Quiz 1

• March 9 (Friday)
• 3:05pm-4:25pm
• Review session:
  – Wed. March 7, 7pm - 9pm
• LA office hour: Thu 7:00-9:00pm
• Open notes/book, no electronic devices
Today’s Recitation

• Keywords: Static

• Array and ArrayList
Static

• Static members:
  – not associated with any particular instance of the class—
    one copy shared by all instances
  – accessible to both static and non-static methods

• Static Methods:
  – may only access static members, **not** instance members
  – should be called using `Classname.methodName()`
public class Number {
    private int num;
    ...
}

Number object 1  Number object 2  Number object 3
num = ...
num = ...
um = ...
static members

```java
public class Number {
    private static int num;
    ...
}
```

Number object 1
```
num = ...
```

Number object 2
```
num = ...
```

Number object 3
```
num = ...
```
public class Number {

    private static int num;
    private int num2;

    public int sum() {
        return num + num2;
    }

    ...
}

Is this ok?
When to Use Static Methods

- When no access to any instance field is required. Usually one of two scenarios:
  - The method takes in all the information it needs as arguments:
    
    \[
    \text{Math.pow(double base, double exp)}
    \]
  - Or, the method needs access to only static variables.
  - Usually you can think of these methods as taking in some information and performing a service for you.
  - Typically, they do not alter the state of the class, as they do not have access to instance variables.
Exercise 1: Static Members

• Write a class Ticket that
  – Keeps track of how many tickets there are
  – Assigns a unique ticket number to each ticket, starting with 100, 101, etc.
  – Has a method to return the number of tickets
  – Has a method to return the ticket number

• Write a class TicketTest that creates some Ticket objects and then prints out how many were created.
Keywords Summary

**public / private**
Control access to data members and methods. **Public** data members can be accessed outside the class, and **public** methods can be invoked outside the class. **Private** data members and methods are not visible outside the class.

**static**
Each instance (object) of a class has its own copy of each non-static data member, but there is only one copy of each **static** data members, shared by all instances (objects) of the class. A **static** method can only use the **static** data members of the class and its input parameters. **Static** methods are invoked on the class name, not on any object name.

**void**
Apart from constructors, every method has a declared return type. If a method does not return anything, its return type must be **void**.

**final**
The value of a **final** data member cannot be modified after it has been initialized.
public class UGrad{
  private double gpa;
  public static final int MAX_GPA = 5;

  public double getGPA() {
    return gpa;
  }

  public void printGPA() {
    System.out.println("GPA: " + gpa + " / " + MAX_GPA);
  }

  public static int getMaxGpa() {
    return MAX_GPA;
  }
}
Array vs. ArrayList

- Size fixed at creation
- Accessed with \texttt{z[ ]}
- Object with no methods
- One public data member: \texttt{z.length}
- Slightly faster

- Size varies as data is added/removed
- Accessed with \texttt{z.get()}
- Object with no data members
- Has lots of methods: e.g., \texttt{z.add()}, \texttt{z.size()}
- More flexible
Array and ArrayList

• Setting and accessing data is different:

```java
int[] array = new int[3];
for(int i = 0; i < array.length ; i++) {
    array[i] = i;
}
```

What does `array` contain?

```java
ArrayList<Integer> arrayList = new ArrayList<Integer>();
for(int i = array.length-1; i > -1  ; i--) {
    arrayList.add(array[i]);
}
```

What does `arrayList` contain?
Array or ArrayList?

• Which would you use for the following problems: an Array or ArrayList?
  – Write a method that returns all the primes between 2 and a specified number
  – Write a method that returns a specified number of random numbers
ArrayList Exercise

• Write a method `findPrimes(int n)` that returns an `ArrayList<Integer>` of all the primes between 2 and n

• Start by putting in all the numbers, then remove the ones that are multiples of each other in the ArrayList

• Some code is provided
public static ArrayList<Integer> findPrimes(int n) {
    //Declare the ArrayList
    for (int i=2; i<=n; i++)
        //Put i in the ArrayList
        int i = 0;
    while //condition to make sure i is a valid index
        int j = i + 1;
    while //condition to make sure j is a valid index
        if // (element j)%(element i)==0
            //remove the proper number from the list
            else
                j++;
        }
    i++;
    }
    //return your ArrayList
}
Array Exercise

• Write a method `makeRandom(int n)` that returns an array of `n` random numbers

• Some code is provided
private static Random r = new Random();
public static int[] makeRandom(int n) {
    //Create your array, then assign random
    //numbers using r.nextInt();
}

<table>
<thead>
<tr>
<th>Skill</th>
<th>Below</th>
<th>Expected</th>
<th>Above</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data types</td>
<td>Types misused. Static not used</td>
<td>Ints and reals used distinguishably. Static used correctly</td>
<td>Int, long, double, boolean, and static used purposefully</td>
</tr>
<tr>
<td>Variables</td>
<td>Numbers and variables used without distinction</td>
<td>Variables used for most quantities</td>
<td>Variables easy to read. Naming is consistent and accurate</td>
</tr>
<tr>
<td>Expressions</td>
<td>Complex expressions not defined correctly; simple expressions ok</td>
<td>Complex expressions organized by parentheses and use of variables</td>
<td>Complex expressions structured for increased clarify</td>
</tr>
<tr>
<td>Loop constructs</td>
<td>Successfully used</td>
<td>Clear and understandable</td>
<td>Appropriate choice of for, while, do-while</td>
</tr>
</tbody>
</table>
# Self assessment, lectures 1-11

<table>
<thead>
<tr>
<th>Skill</th>
<th>Below</th>
<th>Expected</th>
<th>Above</th>
</tr>
</thead>
<tbody>
<tr>
<td>Methods</td>
<td>Methods not clearly defined, or use poor arguments or return values. Wrong use of static</td>
<td>Methods defined but overall structure not always clear. Static used correctly.</td>
<td>Methods organized, named clearly, perform clear behaviors. Static used appropriately</td>
</tr>
<tr>
<td>Method arguments</td>
<td>Few or no arguments used</td>
<td>Appropriate arguments used</td>
<td>Arguments versus data members clearly designed</td>
</tr>
<tr>
<td>Variable scope</td>
<td>Local variable scope inconsistent, often too large</td>
<td>Most variables have appropriate scope</td>
<td>All variables defined just before use and go out of scope after use</td>
</tr>
</tbody>
</table>
# Self assessment, lectures 1-11

<table>
<thead>
<tr>
<th>Skill</th>
<th>Below</th>
<th>Expected</th>
<th>Above</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comments</td>
<td>Some critical comments missing, some not clear</td>
<td>Comments explain basic code</td>
<td>Comments make code self-explanatory</td>
</tr>
<tr>
<td>Indentation</td>
<td>No indentation used</td>
<td>Some indentation used but not consistent</td>
<td>All code properly indented</td>
</tr>
</tbody>
</table>
Homework 4: Scrabble

What are all the two and three letter words that can be made with your hand of 7 letters?

twoLetterWords =

threeLetterWords =