# CS302 - Data Structures

## *using C++*

Topic: Recursion with Arrays

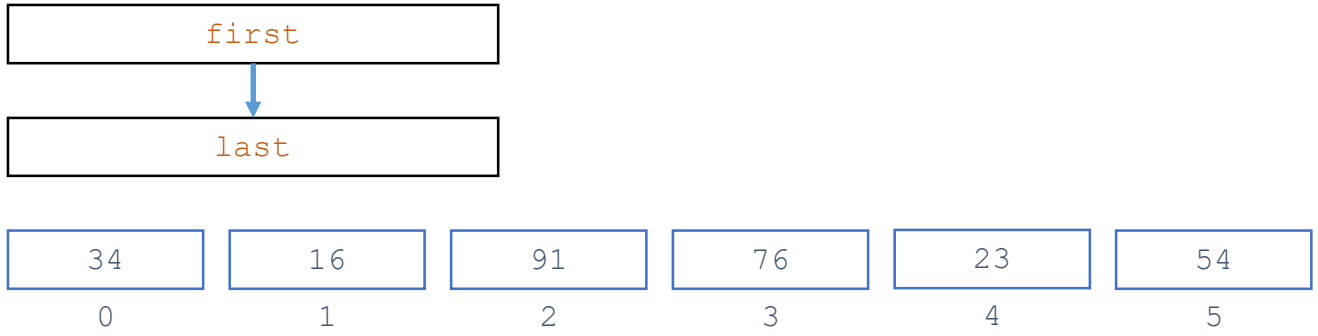
Kostas Alexis

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - **Start with first element**



# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - **Start with first element**


```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - **Start with first element**


```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - **Start with first element**


```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - **Start with first element**


```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

This function has an implicit base case

# Recursive Processing - Arrays

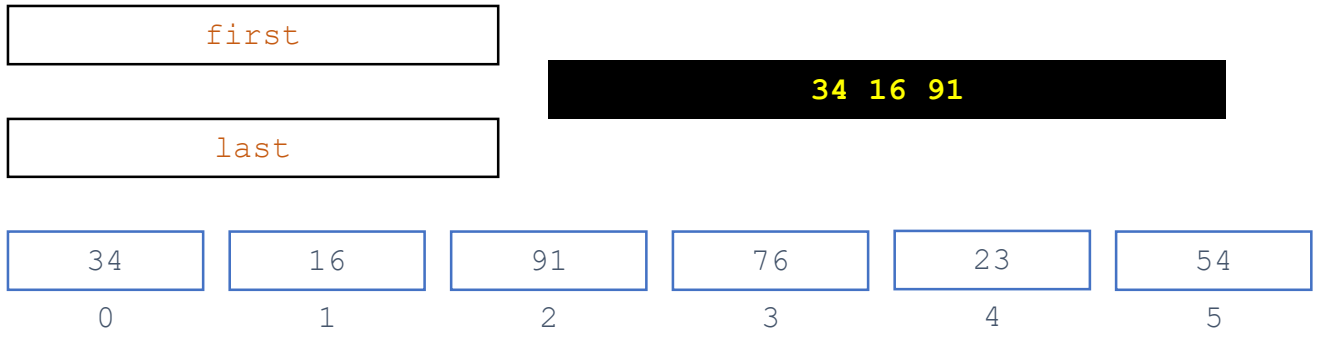
- **Recursively displaying elements stored in an array**
  - **Start with first element**


```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
        displayArray(myArray, first + 1, last);
} // end displayArray
```

This function has an implicit base case

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - **Start with first element**

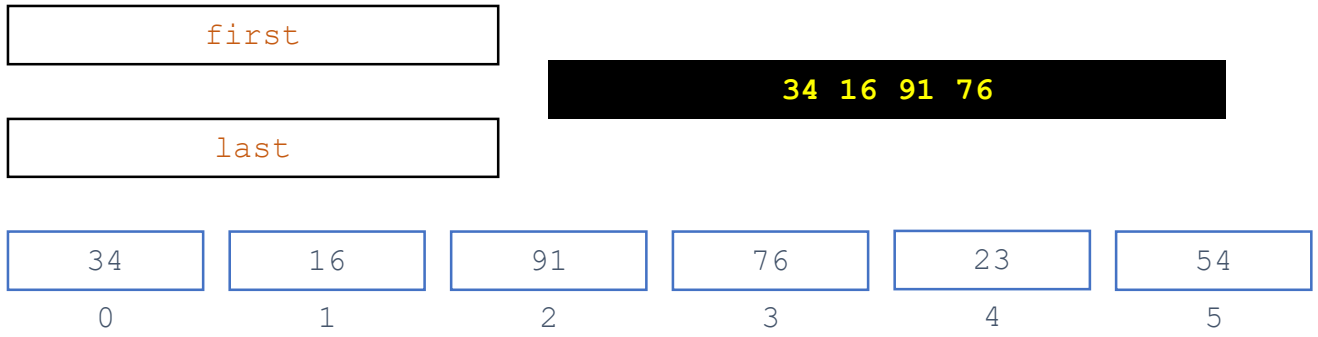



```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
        displayArray(myArray, first + 1, last);
} // end displayArray
```

This function has an implicit base case

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - **Start with first element**

