Extend the Employee class posted as Assignment 3 solution on the course website with the following changes:

1) Write an equals method for Employee class. The method takes an employee object as argument and returns true, if two employee objects have the same values for name, salary, department and numberOfYears.

2) Implement “id” as a static integer variable as part of the Employee class. The “id” static variable will keep track of new employee id numbers starting from 1001 and incrementing by one for each new employee.

3) Write a “toString” method for Employee class which will return a string representation of an Employee object with complete details of their name, idNumber, new salary, and their department.

4) Add javadoc comments to both Employee and TestEmployee classes

All other rules about employee pay raise will remain the same from the previous assignment. Modify the TestEmployee class to accept two employee details, and demonstrate equals and toString method calls.

A sample dialogue is given below. User input is shown in bold

```java
>>java TestEmployee
Enter first employee's information.
Name          (ex. John Smith): Joe Smith
Base Salary     (ex. 30000.00): 40000
Department     (ex. Marketing): Research
Number of Years (ex. 2.5): 1.5

Enter second employee's information.
Name          (ex. John Smith): Jane Texas
Base Salary     (ex. 30000.00): 35000
Department     (ex. Marketing): Accounting
Number of Years (ex. 2.5): 7.5

Demo of equals
employee1.equals(employee2): false

Demo of toString
employee1.toString():
Joe Smith, 1001, 40000.00, Research

Employee2.toString():
Jane Texas, 1002, 36400.00 Accounting
```

The grading criteria for this and future assignments is given by the following table:

<table>
<thead>
<tr>
<th>Grading Criterion</th>
<th>Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>Does not compile</td>
<td>0</td>
</tr>
<tr>
<td>Compiles</td>
<td>8</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>----</td>
</tr>
<tr>
<td>Input, Output and Computation is valid</td>
<td>52</td>
</tr>
<tr>
<td>Test cases</td>
<td>20</td>
</tr>
<tr>
<td>Coding Style</td>
<td>10</td>
</tr>
<tr>
<td>Documentation</td>
<td>10</td>
</tr>
</tbody>
</table>

**Submission**: Combine all source files into one folder, zip the folder and upload the zipped folder on d2l.mu.edu before the class time on the due date, and also submit a paper copy of your submission in the class.