Exam 1

EE363 - Fall 2015
Software Components and Generic Programming

Instructions

Please read each question carefully, and answer each in the space provided. Use the back side of the pages if you need additional space, and be sure to indicate clearly which questions you are answering if you do.

_____

Name

Question 1

William and Theodore are partners in a design project. Instead of meeting to work together all of the time, they decide to use the Revision Control System Git in order to manage all of their updates. Both partners decide that they want to get started, and begin their implementation.

Bill and Ted each create a new Eclipse project with a single Java source file, shown below. They each share their projects to their own local Git repositories.

Ted's Code:

```
1       package com.tedlogan.fall2013.ee363.excellent;
2       public class Journey {
3               private String bogus;
4               public Journey(String bodacious) {
5                       this.bogus = "bodacious";
6               }
7       }
```

Bill's Code

```
1       package com.billpreston.fall2013.ee363.excellent;
2       public class Journey {
3               private String bogus;
4               public Journey() {
5                       bogus = "short dead dude";
6               }
7               public String getBogus() {
8                       return bogus;
9               }
10              public void setBogus(String bogus) {
11                      this.bogus = bogus;
12              }
13 }
```

Both Bill and Ted need to push their changes up stream to the shared Git repository hosted at the repository URL: http://rufus.camp.clarkson.edu/redmine/git/adventure.git

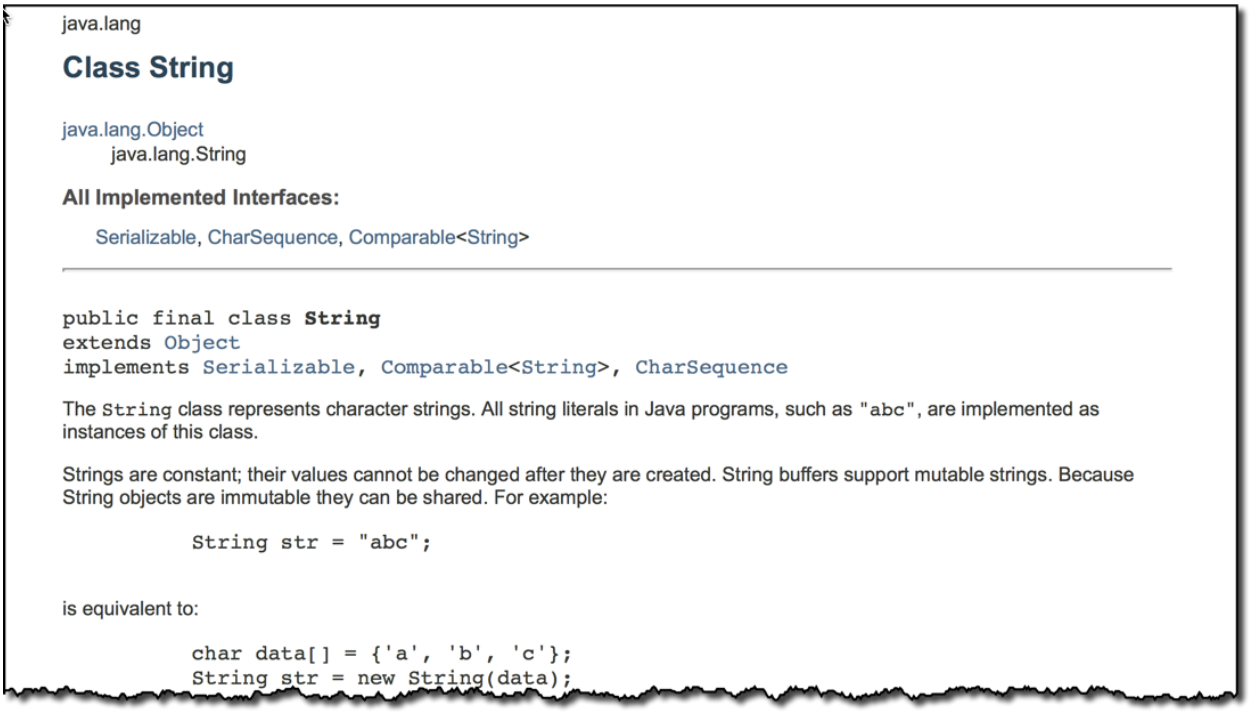
Explain what happens when they both attempt to push their changes upstream simultaneously. Are any problems encountered? How could those problems have been avoided? Discuss any best-practices for coordinating development efforts within a team to avoid otherwise totally bogus issues moving forward.

