Recursive Algorithms

CSC 210 Practice Exercises

Power

Write a recursive method that given two integers (the base, and the exponent) the method returns $base^{power}$

Palindrome

Write a recursive method that takes as argument a word (String) and returns `true` if the word is a palindrome, `false` otherwise.

A palindrome reads the same when inverted. Examples of palindromes: civic, radar, level, rotor, kayak

N-th Fibonacci Number

Fibonacci numbers are a sequence of numbers where every number is the sum of the preceding two numbers.

Write a recursive method that given a number `n`, it returns the `n`th Fibonacci Number.

Exhaustive Search – DNA

DNA is a long molecule made by stringing 4 chemical bases together: adenine (A), guanine (G), cytosine (C), and thymine (T).

Write an enumeration method that given a size `k` it prints out all possible molecules of size `k`.

Example of method call: `dnaSequence(10, "");`

ANSWERS

Power

```
public static int power(int base, int exp) {
        if (exp == 1) return base; // base case, base^1
        else return base * power(base, exp-1);

    }
```

Palindrome

```
public static boolean recPalind(String word, int start, int end) {
        if (start == end) return true; // middle character
        if (word.charAt(start) != word.charAt(end)) return false;
        if (start < end + 1) return recPalind(word, start + 1, end - 1);
        return true;
    }

public static boolean isPalindrome(String word) {
        if (word.length() == 0) return true;
        return recPalind(word, 0, word.length()-1);
    }
```

N-th Fibonacci Number

```
public static int fibonacci(int n) {
        if (n == 0) return n;
        else if (n == 1) return 1;
        else return fibonacci(n-1) + fibonacci(n-2);
    }
```

Exhaustive Search – DNA

```
public static void dnaSequence(int k, String molecule) {
        if (k == 0) System.out.println(molecule);
        else {
            dnaSequence(k-1, molecule + "A");
            dnaSequence(k-1, molecule + "G");
            dnaSequence(k-1, molecule + "C");
            dnaSequence(k-1, molecule + "T");
        }
    }
```