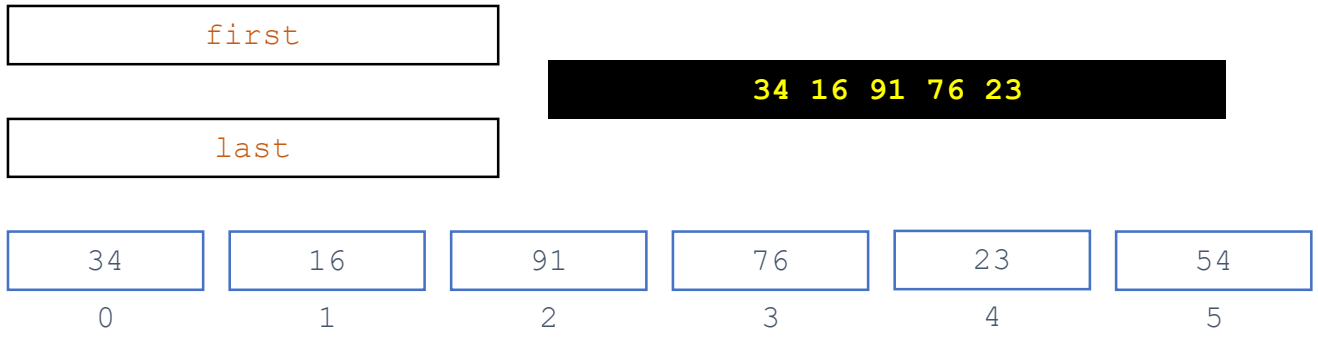



```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
       displayArray(myArray, first + 1, last);
} // end displayArray
```

This function has an implicit base case

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - **Start with first element**

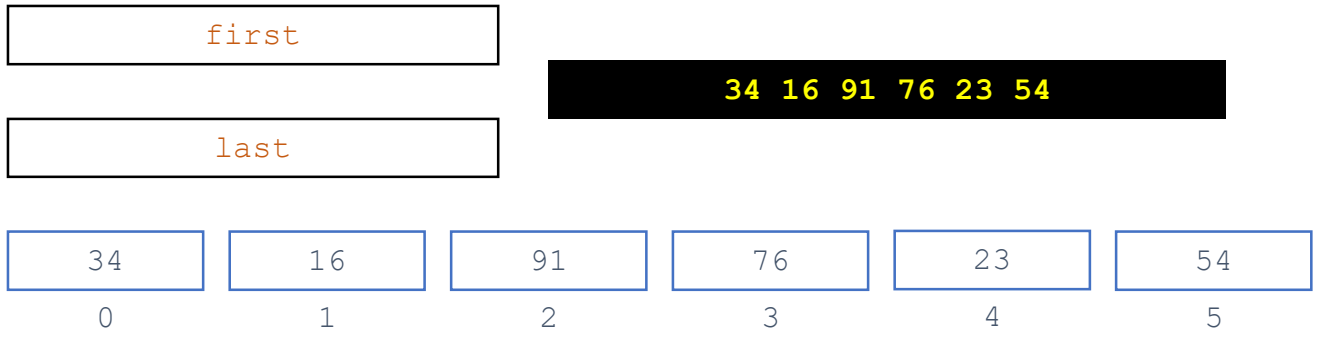



```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
       displayArray(myArray, first + 1, last);
} // end displayArray
```

This function has an implicit base case

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - **Start with first element**

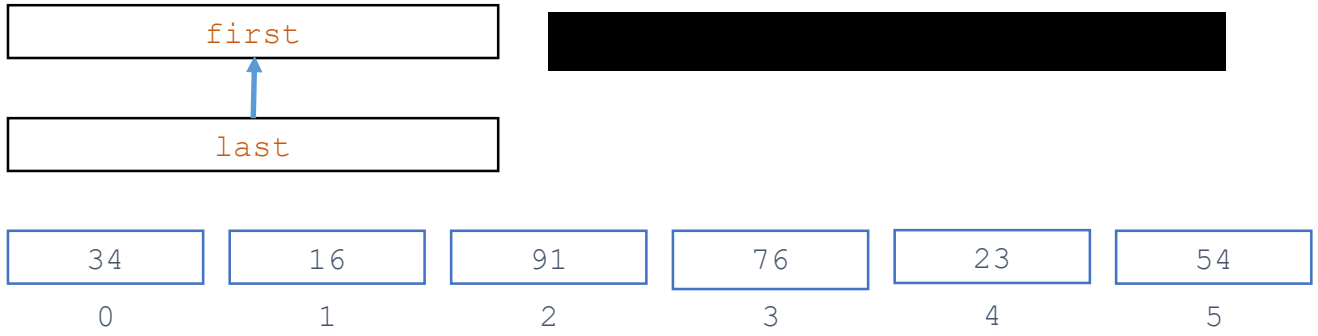



```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
       displayArray(myArray, first + 1, last);
} // end displayArray
```

This function has an implicit base case

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - **Start with last element**

