COSC 310 – Project 1
Duck, Duck, Goose

Goal: Implement the Duck, Duck, Goose simulation describe on page 148 of the *Data Structures & Algorithms in Java*. (P-3.43)

Description: Create a program that plays the children's game Duck, Duck, Goose. The design of the software is up to you, but correctly implementing and using data structures is a key component of this assignment. (So please do not use a data structure that you did not code yourself). Your simulation should also have a graphical component (using Java's Swing library) and be properly documented. The visualization does not need to be complex, but it should show what is happening in the program.

Deliverables:

1. Program source code – Use edu.frostburg.cosc310 as the base package for your code. Use any IDE that you prefer, but only submit the source code (.java files) and a runnable JAR file (rather than the IDE directory). Name your jar LastnameFirstname.jar.
2. Visualization – A main loop can redraw a set of shapes representing the children.
3. Code style – Professional code looks as great as it functions.
4. Documentation – Describe all the important parts of your code. You should not assume I know anything about your assignment (other than Java).
5. Readme.txt – A short file that describes what someone would need to do in order to get your program running.
6. Post-mortem – PDF file describing your assignment: how you approached it, the problems that you ran into, and things you liked or didn't. The goal of a post-mortem is to determine *what was successful and what wasn't*, so you can apply what you have learned towards future projects.

Tips:

- A circular linked-list lends itself naturally to this assignment.
- Design is not a subjective activity. Ask yourself details about your program to see if yours makes sense. (e.g. Should kids refer to their neighbors, or should nodes? What details do we need in order for them to play?)
- Consider the MVC pattern. Keeping your visual aspects separate will make your coding simpler.
- Think about what you need for your program. Should you have a class for kids? For the output? How will you approach this? Draw UML for yourself.
- How do you do something complex? Why not try writing out your pseudocode?
- Separate your program into modules and work on the modules individually.
- This program will take you longer than you think. Start soon.
- In fact, start right now.
- Your textbook is full of helpful details

Checklist:

Check	Component	Grade %
	Duck, Duck, Goose simulation (Design & Implementation)	60%
	Visualization	15%
	Code style (clean and professional)	10%
	Documentation (comment style)	5%
	Other documentation (post mortem)	5%
	Deliverables, packaged in a zip: 1. Source code (.java) 2. YourProgram.jar 3. Readme.txt 4. Post-mortem (PDF)	5%

Avoid:

Check	Component	Grade %
	Compilation errors	-50+%
	Typos	-10%
	Bugs	-x%