```
/* Demonstration of using the map template class */
/* Craig Scratchley, Simon Fraser University! */
/* Help for Assignment 3, ENSC 351 October 2021 */
/* Copyright (c) 2021 Craig Scratchley */

#include <cstdlib>
#include <iostream>
#include <map>

using namespace std;

typedef struct {
    int i;
    char c;
} ValueStruct;

// map is a template class
typedef map< char, ValueStruct > CharValuestructMapT;
//typedef CharValuestructMapT::iterator CharValuestructMapItT;

int main(int argc, char *argv[]) {
    std::cout << "Welcome to the C++ map example" << std::endl;

    CharValuestructMapT myMap;

    ValueStruct myRecord{99, 'N'};

    myMap['n'] = myRecord;  // a copy of myRecord is put in the map

    myRecord.i = 44;   } // update myRecord
    myRecord.c = 'F';  }
    myMap['f'] = myRecord;        // will be
                                  placed in order
                                  of alphabet

    myMap.emplace('g', ValueStruct{33, 'G'});
     // more efficient to add things in map
```

myMap		
char	myRecord	
'n'	99	'N'
'f'	44	'F'
'g'	33	'G'

'f' → 'g' → 'n'

```
    // Note how we can easily access a field of a value.
    cout << "'n' maps to a structure with integer: " << myMap['n'].i << endl;
    cout << "'f' maps to a structure with integer: " << myMap['f'].i << endl;

    // you can test if you don't happen to know if a key is in the map
    if (myMap.find('f') != myMap.end()) {
        cout << "Changing structure mapped to by 'f' so that integer is 444" << endl;
        myMap['f'].i = 444;
    }
    cout << "Now 'f' maps to a structure with integer: " << myMap['f'].i << endl;
```

'f' , 444 , 'F'

```
    // or perhaps more efficiently ...

    // //CharValuestructMapItT
    // auto
    // CharValuestructMapIt0 = myMap.find('f');

    auto CharValuestructMapIt(myMap.find('f'));
    if (CharValuestructMapIt != myMap.end()) {
        cout << "Changing structure mapped to by 'f' so that integer is 4444" << endl;
        CharValuestructMapIt->second.i = 4444;
    }
    cout << "Now 'f' maps to a structure with integer: " << myMap['f'].i << endl;

    // or using CharValuestructMapIt
    cout << "Again, '" << CharValuestructMapIt->first << "' maps to a structure with
integer: " << CharValuestructMapIt->second.i << endl;

    // you can erase a key/value pair from a map (specifying the key)
    cout << "Erasing the key/value pair with key 'f'" << endl;
    myMap.erase('f');

    if (myMap.find('f') == myMap.end()) {
        cout << "Now 'f' is no longer a key in the map" << endl;
    }

    return EXIT_SUCCESS;
}
```