

# Section 8.11

## Induction and Recursive Algorithms

# String Reversal

Name: ReverseString(s)

Input: string s

Output: the reverse of s

```
if (s = "")  
    return ""  
else  
    c := first character of s  
    s2:= the result of removing the first character from s  
    r := ReverseString(s2)  
    return rc, the result of adding c to the end of r  
end-if
```

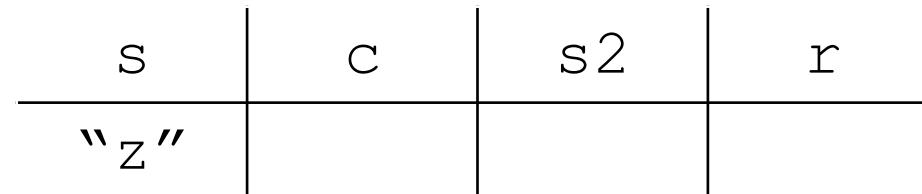
# String Reversal

Name: ReverseString(s)

Input: string s

Output: the reverse of s

```
if (s = "")  
    return ""  
else  
    c := first character of s  
    s2:= the result of removing the first character from s  
    r := ReverseString(s2)  
    return rc, the result of adding c to the end of r  
end-if
```



# String Reversal

Name: ReverseString(s)

Input: string s

Output: the reverse of s

```
if (s = "")  
    return ""
```

else

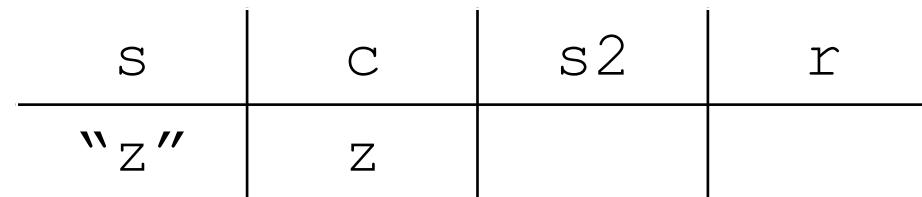
c := first character of s

s2:= the result of removing the first character from s

r := ReverseString(s2)

return rc, the result of adding c to the end of r

end-if



# String Reversal

Name: ReverseString(s)

Input: string s

Output: the reverse of s

```
if (s = "")  
    return ""
```

else

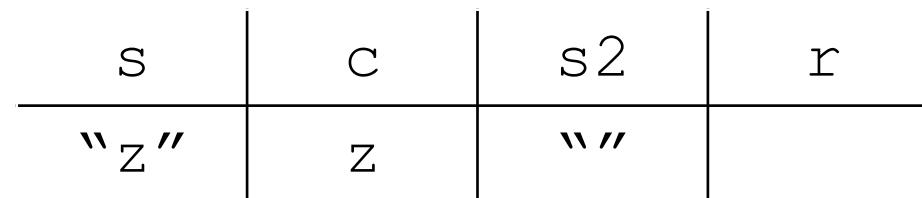
c := first character of s

s2:= the result of removing the first character from s

r := ReverseString(s2)

return rc, the result of adding c to the end of r

end-if



# String Reversal

Name: ReverseString(s)

Input: string s

Output: the reverse of s

```
if (s = "")  
    return ""
```

else

c := first character of s

s2:= the result of removing the first character from s

r := ReverseString(s2)

return rc, the result of adding c to the end of r

end-if

s	c	s2	r
"z"	z	""	""

# String Reversal

Name: ReverseString(s)

Input: string s

Output: the reverse of s

```
if (s = "")  
    return ""
```

else

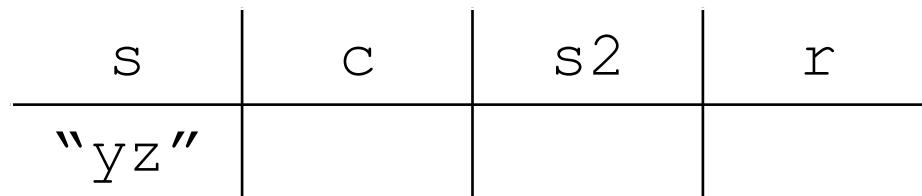
c := first character of s

s2:= the result of removing the first character from s

r := ReverseString(s2)

return rc, the result of adding c to the end of r

end-if



# String Reversal

Name: ReverseString(s)