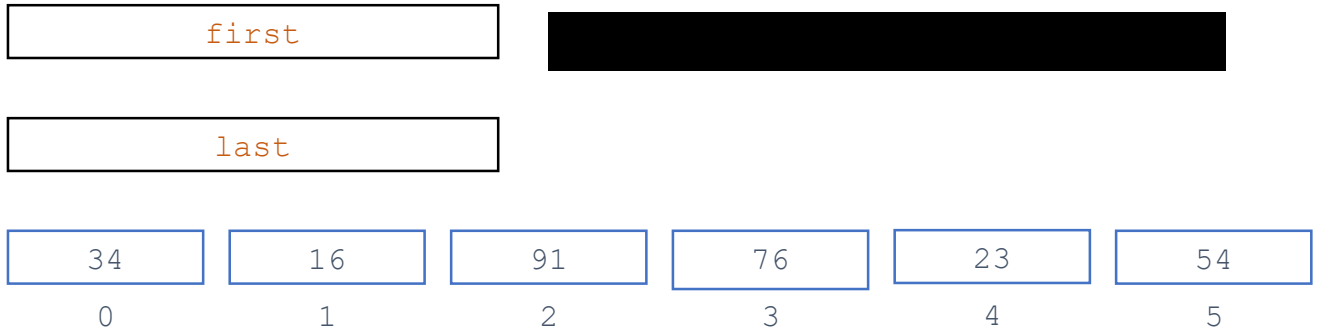



```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - **Start with last element**



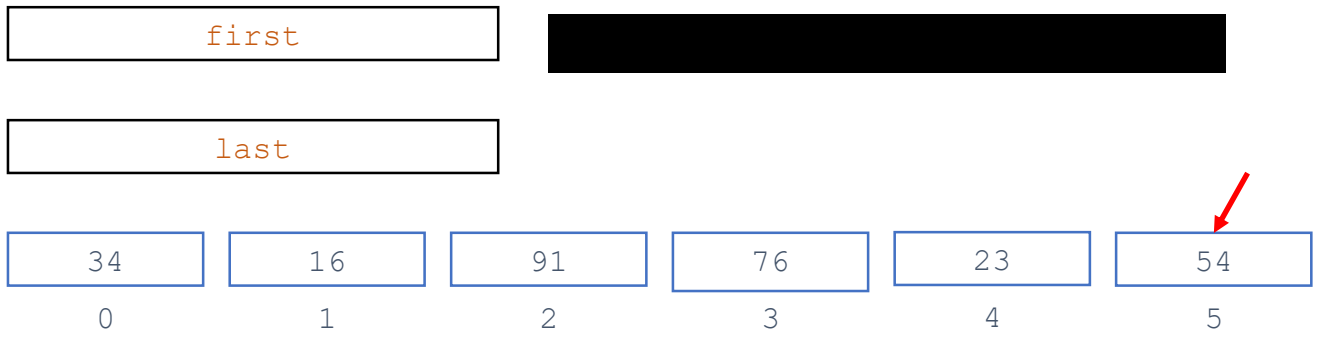

```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

Recursive call prior to display command

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - **Start with last element**



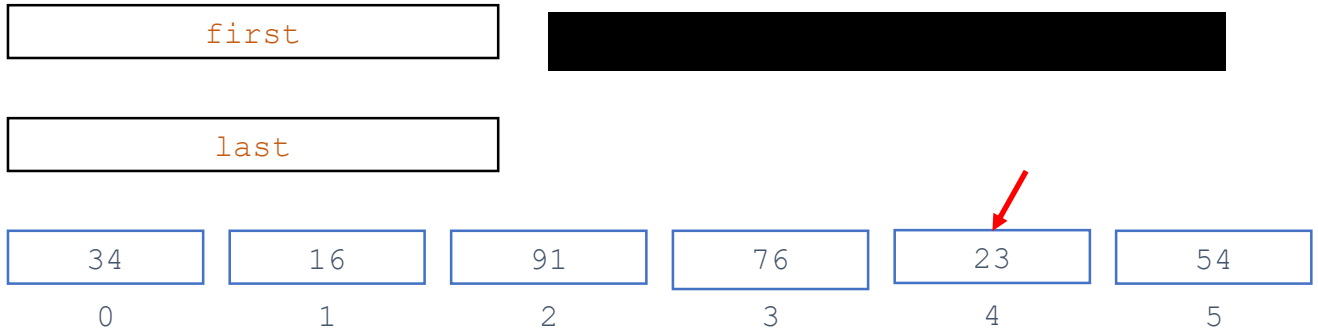

```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

Recursive call prior to display command

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - **Start with last element**



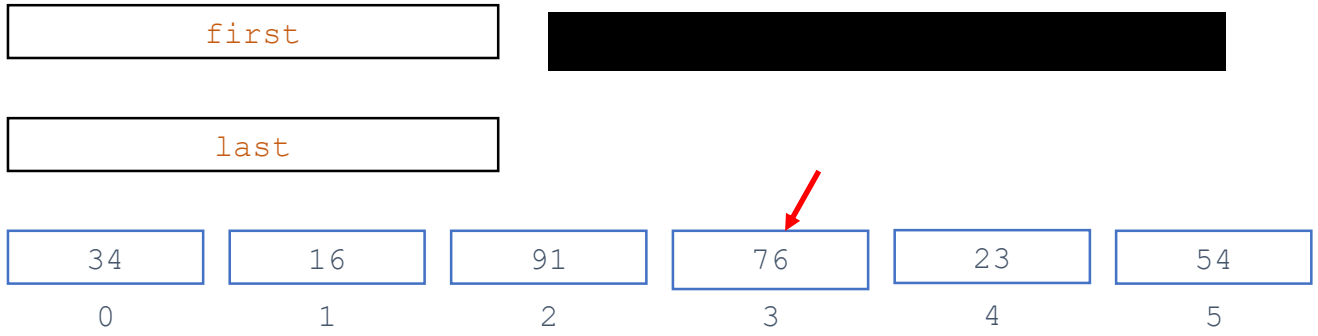

