1. (2 pts) Write a **public static** method matching the prototype \( \text{int} \ \text{min} \ (\text{int}, \text{int}, \text{int}) \) which returns the minimum value of its three arguments. For example, \( \text{min}(1,2,3) \) returns 1 and \( \text{min}(1,1,1) \) also returns 1. Note that the \text{min} method in the \text{Math} class has only two arguments; if you want, you can call it multiple times in the body of the \text{min} method that you are writing.

\[
\text{public static int min (int i1, int i2, int i3) } \\
\text{return Math.min (i1, Math.min(i2,i3));}
\]

2. (3 pts) Assume that the **static** method whose prototype \textbf{boolean} \text{isPrime} (\text{int}) is declared in the \textit{Utility} class (it returns whether or not its argument is a prime number). Write a **public static** method matching the prototype \textbf{int} \text{countPrimesBetween} (\text{int, int}) which returns the number of primes occurring between its first and second argument (inclusive). Thus, \text{countPrimesBetween}(101, 200) returns the number of primes between 101 and 200 (including testing 101 and 200).

\[
\text{public static int countPrimesBetween (int lower, int upper) } \\
\text{int count = 0; } \\
\text{for (int i=lower; i<=upper; i++) } \\
\text{if (Utility.isPrime(i)) count++; } \\
\text{return count;}
\]

3. (4 pts) Write a complete call frame (see the notes) for \text{countPrimesBetween}(2,5), showing its hand simulation. When a variable changes state, draw a line through its old value and write its new one afterward.

<table>
<thead>
<tr>
<th>Don't bother writing your header from above here.</th>
<th>Don't bother writing your code from above here.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
4. (8 pts) Assume that the static min method, from Problem 1, has been defined in the Utility class. You can (and must, in the appropriate place) call this method in the code that you are asked to write below.

Write a public static method matching the prototype

```java
public static int sumMininim(DiceEnsemble d1, DiceEnsemble d2, DiceEnsemble d3, int times)
```

which repeats the following process the number of time specified by the last parameter: roll all three dice ensemble parameters, adding up their minimum pip sum. For example, if we specified throwing the dice ensembles two times, and the first throws were 5, 3, and 10 and the second throws were 11, 5, and 4, the returned values, the sum of the minimums, is 7 (3 from the first throws -second dice, 4 from the second throws –third dice)

In the code that you write, (a) draw an oval around every parameter declaration, (b) draw a box around every argument used, and (c) indicate every local variable declared, if any, by drawing a line to it labeled Local. Get the syntax correct and then hand simulate your code to ensure that it works correctly.

Which parameters/local variables can be declared final?

5. (8 pts) Write a public static method matching the prototype

```java
public static String shuffle(String s1, String s2)
```

which returns a result that shuffles the tokens in its two arguments. For example, if we called this method by writing `shuffle("a b c","x y z")` it would return "a x b y c z". Note that spaces in the arguments and results appear only between the tokens (not after every token; this a bit tricky). This method should throw the IllegalArgumentException with an appropriate error message (and thus not return a result) if its two arguments do not contain the same number of tokens. Use the StringTokenizer class to help you solve this problem.