Input: string s

Output: the reverse of s

```
if (s = "")  
    return ""
```

else

c := first character of s

s2:= the result of removing the first character from s

r := ReverseString(s2)

return rc, the result of adding c to the end of r

end-if

s	c	s2	r
"yz"	y		

# String Reversal

Name: ReverseString(s)

Input: string s

Output: the reverse of s

```
if (s = "")  
    return ""
```

else

c := first character of s

s2:= the result of removing the first character from s

r := ReverseString(s2)

return rc, the result of adding c to the end of r

end-if

s	c	s2	r
"yz"	y	"z"	

# String Reversal

Name: ReverseString(s)

Input: string s

Output: the reverse of s

```
if (s = "")  
    return ""
```

else

c := first character of s

s2:= the result of removing the first character from s

r := ReverseString(s2)

return rc, the result of adding c to the end of r

end-if

s	c	s2	r
"yz"	y	"z"	"z"

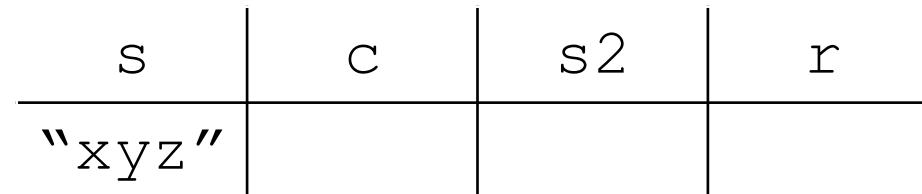
# String Reversal

Name: ReverseString(s)

Input: string s

Output: the reverse of s

```
if (s = "")  
    return ""  
else  
    c := first character of s  
    s2:= the result of removing the first character from s  
    r := ReverseString(s2)  
    return rc, the result of adding c to the end of r  
end-if
```



# String Reversal

Name: ReverseString(s)

Input: string s

Output: the reverse of s

```
if (s = "")  
    return ""
```

else

c := first character of s

s2:= the result of removing the first character from s

r := ReverseString(s2)

return rc, the result of adding c to the end of r

end-if

s	c	s2	r
"xyz"	x		

# String Reversal

Name: ReverseString(s)

Input: string s

Output: the reverse of s

```
if (s = "")  
    return ""
```

else

c := first character of s

s2:= the result of removing the first character from s

r := ReverseString(s2)

return rc, the result of adding c to the end of r

end-if

s	c	s2	r
"xyz"	x	"yz"	

# String Reversal

Name: ReverseString(s)

Input: string s

Output: the reverse of s

```
if (s = "")  
    return ""
```

else

c := first character of s

s2:= the result of removing the first character from s

r := ReverseString(s2)

return rc, the result of adding c to the end of r

end-if

s	c	s2	r
"xyz"	x	"yz"	"zy"

# The Correctness of ReverseString

- Prove by mathematical induction on the length of  $s$  that `ReverseString( $s$ )` returns the reversal of  $s$
- Base case: The length of  $s$  is 0

By the `ReverseString` algorithm, `ReverseString("")` returns ""

# The Correctness of ReverseString

- Induction step: The length of s is k+1

# The Correctness of ReverseString

- Induction step: The length of  $s$  is  $k+1$ 
  1. Assume that for all strings of length  $k$ , ReverseString returns the reversed string

# The Correctness of ReverseString

- Induction step: The length of s is k+1
  1. Assume that for all strings of length k, ReverseString returns the reversed string
  2. ReverseString( $c_1c_2\dots c_{k+1}$ ) returns ReverseString( $c_2\dots c_{k+1}$ ) followed by  $c_1$

# The Correctness of ReverseString

- Induction step: The length of s is  $k+1$ 
  1. Assume that for all strings of length  $k$ , ReverseString returns the reversed string
  2. ReverseString( $"c_1c_2\dots c_{k+1}"$ ) returns ReverseString( $"c_2\dots c_{k+1}"$ ) followed by  $c_1$
  3. ReverseString(s) returns  $"c_{k+1}\dots c_2"$  followed by  $c_1$  by the I.H.

