Array-Lists, Iterators, Comparators, and Collections

Weiss, section 2.4, sections 4.1.9 through 4.1.11, and from the beginning of chapter 6 through section 6.4 (pages 37–47, 121–124, and 229–248)

I am indebted to my colleague, John David Stone, for assistance with this handout.
This work is licensed under the Creative Commons Attribution-ShareAlike 4.0 International License. To view a copy of this license, visit https://creativecommons.org/licenses/by-sa/4.0/ or send a letter to Creative Commons, 543 Howard Street, 5th Floor, San Francisco, California, 94105, USA.
Arrays

- Have fields
- Use reference rules for array assignment and parameter passing
- Use different syntax for declaration and construction
- Resize arrays
  - Compiler needs to know the array size
  - May need to put more data into array
Array Resizing

```java
int [] arr = new int[ 10 ];

int [] original = arr;
arr = new new int[ 12 ];
for( int i = 0; i < 10; i++ )
  arr[ i ] = original[ i ];
```

- **ArrayList**
  - Resize automatically
  - Preserve constant-time access
Multidimensional Arrays

• Simpler to implement compared to C
• Array of arrays
• Can be declared as a ragged array

```c
int[ ][ ] a = {{ 1, 2, 3 }, { 4 }, { 5, 6, 7, 8 }}
```
Command-line Arguments

- Java program receives command line arguments through `—`.

  java Echo Hello World
Enhanced for loop

String[ ] arr = new String[ 10 ];

for( String val : arr ) {
    System.out.println( val );
}

for( int i = 0; i < arr1.length; i++ )
    arr1[ i ] = 0;

for( int i = 0; i < args.length - 1; i++ )
    System.out.println( args[ i ] + " " );
Java Collections Framework

• java.util package contains interfaces and classes (containers) that represent data structures
  – Collection of objects (data storage)
  – Operations applied on the collection
  – Component reuse
  – Iterator object

```java
public static void main( String[] args )
{
    MyContainer v = new MyContainer();
    v.add( "3" );
    v.add( "2" );

    System.out.println( "Container contents: " );
    MyContainerIterator itr = v.iterator();
    while( itr.hasNext() )
        System.out.println( itr.next() );
}
```
CompareTo Method

- Defined by Comparable interface
- Comparable is generic: Comparable<AnyType>
- A class that implements Comparable has to implement `compareTo` method.

```java
package java.lang;

public interface Comparable<AnyType> {
    public int compareTo(AnyType other);
}
```
CompareTo Method (cont.)

```java
public static <AnyType> AnyType findMax( AnyType[] a )
{
    int maxIndex = 0;
    for( int i = 1; i < a.length; i++ )
        if( a[ i ].compareTo( a[ maxIndex ] ) > 0 )
            maxIndex = i;
    return a[ maxIndex ];
}
```

public static <AnyType extends Comparable<Anytype>> ...

Shape implements Comparable<Shape>
Square is a subclass of Shape: Square implements Comparable<Shape>
Square is NOT type-compatible with Comparable<Square>

public static <AnyType extends Comparable <? Super Anytype>> ...