COMP 401 Midterm

Tuesday, Oct 18, 2016 2pm-3:15pm

Instructions

1. Please spread out and try and sit in alternate seats.
2. This is a closed book exam.
3. You will not be penalized for errors in Java syntax.
4. Write on the exam itself. Write on a blank page if there is not enough space to solve a problem.
5. There are:
    - 5 numbered pages including this one and any marked blank pages.
    - 3 questions.
    - 75 possible points. Point values appear in brackets next to each question.
6. You are not required to comment or annotate any code you write, but may get partial credit if you write appropriate comments/annotations but incorrect code.
7. If you need to make any assumptions to clarify a problem, *write your assumptions down.* Only reasonable assumptions get full credit.
8. Please inform the proctor of anything in the exam that you think is a mistake.
9. Your code will be evaluated not only for correctness, but also for time and space efficiency and *style*.
10. You cannot use any Java capabilities not covered in class.
11. To answer questions about some piece of code given here, you can mark the code directly.
12. If you do not understand some English word, do not hesitate to ask the proctor. Naturally, you are expected to know the computer science terms defined in class.
13. Write clearly using a pen/pencil with a dark color – we will be scanning your exams.
14. Please try and write in the allocated space, as that will reduce mistakes in grading of scanned exams.
15. Put your initials/onyen in the header of each page, in case the pages are separated.

NAME , ONYEN , and EMAIL if it is not onyen@live.unc.edu (ALL CAPITALS PLEASE)

(NAME)_____(ONYEN)_____ EMAIL_____

Pledge: I have neither given nor received unauthorized aid on this exam.

(signed)_____

For survey purposes, please indicate the time at which you turned in the exam.

_____

Please do not write below

1. _____/47 2. _____/18 3. _____/10

Total: _____/ 75

[Initials/Onyen:]

1. [47 pts.] REPLACING SPACE CHARACTERS WITH UNDERSCORES

Write a class, ASpaceReplacer, that fulfills the requirements below. You can define other types and refer to them in your class. The requirements for ASpaceReplacer are that it:

1. Implements a method named "replacementChar", which takes a parameter of type char. If the parameter is a space (' '), it returns an underscore ('_'). Otherwise it returns the parameter. The method has *default* rather than public access.
2. Implements a method named "replacementString", which takes a string as a parameter and returns a copy of this string with each space replaced with an underscore. It has *default access*.
3. Provides an editable String property, Input; and a read only String property, Output.
4. Provides constructor that takes a String parameter defining the initial value of Input.
5. Makes sure that the value of the Output property is aways a copy of the Input property with each space replaced with an underscore. Thus, if Input is: "H e L Lo 1!", Output is: "H_e_L_Lo_1! ".
6. Makes sure that the getters of the two properties do not do any scanning, directly or by calling other methods.

All methods in the class are *instance methods*. The relevant String instance methods you should use are:

```
public char charAt(int index) //Returns the char value at the specified index.
public int length()//Returns the length of this string.
```

In addition, you need to use the String "+" operator, which returns a concatenation of its two operands.

The constraints given above describe only some of the conditions your code must satisfy. Certain relevant stylistic rules, covered in class, are not explicitly stated. *A major part of your grade will depend on how well you apply these rules*. In other words, you must adhere to the same requirements that you would if this was a class assignment (with a perfect human grader!), except where explicitly stated otherwise. You can assume all types you define in this exam are in the same package (the default package), and do not need to give any package declaration. You also do not need to give any annotations or comments. Use the last page if you need more space.

[Initials/Onyen:]

[Initials/Onyen:]

2. [18 pts.] REPLACING LOWERCASE LETTERS WITH UPPERCASE LETTERS

(a)[14pts] Write a subclass of ASpaceReplacer, AnUpperCaser, that is like ASpaceReplacer, except that a different character replacement is involved: The value of the Output property is a copy of the Input property with each lowercase letter replaced with the corresponding uppercase letter. Thus, if Input is: "H e L Lo 1!", Output is: " H E L LO 1!".

The relevant Character static methods are.:

```
public static boolean isLowerCase(char ch// Determines if ch is lowercase
public static char toUpperCase(char ch)// Converts ch to uppercase
```

Again, be sure to satisfy these constraints while following relevant stylistic rules, covered in class, not explicitly stated. (It might help you to realize that if m1() and m2() are instance methods in a class, then the call, m2(), by m1() is really the call **this.**m2(), where **this** is the object on which m1() is called. For example, the call, replacementString(), is really the call, **this.** replacementString(). Recall also that which implementation of m2() is called can depend on the class of **this**. If this hint confuses you, ignore it!) *You can get points for this question even if you do not attempt problem 1 as long as you state the assumptions you make about the implementation of ASpaceReplacer.* If this class declares a method that is a variation of a method in the superclass, you can describe how the declared method is different rather than giving the code of the entire method. You must, however, give the header of each method declared in this class. Use the last page if the space below is not enough.

[Initials/Onyen:]

(b)[4pts] Explain whether it makes logical sense for AnUpperCaser to be a subclass of ASpaceReplacer.

3. [10 pts.] MAIN USER OF THE TWO CLASSES

Complete the main method below by *using the two classes above* to convert the value of aString, "H e L Lo 1!", to the replacement strings "H_e_L_Lo_1! " and " H E L LO 1!". The two results should be printed on the console. You can create additional variables to perform the replacements.

This question is intended to help you demonstrate your ability to use the two classes correctly, following style rules you have learned in class. *You can get points for this question even if you do not attempt problems 1 and 2.*

```
public class CharacterReplacerUser {
  public static void main(String[] args) {
      String aString = "H e L Lo 1!";
```