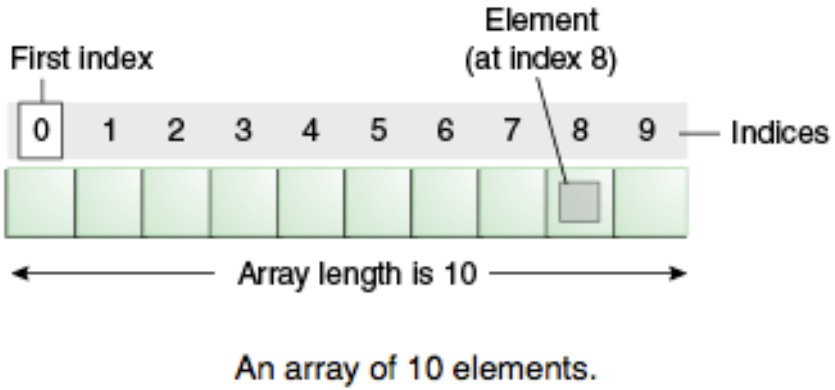
Array Processing

Strings

What is an Array?

- Object used to store a **list** of values
- All values in array are the **same type**
  - E.g. int, double, char, string, etc.



An array of 10 elements.

What is an Array?

- Used to store a large number of data
- Easy to **access** any element in the array

42	jokerwitz@hotmail.com
43	tylerddickinson@gmail.com
44	kevinceltics09@hotmail.com
45	taylorhynes16@yahoo.com
46	moan.rs@hotmail.com
47	nec_nijmegen_05@hotmail.com
48	denice@hotmail.dk
49	cjdjeva@gmail.com
50	joel.dreise@gmail.com
51	smokeunderwater1@hotmail.com
52	itmees@live.com
53	meidla1@hotmail.com
54	cheese_c@hotmail.co.uk
55	Crime@null.net
56	apocalypse55@hotmail.co.uk
57	echo.phyber1@hotmail.com

E.g. array to store a list of emails

How to Declare

Declaring an array to hold up to 10 words:

```
String array[] = new String [10];
```

Declaring an array and initializing variables at the same time:

```
String[] array = {“Hi”,”Hello”,”Welcome”, ”Goodbye”,”Farewell”};
```

String Manipulation

length() to find the length of a string

stringName.charAt(0) will output the first character of a string. Each character in a string has a position. String positions start at 0.

stringName.indexOf(“T”) will output the index number that the letter T belongs to

stringName.substring(3) will output all characters after the index #

Eg: String stringName = “Jim hates math”

Output: “ hates math”

String Manipulation

stringName.substring(#, #)

First # is for the index that you want and last number is for the index that you don't want

stringName2 = " I hate math "

stringName2.trim() Output: "I hate math"

stringName.toLowerCase() makes every character a lower case letter

stringName.toUpperCase() makes every character an upper case letter

String Manipulation

concat()

stringname1.concat(stringName2) will put both strings into 1 word

stringName.replace(“a”, “p”); will replace every a with p

stringName1.equals(stringName2) will return true or false depending on whether the two contain the same value

stringName1.compareTo(stringName2) is for alphabetical order

If the condition above is >0, name1 is before name 2. <0, name1 is before name 2.

Sorting

```
public static void bubbleSort(int[] List){
  boolean swap = true;
  int j=1;
  int temp;
  while(swap==true){
    do{
      swap = false;
      for(int i = 0; i<List.length-j; i++){
        if(List[i]>List[i+1]){
          temp = List[i];
          List[i] = List[i+1];
          List[i+1] = temp;
          swap = true;
        }
      }
      j++;
    }while(swap);

    for(int i = 0; i < List.length; i++){
      System.out.println(List[i]);
    }
  }
```

Searching

```
boolean found;

System.out.println("Enter the word you would like to search within the array");

word = scanner.next();

for (int i = 0; i < array.length; i++) {

        lf (array[i].equals(word))  {

                found = true;

                break;

        }

}
```

Searching (output)

```
if (found == true) {

        System.out.println("The letter entered has been found within the array");

}else{

        System.out.println("The letter entered had not been found within the array");

}
```

Searching (expanding)

```
int pos; // variable to hold position in the array

for (int i = 0; i < array.length; i++) {

        If (array[i].equals(word))  {

                found = true; // setting boolean found to true

                pos = i; // holding the position in the array in pos

                break;

        }

}
```

Searching (output expanding)

```
If (found == true) {

        System.out.println("The word entered has been found within array " + pos);

}else{

        System.out.println("The word entered had not been found within the array");

}
```

Conclusion

- Arrays are very useful when problem solving when a large number of data is involved
- **Organize** large sets of data
- Variety of ways you can access information in an array

The End

Thank you for listening