# The Correctness of ReverseString

- Induction step: The length of  $s$  is  $k+1$ 
  1. Assume that for all strings of length  $k$ , ReverseString returns the reversed string
  2. ReverseString( $"c_1c_2\dots c_{k+1}"$ ) returns ReverseString( $"c_2\dots c_{k+1}"$ ) followed by  $c_1$
  3. ReverseString( $s$ ) returns  $"c_{k+1}\dots c_2"$  followed by  $c_1$  by the I.H.
  4. ReverseString( $s$ ) returns  $"c_{k+1}\dots c_2c_1"$

# Exponentiation

Name: Exponent(a, n)

Input: real number a and nonnegative integer n

Output:  $a^n$

```
if n = 0
    return 1
else
    return a * power(a, n-1)
endif
```

# The Correctness of Exponent

- Prove by mathematical induction on the natural number  $n$  that  $\text{Exponent}(a, n)$  returns  $a^n$
- Base case:  $n$  is 0
  1.  $\text{Exponent}(a, 0)$  returns 1
  2.  $\text{Exponent}(a, 0)$  returns  $a^0$

# The Correctness of Exponent

- Induction step:  $n = k+1$

# The Correctness of Exponent

- Induction step:  $n = k+1$ 
  1. Assume that  $\text{Exponent}(a, k)$  returns  $a^k$

# The Correctness of Exponent

- Induction step:  $n = k+1$ 
  1. Assume that  $\text{Exponent}(a, k)$  returns  $a^k$
  2.  $\text{Exponent}(a, k+1)$  returns  $a * \text{Exponent}(a, k)$

# The Correctness of Exponent

- Induction step:  $n = k+1$ 
  1. Assume that  $\text{Exponent}(a, k)$  returns  $a^k$
  2.  $\text{Exponent}(a, k+1)$  returns  $a * \text{Exponent}(a, k)$
  3.  $\text{Exponent}(a, k+1)$  returns  $a * a^k$  by the I.H.

# The Correctness of Exponent

- Induction step:  $n = k+1$ 
  1. Assume that  $\text{Exponent}(a, k)$  returns  $a^k$
  2.  $\text{Exponent}(a, k+1)$  returns  $a * \text{Exponent}(a, k)$
  3.  $\text{Exponent}(a, k+1)$  returns  $a * a^k$  by the I.H.
  4.  $\text{Exponent}(a, k+1)$  returns  $a^{k+1}$

# The Subsets of $\{a, b, c\}$ and $\{a, b, c\} \cup \{d\}$

The subsets of $\{a, b, c\}$	The subsets of $\{a, b, c\}$ combined with $d$
$\emptyset$	$\emptyset \cup \{d\}$
$\{a\}$	$\{a\} \cup \{d\}$
$\{b\}$	$\{b\} \cup \{d\}$
$\{c\}$	$\{c\} \cup \{d\}$
$\{a, b\}$	$\{a, b\} \cup \{d\}$
$\{a, c\}$	$\{a, c\} \cup \{d\}$
$\{b, c\}$	$\{b, c\} \cup \{d\}$
$\{a, b, c\}$	$\{a, b, c\} \cup \{d\}$
The subsets of $\{a, b, c\} \cup \{d\}$ that do not contain $d$	
The subsets of $\{a, b, c\} \cup \{d\}$ that contain $d$	

