Recursion

CSCI 111

Recursion

Breaking big problems into smaller problems.

- Putting a stack of tests in alphabetical order.

Factorial Example

```
5! = 5 * 4 * 3 * 2 * 1 = 120

x! = x * (x-1) * (x-2) * … * 2 * 1
```

Factorial Example

```
public int factorial(int x)
{






}
```

Factorial Example

```
public int factorial(int x)
{
    int num = 1;
    for (                      )
    {

    }
    return num;
}
```

Factorial Example

```
public int factorial(int x)
{
    int num = 1;
    for (int i = 2; i <= x; i++)
    {

    }
    return num;
}
```

Factorial Example

```
public int factorial(int x)
{
    int num = 1;
    for (int i = 2; i <= x; i++)
    {
        num *= i;
    }
    return num;
}
```

factorial(3)

```
public int factorial(int x)
{
    int num = 1;
    for (int i = 2; i <= x; i++)
    {
        num *= i;
    }
    return num;
}
```

x = 3

factorial(3)

```
public int factorial(int x)
{
    int num = 1;
    for (int i = 2; i <= x; i++)
    {
        num *= i;
    }
    return num;
}
```

x = 3

num = 1

factorial(3)

```
public int factorial(int x)
{
    int num = 1;
    for (int i = 2; i <= x; i++)
    {
        num *= i;
    }
    return num;
}
```

x = 3

num = 1

i = 2

factorial(3)

```
public int factorial(int x)
{
    int num = 1;
    for (int i = 2; i <= x; i++)
    {
        num *= i;
    }
    return num;
}
```

x = 3

num = 1

i = 2 ≤ 3 ?

factorial(3)

```
public int factorial(int x)
{
    int num = 1;
    for (int i = 2; i <= x; i++)
    {
        num *= i;
    }
    return num;
}
```

x = 3
num = 2
i = 2

factorial(3)

```
public int factorial(int x)
{
    int num = 1;
    for (int i = 2; i <= x; i++)
    {
        num *= i;
    }
    return num;
}
```

x = 3

num = 2

i = 3

factorial(3)

```
public int factorial(int x)
{
    int num = 1;
    for (int i = 2; i <= x; i++)
    {
        num *= i;
    }
    return num;
}
```

x = 3

num = 2

i = 3 ≤ 3 ?

factorial(3)

```
public int factorial(int x)
{
    int num = 1;
    for (int i = 2; i <= x; i++)
    {
        num *= i;
    }
    return num;
}
```

x = 3

num = 6

i = 3

factorial(3)

```
public int factorial(int x)
{
    int num = 1;
    for (int i = 2; i <= x; i++)
    {
        num *= i;
    }
    return num;
}
```

x = 3

num = 6

i = 4

factorial(3)

```
public int factorial(int x)
{
    int num = 1;
    for (int i = 2; i <= x; i++)
    {
        num *= i;
    }
    return num;
}
```

x = 3

num = 6

i = 4 ≤ 3 ?

factorial(3)

```
public int factorial(int x)
{
    int num = 1;
    for (int i = 2; i <= x; i++)
    {
        num *= i;
    }
    return num;
}
```

x = 3

num = 6

i = 4

Factorial Using Recursion

```
5! = 5 * 4 * 3 * 2 * 1 = 120
```

- Need to identify a smaller problem

```
5! = 5 * (4 * 3 * 2 * 1) = 120
```

Factorial Using Recursion

Problem 1:

    5 * Problem 2

Problem 2:

    4 * 3 * 2 * 1

Factorial Using Recursion

Problem 1:

5 * Problem 2

Problem 2:

4 * Problem 3

Problem 3:

3 * 2 * 1

Factorial Using Recursion

Problem 1:

    5 * Problem 2

Problem 2:

    4 * Problem 3

Problem 3:

    3 * Problem 4

Problem 4:

    2 * 1

Factorial Using Recursion

Problem 1:

5 * Problem 2

Problem 2:

4 * Problem 3

Problem 3:

3 * Problem 4

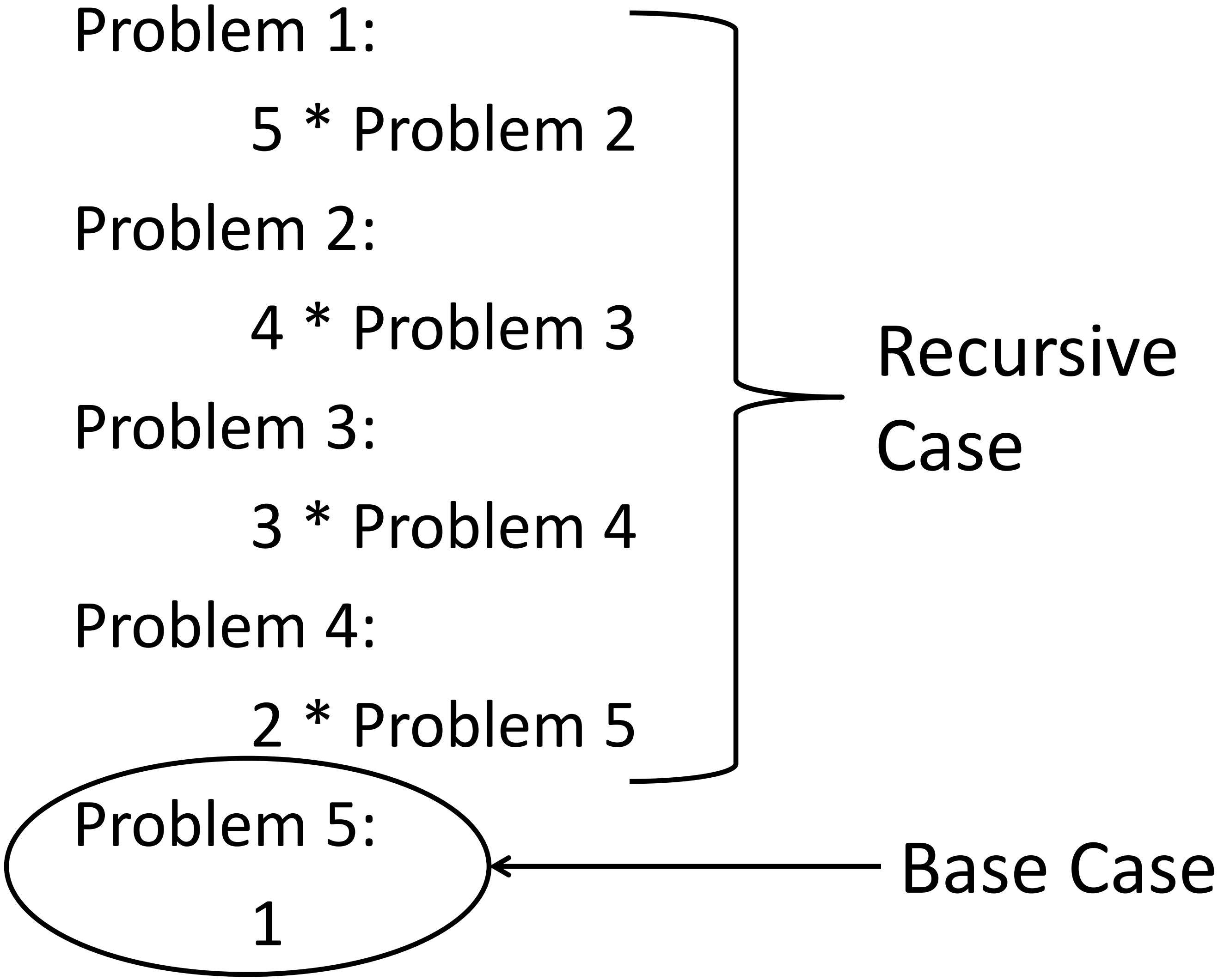
Problem 4:

