- Prove the following algorithm is correct

```
ReverseString(s)

Input  : String s
Output: A string whose characters are the reverse of s
1 l=length(s);
  // Base Case
2 if l equals 1 then
3 |  return s;
4 end
  // Recursive step
5 return Concatenate(s[l],ReverseString(s[1 : l − 1]));
  // s[a] is ath element of s.  s[a : b] is the ath to bth
     elements, inclusive.
```

Set-up and Base Case

- Let $P(n)$ be the predicate that ReverseString correctly reverses any string of length n. We will prove via induction that $P(n)$ is true for all $n \geq 1$.
- Base case: $P(1)$ is true because for strings of length 1, the reverse of the string is the same as the string, so the algorithm should just return the string. This is indeed what the algorithm does in lines $1 - 2$.

Inductive Step

- Inductive step: Let $k \geq 1$. We will prove $P(k+1)$. Consider when the input s is a string of size $k+1$, so $l = k+1$ Since $k \geq 1$, then $k+1 \geq 2$, so the algorithm goes to the recursive step. By our inductive assumption since $s[1: l-1]$ is a string of length k, ReverseString($s[1: l-1]$) works correctly and returns the reverse of the first k characters of s. If we put the last character of s at the beginning of the reverse of the first k characters, we get the full reverse of s. This is what line 4 does, so $P(k+1)$ is true.

Conclusion

- Therefore, by induction, $P(n)$ is true for all $n \geq 1$.