Removing Vowels From a String

- `removeVowels(s)` - removes the vowels from the string `s`, returning its "vowel-less" version!

  `removeVowels("recursive")` should return `"rcrsv"`

  `removeVowels("vowel")` should return `"vwl"`

- Can we take the usual approach to recursive string processing?
  - base case: empty string
  - delegate `s.substring(1)` to the recursive call
  - we're responsible for handling `s.charAt(0)`
    yes!

Which combination is correct?

```
public static String removeVowels(String s) {
    if (s.equals("")) {     // base case
        return __________;
    } else {                // recursive case
        String rem_rest = ____________________;
        // do our one step!
    }
}
```

	first blank	second blank
A.	`""`	`s.substring(1)`
B.	`""`	`removeVowels(s.substring(1))`
C.	`0`	`s.substring(1)`
D.	`0`	`removeVowels(s.substring(1))`

Consider Concrete Cases

`removeVowels("after")` `# first char is a vowel`

- what is its solution? `"ftr"`
- what is the next smaller subproblem? `removeVowels("fter")`
- what is the solution to that subproblem? `"ftr"`
- how can we use the solution to the subproblem?
  What is our one step? just return the subproblem's solution!

`removeVowels("recurse")` `# first char is not a vowel`

- what is its solution? `"rcrs"`
- what is the next smaller subproblem? `removeVowels("ecurse")`
- what is the solution to that subproblem? `"crs"`
- how can we use the solution to the subproblem?
  What is our one step? `"r" + "crs"`

Now write the rest of the function!

removeVowels()

```
public static String removeVowels(String s) {
    if (s.equals("")) {       // base case
        return "";
    } else {                  // recursive case
        String rem_rest = removeVowels(s.substring(1));
        if ("aeiou".indexOf(s.charAt(0)) != -1) {
            return rem_rest;
        } else {
            return s.charAt(0) + rem_rest;
        }
    }
}
```