2 * Problem 5

Problem 5:

1

Factorial Using Recursion



Problem 1:

5 * Problem 2

Problem 2:

4 * Problem 3

Problem 3:

3 * Problem 4

Problem 4:

2 * Problem 5

Problem 5:

1

Recursive Case

Base Case

Factorial Using Recursion

```
5! = 5 * 4 * 3 * 2 * 1 = 120
```

- Need to identify a smaller problem

```
5! = 5 * (4 * 3 * 2 * 1) = 120
5! = 5 * factorial(4) = 120
```

Factorial Using Recursion

Problem 1:

5 * Problem 2

Problem 2:

4 * Problem 3

Problem 3:

3 * Problem 4

Problem 4:

2 * Problem 5

Problem 5:

1

Factorial Using Recursion

```
factorial(5):
    5 * factorial(4)
factorial(4):
    4 * factorial(3)
factorial(3):
    3 * factorial(2)
factorial(2):
    2 * factorial(1)
factorial(1):
    1
```

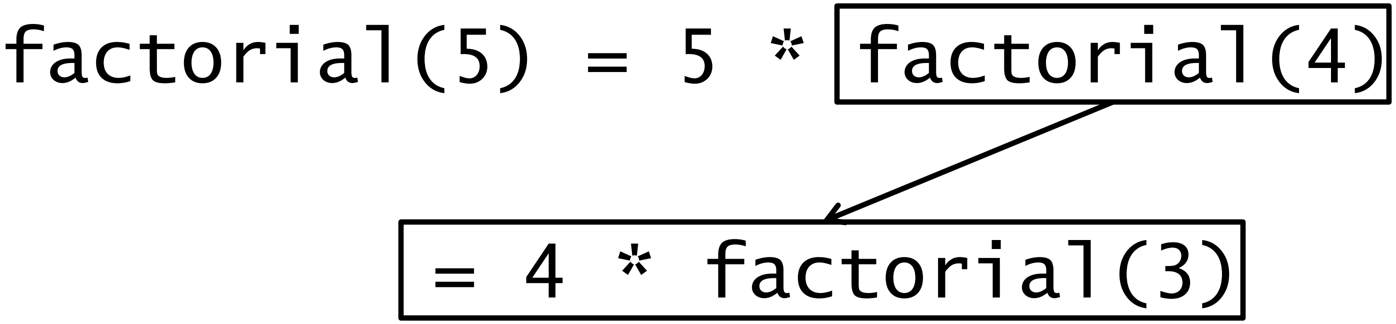
Factorial Using Recursion

```
public int factorial(int x)
{
    if (x == 1)
    {
        return 1;
    }
    else
    {
        return x * factorial(x - 1);
    }
}
```

factorial(5)

```
factorial(5) = 5 * factorial(4)
```

factorial(5)

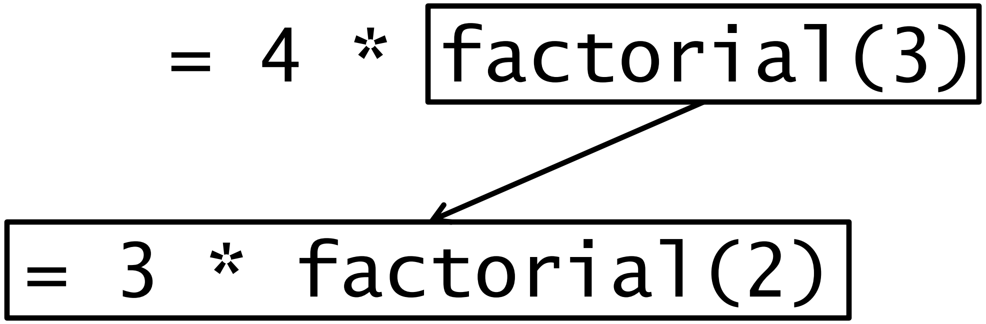


factorial(5) = 5 * factorial(4)

= 4 * factorial(3)

factorial(5)

factorial(5) = 5 * factorial(4)



= 4 * factorial(3)

= 3 * factorial(2)

factorial(5)

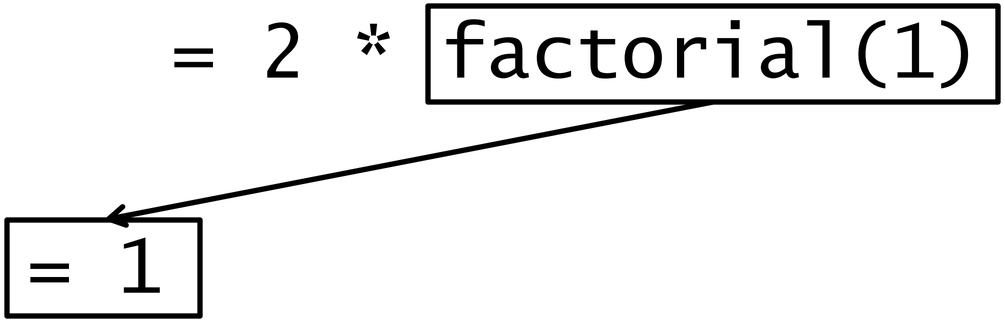
```
factorial(5) = 5 * factorial(4)

               = 4 * factorial(3)

           = 3 * factorial(2)

    = 2 * factorial(1)
```

factorial(5)



```
factorial(5) = 5 * factorial(4)

          = 4 * factorial(3)

      = 3 * factorial(2)

   = 2 * factorial(1)

= 1
```

factorial(5)

factorial(5) = 5 * factorial(4)

= 4 * factorial(3)

= 3 * factorial(2)

= 2 * ~~factorial(1)~~ = 2

= $\boxed{1}$

factorial(5)

factorial(5) = 5 * factorial(4)

= 4 * factorial(3)

= 3 * ~~factorial(2)~~ = 6

= 2 * ~~factorial(1)~~ = $\boxed{2}$

factorial(5)

factorial(5) = 5 * factorial(4)

= 4 * ~~factorial(3)~~ = 24

= 3 * ~~factorial(2)~~ = $\boxed{6}$

factorial(5)

factorial(5) = 5 * ~~factorial(4)~~ = 120

= 4 * ~~factorial(3)~~ = $\boxed{24}$

factorial(5)

```
factorial(5) = 120
```

Recursion – Part 2

In last week's episode…

Breaking big problems into smaller versions of the same problem.

