CompSci 100e
Program Design and Analysis II

January 25, 2011

Prof. Rodger

Announcements

- Apt-0127 due Thursday night
- New apts out due next Tuesday, Feb 1
  - Thesaurus APT done in lab
- Today
  - Finish classScores
  - Arrays vs ArrayLists, Sets

Top 10 ways to survive in CompSci 100

10. Read the book
9. Keep Randy’s Pizza number handy
8. Learn how to spell Rodger
7. Always attend class, even if you have a program due that day
6. Keep working until it is correct

Top 10 ways to survive in CompSci 100

5. Turn it off, Pay attention

4. Visit Professor, TA, UTAs in office/consulting hours

3. Read the CompSci 100 Bulletin Board

2. Seek help when stuck (Abide by “1 hour rule”)

1. Start programming assignments early

ArrayList

- Class vs. primitive
- ArrayList
  - Can grow and shrink
  - Has methods for common tasks (see API)
  - Only holds objects
- Can’t have an ArrayList of int or double
  - Need to use wrapper class like Integer or Double

ArrayList (cont)

- Create an ArrayList – must create memory

```
ArrayList<Integer> idlist = new ArrayList<Integer>();
```

- Add an element to end of the ArrayList

```
idlist.add(8);
```

- Modify kth element in an ArrayList

```
idlist.set(k,8);
```

- Sum the elements in the ArrayList

```
// sum up integers in the ArrayList
int sum = 0;
for (Integer current : idlist)
{
    sum += current;
}
```

ArrayList vs. array

- Methods
  - Sort an arrayList called numbers

    ```
    Collections.sort(numbers);
    ```

  - Sort an array called a

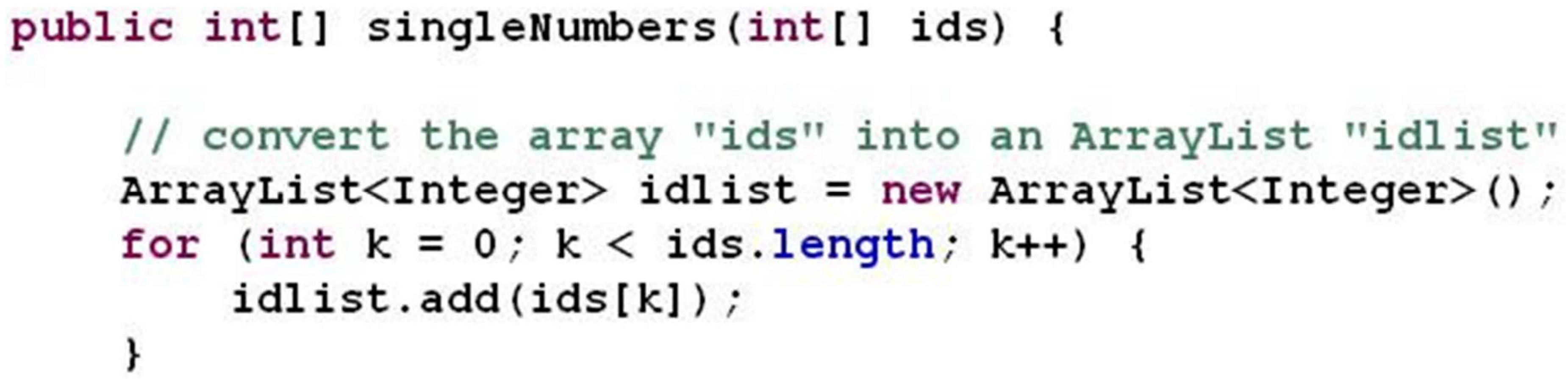
    ```
    Arrays.sort(a);
    ```

- Types
  - Arrays can hold any type
  - ArrayLists only work with objects
- ArrayList’s are dynamic – easy to expand in size
- Can convert from one to the other
- APTs only pass and return arrays

Example: singleNumbers

- Given an integer array that could have duplicates, return an array that has only unique numbers from the original array in the same order (get rid of duplicates!)
- For example if the parameter array is:
  - 8 5 5 8 5
- Then the array to return should be:
  - 8 5

