5. Recursion :: Palindromic Strings

A phrase is a **palindrome** if it reads the same forward as backward:

"Go hang a salami, I'm a lasagna hog."

In this discussion of palindromic strings we shall exclude whitespace and punctuation characters from the string and shall also ignore letter case, so the above sentence would be:

"gohangasalamiimalasagnahog"

Our problem is determine if a string s is a palindrome. First, consider this string:

$s =$ "" and $s.length() = 0$

Is the empty string a palindrome? That depends on how you define palindrome; our definition is:

1. Let s be a string and $s.length() \geq 0$.
2. Let r be the string that results from reversing the characters of s.
3. If s is the empty string then r is the empty string.
4. s is a palindrome if $s.equals(r)$ is true.

By this definition, the empty string is a palindrome.

5. Recursion :: Palindromic Strings (continued)

Now, what about this string:

$s =$ "a"

It should be pretty clear this is a palindrome since $r =$ "a" and $s.equals(r)$ is true. In fact, we can start to formalize this discussion by defining a rule:

Rule 1: A string s, $s.length() \leq 1$, is a palindrome.

What about these strings:

$a =$ "aa"
$b =$ "ab"
$c =$ "aaa"
$d =$ "aba"
$e =$ "abc"

Clearly a, c, and d are palindromes and b and e are not. Now consider these strings:

$f =$ "a??????b"
$g =$ "$a???????????a$"

where ? represents *any* character—we are just obscuring them. It is pretty clear that f is not a palindrome because the leftmost character does not match the rightmost character, which leads to:

Rule 2: A string s, $s.length() > 1$, is not a palindrome if $s_0 \neq s_{s.length()\text{-}1}$.

5. Recursion :: Palindromic Strings (continued)

Is $g =$ "a???????????a" a palindrome? It *could* be: certainly it does not meet Rules 1 or 2. In fact, g **would** be a palindrome **if "???????????" is a palindrome**. Let's formalize this as Rule 3:

Rule 3: A string s, $s.length() \geq 2$, is a palindrome if both of these requirements are met:

a. $s_0 = s_{s.length()\text{-}1}$
b. The substring $t = s_{1:s.length()\text{-}2}$ is a palindrome.

At this point you should note that **Rule 3 is a recursive definition**: we define a string s as a palindrome if ... is a palindrome. And anytime we have a recursive definition, we can employ recursion to derive a solution to a problem.