Recursion – Part 2

In last week’s episode…

Breaking big problems into smaller versions of the same problem.

Factorial:

$$\text{factorial}(x) = \begin{cases} 1 & \text{if } x = 1 \\ x * \text{factorial}(x\text{-}1) & \text{otherwise} \end{cases}$$

```
if (baseCase)
{
        //code
}
else
{
        //recursiveCall
}
```

Finding Max in Array

Goal: Find max of

0	1	2	3	4
1	7	19	3	11

Finding Max in Array

Goal: Find max of

0	1	2	3	4
1	7	19	3	11

Solution 1 (Iteratively):

```
int max = 0;
for (int i = 0; i < array.length; i++)
{
      if (array[i] > max)
      {
            max = array[i];
      }
}
```

Finding Max in Array

Goal: Find max of

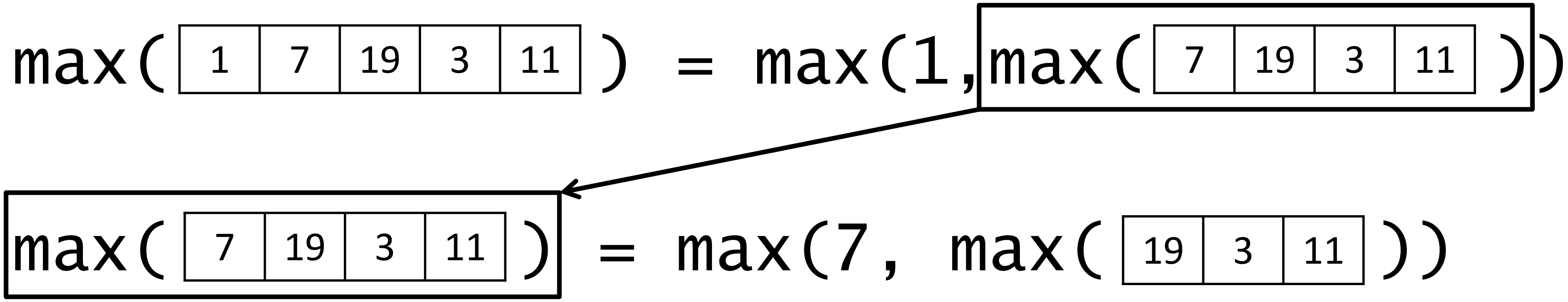
Solution 1 (Recursively):

$$\mathtt{max(}\begin{array}{|c|c|c|c|c|} \multicolumn{1}{c}{0} & \multicolumn{1}{c}{1} & \multicolumn{1}{c}{2} & \multicolumn{1}{c}{3} & \multicolumn{1}{c}{4} \\ \hline 1 & 7 & 19 & 3 & 11 \\ \hline \end{array}\mathtt{)\ =\ max(1,max(}\begin{array}{|c|c|c|c|} \multicolumn{1}{c}{1} & \multicolumn{1}{c}{2} & \multicolumn{1}{c}{3} & \multicolumn{1}{c}{4} \\ \hline 7 & 19 & 3 & 11 \\ \hline \end{array}\mathtt{))}$$

Finding Max in Array

$$\texttt{max(}\begin{array}{|c|c|c|c|c|}\hline 1 & 7 & 19 & 3 & 11 \\ \hline\end{array}\texttt{) = max(1,max(}\begin{array}{|c|c|c|c|}\hline 7 & 19 & 3 & 11 \\ \hline\end{array}\texttt{))}$$

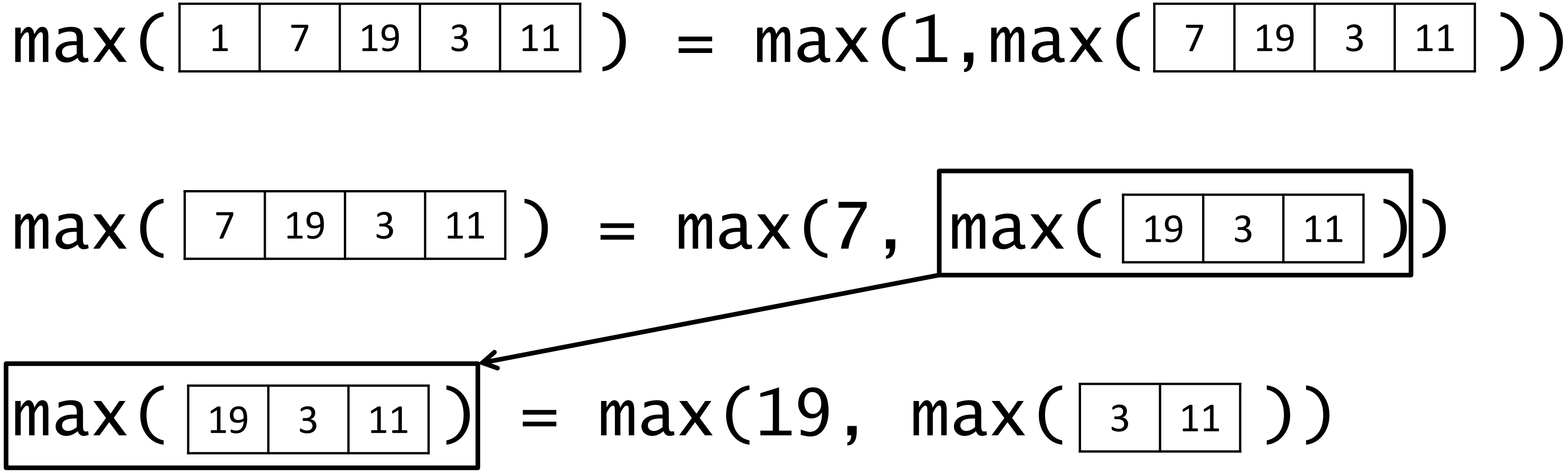
Finding Max in Array



```
max( 1 | 7 | 19 | 3 | 11 ) = max(1, max( 7 | 19 | 3 | 11 ))

max( 7 | 19 | 3 | 11 ) = max(7, max( 19 | 3 | 11 ))
```