Question 2

Consider the following fragment of Java source code, and answer the questions below:

```
1               String test1 = "compare";
2               test1.toUpperCase();
3
4               if (test1 == "COMPARE") {
5                       System.out.println("True");
6               }
7               else {
8                       String test2 = test1;
9                       System.out.println(test2 + " is not \"COMPARE\"");
10              }
```

The JavaDoc documentation of the String class, and String's "toUpperCase()" method are provided here:



java.lang

Class String

java.lang.Object
    java.lang.String

All Implemented Interfaces:

Serializable, CharSequence, Comparable<String>

```
public final class String
extends Object
implements Serializable, Comparable<String>, CharSequence
```

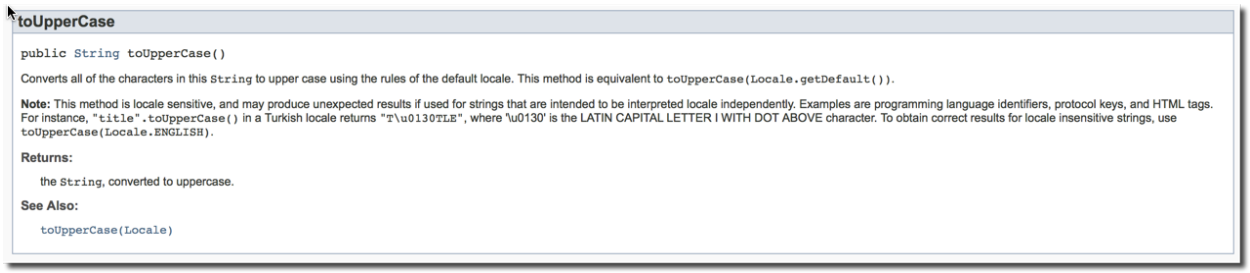
The `String` class represents character strings. All string literals in Java programs, such as `"abc"`, are implemented as instances of this class.

Strings are constant; their values cannot be changed after they are created. String buffers support mutable strings. Because String objects are immutable they can be shared. For example:

```
String str = "abc";
```

is equivalent to:

```
char data[] = {'a', 'b', 'c'};
String str = new String(data);
```



toUpperCase

`public String toUpperCase()`

Converts all of the characters in this `String` to upper case using the rules of the default locale. This method is equivalent to `toUpperCase(Locale.getDefault())`.

Note: This method is locale sensitive, and may produce unexpected results if used for strings that are intended to be interpreted locale independently. Examples are programming language identifiers, protocol keys, and HTML tags. For instance, `"title".toUpperCase()` in a Turkish locale returns `"T\u0130TLE"`, where '\u0130' is the LATIN CAPITAL LETTER I WITH DOT ABOVE character. To obtain correct results for locale insensitive strings, use `toUpperCase(Locale.ENGLISH)`.

Returns:

the `String`, converted to uppercase.

See Also:

`toUpperCase(Locale)`

a. What is printed to the console? Why? Is it the expected value for this test?

b. Step through this code fragment as if you were debugging it. Explain briefly what each line does.

c. Answer true or false, with an explanation for any “false” answer, for each of the following statements about the variable test2:

test2 is a pointer to the zeroth position of a character array containing with the values: [‘C’,’O’,’M’,’P’,’A’,’R’,’E’, ‘\0’]

test2 is a statically allocated object of type String which contains the value “COMPARE”

