Collections in Java

- **Arrays are collections**
  - Arrays are objects
  - Arrays are sequences of elements in base type
  - These elements are collected together in one object: the array
- Java includes may other collection mechanisms
  - Arrays good for some applications (fixed-length sequences), not others (varying-length sequences)
  - Other collections tuned for different purposes
- General observation holds, however:
  - Collections are objects ...
  - ... that contain other objects in a given type
- We’ll study two (more in CMSC132): Stack, ArrayList

Stacks in Java

- Recall: a stack is a data structure (“device” for holding values) – FILO (First In, Last Out)
- Typical operations on a stack
  - **push**: add a new value into the stack
  - **pop**: remove the most recently added value still in stack
  - **top**: return the most recently added value in stack
  - Note: Java calls this “peek”
  - **is empty**: returns true if the stack is currently empty or false otherwise
Example of stack concept (not Java specific)

- Stack s
  - s.isEmpty() == ??
    - true
  - s.push (3);
  - s.isEmpty() == ??
    - false
  - s.push (4);
  - s.peek == ??
    - 4
  - s.pop ();
  - s.push (5);
  - s.peek == ??
    - 5

Stacks in Java (cont.)

- Java includes a generic class for stack objects
- Stack objects contain other objects
- All objects in stack must have same type
- Only objects may be stored in stacks (no primitive-type values)
- Syntax: Stack<E>
  - Stack<E> is a generic class
  - E is a class variable representing the base type
  - Replace E by a specific type to get a stack of that type of elements
  - Class is in java.util package
- Documentation: [http://java.sun.com/j2se/1.5.0/docs/api/java/util/Stack.html](http://java.sun.com/j2se/1.5.0/docs/api/java/util/Stack.html)
- See example: StackExample.java
  - Stack<String> stack = new Stack<String>();
  - Creates a stack of strings
  - extend this to be stack of cats
  - extend this to be stack of integer values

ArrayList Collection

- Like arrays ... but support for inserting/deleting new elements
- Sequences of elements
- All elements must be in same (base) type
- Syntax: ArrayList<E>
- Documentation: [http://java.sun.com/j2se/1.5.0/docs/api/java/util/ArrayList.html](http://java.sun.com/j2se/1.5.0/docs/api/java/util/ArrayList.html)
- See example: ArrayListExample.java
  - ArrayList<String> a = new ArrayList<String>();
  - Creates an ArrayList of strings
  - Collections.sort may be used on ArrayList<String> objects?
  - Reason
    - String implements Comparable interface
    - ArrayList<E> implements List<E> interface
for ... each ... in

- New construct available in Java 1.5 (not available in older versions of Java)
- Use with arrays
- Use with any iterable collection
- Limitations:
  - modifications limited
    - can’t add items to the list being iterated over
    - can’t remove items from the list being iterated over
    - can’t replace items in the list being iterated over
  - access only one
    - only a single collection can be traversed at a time
    - can’t access the one before or the one after on this iteration
  - limited to forward and one at a time
    - can’t traverse the list in the reverse order
    - can’t go to every other element or any variation