First convert array to ArrayList

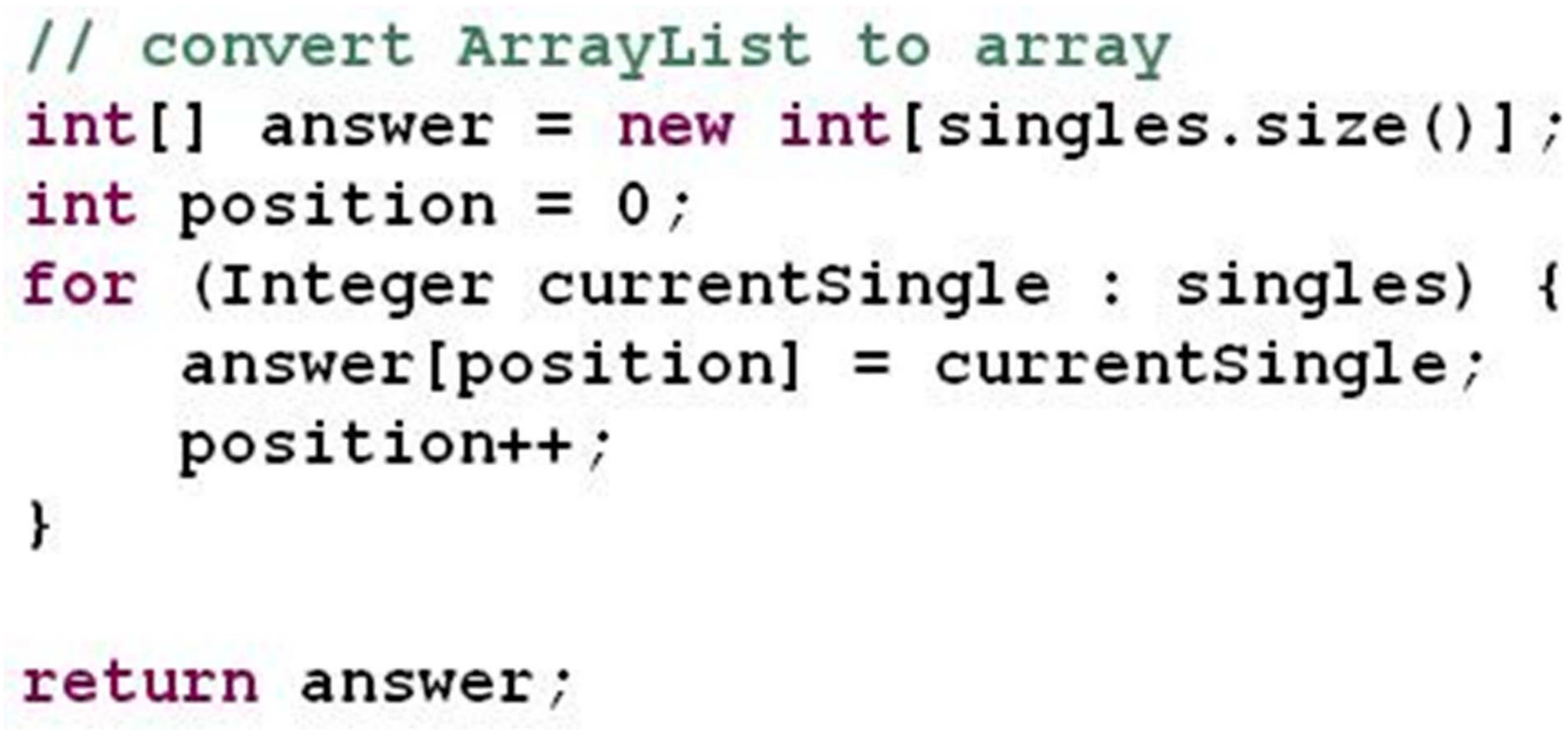


```
public int[] singleNumbers(int[] ids) {

    // convert the array "ids" into an ArrayList "idlist"
    ArrayList<Integer> idlist = new ArrayList<Integer>();
    for (int k = 0; k < ids.length; k++) {
        idlist.add(ids[k]);
    }
```

Second, find unique numbers, how?

Third, convert ArrayList to Array



```
// convert ArrayList to array
int[] answer = new int[singles.size()];
int position = 0;
for (Integer currentSingle : singles) {
    answer[position] = currentSingle;
    position++;
}

return answer;
```

or…

- Convert ArrayList to array

Use ArrayList’s `toArray()` method

```
Integer[] answer =
  (Integer[])singles.toArray();
```

- Convert array to ArrayList

Use Array’s static `asList()` method

```
ArrayList<String> nameList =
  (ArrayList<String>)Arrays.asList(names)
  ;
```

