Unit 7 Assignment

Grading Information: This Program is **due** on **Date Specified**.

Comments are **REQUIRED**; flow charts and pseudocode are **NOT REQUIRED**.

Directions	**Points**
The files must be called <**LastInitialFirstInitialUnit7.java**> (driver) <**LastInitialFirstInitialAddress.java**> (handles Address variables and methods) *Proper coding conventions required the first letter of the class start with a capital letter and the first letter of each additional word start with a capital letter*. Only submit the .**java** files needed to make the program run. Do not submit the .**class** file or any other file.	5%
Style Components Include properly formatted prologue, comments, indenting, and other style elements as shown in Chapter 2 starting page 64 and Appendix 5 page 881-892.	5%
Topics covered in chapter Topics with * are covered in this assignment. **Ensure you use every item listed below with an * in your completed assignment.** *Relationships between reference variables and objects Reference assignment Memory space Equality of Objects Swap data in objects *Method call chaining *Object creating and initialization Calling one constructor from inside another Class variable usage Class methods Class constants	
Basic Requirements Write a driver class and an Address class that gets address data from file, process, and output the results. Refer to **chapter 3** on how to read in a file.	
LiFiUnit7.java (driver) Provide a driver class that demonstrates this Address class. It should contain a main that does the following in the specified order: • Print a welcome message (see sample)	20%

- Print a loading message (see sample)
- Use the following code to load address data from a given text file "addressList.txt".

```
Scanner fileIn = new Scanner(new File("addressList.txt"));
while ( fileIn.hasNext() )
{
  // read info from file
  String street, city, state, zip;

  // read street:
  // break into two reads to handle blank lines between addresses
  // next() will skip blank lines between addresses but stops at
spaces,
  // and nextLine() will read in the rest of the street line
  street = fileIn.next() + fileIn.nextLine();

  city = fileIn.nextLine();
  state = fileIn.nextLine();
  zip = fileIn.nextLine();

  // ADD CODE

} // end file reading while
fileIn.close(); // close file input Scanner object.
```

- Add into the above while loop: declare and initialize an Address object by invoking the 4-parameter constructor. Pass in the street, city, state, zip you just read from the file
- Add into the above while loop: call the display() method on the Address object to print this address (see sample for format)
- After the loop ends, print an end of list message (see sample)
- Print total # of addresses. Need to call the getAddressCount() class method of Address class.
- Print # of PO Box addresses. Need to call the getPOBoxAddressCount() class method of Address class.

This demonstration driver does not call all accessor and mutator methods but it is normal to create them regardless of an immediate use. They may be needed in the future.

Sample output is provided below. Be sure to mimic it exactly.	10%
LiFiAddress.java Write an Address class called LiFiAddress.java that with the follow members: • Four private instance data members, all of String type: street, city, state, and zipCode • Two private *class* data members, both of int type: addressCount, poboxAddressCount. Both should be initialized to 0 within their declaration	60%

- One default constructor with empty body
- One 4-parameter constructor that takes four String type parameters in the order of street, city, state, and zipCode. This constructor should do the following:

```
assign each parameter to the corresponding instance data member

increment class member addressCount by 1

if street String contains "PO Box"
  increment class member poboxAddressCount by 1
```

Hint:
1) Use the indexOf method from String class which takes a String object parameter.
public int indexOf(String str)
See Fig 5.8 on p179 and code example on p183 for detail of this method. You may also check the Java API documentation page for String class.
2) must use method chaining (ch7.6)

- static method getAddressCount(): no parameter, int return type. Return the class member addressCount
- static method getPOBoxAddressCount(): no parameter, int return type. Return the class member poboxAddressCount
- instance method setAddress(): four parameters, all of String type in the order of street, city, state, and zipCode. void return type. This method assign each parameter to the corresponding instance data member.
- Four getXXX methods: no parameter, String return type. Each returns the corresponding instance data member.
- All data members (instance or class) must be private
- All constructors and methods (instance or class) must be public

NOTE: Complete your activity and submit it by clicking "Submit Assignment"	
Total Percentage	100%

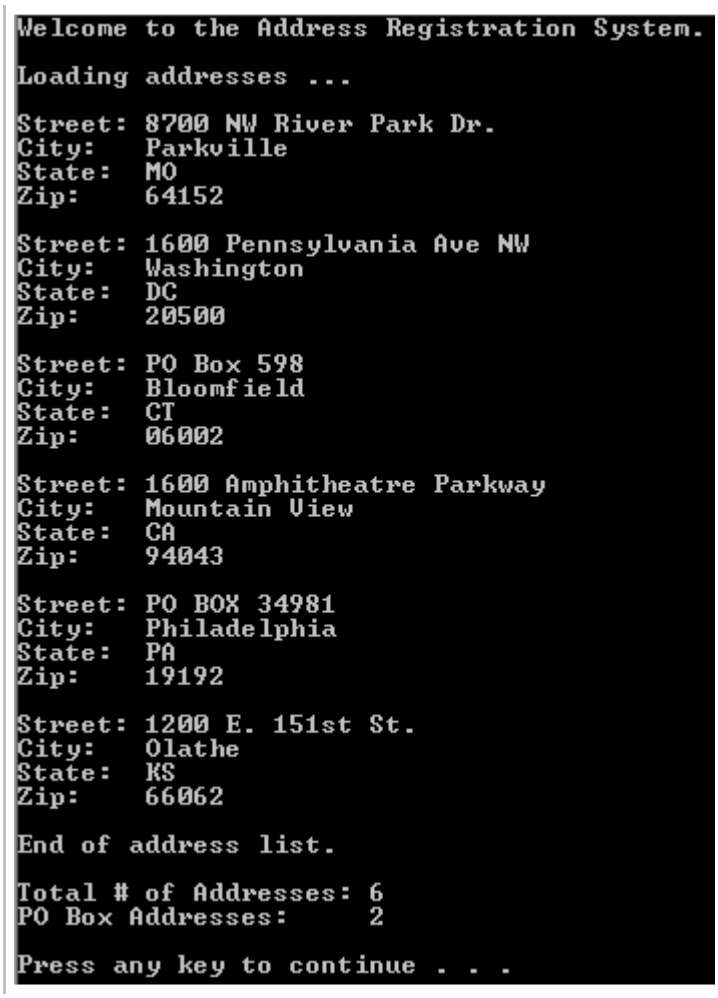
Data for Text File (named addressList.txt. Download it on the assignment page)

```
8700 NW River Park Dr.
Parkville
MO
64152

1600 Pennsylvania Ave NW
Washington
DC
20500

PO Box 598
Bloomfield
```

```
CT
06002

1600 Amphitheatre Parkway
Mountain View
CA
94043

PO BOX 34981
Philadelphia
PA
19192

1200 E. 151st St.
Olathe
KS
66062
```

Sample

Your output should match the sample below. Notice that the two PO Box addresses are both recognized even though they are of different cases: PO Box and PO BOX.


Welcome to the Address Registration System.
Loading addresses ...
Street: 8700 NW River Park Dr.
City: Parkville
State: MO
Zip: 64152
Street: 1600 Pennsylvania Ave NW
City: Washington
State: DC
Zip: 20500
Street: PO Box 598
City: Bloomfield
State: CT
Zip: 06002
Street: 1600 Amphitheatre Parkway
City: Mountain View
State: CA
Zip: 94043
Street: PO BOX 34981
City: Philadelphia
State: PA
Zip: 19192
Street: 1200 E. 151st St.
City: Olathe
State: KS
Zip: 66062
End of address list.
Total # of Addresses: 6
PO Box Addresses: 2
Press any key to continue . . .