Developer Log

Directions: Each time you make a significant advancement or change to your program, or after working on it for a while, add an entry to this log. This will help you and your teacher keep track of your thinking. Now that you've made a copy, change the title to include your name and share this with your teacher.

One example row is done for you in gray.

Date	*Time (roughly)*	*How long did you work on your project?*	*What did you accomplish? What resource(s) did you consult (people or websites) and what did you learn from them?*	*What did you get stuck on (even if you're still stuck on it)?*	*What do you want to work on / get done next time?*
02 Apr	11:00 am	45 mins	I mostly just researched to think about what I want to do or simulate.	I think I have an idea of something with zoos but I'm not sure how deep I want it to be.	I want to make my UML diagram for my program.
05 Apr	1:00pm	80 mins	- Brainstormed the idea for my project. - Started writing my resource classes.	I originally wanted to make a sort of friendship meter for this project and I was stuck on that idea for a long time before I decided I wanted to simulate a day in a game design firm.	Continue writing my resource classes. Write a driver class.
05 Apr	7:00pm	53 mins	- Decided to cut out Employee class and remove Employer class from the hierarchy because they were not useful in their current states. - Created working Worker, Coder, Designer, and Janitor classes.	I got stuck on the inheritance for the Employer constructor and how to use the super method but I ended up getting it to work.	- Talk to Mr. Ayyappan about how I can incorporate more inheritance and recursion into this project. - Add Employer class and get it to count revenue and calculate how much money the firm loses each day to salary.
07 Apr	8:20am	60 mins	- Deleted the Employer Class. Everything I wanted to do with it can be done in the driver	- Having a lot of issues writing recursion methods such as	- Write a working firmOutput Method. - Finish the driver class.

			class. - Transferred code from Worker class to its children to have a more accurate simulation. - Relearned how to use the super method and implemented it in my child classes. - Completed my class hierarchy diagram. - Wrote a completed array in my driver class. - Wrote the firmOutput() method in my driver class (incomplete). - Modified the call to the firmOutput method so that it can be called in 'main' by changing the parameters and changing the method itself to static.	firmOutput() inside of the driver class. The reason for this is that I am having issues writing variables that won't self destruct when the method goes 'deeper'. - Calling methods written in the driver class to 'main'.	
07 Apr	9:55am	25 mins	- Finished the firmOutput() method. - Wrote an incomplete profits() method.	- Found a bug that I am having issues fixing where the amount of time that the employees worked for is completely wrong. (It says that some worked for 22 hours in a day). This probably has something to do with the original workDay method in the Worker class. It may also have something to do with the super method in the constructors of the child classes not working. - The profits method does not work because it	- Update diagram to include the getProfit() method in the Worker class being modified by the child classes. - Fix the profits() method. - Fix the issue with time.

				constantly calls to the Worker class getProfit() instead of the getProfit() method in its child classes.	

09 Apr	12:30	65 mins	- Fixed the profits() method in the main class by actually assigning the value it returns from getProfit() to a variable.
			- Fixed the issue with time by changing “=” to “==” inside of a boolean if statement.
				- My brain breaking after Mr. Ayyappan explained to me how simple the errors I was stuck on for 2 days were.	Phase 2: Make your program fit the rubric.
					- Add inheritance to Worker class from the Object class.
					  - Added an overwrite to toString() and equals()
					- Recursion
					  - Add an overload of a recursion method with an iterative loop
					  - Add a second base case to profits are too low to make sure that it doesn’t go through if the profits are too high (glitch or robbery)
					- Downcast by changing types into subtypes (so you can get rid of the getProfits() method in the worker class.
					
					- Work on 2 MCQs
					- Update the class hierarchy diagram.
					  - Add a sub designer class
					- Compare recursive method with iterative loop (method).

12 Apr	2:10	50 mins	- Overwrote the toString method in the Worker class and its child classes. - I changed the code to be more efficient for all	N/A	- Add an overload of the firmOutput() method to act as an iterative solution to a modified problem of only a select number of employees

			of the doWork() methods to get this to work. - Wrote an overwrite of the equals() method in the main class to only return true if objects were the same type.		being counted. - To count the select number, utilise the overwrite of the equals() method to see if they are the type one wants to include. - Recursion - Add an overload of a recursion method with an iterative loop - Add a second base case to profits are too low to make sure that it doesn't go through if the profits are too high (glitch or robbery) - Downcast by changing types into subtypes (so you can get rid of the getProfits() method in the worker class. - Create a grandchild class (SoundDesigner) - Work on 2 MCQs - Update the class hierarchy diagram. - Add a sub designer class - Compare recursive method with iterative loop (method).
12 Apr	6:30	50 mins	- Add an overload of the firmOutput() method to act as an iterative solution to a modified problem of only a select number of employees being counted. - Made an overload of the same nature for the profit() method	- Had some troubles figuring out how to count the number of each type of worker there are in the array inside of the firmOutput() overload method.	- Downcast by changing types into subtypes (so you can get rid of the getProfits() method in the worker class.) - Create a grandchild class (SoundDesigner) - Work on 2 MCQs

			because why not. - Added a second base case to profit() to print if the profits were unreasonably low or high.		- Update the class hierarchy diagram. - Add a sub designer class - Compare recursive method with iterative loop (method).
13 Apr	8:41	28 mins	- Created a grandchild class (SoundDesigner) and implemented its features into the driver program. - Downcasted by changing types into subtypes to get rid of the getProfits() method in the worker class.	- Had some issues with downcasting as I needed to remember how to do that.	- Work on 2 MCQs - Update the class hierarchy diagram. - Add a sub designer class - Compare recursive method with iterative loop (method).
13 Apr	10:05	25 mins	- Bug fixes. - Updated the class hierarchy diagram. - Add a sub designer class		- Work on 2 MCQs - Compare recursive method with iterative loop (method). - Write a justification for the class hierarchy diagram.
13 Apr	6:50	30 mins	- Compared recursive method with iterative loop (method). - Part 1		- Work on 2 MCQs - Finish Comparing recursive method with iterative loop (method). - Write a justification for the class hierarchy diagram.
13 Apr	7:30	30 mins	- Compared recursive method with iterative loop (method). - Part 2		- Work on 2 MCQs - Write a justification for the class hierarchy diagram.
14 Apr	8:50	110 mins	- Completed both MCQs	Issues with brainstorming the MCQs but I ended up powering through and getting them done.	- Write a justification for the class hierarchy diagram.
14 Apr	12:05	30 mins	- Wrote a justification for the class hierarchy diagram.		- Record and edit video.