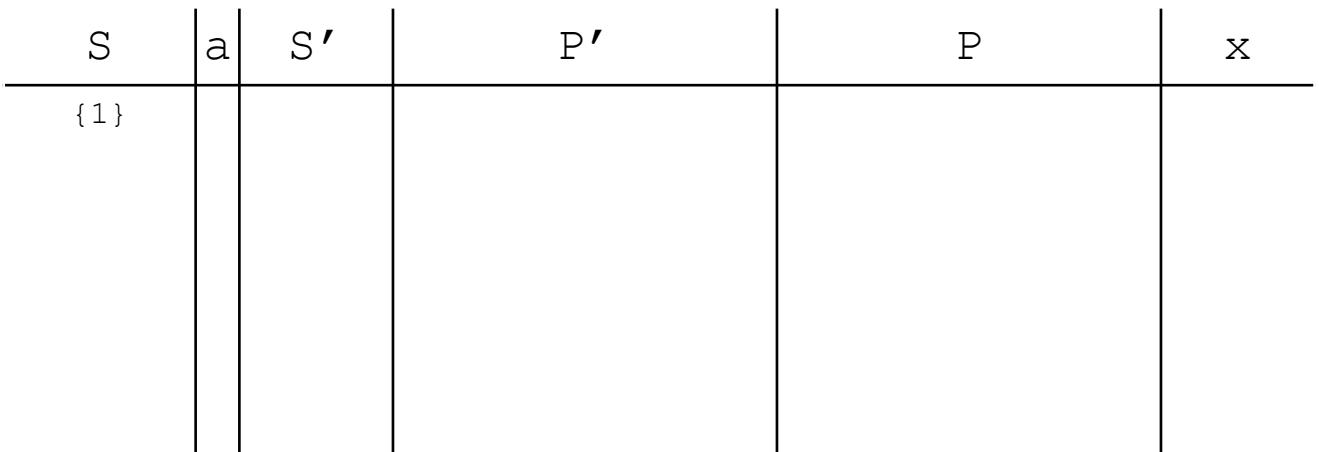
# Power Set

Name: PowerSet(s)

Input: a finite set S

Output: the power set of S

```
if S :=  $\emptyset$ 
    return { $\emptyset$ }
else
    select an element a in S
    S' := S - {a}
    P' := PowerSet(S')
    P := P'
    for each x in P'
        add  $x \cup \{a\}$  to P
    end-for
    return P
end-if
```



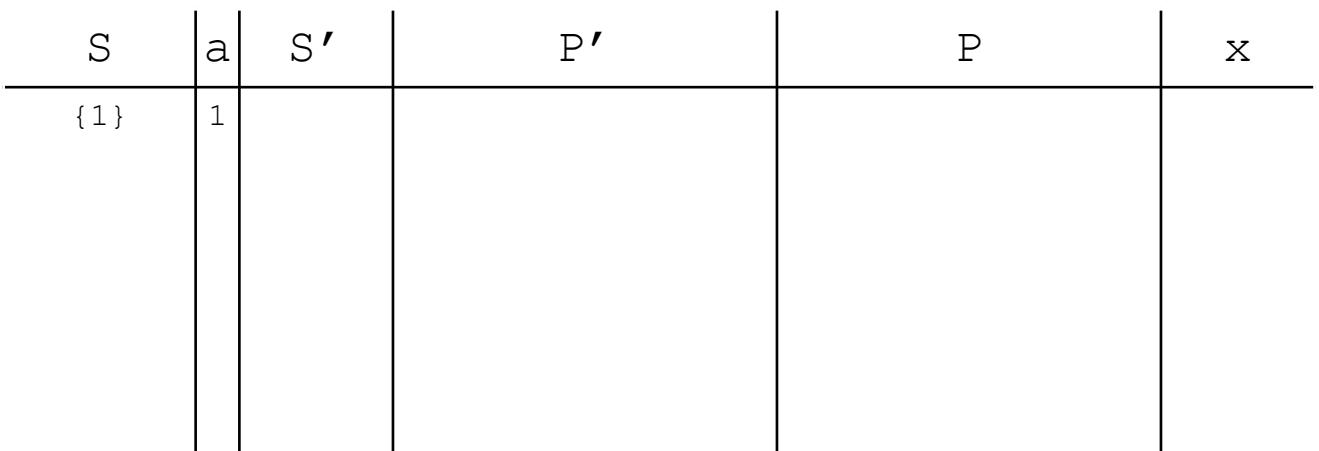
# Power Set

Name: PowerSet(s)

Input: a finite set S

Output: the power set of S

```
if S :=  $\emptyset$ 
    return { $\emptyset$ }
else
    select an element a in S
    S' := S - {a}
    P' := PowerSet(S')
    P := P'
    for each x in P'
        add x  $\cup$  {a} to P
    end-for
    return P
end-if
```



# Power Set

Name: PowerSet(s)

Input: a finite set S

Output: the power set of S

```
if S :=  $\emptyset$ 
    return { $\emptyset$ }
else
    select an element a in S
    S' := S - {a}
    P' := PowerSet(S')
    P := P'
    for each x in P'
        add x  $\cup$  {a} to P
    end-for
    return P
end-if
```

S	a	S'	P'	P	x
{1}	1	{}			

# Power Set

Name: PowerSet(s)

Input: a finite set S

Output: the power set of S

```
if S :=  $\emptyset$ 
    return { $\emptyset$ }
else
    select an element a in S
    S' := S - {a}
    P' := PowerSet(S')
    P := P'
    for each x in P'
        add x  $\cup$  {a} to P
    end-for
    return P
end-if
```