Finding Max in Array

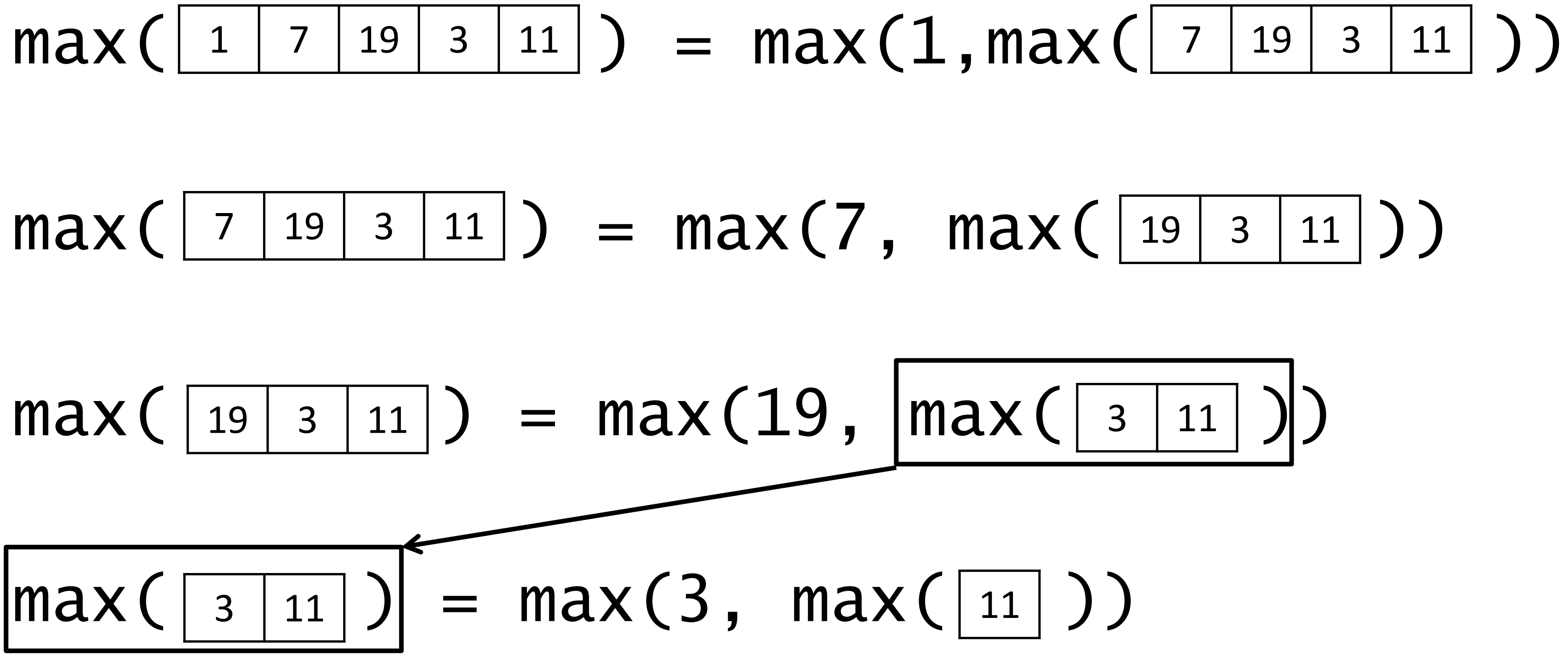


max(1 | 7 | 19 | 3 | 11) = max(1,max(7 | 19 | 3 | 11))

max(7 | 19 | 3 | 11) = max(7, max(19 | 3 | 11))

max(19 | 3 | 11) = max(19, max(3 | 11))

Finding Max in Array



max(1 7 19 3 11) = max(1,max(7 19 3 11))

max(7 19 3 11) = max(7, max(19 3 11))

max(19 3 11) = max(19, max(3 11))

max(3 11) = max(3, max(11))

Finding Max in Array

$\texttt{max(}\boxed{1}\boxed{7}\boxed{19}\boxed{3}\boxed{11}\texttt{) = max(1,max(}\boxed{7}\boxed{19}\boxed{3}\boxed{11}\texttt{))}$

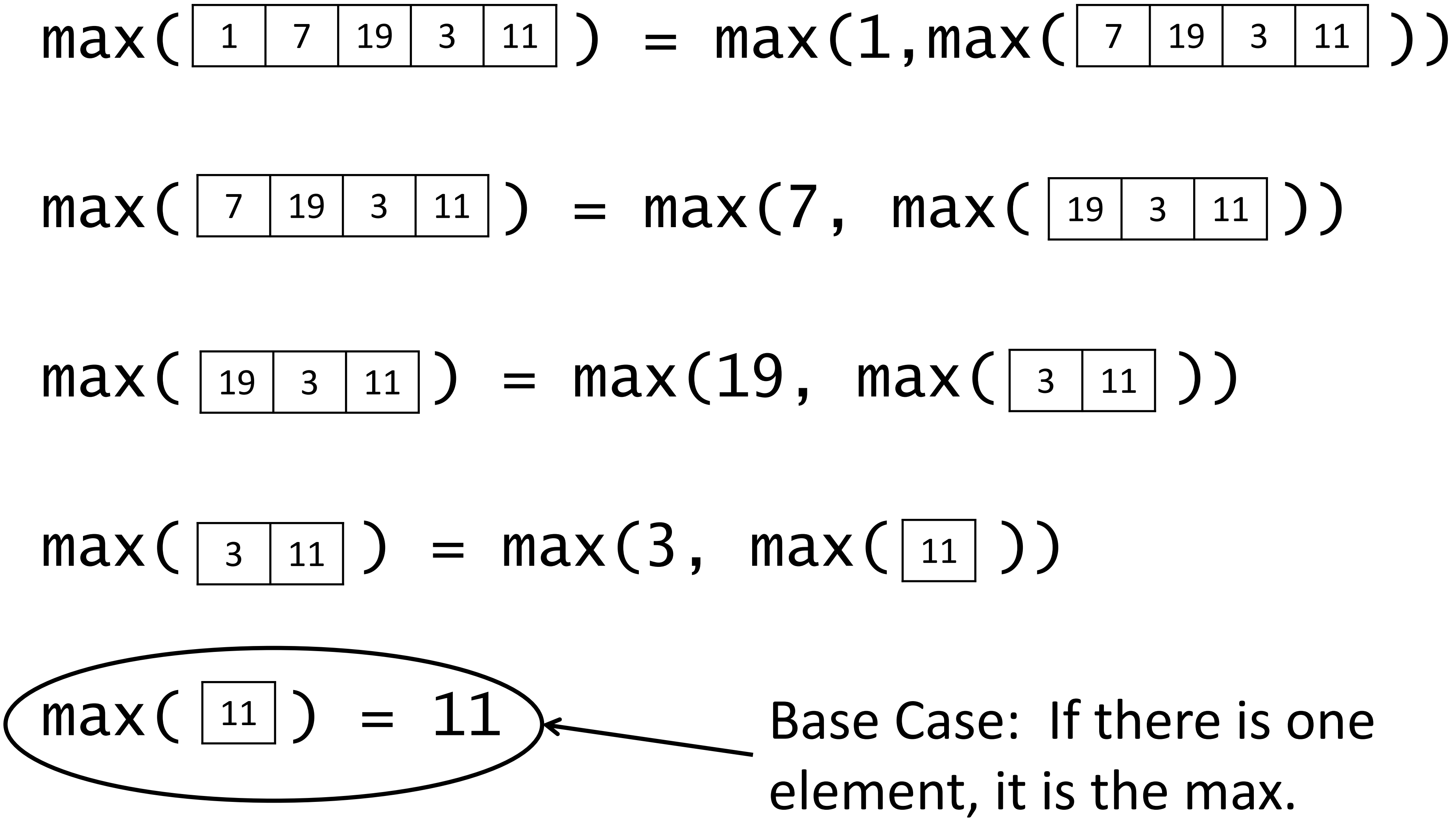
$\texttt{max(}\boxed{7}\boxed{19}\boxed{3}\boxed{11}\texttt{) = max(7, max(}\boxed{19}\boxed{3}\boxed{11}\texttt{))}$

