**HW Refactoring**

You will work in your project group. One person from your group should submit this on Blazeview.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Group |  | | Group Member Names |  |
|  | |  |  | | --- | --- | | 1. |  | | 2. |  | | |  |  | | --- | --- | | 3. |  | | 4. |  | |
|  |  |  |

**Requirements**

1. You will refactor aspects of your project code. You may have already done some, which is fine. I’ll get to that shortly. You must make 3 different refactorings and document them. The refactorings considered in the video lecture are shown below. If I missed any from the video, or there are others you have found out about through reading, you can use those to.

collapse hierarchy, consolidate conditional expression, decompose conditional, extract class, inline class, extract method, feature envy

Comments:

1. The *extract method* refactoring is the easiest and perhaps the most useful. You could probably apply it 10 or more times in your code base (my guess!). And you should. However, you can only use one instance for this assignment.
2. Also, usually easy to spot are: *consolidate conditional expression* and *decompose conditional*
3. *Feature envy* is actually a *code smell* which can be resolved in any number of ways: introducing a method in the class where the “envied” fields exist, sometimes moving the “envied” fields to the “envious” class. And, there are others. I suspect that *feature envy* exists in your code base.
4. As noted in the video, you can highlight code, choose: Refactor, and then choose the proper refactoring technique (I don’t think all are available in Eclipse, but many are)

Finally, if you have already done some refactoring, you can use those too. You’ll just need to roll back your code temporarily so that you can see before and after. Unless you have done extensive refactoring, or you have done one or two that are significant, I recommend using this as a chance to clean your code base up!

**Time spent on this assignment IS logged in the time log.**

**Refactoring 1**

1. *Put name of refactoring here*
2. Discuss why this refactoring is needed and useful. Discuss in HOW you refactored it. What did you specifically do and why.
3. Show the code before the refactoring here (mono-spaced font only):
4. Show the code after the refactoring here (mono-spaced font only):

**Refactoring 2**

1. *Put name of refactoring here*
2. Discuss why this refactoring is needed and useful. Discuss in HOW you refactored it. What did you specifically do and why.
3. Show the code before the refactoring here (mono-spaced font only):
4. Show the code after the refactoring here (mono-spaced font only):

**Refactoring 3**

1. *Put name of refactoring here*
2. Discuss why this refactoring is needed and useful. Discuss in HOW you refactored it. What did you specifically do and why.
3. Show the code before the refactoring here (mono-spaced font only):
4. Show the code after the refactoring here (mono-spaced font only):