S	a	S'	P'	P	x
{1}	1	{}	{ $\emptyset$ }		

# Power Set

Name: PowerSet(s)

Input: a finite set S

Output: the power set of S

```
if S :=  $\emptyset$ 
    return { $\emptyset$ }
else
    select an element a in S
    S' := S - {a}
    P' := PowerSet(S')
    P := P'
    for each x in P'
        add x  $\cup$  {a} to P
    end-for
    return P
end-if
```

S	a	S'	P'	P	x
{1}	1	{}	{ $\emptyset$ }	{ $\emptyset$ }	

# Power Set

Name: PowerSet(s)

Input: a finite set S

Output: the power set of S

```
if S :=  $\emptyset$ 
    return { $\emptyset$ }
else
    select an element a in S
    S' := S - {a}
    P' := PowerSet(S')
    P := P'
    for each x in P'
        add x  $\cup$  {a} to P
    end-for
    return P
end-if
```

S	a	S'	P'	P	x
{1}	1	{}	{ $\emptyset$ }	{ $\emptyset$ }	$\emptyset$

# Power Set

Name: PowerSet(s)

Input: a finite set S

Output: the power set of S

```
if S :=  $\emptyset$ 
    return { $\emptyset$ }
else
    select an element a in S
    S' := S - {a}
    P' := PowerSet(S')
    P := P'
    for each x in P'
        add x  $\cup$  {a} to P
    end-for
    return P
end-if
```

S	a	S'	P'	P	x
{1}	1	{}	{ $\emptyset$ }	{ $\emptyset$ }	$\emptyset$

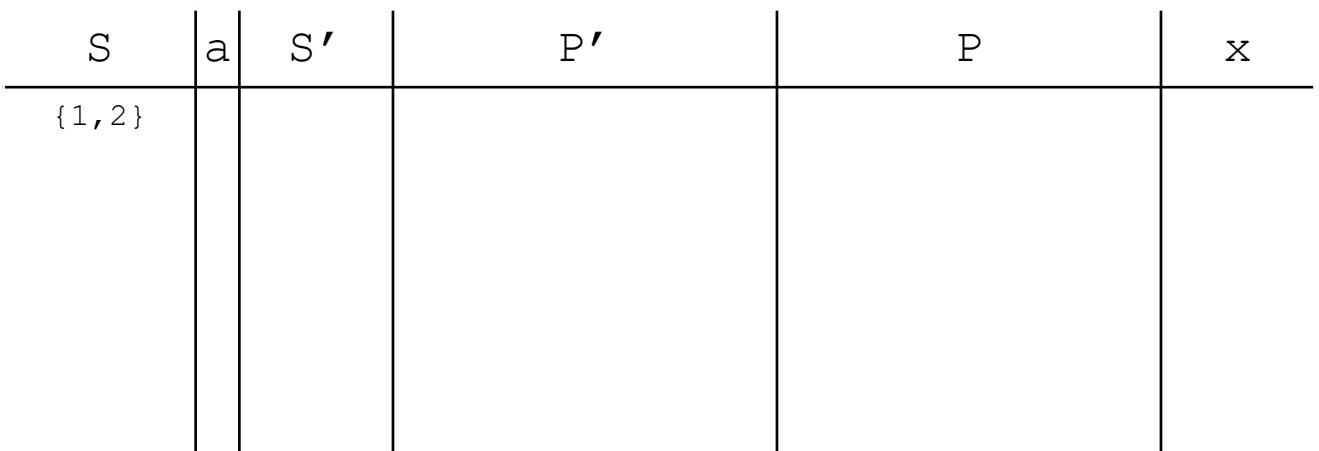
# Power Set

Name: PowerSet(s)

Input: a finite set S

Output: the power set of S

```
if S :=  $\emptyset$ 
    return { $\emptyset$ }
else
    select an element a in S
    S' := S - {a}
    P' := PowerSet(S')
    P := P'
    for each x in P'
        add  $x \cup \{a\}$  to P
    end-for
    return P
end-if
```