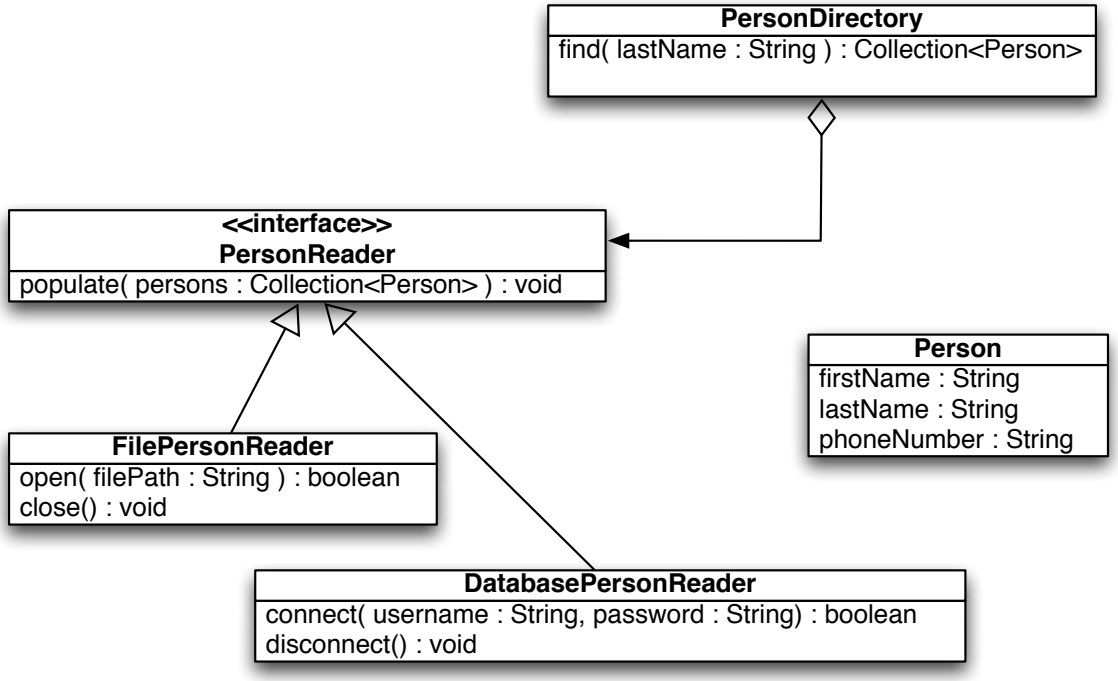
test2 refers to a dynamically allocated object of type String which contains the value “COMPARE”

test2 refers to the same memory location on the stack as test1

test2 refers to the same memory location on the heap as test1

Question 3

Consider the following UML diagram and implementation for the PersonDirectory class.




```
package com.timfanelli.ee363.exam1;

public class PersonDirectory {
	private List<Person> allPeople;
	private PersonReader personReader;

	public PersonDirectory() {
		this.allPeople = new ArrayList();
		personReader = new FilePersonReader( );

		personReader.open( “C:\\TEMP\\people.txt” );
		personReader.populate(allPeople);
		personReader.close();
	}

	public Collection<Person> find( String lastName ) {
		Collection<Person> results = new ArrayList<Person>();
		for ( Person p : allPeople ) {
			if ( p.getLastName().equals(lastName) )
				results.add( p );
		}

		return results;
	}

}
```

The system above was written by Samir to simplify his life... when his colleague Peter started blowing off work at Initech, he was constantly having to chase down employees to get work done to keep their boss, Lumbergh, off their backs. This directory system made it easy for Samir to lookup his colleagues numbers instead of having to go find their cubicles to talk to them.

When Initech started modernizing their infrastructure, though, they provided a web-based directory where Samir could look up phone numbers instead of from his own database or local file (which he kept updated himself). He decides to add a new "WebDirectoryPersonReader" to his system.

Answer the following questions:

a. On the class diagram above, draw in the new WebDirectoryPersonReader class in the appropriate place.

b. Does the addition of this class violate either the Liskov Substitution Principle, or the Open Closed Principles? Explain, and propose a solution if necessary.

c. Does PersonDirectory need to be modified in order to take advantage of the new WebDirectoryPersonReader, and if so, how?

d. Does your answer to (c) violate either the Liskov Subsitution Principle, or the Open Closed Principles? Explain, and propose a solution if necessary.