Manipulating One-dimensional Arrays

Mitsu Ogihara

Department of Computer Science
University of Miami

Table of Contents

1 For each

2 Exchanging Values

3 Reversing

For-each iteration

For enumerating the elements of an array from the beginning to the end, one can use for each expression
Suppose an array `a` is declared to be of <type>[], then

```
for ( <type> <name> :  a ) { ...  }
```

retrieves the elements of `a` one after another and stores in the variable `<name>`

For examples, if `theList` is a `String` array with elements `"Abraham"`, `"Brigit"`, `"Carlos"`, `"Dwayne"`, and so on,

```
for ( String xyz : arrayX ) {
  BODY
}
```

executes BODY with the value of `xyz` equal to `"Abraham"`, equal to `"Brigit"`, equal to `"Carlos"`, equal to `"Dwayne"`, and so on.

This comes handy when performing a uniform action to the elements in the array in the order then appear

Benefits of “For Each”

No need to speficy the index value since the elements are generated one after the other

If the index value needs to be referred, better use the regular for-loop

If `s` is an array of String objects, then

```
for ( int i = 0; i < s.length; i++ ) {
  String w = s[i];
  ...
}
```

is essentially the same as:

```
int i = 0;
for ( String w : s ) {
  ...
  i++;
}
```

What are the differences?

String toCharArray

The method `toCharArray` of class `String` returns an array of char that corresponds to the character sequence of the String

For example, if `w` holds the value `"CSC120"` (excluding the double quote), then `w.toCharArray()` returns the six element array
`['C', 'S', 'C', '1', '2', '0']`

Processing Letters in a String

Scan the letters of a given input String and then converts any letter other than the apostrophe, the alphabet, and the digits to white space, but minimize the number of white space appearing between non-white space to 1

- Use a boolean to remember whether the last char printed is a space
- Convert the string to a char-array and then process the elements using "for each"
- Given a char `c` at hand
  - If `c` needs to be printed, print it and set the boolean to false
  - Otherwise, print a white space if the boolean is false and then set the boolean to true

The Code: CharArray.java

```
1  import java.util.Scanner;
2  public class CharArray {
3    public static void main(String[] args ) {
4      Scanner console = new Scanner(System.in);
5      System.out.print( "Enter a text line: " );
6      String input = console.nextLine();
7      char[] cArray = input.toCharArray();
8      boolean isPreviousSpace = true;
9      for ( char c : cArray ) {
10       if ( Character.isLowerCase( c )
11           || Character.isUpperCase( c )
12           || Character.isDigit( c ) || c == '\'' ) {
13         System.out.print( c );
14         isPreviousSpace = false;
15       }
16       else if ( !isPreviousSpace ) {
17         System.out.print( " " );
18         isPreviousSpace = true;
19       }
20     }
21     System.out.println();
22   }
23 }
```

Create a scanner by the name of `console` and receive input from the user

The Code: CharArray.java

```
1  import java.util.Scanner;
2  public class CharArray {
3    public static void main(String[] args ) {
4      Scanner console = new Scanner(System.in);
5      System.out.print( "Enter a text line: " );
6      String input = console.nextLine();
7      char[] cArray = input.toCharArray();
8      boolean isPreviousSpace = true;
9      for ( char c : cArray ) {
10       if ( Character.isLowerCase( c )
11           || Character.isUpperCase( c )
12           || Character.isDigit( c ) || c == '\'' ) {
13         System.out.print( c );
14         isPreviousSpace = false;
15       }
16       else if ( !isPreviousSpace ) {
17         System.out.print( " " );
18         isPreviousSpace = true;
19       }
20     }
21     System.out.println();
22   }
23 }
```

Convert the input to a char array

The Code: CharArray.java

```
1  import java.util.Scanner;
2  public class CharArray {
3    public static void main(String[] args ) {
4      Scanner console = new Scanner(System.in);
5      System.out.print( "Enter a text line: " );
6      String input = console.nextLine();
7      char[] cArray = input.toCharArray();
8      boolean isPreviousSpace = true;
9      for ( char c : cArray ) {
10       if ( Character.isLowerCase( c )
11           || Character.isUpperCase( c )
12           || Character.isDigit( c ) || c == '\'' ) {
13         System.out.print( c );
14         isPreviousSpace = false;
15       }
16       else if ( !isPreviousSpace ) {
17         System.out.print( " " );
18         isPreviousSpace = true;
19       }
20     }
21     System.out.println();
22   }
23 }
```

