CS206

ArrayList
Array

- An array is a sequenced collection of homogenous variables (elements)
- Each element of an array has an index
- The entire array is contiguous in memory
- The length of an array is fixed and can not be changed
Array/List

• Dynamically-sized array
• Stores an ordered sequence of objects
  • Not sorted, ordered in the sense that arrays are ordered
• Can grow and shrink when items are added/removed
• Standard array features all supported, but with different syntax
Array/List

- ArrayList is implemented with an array
- A variable keeps track of the current size
  - initially it is equal to the actual size
  - deletion
    - elements are shifted to the left and size is decremented
  - addition, if not enough space
    - Create new, bigger array
    - Copy elements of old array into new one
Insertion

- In an operation \( \text{add}(i, o) \), we make room for the new element by shifting forward/to the right the elements \( A[i] \), ..., \( A[n - 1] \).
Deletion

• In an operation \texttt{remove}(i), we fill the hole by shifting backward/to the left the left the elements $A[i + 1], \ldots, A[n - 1]$
Java Interfaces

- Java allows only single inheritance.
  - A class can only extend one class
  - As a result, Java does not need any collision resolution.
- BUT a class can “implement” any number of Interfaces
  - Interfaces only define methods
    - they do not provide method bodies so no collision resolution required.
public interface ArraListInterface<T> {
    boolean add(T t);
    void add(int index, T t) throws IndexOutOfBoundsException;
    T get(int index) throws IndexOutOfBoundsException;
    void remove(int index) throws IndexOutOfBoundsException;
    boolean set(int index, T t) throws IndexOutOfBoundsException;
    int size();
    int indexOf(T t);
    void clear();
}
public class ArraList<T> implements ArraListInterface<T> {
    private int capacity = 10;
    private static final double GROWTH_RATE = 1.618033; // the golden mean
    private int count; // number of items currently in ArraList
    private T[] arra; // the array underlying the ArraList
    public ArraList() {
        arra = (T[]) new Object[capacity];
        count = 0;
    }
    public ArraList(int initialCapacity) {
        capacity = initialCapacity;
        arra = (T[]) new Object[capacity];
        count = 0;
    }
}
Class implements add(t,i), remove(i)
Creation with Type Parameters

- When constructing an `ArrayList`, you must specify the type of elements via `<>

```java
ArrayList<String> l1 = new ArrayList<>();
ArrayList<Integer> l2 = new ArrayList<>();
```
Example usage

- Write a program to collect then print all unique words in a file
- Problem: you do not know the number of distinct words!
  - Solution
    - allocate a really big array
    - Use ArraList!
WordCounter —
Count the unique words in file!

WordCounter.java
java.util.ArrayList

- Implemented exactly as ours
- Part of Java collections framework
- import java.util.ArrayList
- Use this one rather than ours for Homework 3