```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

Recursive call prior to display command

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - **Start with last element**



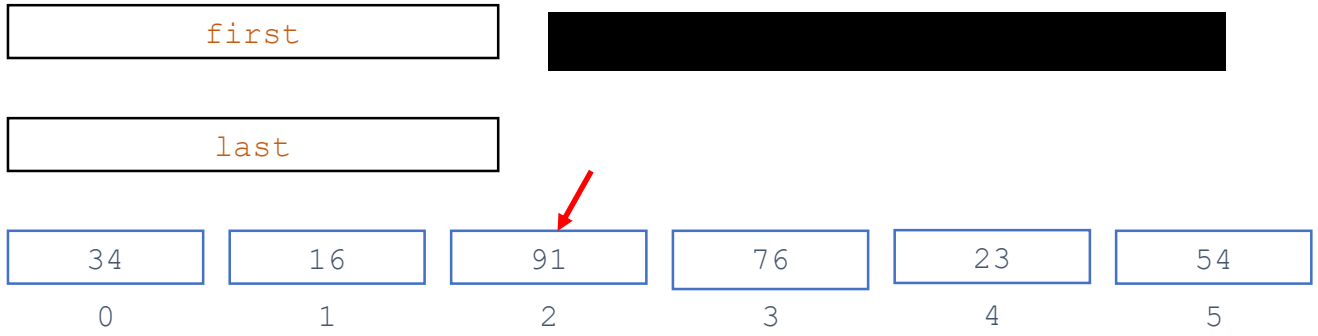

```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

Recursive call prior to display command

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - **Start with last element**



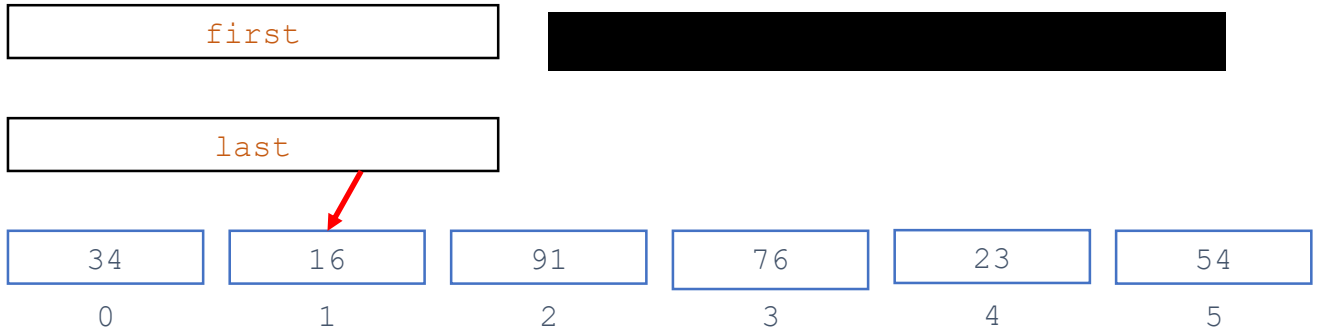

```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

Recursive call prior to display command

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - **Start with last element**



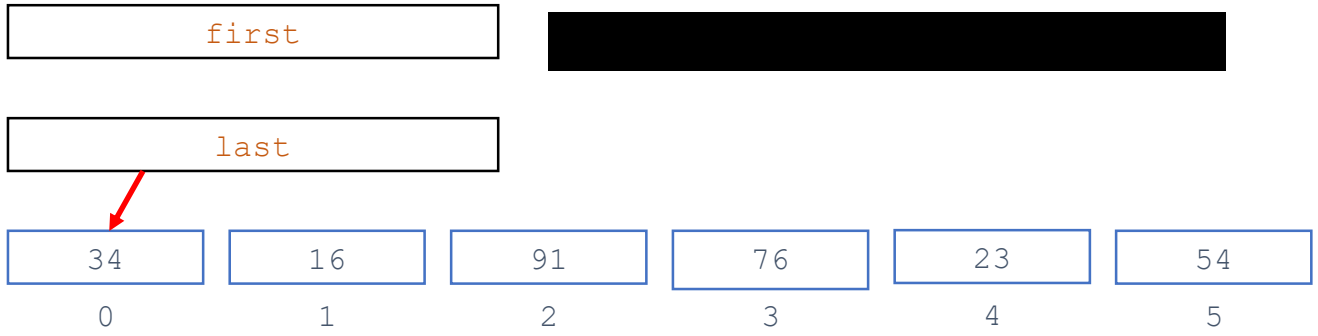

```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

Recursive call prior to display command

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - **Start with last element**



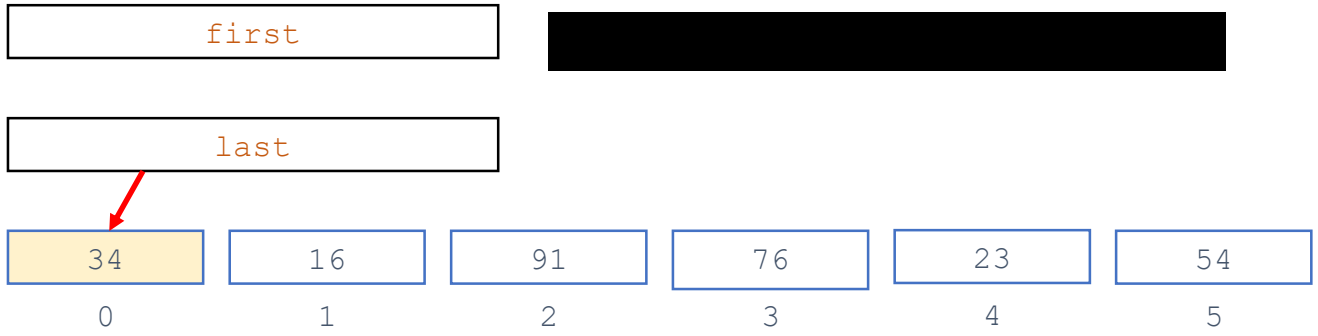

