Recall the Academic Integrity statement that you signed. Write all answers clearly on these pages, ensuring your final answers are easily recognizable. The number of points for each problem is clearly marked, for a total of 25 points. I will post my solutions on the web on Monday, off the Solutions link, after class.

1. (2 pts) What does the following code print when it reads a file containing: 1 2 x 4 5
   ```java
   for (;;)
       try { System.out.print( inputFile.readInt() ); }
       catch (NumberFormatException nfe ) {System.out.print("B"); inputFile.readInt();}
       catch (EOFException eofe) {System.out.print("E"); break;}
   System.out.print("D");
   ```

   It prints:

2. (4 pts) (a) Translate the following for/if/break loop into both a while loop and a do loop. Write below only whichever code you think is simpler.
   ```java
   int x;
   for (;;) {
       x = Prompt.forInt("Enter value");
       if (Math.isPrime(x))
           break;
   }
   System.out.println(x + " is a prime number");
   ```

   (b) Write a general rule (in English) for deciding when a for/if/break loop more easily translates into a while loop and when into a do loop, and when it should probably stay a for/if/break loop. Hint: location of the break.

3. (3 pts) (a) Briefly describe the difference between testing and debugging. (b)What is the golden rule of debugging? (c) What is Job #1 in debugging? Memorize these after you look them up in the notes.

4. (2 pts) Rewrite/simplify the following if statement (from the Collatz program) by using a conditional expression.
   ```java
   if (currentValue%2 == 0)
       currentValue = currentValue/2;
   else
       currentValue = 3*currentValue + 1;
   ```
5. (4 pts) The Eclipse debugger has two major (and quite different) functions that aid in debugging. What are they? One of these functions can be done in two quite different ways. What are these two ways?

6. (3 pts) One of my pet-peeves is to not write code that tests a boolean value to see if it is `== true`. Assume that we both declare and initialize `boolean traceInDetail = Prompt.forBoolean("Trace in detail?");` fill in the following truth table to prove that the expressions `traceInDetail` and `traceInDetail == true` are equivalent: they always evaluate to the same value.

<table>
<thead>
<tr>
<th>traceInDetail</th>
<th>traceInDetail</th>
<th>traceInDetail == true</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

I also don’t like writing `traceInDetail == false`. Rewrite the code on the left to avoid checking this equality with a `boolean` literal (hint: use a logical operator to compute the right truth value).

```java
if (traceInDetail == false)   if (                     )
...        ...
```

7. (4 pts) Rewrite the following code (concentrate on the loop) as general `for` loop with as much information in the parentheses as possible; the answer is a bit weird. When you translate the termination condition into a continuation condition, use DeMorgan’s laws (and other relational equivalences) to simplify this condition.

```java
int x = Prompt.forInt("Input x");
int c = 0;
for (;;) {
  if (x > 5 || x == 0)
    break;
  c++;
  x = Prompt.forInt("Input x");
}
System.out.println("looped " +c+ "times");
```

8. (3 pts) When the code on the left runs, nothing is printed. (a) Explain why nothing is printed. Hint: It is really screwy; look carefully at this code. Feel free to run it (in Java or the Eclipse debugger) to help you find the problem. (b) If we rewrote this code as the `for` loop on the right, it won’t even compile. Explain why. (c) What conclusion can one draw from (a) and (b)? PS: I don’t like the indentation style used here.

```java
int i = 1;
while (i<=10);
{
  System.out.println("i is now " + i);
  i++;
}
for (int i = 1; i<=10; i++);
{            System.out.println("i is now "+ i);
  System.out.println("i is now " + i);
  i++;
}

(a)  (b)  (c)