Use “for each” on the array, the iterator variable is `c`

The Code: CharArray.java

```
1  import java.util.Scanner;
2  public class CharArray {
3    public static void main(String[] args ) {
4      Scanner console = new Scanner(System.in);
5      System.out.print( "Enter a text line: " );
6      String input = console.nextLine();
7      char[] cArray = input.toCharArray();
8      boolean isPreviousSpace = true;
9      for ( char c : cArray ) {
10       if ( Character.isLowerCase( c )
11           || Character.isUpperCase( c )
12           || Character.isDigit( c ) || c == '\'' ) {
13         System.out.print( c );
14         isPreviousSpace = false;
15       }
16       else if ( !isPreviousSpace ) {
17         System.out.print( " " );
18         isPreviousSpace = true;
19       }
20     }
21     System.out.println();
22   }
23 }
```

The case in which `c` needs to be printed no matter what

The Code: CharArray.java

```
1  import java.util.Scanner;
2  public class CharArray {
3    public static void main(String[] args ) {
4      Scanner console = new Scanner(System.in);
5      System.out.print( "Enter a text line: " );
6      String input = console.nextLine();
7      char[] cArray = input.toCharArray();
8      boolean isPreviousSpace = true;
9      for ( char c : cArray ) {
10       if ( Character.isLowerCase( c )
11           || Character.isUpperCase( c )
12           || Character.isDigit( c ) || c == '\'' ) {
13         System.out.print( c );
14         isPreviousSpace = false;
15       }
16       else if ( !isPreviousSpace ) {
17         System.out.print( " " );
18         isPreviousSpace = true;
19       }
20     }
21     System.out.println();
22   }
23 }
```

The remaining case; the `else` is not needed because the only action needed is to set the boolean to true, which is already is the case

The Code: CharArray.java

```
1  import java.util.Scanner;
2  public class CharArray {
3    public static void main(String[] args ) {
4      Scanner console = new Scanner(System.in);
5      System.out.print( "Enter a text line: " );
6      String input = console.nextLine();
7      char[] cArray = input.toCharArray();
8      boolean isPreviousSpace = true;
9      for ( char c : cArray ) {
10       if ( Character.isLowerCase( c )
11           || Character.isUpperCase( c )
12           || Character.isDigit( c ) || c == '\'' ) {
13         System.out.print( c );
14         isPreviousSpace = false;
15       }
16       else if ( !isPreviousSpace ) {
17         System.out.print( " " );
18         isPreviousSpace = true;
19       }
20     }
21     System.out.println();
22   }
23 }
```

Print a new line

Table of Contents

1 For each

2 Exchanging Values

3 Reversing

Exchanging Values Between Two Elements of an Array

The problem: given an array `X` of `TYPE` and two indices `p` and `q` exchange values between the two elements `X[p]` and `X[q]`

Exchanging Values Between Two Elements of an Array

The problem: given an array `X` of `TYPE` and two indices `p` and `q` exchange values between the two elements `X[p]` and `X[q]`

A naive approach: execute

```
X[p] = X[q];
X[q] = X[p];
```

What's wrong with this?

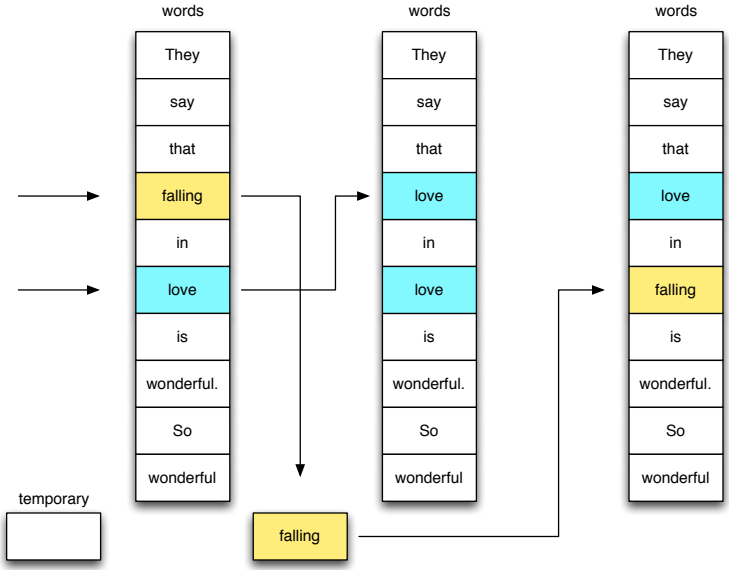
Exchanging Values Between Two Elements of an Array