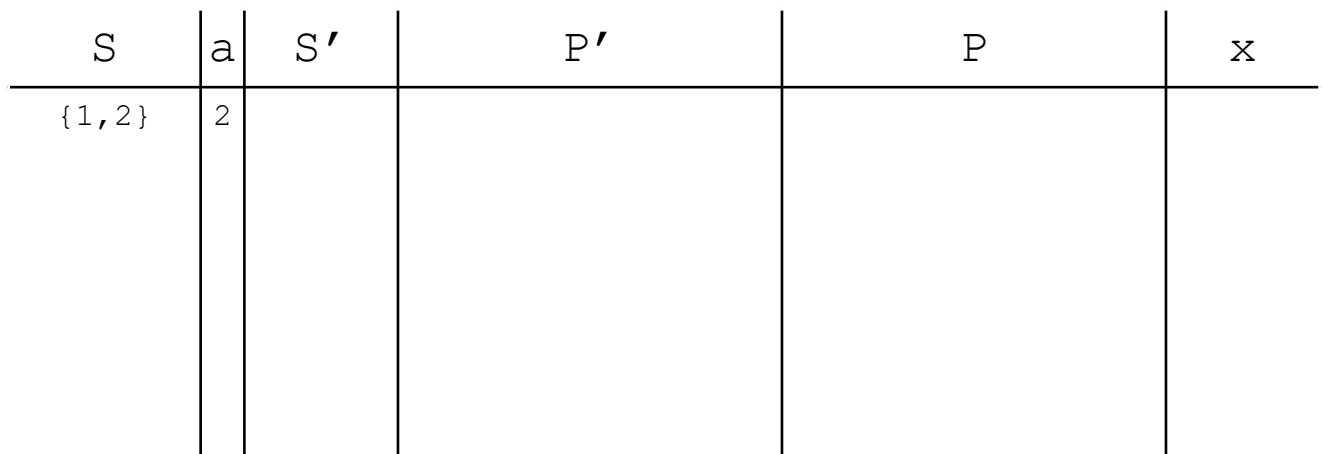
# Power Set

Name: PowerSet(s)

Input: a finite set S

Output: the power set of S

```
if S :=  $\emptyset$ 
    return { $\emptyset$ }
else
    select an element a in S
    S' := S - {a}
    P' := PowerSet(S')
    P := P'
    for each x in P'
        add  $x \cup \{a\}$  to P
    end-for
    return P
end-if
```



# Power Set

Name: PowerSet(s)

Input: a finite set S

Output: the power set of S

```
if S :=  $\emptyset$ 
    return { $\emptyset$ }
else
    select an element a in S
    S' := S - {a}
    P' := PowerSet(S')
    P := P'
    for each x in P'
        add x  $\cup$  {a} to P
    end-for
    return P
end-if
```

S	a	S'	P'	P	x
{1, 2}	1	{2}			

# Power Set

Name: PowerSet(s)

Input: a finite set S

Output: the power set of S

```
if S :=  $\emptyset$ 
    return { $\emptyset$ }
else
    select an element a in S
    S' := S - {a}
    P' := PowerSet(S')
    P := P'
    for each x in P'
        add  $x \cup \{a\}$  to P
    end-for
    return P
end-if
```

S	a	S'	P'	P	x
{1, 2}	2	{1}	{ $\emptyset, \{1\}$ }		

# Power Set

Name: PowerSet(s)

Input: a finite set S

Output: the power set of S

```
if S :=  $\emptyset$ 
    return { $\emptyset$ }
else
    select an element a in S
    S' := S - {a}
    P' := PowerSet(S')
    P := P'
    for each x in P'
        add x  $\cup$  {a} to P
    end-for
    return P
endif
```

S	a	S'	P'	P	x
{1, 2}	2	{1}	{ $\emptyset$ , {1}}	{ $\emptyset$ , {1}}	

# Power Set

Name: PowerSet(s)

Input: a finite set S

Output: the power set of S

```
if S :=  $\emptyset$ 
    return { $\emptyset$ }
else
    select an element a in S
    S' := S - {a}
    P' := PowerSet(S')
    P := P'
    for each x in P'
        add x  $\cup$  {a} to P
    end-for
    return P
end-if
```

S	a	S'	P'	P	x
{1, 2}	2	{1}	{ $\emptyset$ , {1}}	{ $\emptyset$ , {1}}	$\emptyset$

# Power Set

Name: PowerSet(s)

Input: a finite set S

Output: the power set of S

```
if S :=  $\emptyset$ 
    return { $\emptyset$ }
else
    select an element a in S
    S' := S - {a}
    P' := PowerSet(S')
    P := P'
    for each x in P'
        add x  $\cup$  {a} to P
    end-for
    return P
end-if
```