```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

Recursive call prior to display command

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - **Start with last element**




**Base case reached**

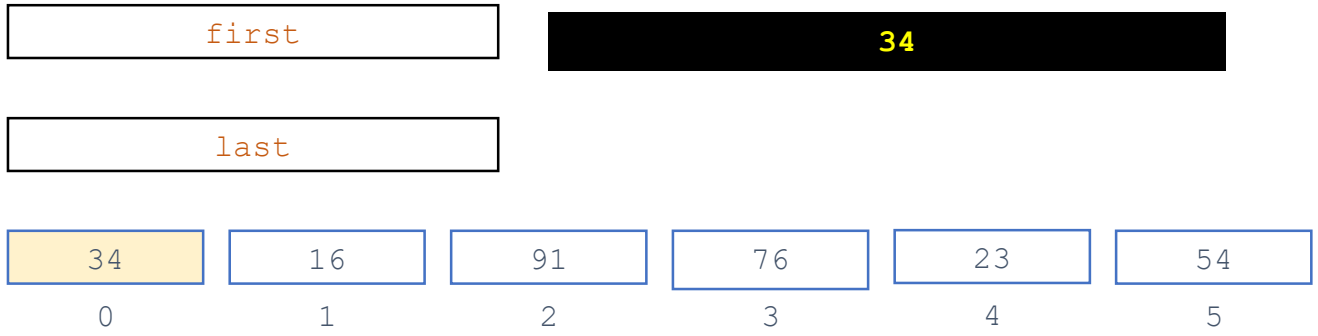
```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

Recursive call prior to display command

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - **Start with last element**



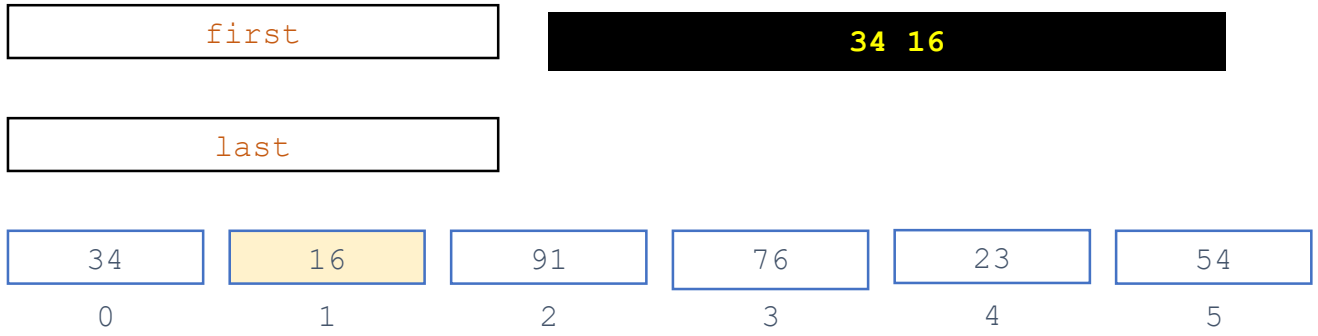

```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

Recursive call prior to display command

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - **Start with last element**