A correct generic approach: execute

```
TYPE temporary = X[p];
X[p] = X[q];
X[q] = temporary;
```

This method saves the value `X[p]` in a storage `temporary`

Graphical View



Swapping Elements in an Array of `String` Objects

- Receive the user the number of elements
- Create a `String` array of that number of receive data from the user to store in the array
- Present the data with there position values
- Present the data with there position values
- Receive from the user two position values and perform exchange

Swapping Elements in an Array of Strings

```
1  import java.util. * ;
2  public class WordExchange {
3    public static void main( String[] args ) {
4      Scanner console = new Scanner( System.in );
5      String[] theWords;
6      int size, pos, index1, index2;
7      String temporary;
8      //------ receive size
9      System.out.print( "How many words? " );
10     size  = Integer.parseInt( console.nextLine() );
11     //------ declare the array and receive words
12     theWords = new String[ size ];
13     for ( pos = 0; pos < size; pos ++ ) {
14       System.out.printf( "Enter data %d: ", pos );
15       theWords[ pos ] = console.nextLine();
16     }
17     //------ print the words
18     for ( pos = 0; pos < theWords.length; pos ++ ) {
19       System.out.printf( "The word no. %d is %s%n",
20           pos, theWords[ pos ] );
21     }
```

Ask the user to enter the size and create an array

Use `Integer.parseInt`

Swapping Elements in an Array of Strings

```
1  import java.util. * ;
2  public class WordExchange {
3    public static void main( String[] args ) {
4      Scanner console = new Scanner( System.in );
5      String[] theWords;
6      int size, pos, index1, index2;
7      String temporary;
8      //------ receive size
9      System.out.print( "How many words? " );
10     size  = Integer.parseInt( console.nextLine() );
11     //------ declare the array and receive words
12     theWords = new String[ size ];
13     for ( pos = 0; pos < size; pos ++ ) {
14       System.out.printf( "Enter data %d: ", pos );
15       theWords[ pos ] = console.nextLine();
16     }
17     //------ print the words
18     for ( pos = 0; pos < theWords.length; pos ++ ) {
19       System.out.printf( "The word no. %d is %s%n",
20           pos, theWords[ pos ] );
21     }
```

As before, interact with the user to receive the data

Swapping Elements in an Array of Strings

```
1  import java.util. * ;
2  public class WordExchange {
3    public static void main( String[] args ) {
4      Scanner console = new Scanner( System.in );
5      String[] theWords;
6      int size, pos, index1, index2;
7      String temporary;
8      //------ receive size
9      System.out.print( "How many words? " );
10     size  = Integer.parseInt( console.nextLine() );
11     //------ declare the array and receive words
12     theWords = new String[ size ];
13     for ( pos = 0; pos < size; pos ++ ) {
14       System.out.printf( "Enter data %d: ", pos );
15       theWords[ pos ] = console.nextLine();
16     }
17     //------ print the words
18     for ( pos = 0; pos < theWords.length; pos ++ ) {
19       System.out.printf( "The word no. %d is %s%n",
20           pos, theWords[ pos ] );
21     }
```

