CS 1428
Fall 2019
Gentry Atkinson
Lab 10

Introduction:

By now you are very familiar with creating and using variables of different types. But C++ does not limit you to using built in data types. You can also create collections of primitive data types called **structs** which can represent more complex objects from the real world. So if, for instance, you knew that every student has a name, an age, and a major then a student could be represented like this:

```
struct Student {
        string      name;
        int         age;
        string      major;
};
Student student1;
student1.name = "Gentry";
```

This allows us to collect data into logical collections which are easier to understand than declaring big groups of primitive variables separately. This is also the first step towards "Object Orientation" which collects all of your code into logically related packets rather than just the data.

The purpose of today's lab is to familiarize you with defining, declaring, and using **struct**s.

Directions:

1- Launch Code::Blocks and start a new file. Name it your_last_name_lab9.cpp.

2- Include the standard header for this lab:

```
//Your Name
//CS1428 Fall 2019
//Lab 10
```

3- Include the iostream and string standard libraries and declare some functions that we will be using in this lab. Start your main function:

```
#include <iostream>
#include <string>
```

```
using namespace std;
int main() {
```

4- You have been hired as the in-house developer for a real estate broker. They need to be able to track a large number of Properties which they are currently handling. Define a **struct** *before your main function* called Property which will hold the following pieces of information:

address	a string of numbers and letters
city	a string of letters
price	a number, possibly with a decimal
available	true or false

5- Create an **array** of **Properties** with the following values:

123 Lovely Ln San Marcos 155000.00 true	246 Awful Way Buda 144000.25 true	135 Bigol Hill Kyle 123888.99 true
235 LBJ Court San Marcos 200000.00 true	1123 Van Zandt Dr Austin 900000.00 true	136 Triangle Pl New Braunfels 66000.00 true

You can use this code to create and initialize your array:

```
Property whateverNameYouWant[] = {
        {"123 Lovely Ln", "San Marcos", 155000.0, true},
        {"246 Awful Way", "Buda", 144000.25, true},
        {"135 Bigol Hill", "Kyle", 123888.99, true},
        {"235 LBJ Court", "San Marcos", 200000.0, true},
        {"1123 Van Zandt Dr", "Austin", 900000.0, true},
        {"136 Triangle Pl", "New Braunfels", 66000.0, true}
}
```

6- Write a function called **printProperty** which will take a Property as a parameter and neatly print the values of the Property to the console. Your output should be similar to the following sample:

```
Address:        123 Lovely Ln
City:           San Marcos
Price:          155000.00
Available:      true
```

The prototype of this function will be:

```
void printProperty(Property toPrint);
```

7- Your shop has been working hard and has sold several properties. They want to be able to easily update your array so that the **available** value in the array gets set to **false** instead of **true.** Write a function with the following signature:

- ❏ **Parameters:** a string **address** which holds the address of a Property, an array of Properties called **properties**, and an int **size** which holds the size of the array **properties**
- ❏ **Side Effects:** one value in properties with the same address as given in the parameters should have its **available** value set to **false.** Only one value should be changed in the properties array every time this function is called.
- ❏ **Return: true** if a value has been updated and **false** otherwise.

The prototype of this function will be:

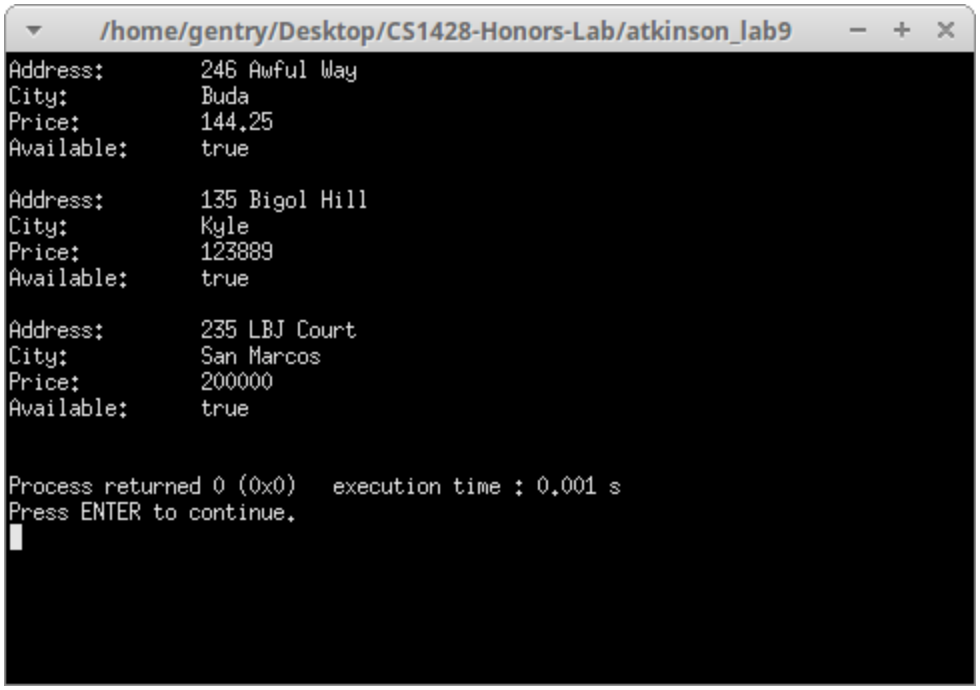
bool updateProperty(string address, Property properties[], int size);

8- Call **updateProperty** from your **main** function to mark the following properties as sold:

- 1123 Van Zandt Dr
- 123 Lovely Ln
- 136 Triangle Pl

9- Finally, add a loop to your **main** function which will print every property which is still available for sale.

10- Build and Run your program. Fix any errors. Your output should look something like this:



11- Submit your cpp file through TRACS. You can leave when you're done.