$\texttt{max(}\boxed{19}\boxed{3}\boxed{11}\texttt{) = max(19, max(}\boxed{3}\boxed{11}\texttt{))}$

$\texttt{max(}\boxed{3}\boxed{11}\texttt{) = max(3, }\boxed{\texttt{max(}\boxed{11}\texttt{)}}\texttt{)}$

$\boxed{\texttt{max(}\boxed{11}\texttt{)}}\texttt{ = 11}$

Finding Max in Array



max(1 | 7 | 19 | 3 | 11) = max(1,max(7 | 19 | 3 | 11))

max(7 | 19 | 3 | 11) = max(7, max(19 | 3 | 11))

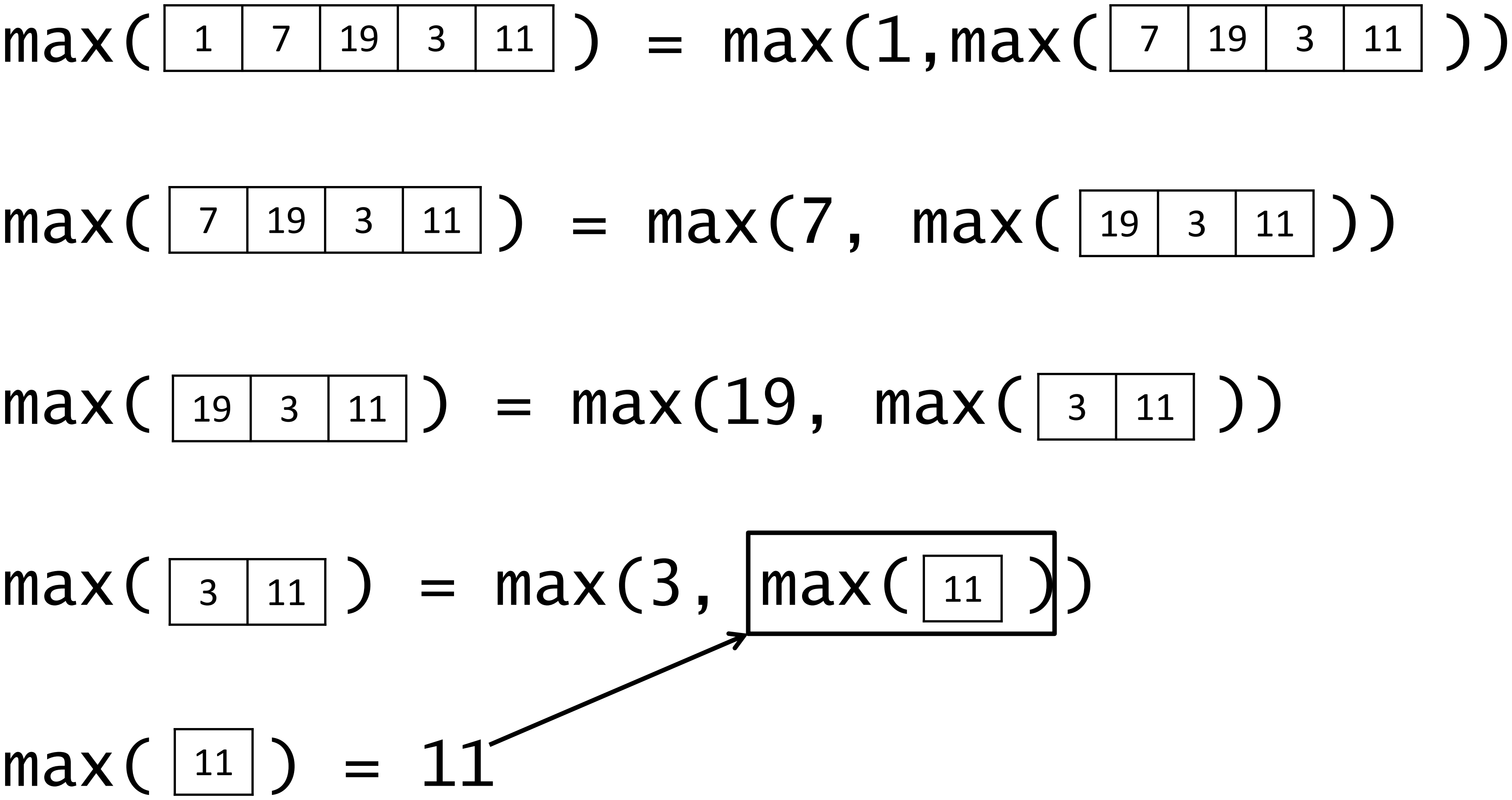
max(19 | 3 | 11) = max(19, max(3 | 11))

max(3 | 11) = max(3, max(11))

max(11) = 11

Base Case: If there is one element, it is the max.

Finding Max in Array



max(1 7 19 3 11) = max(1,max(7 19 3 11))