Print the data

Swapping Elements in an Array of Strings

```
22      //------ perform exchange
23      System.out.print( "Enter index 1: " );
24      index1 = console.nextInt();
25      System.out.print( "Enter index 2: " );
26      index2 = console.nextInt();
27      if ( index1 >= 0 && index1 < size &&
28          index2 >= 0 && index2 < size ) {
29        temporary = theWords[ index1 ];
30        theWords[ index1 ] = theWords[ index2 ];
31        theWords[ index2 ] = temporary;
32      }
33      System.out.println( "We have made the switch." );
34      //------ print the words
35      for ( pos = 0; pos < theWords.length; pos ++ ) {
36        System.out.printf( "The word no. %d is %s%n",
37            pos, theWords[ pos ] );
38      }
39    }
40  }
```

Receive indices; use `Integer.parseInt`

Swapping Elements in an Array of Strings

```
22      //------ perform exchange
23      System.out.print( "Enter index 1: " );
24      index1 = console.nextInt();
25      System.out.print( "Enter index 2: " );
26      index2 = console.nextInt();
27      if ( index1 >= 0 && index1 < size &&
28          index2 >= 0 && index2 < size ) {
29        temporary = theWords[ index1 ];
30        theWords[ index1 ] = theWords[ index2 ];
31        theWords[ index2 ] = temporary;
32      }
33      System.out.println( "We have made the switch." );
34      //------ print the words
35      for ( pos = 0; pos < theWords.length; pos ++ ) {
36        System.out.printf( "The word no. %d is %s%n",
37            pos, theWords[ pos ] );
38      }
39    }
40  }
```

Exchanging elements between two indices entered
Do this only if they are valid indices

Swapping Elements in an Array of Strings

```
22      //------ perform exchange
23      System.out.print( "Enter index 1: " );
24      index1 = console.nextInt();
25      System.out.print( "Enter index 2: " );
26      index2 = console.nextInt();
27      if ( index1 >= 0 && index1 < size &&
28          index2 >= 0 && index2 < size ) {
29        temporary = theWords[ index1 ];
30        theWords[ index1 ] = theWords[ index2 ];
31        theWords[ index2 ] = temporary;
32      }
33      System.out.println( "We have made the switch." );
34      //------ print the words
35      for ( pos = 0; pos < theWords.length; pos ++ ) {
36        System.out.printf( "The word no. %d is %s%n",
37            pos, theWords[ pos ] );
38      }
39    }
40  }
```

Print the data

Table of Contents

1 For each

2 Exchanging Values

3 Reversing

Reversing an Array

Suppose we have an array of integers in which the elements appear in the ascending order. Suppose we want to reverse the order. How do you do it?

Solution No.1 : Creating a new array

- Create a new array of the same type and the same dimension
- Copy the elements from the original to this new array in the reverse order
- Rename the new array with the original name

Solution No.1 : Creating a new array

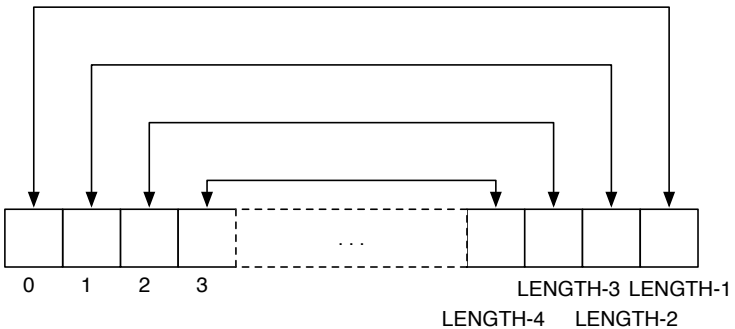
- Create a new array of the same type and the same dimension
- Copy the elements from the original to this new array in the reverse order
- Rename the new array with the original name

```
TYPE[] SUBSTITUTE = new TYPE[OLD.length];
for (int index = 0; index < OLD.length; index ++) {
  SUBSTITUTE[index] = OLD[OLD.length - 1 - index];
}
OLD = SUBSITUTE;
```

Solution No. 2: Reversing in place

- Exchange between element `0` and element `LENGTH - 1`
- Exchange between element `1` and element `LENGTH - 2`
- Exchange between element `2` and element `LENGTH - 3`
- . . .
- Exchange between element `M` and element `LENGTH - 1 - M`

Solution No. 2: Reversing in place



Solution No. 2: Reversing in place

- Exchange between element `0` and element `LENGTH - 1`
- Exchange between element `1` and element `LENGTH - 2`
- Exchange between element `2` and element `LENGTH - 3`
- . . .
- Exchange between element `M` and element `LENGTH - 1 - M`

Solution No. 2: Reversing in place

- Exchange between element `0` and element `LENGTH - 1`
- Exchange between element `1` and element `LENGTH - 2`
- Exchange between element `2` and element `LENGTH - 3`
- . . .
- Exchange between element `M` and element `LENGTH - 1 - M`

What is the right value for `M`?

