

Arrays and Collections

COP3330 - Fall 2006
University of Central Florida

H. Schwartz

Outline

- Arrays
- Iterator for loop
- Searching and Sorting
- Collections
- ArrayList and Vector
- CompareTo

Arrays

- A single object made up of elements
- Definition:
 - `<ElementType>[] <name>;`

COP 3330 - Fall 2006

Arrays

- To construct an array object:
`= new <type>[<size>;`

```
int[] numbers = new int[10]; //creates a new array of integers
String[] sentences = new String[5]; //creates an array of references to Strings
//the Strings themselves have not been created yet
```

COP 3330 - Fall 2006

Arrays

- Accessing an element

```
numbers[0] = 1;
numbers[1] = 35;

//constructing an array to go into the first element:
sentences[0] = new String("This is a sentence.");

if (numbers[0] == 1){
    //if the first number is 0 do something
}
...
```

COP 3330 - Fall 2006

Arrays

- Explicit Initiation and Constants

```
<ElementType>[] name = {<exp1>, <exp2>,
    <exp3>, ...};
```

```
int[] sequence = {1, 7, 19, 259, 65539}; //explicit definition
//constant:
final int[] binarySequence = {2, 4, 8, 16, 32, 64, 128, 256};
```

COP 3330 - Fall 2006

Arrays

- Methods with arrays

```
File []files = new File[3]; //creates an array of files
//create File objects for given files / directories
files[0] = new File("something.txt");
files[1] = new File("/home/user");
files[2] = new File("/home/user/oop.txt");

//perform File instance methods on objects:
if (files[0].exists())
    System.out.println("it exists");
String []fNames = files[1].list(); //lists contents on directory
files[2].setLastModified(123154623);
```

COP 3330 - Fall 2006

Iterator for loop

```
for ( <Iterator> : Array )
    <Action>
```

```
...
//computes how many integers are greater than 50 in intAr
int numGreater = 0;
for (int i = 0; i < intAr.length; i++){
    if (intAr[i] > 50)
        numGreater++;
}

//equivalent:
for (int num: intAr){
    if (num > 50)
        numGreater++;
}
```

COP 3330 - Fall 2006

Iterator for loop

- Advantages (over traditional for):
 - Stays in range
 - Clear code
- Disadvantages:
 - Do not know index
 - New to Java
 - Not as flexible

COP 3330 - Fall 2006

Searching and Sorting

- `sequentialSearch`
 - order will not help (array is unsorted)
 - or you'd like to check every element anyway
- `binarySearch`
 - exploit a sorted list to find answer faster
 - checks midpoint

COP 3330 - Fall 2006

Searching and Sorting

- SelectionSort
 - select the smallest (or greatest) element until every spot is full

```
for (int i = 0; i < intArray.length; i++){
    //selecting for intArray[i]
    int indexOfLeast = i;
    for (int check = i+1; check < intArray.length; check++){
        //check the remaining indices
        if (intArray[check] < intArray[indexOfLeast])
            indexOfLeast = check;
    }
    //swap elements in indexOfLeast and i positions:
    int temp = intArray[i];
    intArray[i] = intArray[indexOfLeast];
    intArray[indexOfLeast] = temp;
}
```

COP 3330 - Fall 2006

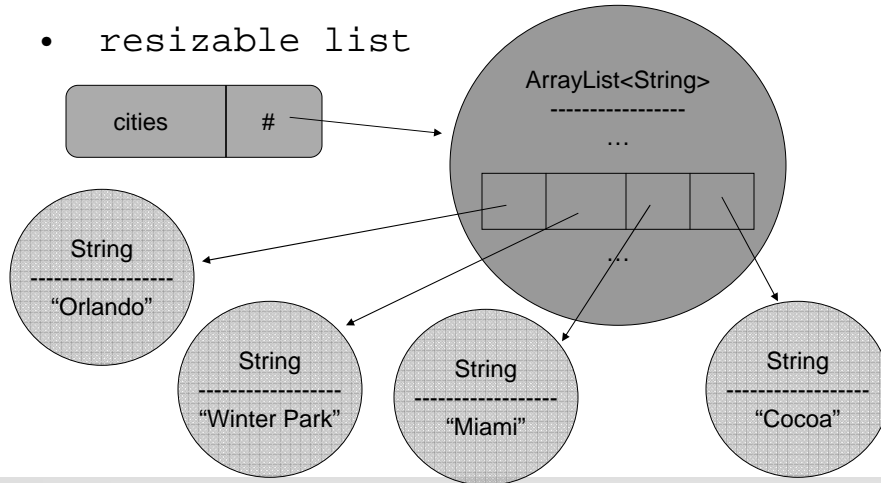
Collections

- Framework for many classes that contain a collection of something and perform more advanced array-like operations.
 - ArrayList
 - Vector
 - TreeSet
 - Priority Queue

COP 3330 - Fall 2006

ArrayList

- resizable list



COP 3330 - Fall 2006

ArrayList

- `ArrayList<String> cities`
= `new ArrayList<String>();`
- Type of elements in the collection

```
public class UsesAnArrayList{  
    private ArrayList<Integer> intList;  
  
    public UsesAnArrayList(){  
        //constructor  
        intList = new ArrayList<Integer>();  
    }  
}
```

*Tangent:
Java Number
classes*

COP 3330 - Fall 2006

ArrayList

- Methods
 - void add(int i, T v)
 - boolean add(T v)
 - T get(int i)
 - T remove(int i)
 - T set(int i, T v)
 - boolean isEmpty()
 - int size()

(p. 426 of book)

COP 3330 - Fall 2006

CompareTo

- public int compareTo(T v)
 - Used to compare elements of a Collection
 - Must "implements Comparable<T>"
 - Common classes which implement Comparable:
 - String
 - All Number classes (Integer, Double...etc)

COP 3330 - Fall 2006

CompareTo

- Collections algorithms which use compareTo:
 - static void sort(List<T> a)
 - max
 - min
 - reverse
 - shuffle