Unix commands for use in PuTTy:

<https://en.wikipedia.org/wiki/List_of_Unix_commands>

Recursion Practice:

**public** **class** RecursiveMethods {

**public** **static** **void** main(String[] args) {

//TEST C-5.13

**int** result = *recursiveProduct*(4,4);

System.***out***.println("A recursive product: " + result);

//TEST C-5.18

String string1 = "gohangasalamiimalasagnahog";

**boolean** isPalindrome = *isPalindrome*(string1);

System.***out***.println("Result of palindrome testing: " + isPalindrome);

//TEST C-5.19

String string2 = "aeioubcs";

**boolean** hasMoreVowels = *hasMoreVowels*(string2, 0, 0, 0);

System.***out***.println("More Vowels than Consonants?: " + hasMoreVowels);

}

//C-5.13

//Recursive algorithm used to compute the product of two positive integers, m and n,

//using only addition and subtraction

**public** **static** **int** recursiveProduct(**int** m, **int** n) **throws** UnsupportedOperationException

{

//Anything multiplied by 0 is 0.

**if**(n == 0 || m == 0) {

**return** 0;

}

//This method does not support negative arguments

**else** **if**(n < 0 || m < 0) {

**throw** **new** UnsupportedOperationException("Only accepts nonnegative inputs.");

}

//This instance is the base case. As well as the

//main branch of the recursion.

//When n is 1, the recursion should stop because

//a number multiplied by one is the number itself.

**else**

{

**return** m + *recursiveProduct*(m, (n - 1));

}

}

//C-5.18

//Recursive Java method that determines if a string is a palindrome

**public** **static** **boolean** isPalindrome(String s)

{

//This is the base case and the input validation. Both nothing and one

//are the same forwards and backwards in my mind. Additionally, for the

//program to work, the base case needs to be considered true.

**if**(s.length() == 0 || s.length() == 1)

**return** **true**;

//Check the first position and the last position indices for equivalence.

**else** **if**(s.charAt(0) == s.charAt(s.length()-1)) {

**return** *isPalindrome*(s.substring(1, s.length()-1));

}

//If neither of these first two cases is true, then it is necessarily false.

**else**

**return** **false**;

}

//C-5.18

//Recursive Java method that determines if a string has more

//vowels than consonants

**public** **static** **boolean** hasMoreVowels(String s, **int** index, **int** vowelCount,

**int** consonantCount) **throws** UnsupportedOperationException

{

**if**(index < s.length() && s.toUpperCase().charAt(index) >= 'A' &&

s.toUpperCase().charAt(index) <= 'Z') {

**switch** (s.toUpperCase().charAt(index)) {

**case** 'A':

**case** 'E':

**case** 'I':

**case** 'O':

**case** 'U':

vowelCount++;

**break**;

**default**:

consonantCount++;

}

index++;

**return** *hasMoreVowels*(s, index, vowelCount, consonantCount);

}

**else** **if**(index == s.length())

**if** (vowelCount > consonantCount) {

**return** **true**;

}

**else**

**return** **false**;

**else**

**return** **false**;

}

}