Ryan Ellis
COSC 320 --- 001 Lab 6 3/11/2025

Lab report: In this lab, we were given the header file d_state.h for the stateCity class structure and were asked to write two programs: one to implement the STL set library and include the stateCity class structure, and another to implement the STL map library to prompt the user for a string "state" input and check to see if the input was apart of the set or map. This lab took me about 1 hour to complete cumulatively.

Pre-lab:

1. Review the contents we learned about set and map.

Lab 3.1

Exercise 1:

Header file (d_state.h)

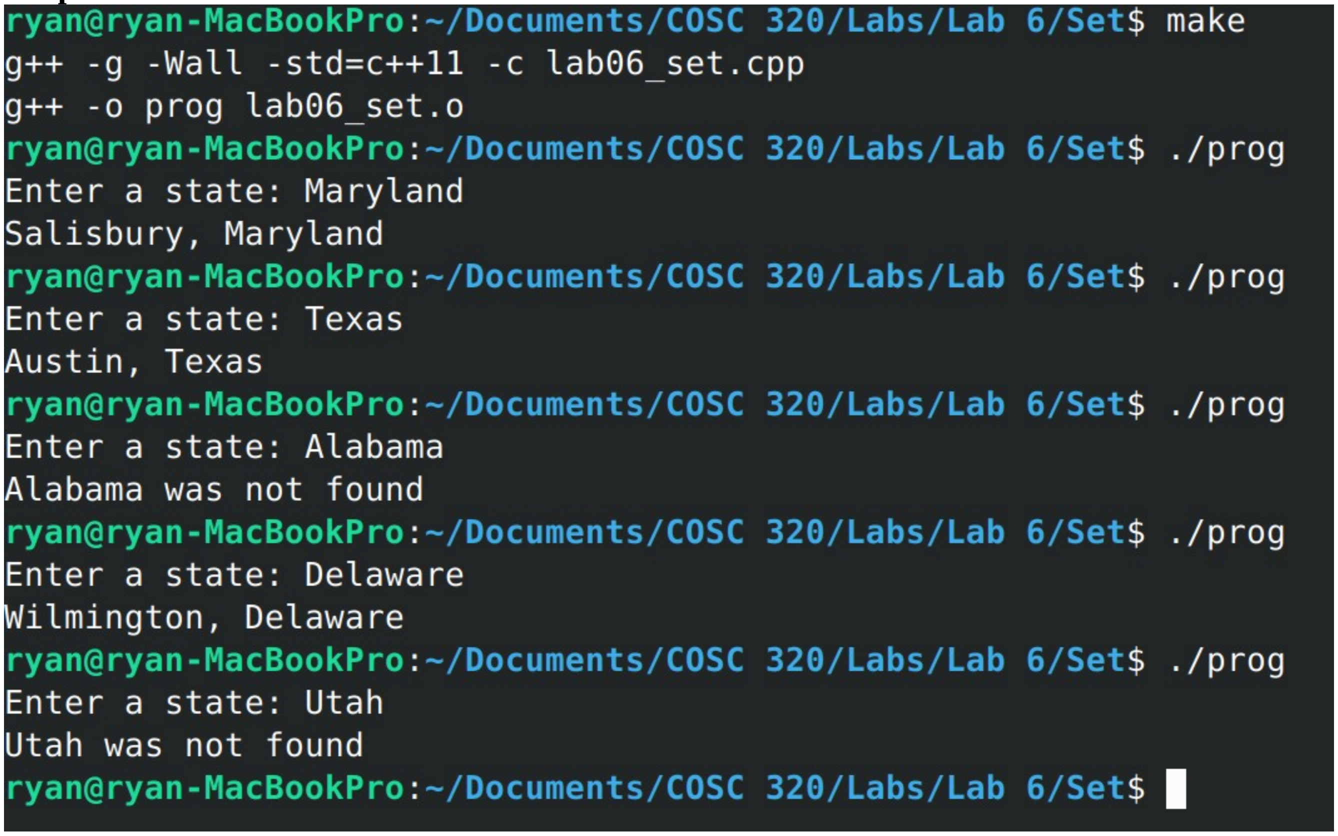
```
#ifndef STATECITY_CLASS
#define STATECITY_CLASS

#include <iostream>
#include <string>

using namespace std;

// object stores the state name and city in the state
class stateCity
{
	public:
		stateCity (const string& name = "", const string& city = "");

		// output the state and city name in the format
		//    cityName, stateName
		friend ostream& operator<< (ostream& ostr, const stateCity& state);

		// operators < and == must be defined to use with set object.
		// operators use only the stateName as the key
		friend bool operator< (const stateCity& a, const stateCity& b);

		friend bool operator== (const stateCity& a, const stateCity& b);

	private:
		string stateName, cityName;
};

#endif  // STATECITY_CLASS

//=================================================
// Description: Constructor for the stateCity class structure, sets
// the passed parameters to the private values of the city and state name.
```

```
//=====================================================
stateCity :: stateCity(const string &name, const string &city){
   stateName = name;
   cityName = city;
}

//=====================================================
// Description: Overloaded stream operator for the stateCity class structure,
// which will output the city and state name in a formatted output.
//=====================================================
ostream &operator<<(ostream &ostr, const stateCity& state){
   ostr<< state.cityName <<", "<<state.stateName<<endl;
   return ostr;
}

//=====================================================
// Description: Overloaded equivalency operator for the stateCity class
// structure, which returns a true boolean value if the two states are
// equivalent.
//=====================================================
bool operator==(const stateCity &a, const stateCity &b){
   bool status;

   if(a.stateName == b.stateName)
      status = true;
   else
      status = false;
   return status;
}

//=====================================================
// Description: Overloaded less than operator for the stateCity class
// structure, which returns a true boolean value if the left state
// is less than the right state.
//=====================================================
bool operator< (const stateCity &a, const stateCity &b){
   bool status;

   if(a.stateName < b.stateName)
      status = true;
   else
      status = false;
   return status;

}
```