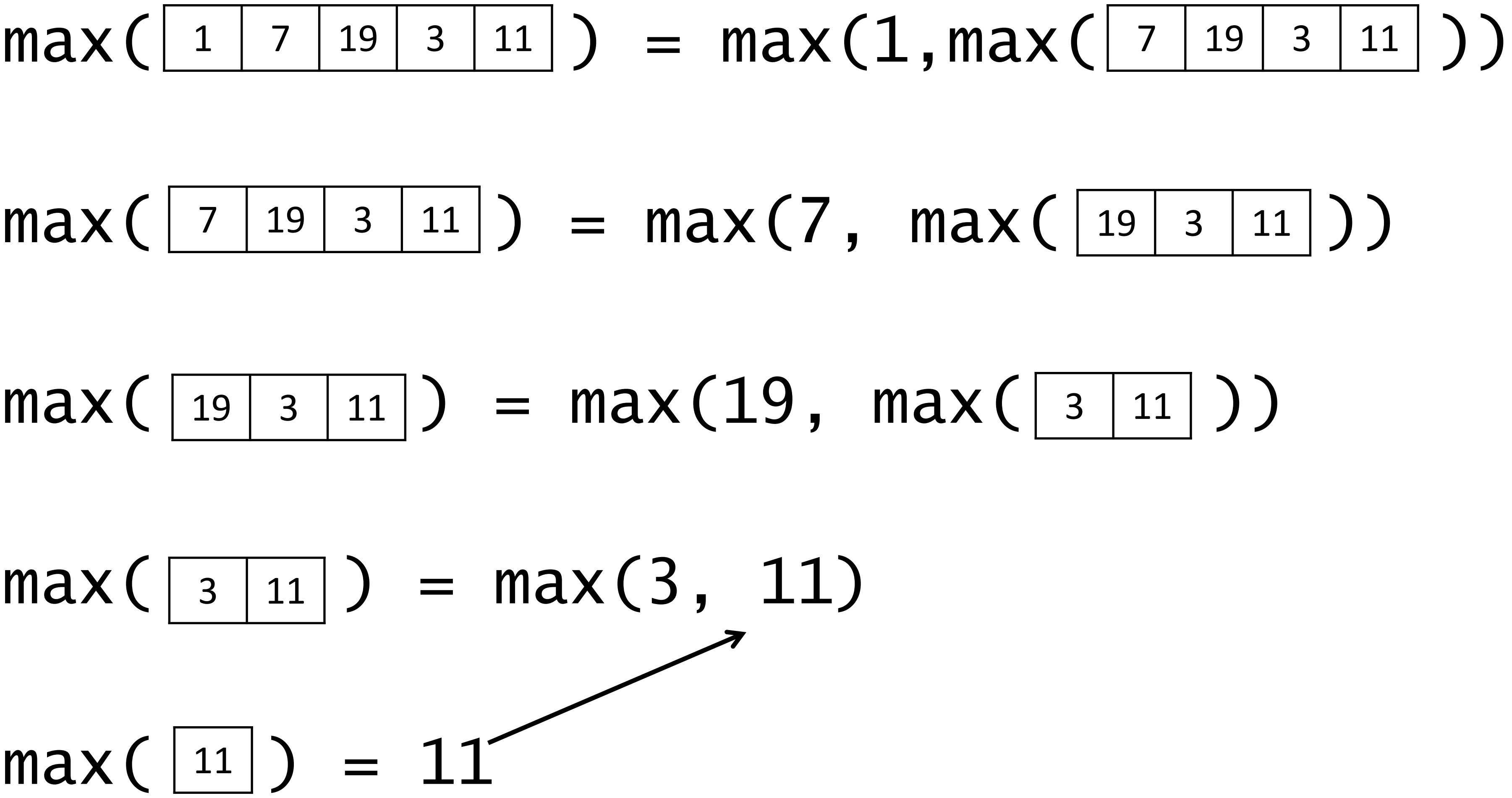
max(7 19 3 11) = max(7, max(19 3 11))

max(19 3 11) = max(19, max(3 11))

max(3 11) = max(3, max(11))

max(11) = 11

Finding Max in Array



$$\texttt{max(}\boxed{1}\boxed{7}\boxed{19}\boxed{3}\boxed{11}\texttt{) = max(1,max(}\boxed{7}\boxed{19}\boxed{3}\boxed{11}\texttt{))}$$

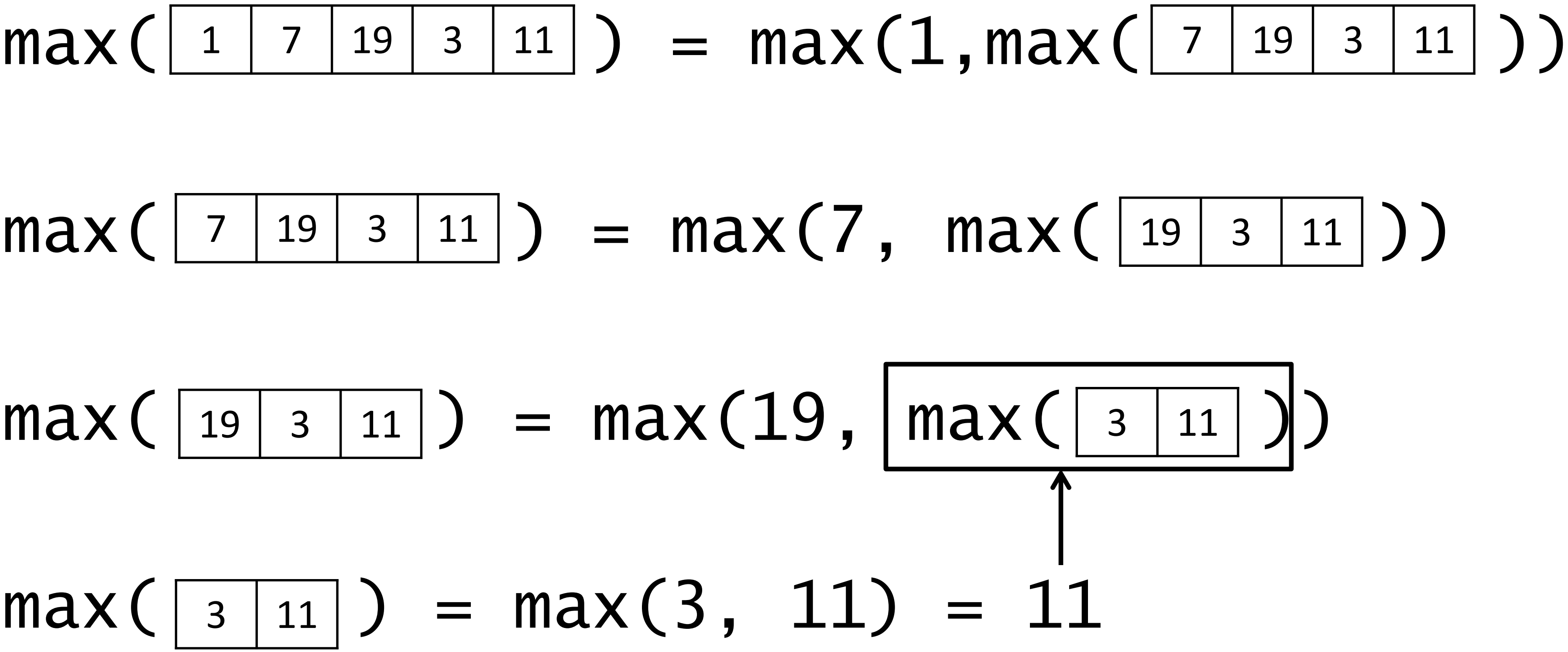
$$\texttt{max(}\boxed{7}\boxed{19}\boxed{3}\boxed{11}\texttt{) = max(7, max(}\boxed{19}\boxed{3}\boxed{11}\texttt{))}$$

$$\texttt{max(}\boxed{19}\boxed{3}\boxed{11}\texttt{) = max(19, max(}\boxed{3}\boxed{11}\texttt{))}$$