Solution No. 2: Reversing in place

- Exchange between element `0` and element `LENGTH - 1`
- Exchange between element `1` and element `LENGTH - 2`
- Exchange between element `2` and element `LENGTH - 3`
- . . .
- Exchange between element `M` and element `LENGTH - 1 - M`

What is the right value for `M`? `M = LENGTH/2 - 1`

Demonstration using an Eleven Element Array

```
1  public class ReverseByExchange {
2    public static void print( int[] data ) {
3      for ( int i = 0; i < data.length; i ++ ) {
4        System.out.print( data[ i ] );
5        if ( i < data.length - 1 ) {
6          System.out.print( " " );
7        }
8      }
9      System.out.println();
10   }
11   public static void main( String[] args ) {
12     int[] numbers = new int[ 11 ];
13     for ( int index = 0; index < 11; index ++ ) {
14       numbers[ index ] = index;
15     }
16     print( numbers );
17     for ( int index = 0; index < numbers.length / 2; index ++ ) {
18       int temporary = numbers[ index ];
19       numbers[ index ] = numbers[ numbers.length - 1- index ];
20       numbers[ numbers.length - 1 - index ] = temporary;
21     }
22     print( numbers );
23   }
24 }
```

A method for printing an `int` array

Demonstration using an Eleven Element Array

```
1  public class ReverseByExchange {
2    public static void print( int[] data ) {
3      for ( int i = 0; i < data.length; i ++ ) {
4        System.out.print( data[ i ] );
5        if ( i < data.length - 1 ) {
6          System.out.print( " " );
7        }
8      }
9      System.out.println();
10   }
11   public static void main( String[] args ) {
12     int[] numbers = new int[ 11 ];
13     for ( int index = 0; index < 11; index ++ ) {
14       numbers[ index ] = index;
15     }
16     print( numbers );
17     for ( int index = 0; index < numbers.length / 2; index ++ ) {
18       int temporary = numbers[ index ];
19       numbers[ index ] = numbers[ numbers.length - 1- index ];
20       numbers[ numbers.length - 1 - index ] = temporary;
21     }
22     print( numbers );
23   }
24 }
```

Print all elements in one line with one white space in between and go to newline at the end

Demonstration using an Eleven Element Array

```
1  public class ReverseByExchange {
2    public static void print( int[] data ) {
3      for ( int i = 0; i < data.length; i ++ ) {
4        System.out.print( data[ i ] );
5        if ( i < data.length - 1 ) {
6          System.out.print( " " );
7        }
8      }
9      System.out.println();
10   }
11   public static void main( String[] args ) {
12     int[] numbers = new int[ 11 ];
13     for ( int index = 0; index < 11; index ++ ) {
14       numbers[ index ] = index;
15     }
16     print( numbers );
17     for ( int index = 0; index < numbers.length / 2; index ++ ) {
18       int temporary = numbers[ index ];
19       numbers[ index ] = numbers[ numbers.length - 1- index ];
20       numbers[ numbers.length - 1 - index ] = temporary;
21     }
22     print( numbers );
23   }
24 }
```

Create an eleven element array

Demonstration using an Eleven Element Array

```
1  public class ReverseByExchange {
2    public static void print( int[] data ) {
3      for ( int i = 0; i < data.length; i ++ ) {
4        System.out.print( data[ i ] );
5        if ( i < data.length - 1 ) {
6          System.out.print( " " );
7        }
8      }
9      System.out.println();
10   }
11   public static void main( String[] args ) {
12     int[] numbers = new int[ 11 ];
13     for ( int index = 0; index < 11; index ++ ) {
14       numbers[ index ] = index;
15     }
16     print( numbers );
17     for ( int index = 0; index < numbers.length / 2; index ++ ) {
18       int temporary = numbers[ index ];
19       numbers[ index ] = numbers[ numbers.length - 1- index ];
20       numbers[ numbers.length - 1 - index ] = temporary;
21     }
22     print( numbers );
23   }
24 }
```