Implementation File (lab06_set.cpp)

```
//===============================================================
// Filename: lab06_set.cpp
// Author: Ryan Ellis
// Creation Date: 3/11/2025
// Last Update: 3/11/2025
// Description: Main program that implements STL set and stateCity class structure that builds a set of
// stateCity objects and prompts the user for a state and searches the set and returns if it is found or not.
//=================================================================

#include <iostream>
#include <set>
#include "d_state.h"

using namespace std;

int main(){

    set<stateCity> s;
    set<stateCity>::iterator pos;              //define set and iterator

    stateCity md("Maryland", "Salisbury");      //define stateCity objects
    stateCity de("Delaware", "Wilmington");
    stateCity tx("Texas", "Austin");
    stateCity fl("Florida", "Miami");
    stateCity md1("Pennsylvania", "York");

    string input;          //user input

    s.insert(md);          //insert set with stateCity objects
    s.insert(de);
    s.insert(tx);
    s.insert(fl);
    s.insert(md1);

    cout<<"Enter a state: ";       //prompt for user input
    cin>> input;

    stateCity flag(input, "");     //set input as flag for search

    pos = s.find(flag);            //set iterator for flag

     if(pos != s.end())            //if iterator isn't pointing at end of set it's found
            cout<< *pos;
     else
            cout<<input <<" was not found"<<endl;       //otherwise it's not found



    return 0;
}
```

Output:

