Announcements

• You may only use 1 slip day!! even if you have 2, you can just use 1 slip day per PSA.
• You may “buy” a third one if you run out of slip days (with an additional penalization of -3 points)
The CSE 8B central skills

- **Write**
  - Programs/algorithms
  - Using loops, arrays, conditionals, variables, objects, subclasses, GUIs, recursion, etc (see review sheet)

- **Trace/debug**
  - Draw memory models
  - Given some code, what does it do?
  - Find the errors and fix them

- **Explain**
  - Why things go wrong
  - How things work/why code functions the way it does
  - Why the answer to a question is right or wrong

You will be given reference material. Memorize *concepts* not details.
Write: code that solves an array problem

Write a method that takes an array of ints and returns a new array of ints that contains only a single instance of each of the numbers in the input array.

Let’s assume, to start, that you have the following helper method:

```java
public int countUnique( int[] a )
```

Which will return the number of unique elements in a.

**Hint:**
consider writing a helper method that will check whether an array contains a given element.

```java
public class ArrayPlay
    public int[] removeDuplicates( int[] arr )
{
```
Trace/debug: behavior and memory models
Trace/debug: interpreting Java errors

In what method that you wrote does the error occur?
A. main
B. sum
C. main and sum
D. neither

java.lang.ArrayIndexOutOfBoundsException: 5
    at ArrayPlayReview.sum(ArrayPlayReview.java:7)
    at ArrayPlayReview.main(ArrayPlayReview.java:15)
    at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
    at sun.reflect.NativeMethodAccessorImpl.invoke(Unknown Source)
    at sun.reflect.DelegatingMethodAccessorImpl.invoke(Unknown Source)
    at java.lang.reflect.Method.invoke(Unknown Source)
$\text{Trace/debug: interpreting Java errors}$

```
java.lang.ArrayIndexOutOfBoundsException: 5
    at ArrayPlayReview.sum(ArrayPlayReview.java:7)
    at ArrayPlayReview.main(ArrayPlayReview.java:15)
    at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
    at sun.reflect.NativeMethodAccessorImpl.invoke(Unknown Source)
    at sun.reflect.DelegatingMethodAccessorImpl.invoke(Unknown Source)
    at java.lang.reflect.Method.invoke(Unknown Source)
```

What line is the $\textit{source}$ of the error (i.e. where you should go to fix it? 

A. 5  
B. 7  
C. 15  
D. You cannot tell without looking at the code  
E. Other
Explain: Object Oriented Programming

public class MorePeople {
    public static void main( String[] args )
    {
        Person p = new Person( "Sally" );
        Person s = new Student( "Sam" );
        Student t = new Person( "John" );

        s.setName( "Steve" );
        System.out.println( p.getName() );
        System.out.println( s.getName() );
        System.out.println( t.getName() );
    }
}

class Person {
    private String name;
    public Person( String n ) { name = n; }
    public String getName() { return this.name; }
}

class Student extends Person {
    public Person( String n ) { name = n; }
    public void setName( String newName )
    { name = newName; }
}
public static int find( String[] myList, String toFind, int startIndex )
{
    if ( startIndex >= myList.length ) {
        return -1;
    }
    if ( toFind.equals( myList[startIndex] ) ) {
        return startIndex;
    }
    // what goes here _______
}

A. return find( myList, toFind, startIndex + 1 );
B. return find( myList, toFind, startIndex - 1 );
C. return startIndex + 1;
D. find( myList, toFind, startIndex + 1 );

Explain why each answer is wrong or right!
The CSE 8B central skills

- Write
  - Programs/algorithms
  - Using loops, arrays, conditionals, variables objects, subclasses, GUIs, recursion, etc (see review sheet)

- Trace/debug
  - Draw memory models
  - Given some code, what does it do?
  - Find the errors and fix them

- Explain
  - Why things go wrong
  - How things work/why code functions the way it does
  - Why the answer to a question is right or wrong

You will be given reference material. Memorize *concepts* not details.