Store `index` to slot `index` for all `index=0` to `10`

Demonstration using an Eleven Element Array

```
1  public class ReverseByExchange {
2    public static void print( int[] data ) {
3      for ( int i = 0; i < data.length; i ++ ) {
4        System.out.print( data[ i ] );
5        if ( i < data.length - 1 ) {
6          System.out.print( " " );
7        }
8      }
9      System.out.println();
10   }
11   public static void main( String[] args ) {
12     int[] numbers = new int[ 11 ];
13     for ( int index = 0; index < 11; index ++ ) {
14       numbers[ index ] = index;
15     }
16     print( numbers );
17     for ( int index = 0; index < numbers.length / 2; index ++ ) {
18       int temporary = numbers[ index ];
19       numbers[ index ] = numbers[ numbers.length - 1- index ];
20       numbers[ numbers.length - 1 - index ] = temporary;
21     }
22     print( numbers );
23   }
24 }
```

Print the elements

Demonstration using an Eleven Element Array

```
1  public class ReverseByExchange {
2    public static void print( int[] data ) {
3      for ( int i = 0; i < data.length; i ++ ) {
4        System.out.print( data[ i ] );
5        if ( i < data.length - 1 ) {
6          System.out.print( " " );
7        }
8      }
9      System.out.println();
10   }
11   public static void main( String[] args ) {
12     int[] numbers = new int[ 11 ];
13     for ( int index = 0; index < 11; index ++ ) {
14       numbers[ index ] = index;
15     }
16     print( numbers );
17     for ( int index = 0; index < numbers.length / 2; index ++ ) {
18       int temporary = numbers[ index ];
19       numbers[ index ] = numbers[ numbers.length - 1- index ];
20       numbers[ numbers.length - 1 - index ] = temporary;
21     }
22     print( numbers );
23   }
24 }
```

The range of index for the reversal is 0 through `numbers.length/2 - 1`

Demonstration using an Eleven Element Array

```
1  public class ReverseByExchange {
2    public static void print( int[] data ) {
3      for ( int i = 0; i < data.length; i ++ ) {
4        System.out.print( data[ i ] );
5        if ( i < data.length - 1 ) {
6          System.out.print( " " );
7        }
8      }
9      System.out.println();
10   }
11   public static void main( String[] args ) {
12     int[] numbers = new int[ 11 ];
13     for ( int index = 0; index < 11; index ++ ) {
14       numbers[ index ] = index;
15     }
16     print( numbers );
17     for ( int index = 0; index < numbers.length / 2; index ++ ) {
18       int temporary = numbers[ index ];
19       numbers[ index ] = numbers[ numbers.length - 1- index ];
20       numbers[ numbers.length - 1 - index ] = temporary;
21     }
22     print( numbers );
23   }
24 }
```

Exchange between two elements using `temporary` as a temporary storage

Demonstration using an Eleven Element Array

```
1  public class ReverseByExchange {
2    public static void print( int[] data ) {
3      for ( int i = 0; i < data.length; i ++ ) {
4        System.out.print( data[ i ] );
5        if ( i < data.length - 1 ) {
6          System.out.print( " " );
7        }
8      }
9      System.out.println();
10   }
11   public static void main( String[] args ) {
12     int[] numbers = new int[ 11 ];
13     for ( int index = 0; index < 11; index ++ ) {
14       numbers[ index ] = index;
15     }
16     print( numbers );
17     for ( int index = 0; index < numbers.length / 2; index ++ ) {
18       int temporary = numbers[ index ];
19       numbers[ index ] = numbers[ numbers.length - 1- index ];
20       numbers[ numbers.length - 1 - index ] = temporary;
21     }
22     print( numbers );
23   }
24 }
```

