Word Processor

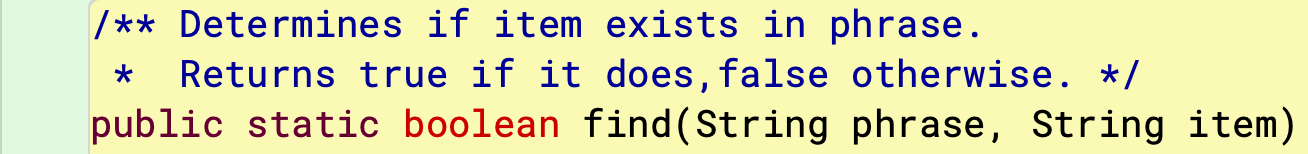
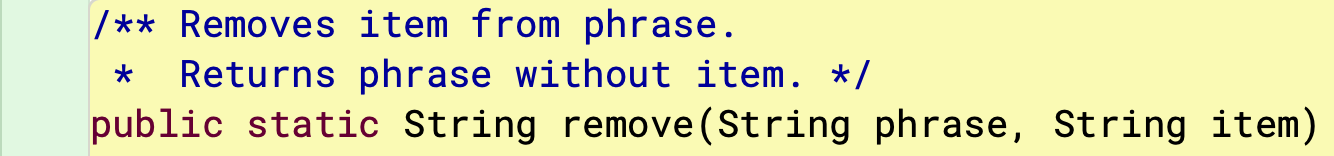
A word processor is a device or computer program that provides for input, editing, formatting, and output of text, often with some additional features. Early word processors were stand-alone devices dedicated to the function, but current word processors are word processor programs running on general-purpose computers. ([Wikipedia](https://en.wikipedia.org/wiki/Word_processor)).

Computer users access word processing applications every day but probably have no idea how basic word processing string manipulations work.

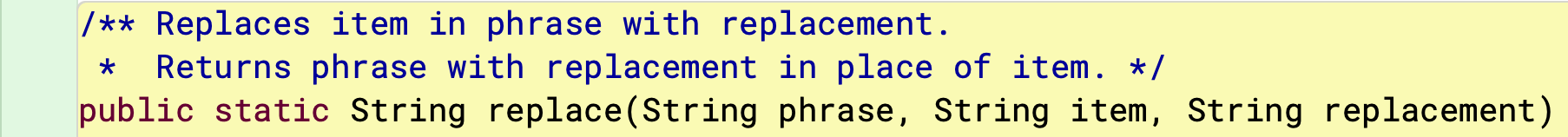
In this project, you will be recreating three basic word processing methods for manipulating strings of characters, namely, **find**, **remove**, and **replace**.

* **find** determines if a given String exists in another String.
* **remove** eliminates a given String from another String.
* **replace** changes a given String found in another String with a replacement String.

To get started on this project, download, extract, and save the [Word Processor](https://drive.google.com/file/d/1a0mhygu5jGAg1B2bjrFYjwZ0tZFVGxyS/view?usp=sharing) project to your computer. Your task is to write the find, remove, and replace from scratch in the WordProcessor class as follows.

* Here are the find method comments and signature.  
    
    
    
  Add opening and closing curly braces, and between them return the correct outcome (hint: use indexOf, recalling what indexOf returns when what is being searched for is not found).
* Here are the remove method comments and signature.  
    
    
    
  The method returns a String, which is phrase without item. If item does not exist in phrase, then the method just returns the original phrase. Be sure to check first if item is in phrase (hint: use the method you just wrote). If it is,  
  + find the position that item is in phrase
  + create a substring of phrase from zero to position
  + create another substring of the remainder of phrase from position + item.length(
  + return the two concatenated String objects.

If item is not in phrase, simply return phrase.

* Here are the replace method comments and signature.  
    
    
    
  The method returns a String which is item in phrase replaced by replacement. Similar to remove, if item does not exist in phrase, then the method just returns the original phrase. Again, be sure to check first if item is in phrase. If it is,  
  + find the position that item is in phrase
  + create a substring of phrase from zero to position
  + create another substring of the remainder of phrase from position + item.length()
  + concatenated the first String + the replacement String + the remainder String.
  + return the concatenated String.

If item is not in phrase, simply return phrase.

BTW, no need to worry about the possibility of an item being repeated more than once in a phrase; each method is intended to deal with just a single occurrence.

Take each method in order, one at a time. Run WordProcessorTest to check your work (and to offer helpful hints).