignment 5-1

void display(const std::vector<int> & V) {
    for (auto n : V)
        std::cout << n << " ";
}








// Assume the stream has the correct format so that the
// stringstream can be read alternatively to get keys and values

bool createSets(const string & input, map<char , vector<int>> & Dict) {
    int n;
    char c;

    std::stringstream stream(input);

    while (stream >> c) {
        stream >> n;
        Dict[c].push_back(n);
    }
    return true;
}
```