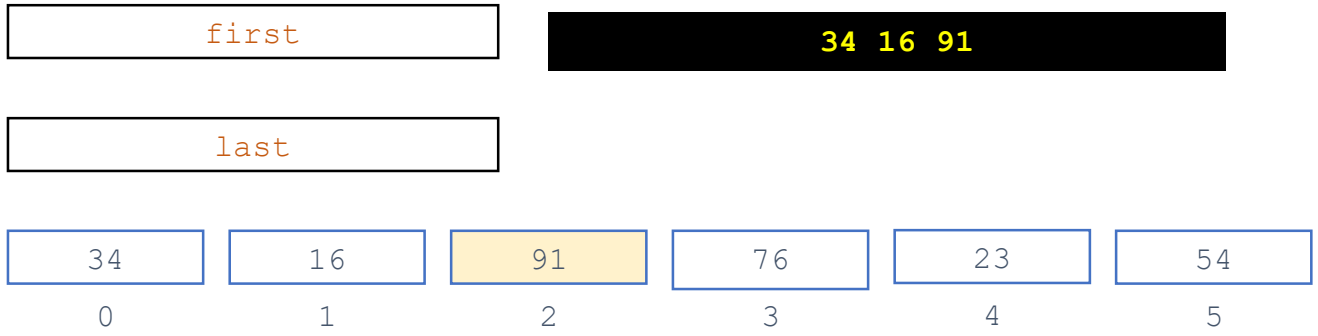

```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

Recursive call prior to display command

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - **Start with last element**



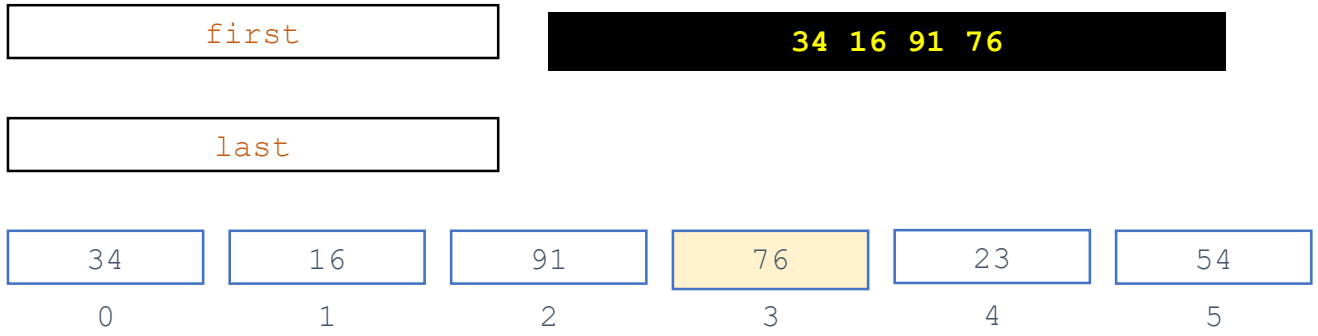

```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

Recursive call prior to display command

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - **Start with last element**



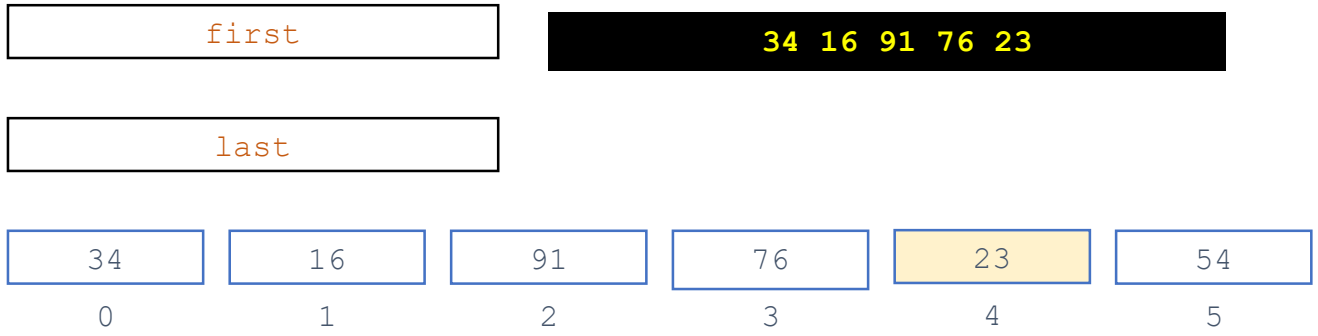

```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

Recursive call prior to display command

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - **Start with last element**



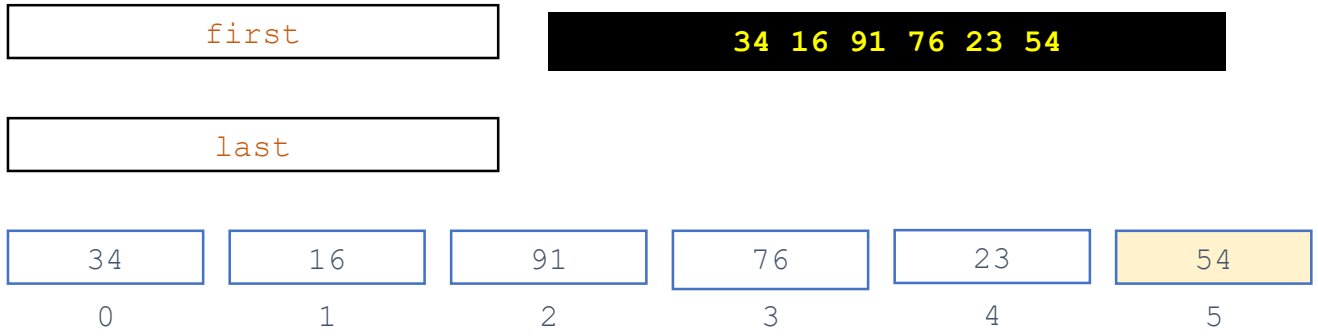