$$\texttt{max(}\boxed{3}\boxed{11}\texttt{) = max(3, 11)}$$

$$\texttt{max(}\boxed{11}\texttt{) = 11}$$

Finding Max in Array



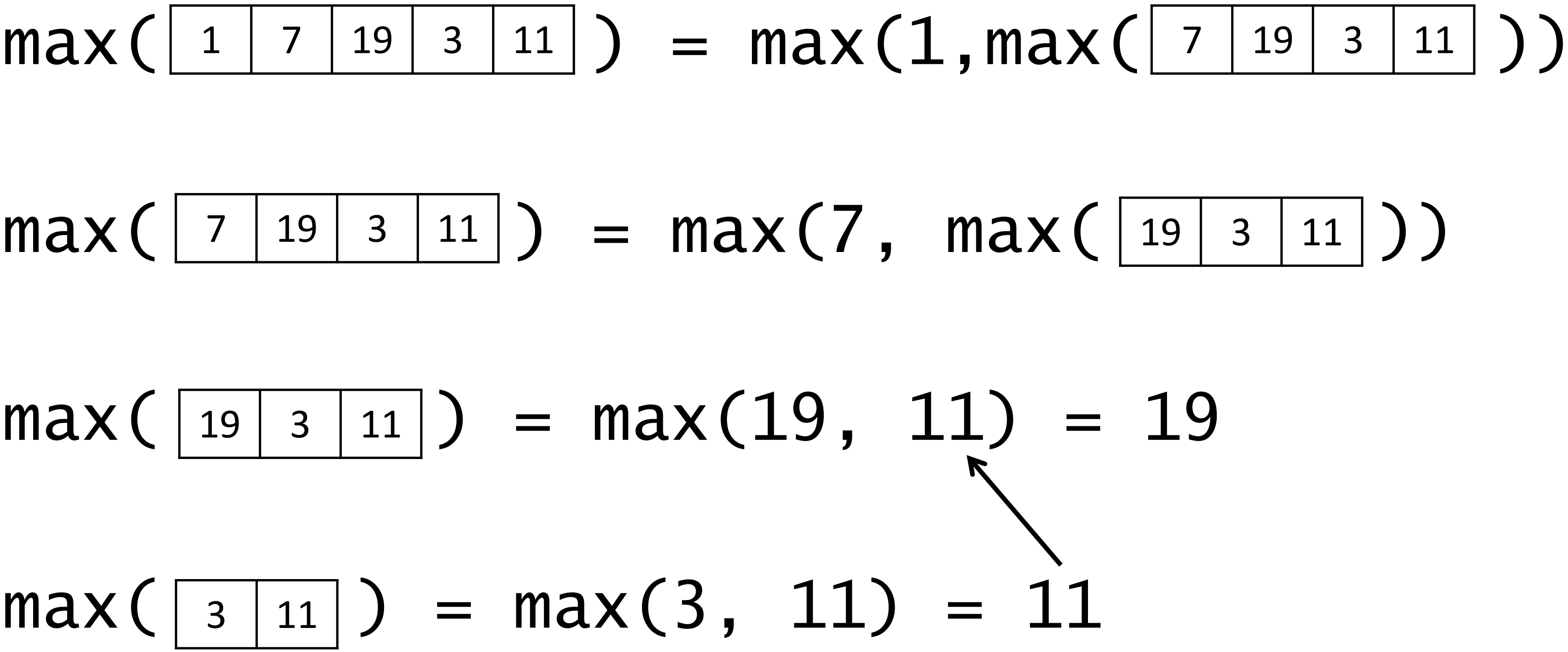
$$\mathtt{max}(\boxed{1}\boxed{7}\boxed{19}\boxed{3}\boxed{11}) = \mathtt{max}(1,\mathtt{max}(\boxed{7}\boxed{19}\boxed{3}\boxed{11}))$$

$$\mathtt{max}(\boxed{7}\boxed{19}\boxed{3}\boxed{11}) = \mathtt{max}(7,\ \mathtt{max}(\boxed{19}\boxed{3}\boxed{11}))$$

$$\mathtt{max}(\boxed{19}\boxed{3}\boxed{11}) = \mathtt{max}(19,\ \boxed{\mathtt{max}(\boxed{3}\boxed{11})})$$

$$\mathtt{max}(\boxed{3}\boxed{11}) = \mathtt{max}(3,\ 11) = 11$$

Finding Max in Array



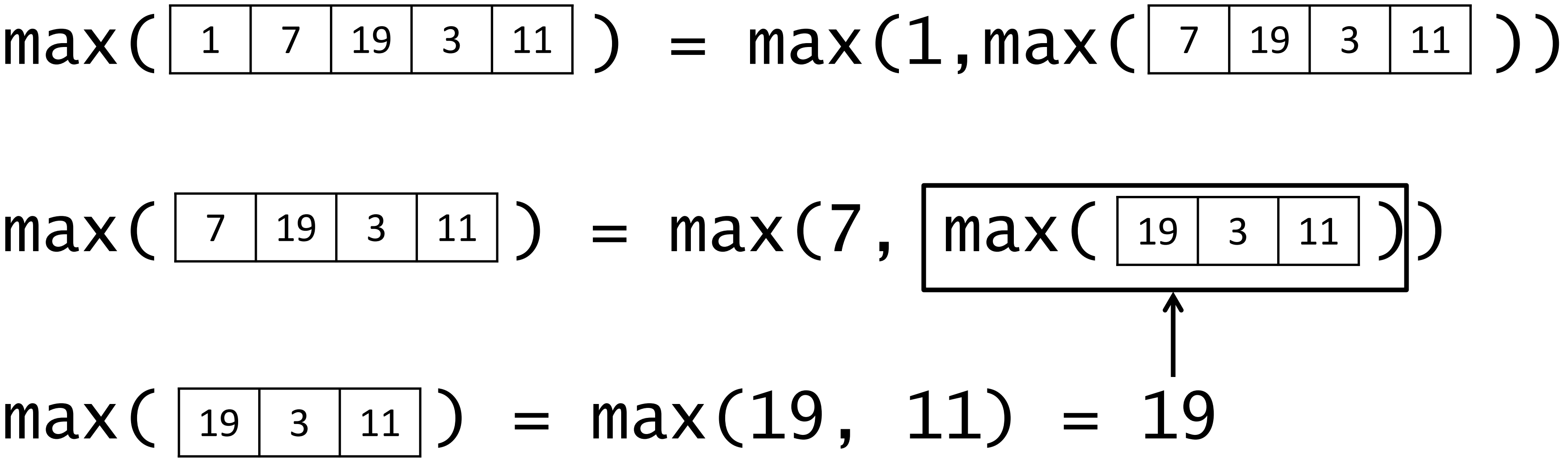
max(1 7 19 3 11) = max(1,max(7 19 3 11))

max(7 19 3 11) = max(7, max(19 3 11))