S	a	S'	P'	P	x
{1, 2}	2	{1}	{ $\emptyset$ , {1}}	{ $\emptyset$ , {1}}	$\emptyset$
				{ $\emptyset$ , {1}, {2}}	

# Power Set

Name: PowerSet(s)

Input: a finite set S

Output: the power set of S

```
if S :=  $\emptyset$ 
    return { $\emptyset$ }
else
    select an element a in S
    S' := S - {a}
    P' := PowerSet(S')
    P := P'
    for each x in P'
        add x  $\cup$  {a} to P
    end-for
    return P
end-if
```

S	a	S'	P'	P	x
{1, 2}	2	{1}	{ $\emptyset$ , {1}}	{ $\emptyset$ , {1}}	$\emptyset$
				{ $\emptyset$ , {1}, {2}}	{1}

# Power Set

Name: PowerSet(s)

Input: a finite set S

Output: the power set of S

```
if S :=  $\emptyset$ 
    return { $\emptyset$ }
else
    select an element a in S
    S' := S - {a}
    P' := PowerSet(S')
    P := P'
    for each x in P'
        add x  $\cup$  {a} to P
    end-for
    return P
end-if
```

S	a	S'	P'	P	x
{1, 2}	2	{1}	{ $\emptyset$ , {1}}	{ $\emptyset$ , {1}}	$\emptyset$
				{ $\emptyset$ , {1}, {2}}	{1}
				{ $\emptyset$ , {1}, {2}, {1, 2}}	

# The Correctness of PowerSet

- Prove by mathematical induction on the natural number  $n$  that for any set  $S$  of  $n$  elements  $\text{PowerSet}(S)$  returns the power set of  $S$
- Base case:  $n$  is 0 and  $S=\emptyset$

$\text{PowerSet}(\emptyset)$  returns  $\{\emptyset\}$

# The Correctness of PowerSet

- Induction step
  1. Assume that PowerSet returns the power set of any set of  $k$  elements
  2. Let  $S$  be a set of  $k+1$  elements and let  $a$  be the element selected from  $S$  by the PowerSet algorithm
  3. First, show that  $\text{PowerSet}(S) \subseteq \mathcal{P}(S)$
  4. Assume  $T \in \text{PowerSet}(S)$
  5. There are 2 cases:  $a \notin T$  and  $a \in T$

# The Correctness of PowerSet

6. Case 1:  $a \notin T$
7.  $T \in P' = \text{Powerset}(S') = \mathcal{P}(S')$  by the I.H.
8.  $T \subseteq S' \subseteq S$
9.  $T \in \mathcal{P}(S)$
10. Case 2:  $a \in T$
11.  $T$  was added to PowerSet( $S$ ) by adding  $a$  to an element of  $P'$
12. By the I.H.,  $P' = \mathcal{P}(S')$  so  $T \in \mathcal{P}(S)$  since  $a \in S$
13. In each case ,  $T \in \mathcal{P}(S)$
14.  $\text{PowerSet}(S) \subseteq \mathcal{P}(S)$

# The Correctness of PowerSet

15. Second, show that  $\mathcal{P}(S) \subseteq \text{PowerSet}(S)$

16. Assume  $T \in \mathcal{P}(S)$

17. There are 2 cases:  $a \notin T$  and  $a \in T$

18. Case 1:  $a \notin T$

19.  $T \subseteq S'$  and hence  $T \in \mathcal{P}(S')$

20.  $T \in \text{PowerSet}(S')$  by the I.H.

21.  $T \in \text{PowerSet}(S)$  since  $S' \subseteq S$

# The Correctness of PowerSet

22. Case 2:  $a \in T$
23.  $T - \{a\} \subseteq S'$  and hence  $T - \{a\} \in \mathcal{P}(S')$
24.  $T - \{a\} \in \text{PowerSet}(S') = P'$  by the I.H.
25.  $(T - \{a\}) \cup \{a\} = T$  is added to  $P$  in the for loop
26.  $T \in \text{PowerSet}(S)$
27. In each case ,  $T \in \text{PowerSet}(S)$
28.  $\mathcal{P}(S) \subseteq \text{PowerSet}(S)$
29.  $\text{PowerSet}(S) = \mathcal{P}(S)$