```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

Recursive call prior to display command

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - **Start with last element**



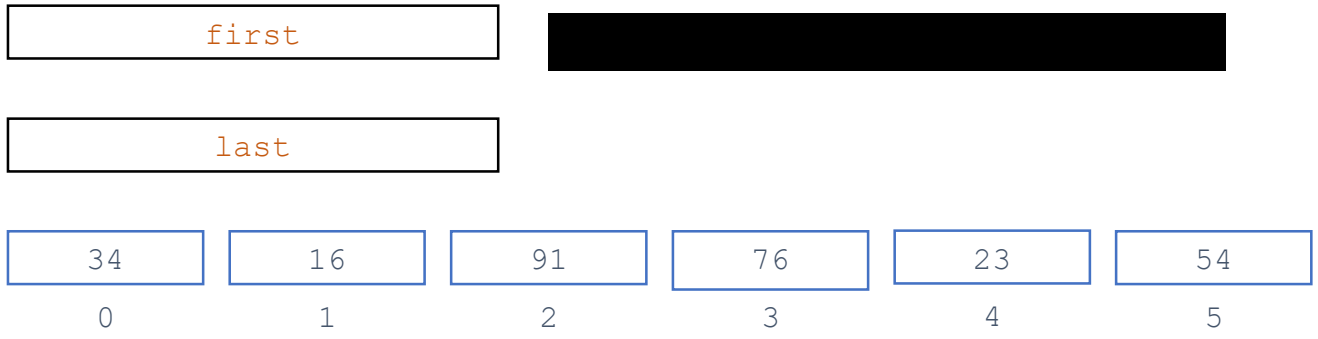

```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

Recursive call prior to display command

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - Start with last element
  - Divide the array in half




```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

```
// Divide the array in half
void displayArray(int myArray[], int first, int last)
{
     if (first == last)
          std::cout <<myArray[first] << " ";
     else
     {
          int mid = (first + last)/2;
          displayArray(myArray, first, mid);
          displayArray(myArray, mid + 1, last);
     } // end if
} // end displayArray
```

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - Start with last element
  - Divide the array in half


```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

```
// Divide the array in half
void displayArray(int myArray[], int first, int last)
{
     if (first == last)
          std::cout <<myArray[first] << " ";
     else
     {
          int mid = (first + last)/2;
          displayArray(myArray, first, mid);
          displayArray(myArray, mid + 1, last);
     } // end if
} // end displayArray
```

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - Start with last element
  - Divide the array in half


```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

```
// Divide the array in half
void displayArray(int myArray[], int first, int last)
{
     if (first == last)
          std::cout <<myArray[first] << " ";
     else
     {
          int mid = (first + last)/2;
          displayArray(myArray, first, mid);
          displayArray(myArray, mid + 1, last);
     } // end if
} // end displayArray
```

**two calls**

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - Start with last element
  - Divide the array in half


```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

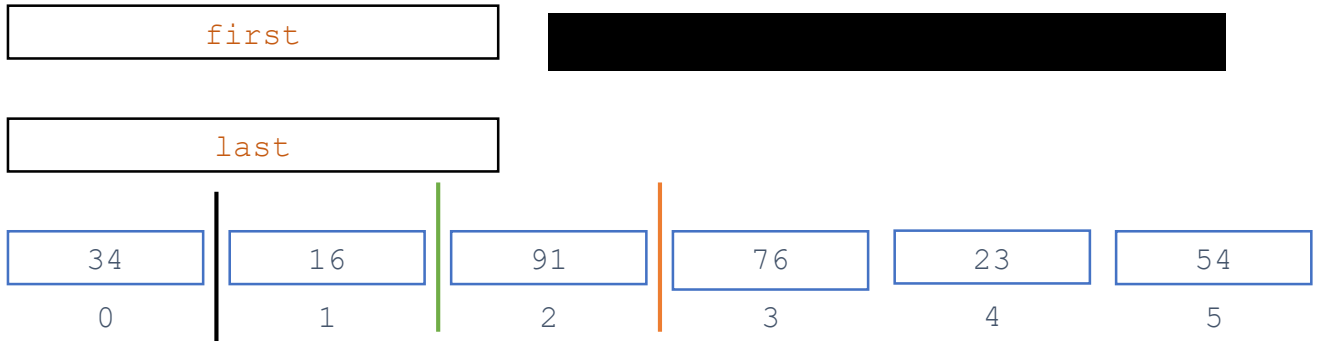
```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

```
// Divide the array in half
void displayArray(int myArray[], int first, int last)
{
     if (first == last)
          std::cout <<myArray[first] << " ";
     else
     {
          int mid = (first + last)/2;
          displayArray(myArray, first, mid);
          displayArray(myArray, mid + 1, last);
     } // end if
} // end displayArray
```

**two calls**

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - Start with last element
  - Divide the array in half




```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

```
// Divide the array in half
void displayArray(int myArray[], int first, int last)
{
     if (first == last)
          std::cout <<myArray[first] << " ";
     else
     {
          int mid = (first + last)/2;
          displayArray(myArray, first, mid);
          displayArray(myArray, mid + 1, last);
     } // end if
} // end displayArray
```

**two calls**

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - Start with last element
  - Divide the array in half