Print the elements

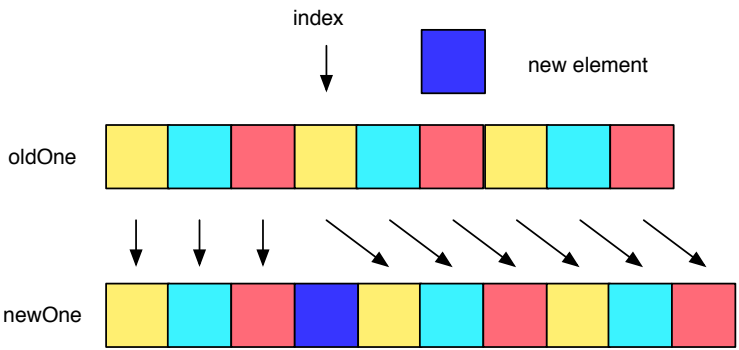
Adding an Element to an Array

Suppose you want to add an element to an existing array at a particular location. How do you do it?

General Solution for Insertion

- Create a new temporary array with a different name, say newArray
- Copy all the elements before the insertion point at the same element location
- Add the new element at the designated location
- Copy all the elements on and after the inserttion point to one higher position
- Set the current array to the temporary

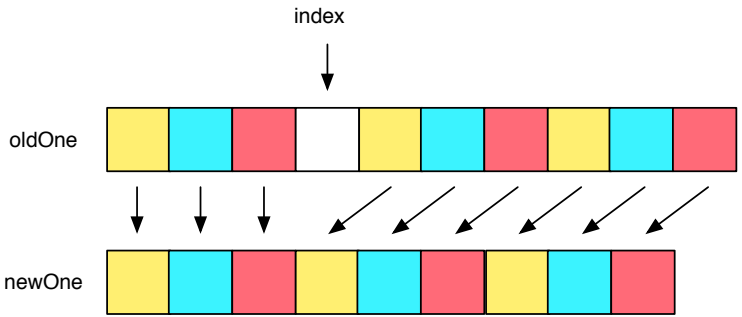
Graphical View



General Solution for Deletion

- Create a new temporary array with a different name, say newArray
- Copy all the elements before the removal point at the same element location
- Copy all the elements on and after the inserttion point to one lower position
- Set the current array to the temporary

Graphical View



The Code: InsertAndDelete.java

```
1  public class InsertAndDelete {
2    //--- print the elements of a given array
3    public static void print( String[] names ) {
4      for ( String w : names ) {
5        System.out.print( " " + w );
6      }
7      System.out.println();
8    }
```

Method that prints the elements of an array

The Code: InsertAndDelete.java

```
1  public class InsertAndDelete {
2    //--- print the elements of a given array
3    public static void print( String[] names ) {
4      for ( String w : names ) {
5        System.out.print( " " + w );
6      }
7      System.out.println();
8    }
```

Using “for each” and prints each element with a " " in front

The Code: InsertAndDelete.java (insert)

```
9   //--- main
10  public static void main(String[] args) {
11    String[] names = new String[]{
12        "Sarah", "Peggy", "Helen", "Dinah", "Ella", "Billie" };
13    print( names );
14    //------- removing element 3
15    String[] namesLess = new String[ names.length - 1 ];
16    for ( int i = 0; i <= 2; i ++ ) {
17      namesLess[i] = names[i];
18    }
19    for ( int i = 3; i < names.length; i ++ ) {
20      namesLess[i-1] = names[i];
21    }
22    print( namesLess );
```

The names array; print the array contents using the method print

The Code: InsertAndDelete.java (insert)

```
9   //--- main
10  public static void main(String[] args) {
11    String[] names = new String[]{
12        "Sarah", "Peggy", "Helen", "Dinah", "Ella", "Billie" };
13    print( names );
14    //------- removing element 3
15    String[] namesLess = new String[ names.length - 1 ];
16    for ( int i = 0; i <= 2; i ++ ) {
17      namesLess[i] = names[i];
18    }
19    for ( int i = 3; i < names.length; i ++ ) {
20      namesLess[i-1] = names[i];
21    }
22    print( namesLess );
```

