

Part I – Multiple Choice

There will be 20 AP Exam-style multiple choice questions. See attached handout.

Part II – Free Response

There will be 6-10 methods for you to implement similar to the following methods.

1. Write a method named `remove` as started below. The method finds the first occurrence of a specified `String` named `word` in a given `String` named `text` and returns `text` with `word` removed. There should only be one space in the position where `word` is removed. In other words, there should not be any occurrences of two consecutive blank spaces. If `word` is not found, the method returns `text` unchanged.

```
// precondition: text is a non-empty string that consists of words separated by single blank spaces
// postcondition: if word is found in text, its first occurrence is removed from text and text is then
//                returned; otherwise text is returned unchanged

public static String remove(String text, String word)
```

2. Assume that you are given the following `Sentence` class which has a class invariant that `myWords` will always contain a set of English words that are separated by single blank spaces. Each word in `myWords` only contains letters. That is, there are no punctuation symbols (such as periods, question marks, or even hyphens) or digits within `myWords`. There is no leading blank space in front of the first word in `myWords` and there is no trailing blank space behind the last word in `myWords`. You are also guaranteed as a class invariant that `myWords` will always contain at least 2 or more words.

```
public class Sentence
{
    private String myWords;

    default and "other" constructors are fully implemented and available
}
```

```
// postcondition: the number of separate words in myWords is returned
public int numWords()
{
    a) implement this method here
```

```
}
```

```
// postcondition: the number of separate letters in myWords is returned
public int numLetters()
{
    b) implement this method here
```

```
}
```

```
// precondition: key will not be a null string and it will be a word
//                that only contains letters
// postcondition: if the String key is found anywhere within myWords,
//                the boolean value true will be returned;
//                otherwise false will be returned.
public boolean find(String key)
{
    c) implement this method here
```

```
}
```

```
// precondition: num > 0 and num will be less than or equal to
// the number of words in myWords
// postcondition: return the numth word in myWords where the very
// first word would be returned if num is 1
public String getWord(int num)
{
    d) implement this method here
```

```
}
```

```
// postcondition: return the boolean value true if any word
// occurs twice or more in myWords
public boolean containsDouble()
{
    e) implement this method here
```

```
}
```

```
// precondition: return the boolean value true if any word
// ends with the letter lowercase 's'
public boolean containsWordThatEndsWithS()
{
    f) implement this method here

```

```
}
```

```
}
```