# CSA Mock Exam 2: administered on AP Live 05/06/2020

## Question 2: Methods & Control Structures plus open-ended question

---

**Directions: SHOW ALL YOUR WORK. REMEMBER THAT PROGRAM SEGMENTS ARE TO BE WRITTEN IN JAVA.**

Notes:

- Assume that the classes listed in the Java Quick Reference have been imported where appropriate.
- Unless otherwise noted in the question, assume that parameters in method calls are not `null` and that methods are called only when their preconditions are satisfied.
- In writing solutions for each question, you may use any of the accessible methods that are listed in classes defined in that question. Writing significant amounts of code that can be replaced by a call to one of these methods will not receive full credit.

2.

```
public class WordScrambler
{
  private String[] scrambledWords;

  /** @param wordArr an array of String objects
   *          Precondition: wordArr.length is even
   */
  public WordScrambler(String[] wordArr)
  {
    scrambledWords = mixedWords(wordArr);
  }


  /** @param word1 a String of characters
   *  @param word2 a String of characters
   *  @return a String that contains the first half of word1 and the second half of word2
   */
  private String recombine(String word1, String word2)
  {  /* to be implemented in part (a) */  }
```

(a) Write the `WordScrambler` method `recombine`. This method returns a `String` created from its two `String` parameters as follows.

- take the first half of `word1`
- take the second half of `word2`
- concatenate the two halves and return the new string.

For example, the following table shows some results of calling `recombine`. Note that if a word has an odd number of letters, the second half of the word contains the extra letter.

| word1 | word2 | recombine(word1, word2) |
|---|---|---|
| "apple" | "pear" | "apar" |
| "pear" | "apple" | "peple" |

Complete method `recombine` below.

```
/** @param word1 a String of characters
 *  @param word2 a String of characters
 *  @return a String that contains the first half of word1 and the second half of word2
 */
private String recombine(String word1, String word2)
```

(b)

A programmer would like to add a method `checkValidLength` that will utilize the `recombine` method and determine if the newly formed string is considered *valid*. A recombined string is considered *valid* if its length is less than at least one of the originally paired words.

For example, the original words "apple" and "pear" would result in a recombined word "apar". The recombined "apar" would be considered *valid* because the length of "apar" is less than the length of "apple". The original words "pear" and "apple" would result in the recombined word "peple". The recombined word "peple" would not be considered *valid* because its length is not less than "pear" or "apple".

Write a description of how you would implement the `checkValidLength` so that it returns true if a pair of words is valid.

Make sure to include the following in your response.

- Write the method header for the `checkValidLength` method.
- Identify any new, existing, or modified variables or methods from the `WordScrambler` class that could be called to support this modification. **Do not write the program code for this change.**
- Identify any local variables that would be required within the `checkValidLength` method and describe how each would be implemented. **Do not write the program code for this change.**

**STOP**

**END OF THE MOCK EXAM**