max(19 3 11) = max(19, 11) = 19

max(3 11) = max(3, 11) = 11

Finding Max in Array

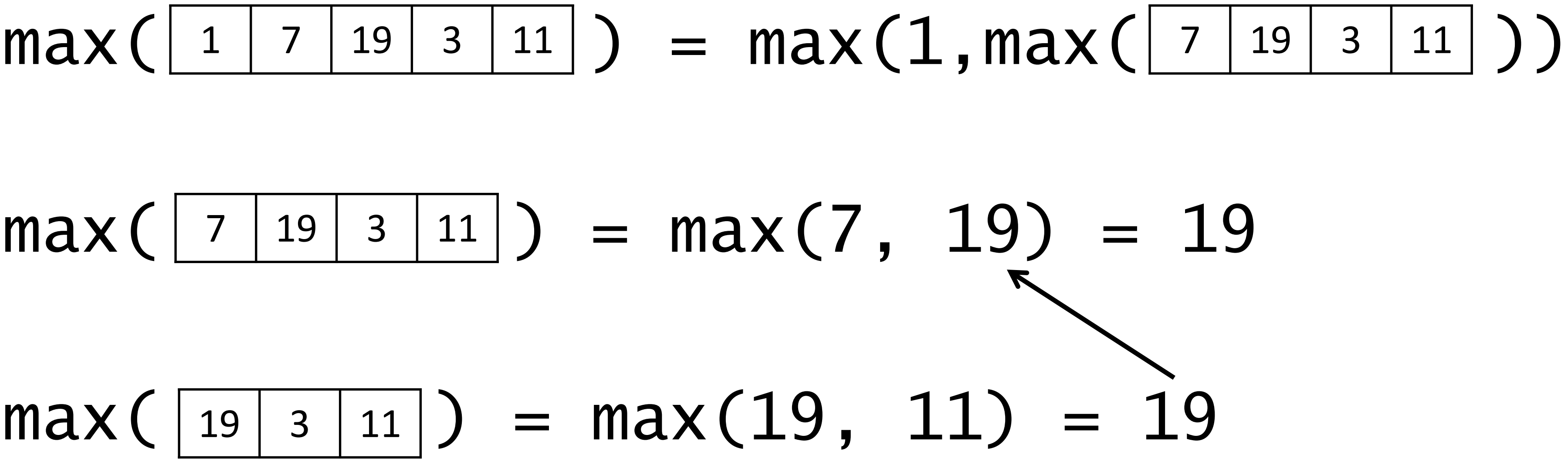


$$\texttt{max(}\begin{array}{|c|c|c|c|c|}\hline 1 & 7 & 19 & 3 & 11 \\ \hline\end{array}\texttt{) = max(1,max(}\begin{array}{|c|c|c|c|}\hline 7 & 19 & 3 & 11 \\ \hline\end{array}\texttt{))}$$

$$\texttt{max(}\begin{array}{|c|c|c|c|}\hline 7 & 19 & 3 & 11 \\ \hline\end{array}\texttt{) = max(7, }\boxed{\texttt{max(}\begin{array}{|c|c|c|}\hline 19 & 3 & 11 \\ \hline\end{array}\texttt{)}}\texttt{)}$$

$$\texttt{max(}\begin{array}{|c|c|c|}\hline 19 & 3 & 11 \\ \hline\end{array}\texttt{) = max(19, 11) = 19}$$

Finding Max in Array

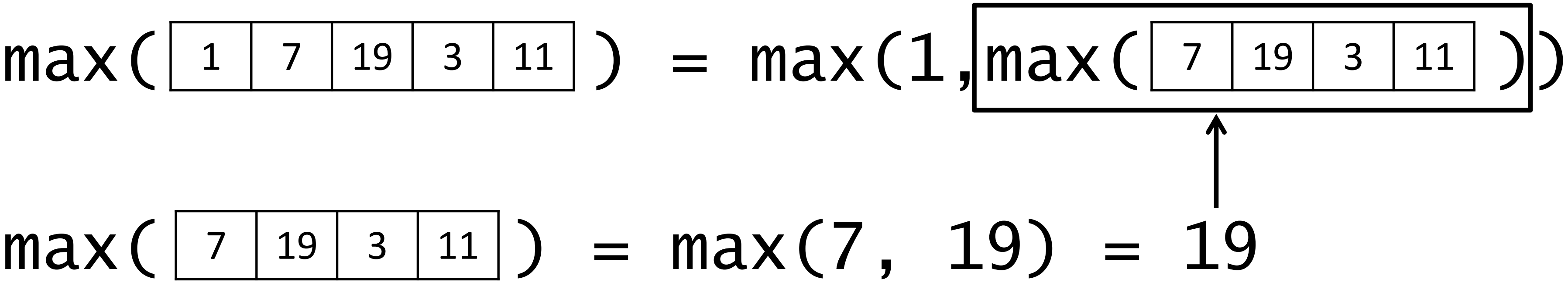


```
max( |1|7|19|3|11| ) = max(1,max( |7|19|3|11| ))

max( |7|19|3|11| ) = max(7, 19) = 19

max( |19|3|11| ) = max(19, 11) = 19
```

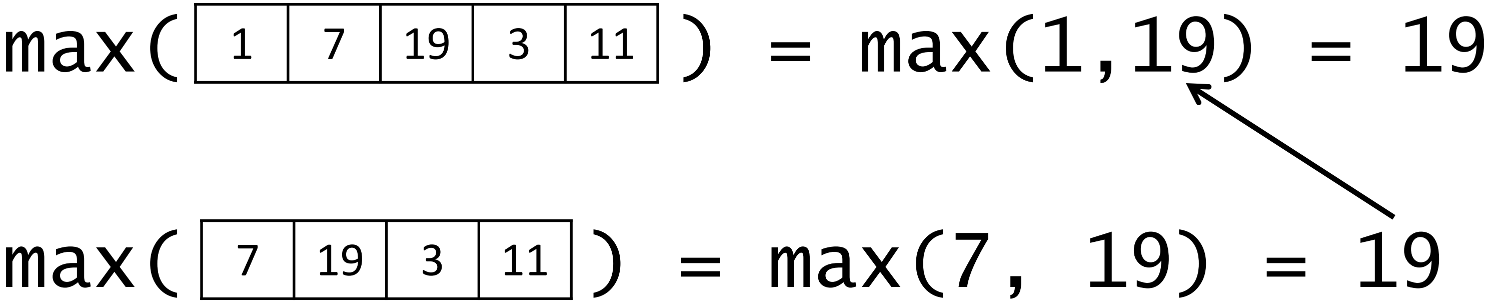
Finding Max in Array



max(1 | 7 | 19 | 3 | 11) = max(1, max(7 | 19 | 3 | 11))

max(7 | 19 | 3 | 11) = max(7, 19) = 19

Finding Max in Array



max(1 | 7 | 19 | 3 | 11) = max(1,19) = 19

max(7 | 19 | 3 | 11) = max(7, 19) = 19

Finding Max in Array

max(

1	7	19	3	11

) = 19

Finding Max in Array

What will the code look like?

- Is it efficient to keep creating new arrays?
- Can we use the same array? How?