1 More Practice with Recursion

ANSWER KEY

For each problem, write a recursive method with code. Use appropriate comments.

1. A function to tell if b is prime. By the math definition, b is prime if b is only (evenly) divisible by 1 and itself. Here is a math formula. It uses two functions.

   $isPrime(b) = isPrime(b, b-1)$

   $$isPrime(b,x) = \begin{cases} yes & \text{if } x == 1 \\ no & \text{if } (b\%x) == 0 \\ isPrime(b,x-1) & \text{otherwise} \end{cases}$$

```
boolean isPrime(int b) {
  return isPrimt(b, b-1);
}

boolean isPrime(int b, int x) {
  if (x == 1) return true;
  else if ((b % x) == 0) return false;
  else return isPrime(b, x-1);
}
```

2. A function to count the number of primes between a and b. Here is a math formula.

   $$count\_primes(a,b) = \begin{cases} 0 & \text{if } a > b \\ count\_primes(a,b-1)+1 & \text{if } isPrime(b) \\ count\_primes(a,b-1) & \text{otherwise} \end{cases}$$

```
int countPrimes(int a, int b) {
  if (a<b) return 0;
  else if (isPrime(b)) return countPrimes(a, b-1) + 1;
  else return countPrimes(a, b-1);
}
```

3. A function to remove consecutive duplicates from a string For example, convert ”aabccba” to ”abcba”. Convert ”ww” to ”w”. Convert ”” to ””. Convert ”v” to ”v”. Convert ”qvq” to ”qvq”.

```
// words of length 0 and 1 have no duplicates to remove
// otherwise, if the first 2 letters are the same,
//                 ignore the first letter and recurse on letters 1...end
//            else keep the first letter (0th) and recurse on 1..end
String removeDuplicates(String word) {
     if(word == null || word.length() <= 1)
        return word;
     else if( word.charAt(0) == word.charAt(1) )
```

[1] Borrowed from Professor Wittie

```
        return removeDuplicates(word.substring(1, word.length()));
    else
        return word.charAt(0) +
            removeDuplicates(word.substring(1, word.length()));
}
```