Programming Project 2 Grading Rubric

1. Testing: 20 points

Points	Metric
20	A description of how to thoroughly test all the methods of the assignment. Completed tests for each method written in the student's submission. Each test gives the true output of the student's submission. A note on any test that gives an incorrect result.
18	Tested all the methods, but may have missed a few obvious cases.
15	For at least three quarters of the methods, the tests and/or descriptions of what to test covered.
12	For at least half of the methods, multiple tests and/or descriptions of what to test cover.
10	Some testing done.
0	No testing done.
-10	Includes output for methods that were not in the student's submission OR gives any results misrepresenting how the student's code works.

2. Readability: 20 points

Basic program structure

Points	Metric
10	Correct comments at top of class, above each method and above each field. The comments are at most one to three sentences and are a good description. Class has fields first, then constructors, then the remaining methods. loop. All code follows the proper indentation and naming style. (See 2.16 of the recommended textbook.) There is a blank line between each method.
9	Mostly indented well (up to two places not indented correctly) AND most names correctly follow the naming convention (up to two poorly named variables or methods) AND comments on most methods, constructors, fields, and the top of the file (up to two missing comments). Each comment is a good description. Also, the code is organized correctly with at most two methods in the wrong location.
7	More than two places of poor indenting OR more than two method or variable names that do not follow the Java convention OR more than two missing comments OR comments are longer than one to three sentences OR comments are direct copies of the homework description OR the fields are not at the top of the file followed by the constructors and the instance methods.
6	Comments missing from at least half of the places that require them OR more than half of the methods are poorly indented OR the code is very disorganized
2	Missing comments in the code OR poor indentation throughout the code
0	No comments in the code AND poor indentation in the code.

JavaDoc Comments

Points	Metric
10	All comments above the class, fields, and methods follow the JavaDoc comment style. Correct tags are added for the author, all method parameters and return types.
9	All comments above the class, fields, and methods follow the JavaDoc comment style. Missing only a few tags or at most 5 places the tags are misspelled.
7	Most comments above the class, fields, and methods follow the JavaDoc comment style, but many places are missing tags or the tags are misspelled so the comments are not formatted properly on the webpage.
5	Most comments in JavaDoc style of /** but missing most tags OR has most tags but the comments do not start with /** and so do not appear on the webpage.
2	Comments only provided for one class.
0	No JavaDoc comments provided.

3. Correctness: 60 points

Performance correctness: 40 points

Points	Metric
40	All classes completed. All methods work correctly and produce appropriate output. The toString and equals methods correctly override the Object method and the equals correctly compares the objects to determine equality.
36	All but one class is incorrect OR one method of `value` or `toString` is incorrect across many classes.
30	All but two classes are incorrect OR one method of `equals` or `derivative` is incorrect across many classes OR two other methods are incorrect across many classes.
25	At least four classes correct OR at least one of `equals` and `derivative` is correct across all the classes.
20	At least three classes are correct OR at least two of the `value` and `toString` methods are correct on all the classes.
10	At least one class is correct.
0	No working methods or constructors.

Organization correctness: 20 points

Points	Metric
20	The class heirarchy is well organized and intuitive and polymorphism is used to simplify the code everywhere that is appropriate.
	The class heirarchy is well organized but one class is not in the appropriate spot and polymorphism is

18	used to simplify the code everywhere that is appropriate.
14	A hierarchy is created but two classes may be in the wrong place and polymorphism is correctly used OR a good hierarchy is created but polymorphism is only used a few places where it can simplify the code.
10	A hierarchy is created that has errors AND some polymorphism is used for the problem, but the code could be simplified.
5	Some attempt at creating a hierarchy but no polymorphism is used.
0	No class hierarchy is created, no polymorphism is used.