```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

```
// Divide the array in half
void displayArray(int myArray[], int first, int last)
{
     if (first == last)
          std::cout <<myArray[first] << " ";
     else
     {
          int mid = (first + last)/2;
          displayArray(myArray, first, mid);
          displayArray(myArray, mid + 1, last);
     } // end if
} // end displayArray
```

**two calls**

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - Start with last element
  - Divide the array in half


```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

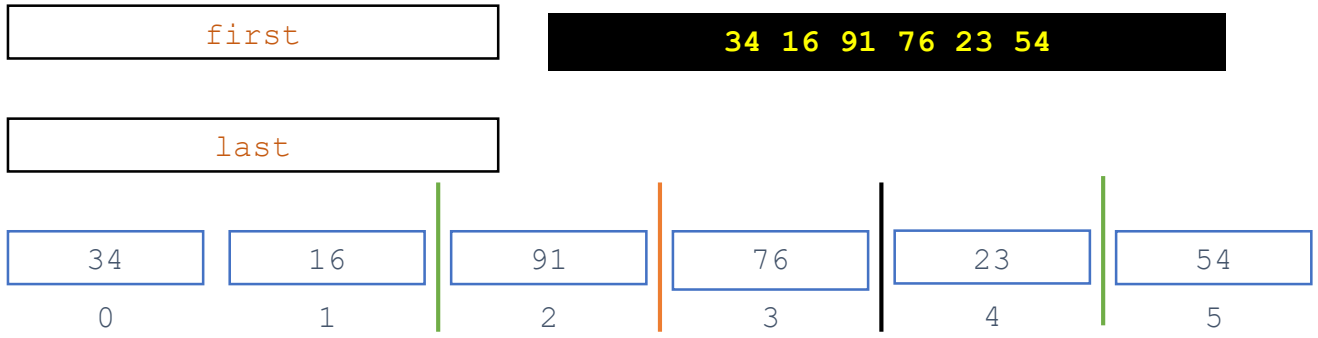
```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

```
// Divide the array in half
void displayArray(int myArray[], int first, int last)
{
     if (first == last)
          std::cout <<myArray[first] << " ";
     else
     {
          int mid = (first + last)/2;
          displayArray(myArray, first, mid);
          displayArray(myArray, mid + 1, last);
     } // end if
} // end displayArray
```

**two calls**

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - Start with last element
  - Divide the array in half



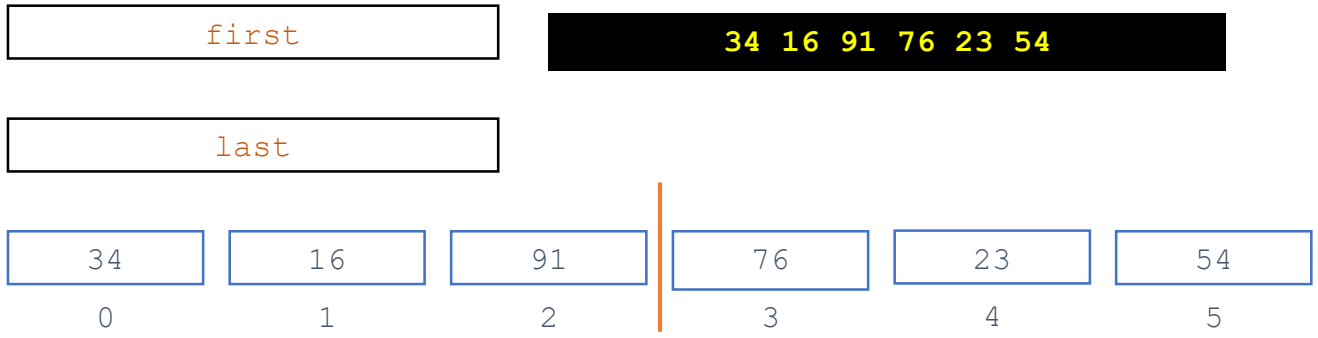

```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

```
// Divide the array in half
void displayArray(int myArray[], int first, int last)
{
     if (first == last)
          std::cout <<myArray[first] << " ";
     else
     {
          int mid = (first + last)/2;
          displayArray(myArray, first, mid);
          displayArray(myArray, mid + 1, last);
     } // end if
} // end displayArray
```

# Recursive Processing - Arrays

- **Recursively displaying elements stored in an array**
  - Start with first element
  - Start with last element
  - Divide the array in half




```
// Start with myArray[first]
void displayArray(int myArray[], int first, int last)
{
     std::cout << myArray[first] << " ";
     if (first < last)
          displayArray(myArray, first + 1, last);
} // end displayArray
```

Implicit base case

```
// Start with myArray[last]
void displayArray(int myArray[], int first, int last)
{
     if (first <= last)
     {
          displayArray(myArray, first, last-1);
          std::cout << myArray[last] << " ";
     } // end if
} // end displayArray
```

```
// Divide the array in half
void displayArray(int myArray[], int first, int last)
{
     if (first == last)
          std::cout <<myArray[first] << " ";
     else
     {
          int mid = (first + last)/2;
          displayArray(myArray, first, mid);
          displayArray(myArray, mid + 1, last);
     } // end if
} // end displayArray
```

explicit base case

# Thank you