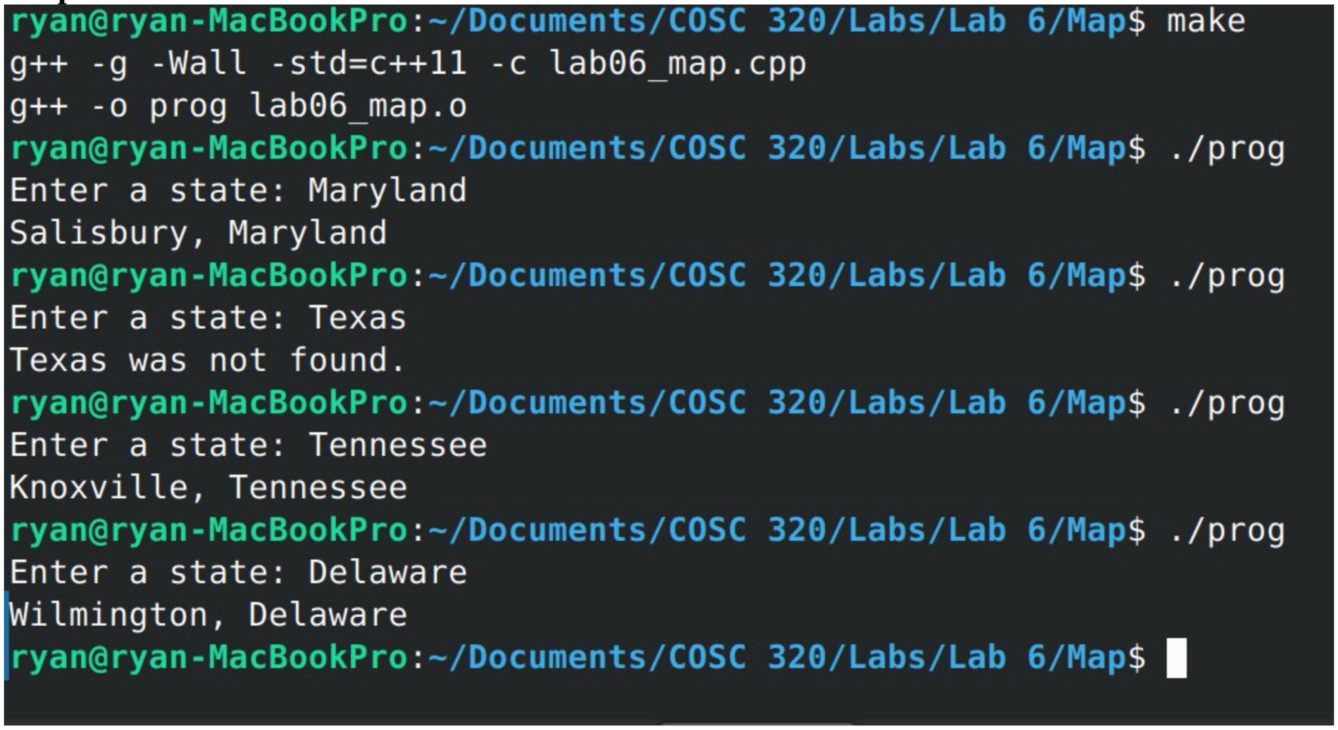


```
ryan@ryan-MacBookPro:~/Documents/COSC 320/Labs/Lab 6/Set$ make
g++ -g -Wall -std=c++11 -c lab06_set.cpp
g++ -o prog lab06_set.o
ryan@ryan-MacBookPro:~/Documents/COSC 320/Labs/Lab 6/Set$ ./prog
Enter a state: Maryland
Salisbury, Maryland
ryan@ryan-MacBookPro:~/Documents/COSC 320/Labs/Lab 6/Set$ ./prog
Enter a state: Texas
Austin, Texas
ryan@ryan-MacBookPro:~/Documents/COSC 320/Labs/Lab 6/Set$ ./prog
Enter a state: Alabama
Alabama was not found
ryan@ryan-MacBookPro:~/Documents/COSC 320/Labs/Lab 6/Set$ ./prog
Enter a state: Delaware
Wilmington, Delaware
ryan@ryan-MacBookPro:~/Documents/COSC 320/Labs/Lab 6/Set$ ./prog
Enter a state: Utah
Utah was not found
ryan@ryan-MacBookPro:~/Documents/COSC 320/Labs/Lab 6/Set$
```

Exercise 2:

```
//=================================================================
// Filename: lab06_map.cpp
// Author: Ryan Ellis
// Creation Date: 3/11/2025
// Last Update: 3/11/2025
// Description: Main program that implements STL map structure that builds a map of
// key and data values using strings for state and city names, then prompts the user for a state and
searches
// the map and returns if it is found or not.
//=================================================================

#include <iostream>
#include <map>
#include "d_state.h"

using namespace std;

int main(){
    map<string, string> m;                  //define map and iterator
    map<string, string> ::iterator miter;

    m["Maryland"] = "Salisbury";        //populate map with city, state names
    m["Delaware"] = "Wilmington";
    m["Florida"] = "Miami";
    m["Tennessee"] = "Knoxville";
    m["Pennsylvania"] = "Philadelphia";


    string input;       //user input

    cout<<"Enter a state: ";        //prompt for user input
    cin>>input;

    miter = m.find(input);          //
    if(miter != m.end())
        cout<<miter->second<<", "<<miter->first<<endl;
    else
        cout<<input<< " was not found."<<endl;

    return 0;
}
```

Output:



```
ryan@ryan-MacBookPro:~/Documents/COSC 320/Labs/Lab 6/Map$ make
g++ -g -Wall -std=c++11 -c lab06_map.cpp
g++ -o prog lab06_map.o
ryan@ryan-MacBookPro:~/Documents/COSC 320/Labs/Lab 6/Map$ ./prog
Enter a state: Maryland
Salisbury, Maryland
ryan@ryan-MacBookPro:~/Documents/COSC 320/Labs/Lab 6/Map$ ./prog
Enter a state: Texas
Texas was not found.
ryan@ryan-MacBookPro:~/Documents/COSC 320/Labs/Lab 6/Map$ ./prog
Enter a state: Tennessee
Knoxville, Tennessee
ryan@ryan-MacBookPro:~/Documents/COSC 320/Labs/Lab 6/Map$ ./prog
Enter a state: Delaware
Wilmington, Delaware
ryan@ryan-MacBookPro:~/Documents/COSC 320/Labs/Lab 6/Map$
```