Since the advent of worldwide audio communications, the international telecommunications industry has adopted the Radiotelephony Spelling Alphabet (RSA) to aid in understanding garbled audio messages. In other words, RSA is a way of verbally spelling out the individual letters in an audio message. For example, the letters in the phrase **Hello World** could be verbally spelled out using RSA as **Hotel Echo Lima Lima Oscar Whiskey Oscar Romeo Lima Delta**. Notice how the first letter of each RSA word corresponds to a letter of the alphabet.

This project involves writing two methods that will take a `String` message and convert it into RSA words. To get started on this project, download, extract, and save the Telephony projected onto your laptop. The project download includes the method headers and a `static` block (technically not a method) that will automatically execute when the class loads. This static block will retrieve the RSA words from a file and store them in a class (static) array of `String` objects called `phonicAlphabet`.

Begin by completing the `convertLetter` method, which converts a single letter into the corresponding RSA word. For example, a call to `convertLetter("F")` should return the RSA word **Foxtrot** as a `String`. To keep things simple, note that the first line of the method body (provided for you) ensures the parameter `letter` is uppercase. Loop through the `phoneticAlphabet` array and see if the first letter of each entry matches `letter`. If so, return that array element. (Remember the proper way to compare two `String` objects for equivalence.) If you get through the entire array without a match, then return an empty string (`""`). Run the tester to make sure this method is working correctly before continuing.

Next, complete the `convertPhrase` method, which converts a phrase into an array of RSA words, one RSA word for each letter in the phrase. This method returns an array of `String` objects, so start by declaring and instantiating what you need to return (a new array of `String` objects). Recall that every time you create a new array, you must specify how big it is. How big should this one be? The array's length should be the same as the length of the phrase. Then, go ahead and return the array at the end of the method. Now, all that needs to be done is to write the lines of code in between that fill the array appropriately. You will want to loop through each character in `phrase`, one letter at a time. Call `convertLetter` each time through the loop, using each letter in `phrase`, and store the results in the array you created.

That's all there is to it. Run the tester to make sure this method is working correctly, and then submit a screenshot of the tester success message.

When you are finished, run the `main` method in `TelephonyDriver` and enter a message of your choice to see it translated into RSA.