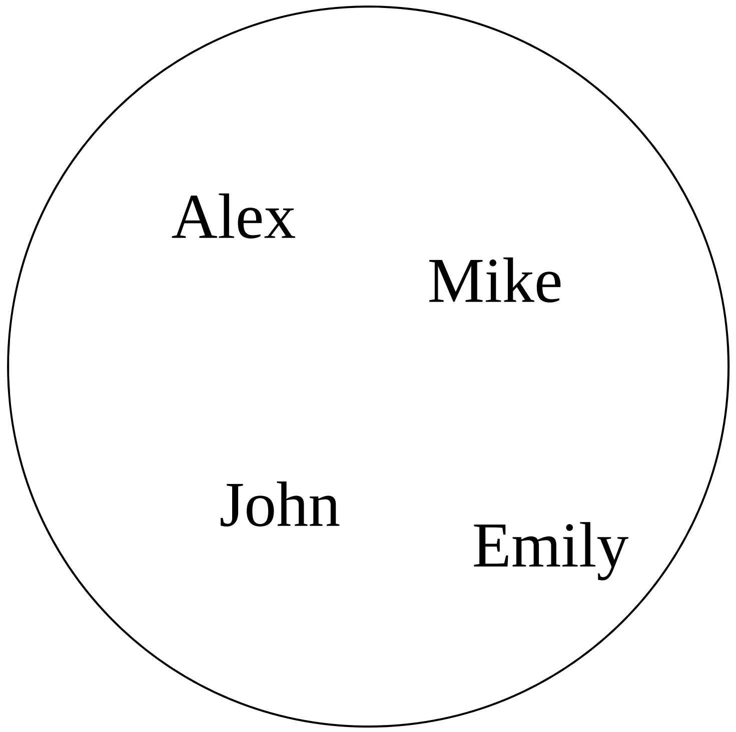
  – *Only works with Objects not primitive types*
  – *names is an array of Strings*

Sets

- Set is an unordered list of items
  - Items are unique! Only one copy of each item in set!
- In Java we will use TreeSet to manipulate a set
  - A TreeSet is a a particular implementation of a Set
  - A TreeSet just happens to store the items in increasing order
  - Another implementation we will use is HashSet – more on that later
- Operations:
  - Create a set
  - Add an item to a set
  - Check if item is in a set
  - Is set empty?
  - Remove item from set

Example – Create and add to Set

```
TreeSet<String> firstnames = new
   TreeSet<String>();
firstnames.add("John");
firstnames.add("Emily");
firstnames.add("Alex");
firstnames.add("Mike");
firstnames.add("John");
firstnames.add("Mike");
```



Example – Is object in set?

```
if (firstnames.contains("Zed"))
   System.out.println("Zed is in the set.");
else
   System.out.println("Zed is not in the set.");
if (firstnames.contains("Mike"))
   System.out.println("Mike is in the set.");
else
   System.out.println("Mike is not in the set.");
```

Iterator – Look at each element in a Set

- Can create an iterator to look at each element in the set
- In general don’t know the order of the elements, however TreeSet implementation does give the elements in order.
- Guaranteed to give you all the elements in the set – one at a time
  - What is this similar to that we have done before?

Iterate over elements in Set firstnames

With collections loop, iterator is Automatically created for you!

```
// Print elements in set
for (String name: firstnames)
{
        System.out.println(name);
}
```

Alternative way to use Iterator

```
// you must create iterator for set
Iterator<String> iter2 =
  firstnames.iterator();
// use iterator to print elements in set

while (iter2.hasNext())
{
   System.out.println(iter2.next());
}
```

Example – Other Operations on Sets

- size() – returns size of set
  - System.out.println("Size of set is " + firstnames.size());
- remove(object) – remove object from set if there
- isEmpty() – return true if set is empty

- See "Sets" and "Iterator" on Java API page

Problems

- Unique Names classwork
- Can you use a set to help in solving classScores?

Set Operations

- Union of two sets
  - all the elements from both sets
- Intersection of two sets
  - the elements that are in both sets
- Difference of two sets (A – B)
  - the elements in A that are not in B

Collections: ArrayList vs Set

- ArrayList
  - directly access an item
  - keep items ordered
  - Can have duplicates of items
  - What are operations on an ArrayList?
- Sets
  - Keeps items unordered
  - No duplicate items
  - Easily remove duplicates
  - What are operations on a Set?

Using Both ArrayList and Sets

- You may want to use a set to get rid of duplicates, then put the items in an ArrayList and sort them!
- Using TreeSet items are stored in sorted order

APT Thesaurus – in lab this week

- Thesaurus
  - Entries of synonyms
  - Entries with two or more common words merge
  - Example:
  - “fa so la” “fa ti do” “la fa me”
  - Results in “fa so la me” and “fa ti do”
  - Sorted is “do fa ti” “fa la me so”