Create a new array with one fewer elements

The Code: InsertAndDelete.java (insert)

```
9   //--- main
10  public static void main(String[] args) {
11    String[] names = new String[]{
12        "Sarah", "Peggy", "Helen", "Dinah", "Ella", "Billie" };
13    print( names );
14    //------- removing element 3
15    String[] namesLess = new String[ names.length - 1 ];
16    for ( int i = 0; i <= 2; i ++ ) {
17      namesLess[i] = names[i];
18    }
19    for ( int i = 3; i < names.length; i ++ ) {
20      namesLess[i-1] = names[i];
21    }
22    print( namesLess );
```

Copy elements before the point of elimination

The Code: InsertAndDelete.java (insert)

```
9   //--- main
10  public static void main(String[] args) {
11    String[] names = new String[]{
12        "Sarah", "Peggy", "Helen", "Dinah", "Ella", "Billie" };
13    print( names );
14    //------- removing element 3
15    String[] namesLess = new String[ names.length - 1 ];
16    for ( int i = 0; i <= 2; i ++ ) {
17      namesLess[i] = names[i];
18    }
19    for ( int i = 3; i < names.length; i ++ ) {
20      namesLess[i-1] = names[i];
21    }
22    print( namesLess );
```

Copy the rest to one lower position

The Code: InsertAndDelete.java (insert)

```
9   //--- main
10  public static void main(String[] args) {
11    String[] names = new String[]{
12        "Sarah", "Peggy", "Helen", "Dinah", "Ella", "Billie" };
13    print( names );
14    //------- removing element 3
15    String[] namesLess = new String[ names.length - 1 ];
16    for ( int i = 0; i <= 2; i ++ ) {
17      namesLess[i] = names[i];
18    }
19    for ( int i = 3; i < names.length; i ++ ) {
20      namesLess[i-1] = names[i];
21    }
22    print( namesLess );
```

Print

The Code: InsertAndDelete.java (delete)

```
23      //------- adding "Nancy" as element 4
24      String[] namesMore = new String[ names.length + 1 ];
25      for ( int i = 0; i <= 3; i ++ ) {
26        namesMore[i] = names[i];
27      }
28      namesMore[4] = "Nancy";
29      for ( int i = 4; i < names.length; i ++ ) {
30        namesMore[i+1] = names[i];
31      }
32      print( namesMore );
33    }
34  }
```

Create a new array with one more elements

The Code: InsertAndDelete.java (delete)

```
23      //------- adding "Nancy" as element 4
24      String[] namesMore = new String[ names.length + 1 ];
25      for ( int i = 0; i <= 3; i ++ ) {
26        namesMore[i] = names[i];
27      }
28      namesMore[4] = "Nancy";
29      for ( int i = 4; i < names.length; i ++ ) {
30        namesMore[i+1] = names[i];
31      }
32      print( namesMore );
33    }
34  }
```

Copy elements before the point of insertion

The Code: InsertAndDelete.java (delete)

```
23      //------- adding "Nancy" as element 4
24      String[] namesMore = new String[ names.length + 1 ];
25      for ( int i = 0; i <= 3; i ++ ) {
26        namesMore[i] = names[i];
27      }
28      namesMore[4] = "Nancy";
29      for ( int i = 4; i < names.length; i ++ ) {
30        namesMore[i+1] = names[i];
31      }
32      print( namesMore );
33    }
34  }
```

Place the new element

The Code: InsertAndDelete.java (delete)

```
23      //------- adding "Nancy" as element 4
24      String[] namesMore = new String[ names.length + 1 ];
25      for ( int i = 0; i <= 3; i ++ ) {
26        namesMore[i] = names[i];
27      }
28      namesMore[4] = "Nancy";
29      for ( int i = 4; i < names.length; i ++ ) {
30        namesMore[i+1] = names[i];
31      }
32      print( namesMore );
33    }
34  }
```

Copy the rest to one higher position

The Code: InsertAndDelete.java (delete)

```
23      //------- adding "Nancy" as element 4
24      String[] namesMore = new String[ names.length + 1 ];
25      for ( int i = 0; i <= 3; i ++ ) {
26        namesMore[i] = names[i];
27      }
28      namesMore[4] = "Nancy";
29      for ( int i = 4; i < names.length; i ++ ) {
30        namesMore[i+1] = names[i];
31      }
32      print( namesMore );
33    }
34  }
```

Print

The End