SWE 619 Fall 2020
Final Paul Ammann

This is an open-book/open-notes exam. This means that you can access course materials on paper or on the internet. The time limit on this exam is **2 hours and 30 minutes**.

It is a violation of the honor code to communicate with any other person (except me, the instructor) about this exam while you are taking it.

It is a violation of the honor code to discuss or share the contents of this exam in any way with any student who is currently registered for this course but who has not yet completed this exam.

When you see a reference to code, you should map that to the relevant Java file I provided as part of your study guide for this exam.

Capture your answers as a PDF document and submit on Blackboard by 7:00PM. If, for any reason, you have a problem submitting to BB, submit your final on Piazza in a private post. Your post should also explain your problem.

Please make sure your **NAME** is at the top of your submission.

Code for the examples used in this exam is published on guide discussed in class. See the course web page for the relevant link. Make sure you are getting the right code!

Section	Points	Score
Question 1	30	
Question 2	35	
Question 3	35	
Total	100	

Question 1

Consider `StackInClass.java`. You should take special note of the `push()` method; it is a variation on Bloch's code.

1. Write a total contract for `push()`
2. What is a good rep-invariant for this version of the `Stack` class?
3. Is `pop()` correct? Be sure to explain your reasoning.

4. As written, `pushAll()` violates Bloch's guidance on exception handling. Why? Fix it.
5. Write a reasonable implementation of `clone()`. Note: Don't worry about Java syntax; ”exam” Java is fine.

Question 2

Consider `Queue.java`.

1. Write a reasonable `toString()` implementation.
2. Consider a new method, `deQueueAll()`, which does exactly what the name suggests. Write a reasonable contract for this method and then implement it. Be sure to follow Bloch's advice with respect to generics.
3. Rewrite the `deQueue()` method for an immutable version of this class.
4. Write a reasonable implementation of `clone()`. Note: Don't worry about Java syntax; ”exam” Java is fine.

Question 3

Consider `MapPoly.java`.

1. Bloch would not accept that the `MapPoly` class is immutable. Why not? Show how it would be possible to provide mutable behavior with the class if Bloch's problem isn't fixed. Fix the problem.
2. Consider implementing `Cloneable` for this class. Decide whether Bloch would think this is a good idea and provide justification for your answer. Note: You don't have to actually implement anything for this question.
3. Provide reasonable implementations of `equals()` and `hashCode()`. Explain why you believe your implemetations are appropriate.