**Requirements**

You must build atop the vending machine hardware already provided.  You may not alter the hardware classes.

You must implement fully functional logic for the vending machine:

* Customers must be able to press pop selection buttons and enter coins, in order to ultimately purchase pops.
* Again, Canadian coin denominations must be supported.
* When the machine contains no credit for the user, a message "Hi there!" should be displayed for 5 seconds and then should be erased for 10 seconds, with this cycle repeating.
* When the user enters valid coins, the message "Credit: " and the amount of credit should be displayed.
* The machine should provide change once a pop is vended.  If the exact change can be provided, it should be, and the credit should go to zero.  If the exact change cannot be provided, the amount as close as possible to the exact change should be returned, but without going over; any remaining amount that cannot be returned should still exist as credit
* Whenever exact change cannot be guaranteed for all possible transactions, the "exact change only" light should be turned on; otherwise, it should be off.
* If the machine cannot store additional coins or otherwise becomes aware of a problem that cannot be recovered from (including being out of all pop), the "out of order" light should be turned on; if the safety is enabled, this light should also be turned on; otherwise, it should be turned off.
* Each action of the user and the actions of the machine that are visible to the user should be logged in a text file (called the "event log").  You can determine the format of this text file.  Be aware that the vending machine possesses an internal clock, as per a standard Java virtual machine.

In addition, the design goals for this vending machine include:

* easy extensibility to other forms of payment, including mixed modes (e.g., paying partially with coins, partially with credit card)
* ease of changing details on communication with the user and to the event log
* ease of supporting alternative hardware

"Ease" means that code simply has to be added (e.g., one new subclass) and other modifications are (mostly) not needed.  Having to rewrite your entire system does not constitute "ease", in my opinion, for example.

**DO NOT** implement other forms of payment (for now).  Your design ought to support such new forms of payment when they are added "in future".

**Deliverables**

Your team will provide **five**items in your submission:

1. your source code that supports the above requirements and design goals;
2. a JUnit test suite that (a) unit tests your logic, and (b) system tests the vending machine with the logic installed;
3. one or more diagrams describing how the hardware and your logic support the user's actions;
4. a 1-page justification of your design, relative to the design goals above; and
5. the log file from your team's Git repository showing the history of changes that you make as you go.

In addition, each student must submit a **peer-/self-evaluation** of your own and your teammates' contributions to the solution, as an Excel spreadsheet.  You will give a score to each of your teammates and yourself.  **The scores must sum to zero.**Someone with a positive score was more beneficial than average; a negative score is less beneficial than average.  In addition, you should provide a short justification at the bottom of the spreadsheet as to why these were your opinions.   These evaluations will be strictly confidential.

Your opinions should be supportable by evidence in the Git log file.

Submission must be made to this Dropbox folder by the deadline.  For technical reasons, it is an individual submission folder: one person on your team can submit the team deliverables or multiple people can do so: we will use the latest one for evaluation.  Each person will have to individually submit the peer/self evaluations.

**Mr. Client**

Mr. Client cautions: "This description is good enough for you to get started, but we're working on certain changes to the hardware that will likely impact the requirements.  Be prepared to make adjustments to your design and implementation.  I'm not sure when our hardware changes will be ready to roll out for you.  We need to work out some issues with the vendors of some of the hardware components."

"Any time you find any issues with the hardware, we'll have to fix it on our end.  Make sure you give us some time to achieve this.  I know: it's a moving target, but that can't be helped."

"Oh, did I remember to tell you that the vending machine absolutely must be purple!  The marketing team will have my head if I forget that."

**Grading**

10% of your grade will be for the quality of your peer/self evaluation.

The other 90% of your grade will be based on the team's success, but adjusted according to individual contributions:

* 20% of your grade will be for the models.
* 20% will be for completeness in meeting the requirements.
* 20% will be for design.
* 30% will be for testing.

**Live Demos**

We are considering having each of the 9 teams give a live demo of their tool to Sajjad.  We are figuring out whether this is feasible.  ("Heads up": Expect to have to give a live demo in connection with the final assignment, probably in lecture time.)