UNBC **CPSC 101 Team Term Project** Winter 2009

version 1 **David Casperson**

Teams: The list of teams and members will be distributed later this week. It is each team's responsibility to distribute work amongst itself and find appropriate times to meet.

Project: *Design* and *implement* a vending machine simulator using the techniques of object-oriented programming in JAVA. The specifications and the problem statement are attached.

Design (30%): Read the sheet "Thinking about Objects". As a group, discuss how you might apply these concepts to your programs. Submit a design document that includes

- table of contents.
- a list of nouns,
- precisely worded paragraphs describing each noun,
- a list of facts,
- a list (by class) of attributes, behaviours, and messages sent to other objects,
- a cover, and a
- the percentage of work completed by each member of the group and provide the proposed distribution of workload for the implementation part of the project.

Some textbooks emphasize formalized design using UML diagrams. While I strong encourage you to learn as much UML as you can on your own, I am not going to teach it, or require it of you. In particular, *UML is not necessary in your design document*.

The above list gives minimal requirements for your design document. Include other information that you think will help you design and implement your project.

Due Date: February 06, 2009

Revised Design (0%): I shall attempt to constructively criticize the design documents and return them to the project groups by early in the week of Februabry 13. I may insist in some cases that the design be re-submitted before work on the implementation begins.

Due Date: February 13, 2009

Implementation (50%): Complete the implementation of your simulator by writing code as outlined in Elevator Laboratory Assignments 5 and 6. Thoroughly test your program and provide the following:

1. Your revised design document. **Due Date: March 20, 2009**
2. Complete listings of your program. **Due Date: March 20, 2009**
3. Output for three representative simulations showing how to use your program. **Due Date: March 20, 2009**
4. A short users' guide explaining how to find and use your program(s). **Due Date: March 20, 2009**
5. An itemized list of contributions of each member. **Due Date: March 20, 2009**

6. A 5–10 minute presentation of your work and a demonstration of the simulation (to be scheduled).

Note: As you proceed with the implementation, you may want to revise your design. This is acceptable, and even encouraged. However, *all changes to the original design must be clearly documented with the reason as to why thoses changes were necessary.*

Testing (10%): Your group will make your program(s) (but not the object code or source) and users' manual available to another group in the class assigned by the instructor. That group will thoroughly test your code and provide feedback for improvements in the design, interface, *etc.* You will do the same for a different group, and supply a copy of your written report to the group you review and to the instructor.

Due Date: March 30, 2009

Revision (10%): You may agree or disagree with the suggested changes and criticisms contained in the report reviewing your project. In case you agree, you may choose to revise your design and implementation. If you disagree, you may submit a critique of the proposed changes. In either case, you will submit a formal final report.

Due Date: April 06, 2009

Notes:

1. You will work on this project as a group. It is your responsibility to divide the work within the group and to make sure that your group functions effectively.
2. All submissions *must be* formal reports, meaning that they must be neatly presented and bound, grammatically correct, correctly spelled, and professional looking.
3. Both the testing report and the final written response should be written as objectively and professionally as possible. In particular: they should have opening and concluding paragraphs; and they should state who is reviewing or responding to what (by whom). Statements like "*Obviously, the group who reviewed our project never bothered reading p.96 of our user manual*" automatically lose marks (whether or not they are true).

Problem statement

A company is designing a new vending machine. The company wants you to develop an object-oriented software simulator so that they can see whether the machine that they are developing will meet the customer's and service people's needs.

The company's vending machine is similar to many of the candy vending machines at UNBC, but smaller. There are ten rows and eight columns of spring coils that can hold chips, candy, and other merchandise. Each spring coil can hold up to 10 items.

The machine operates in two modes, depending on whether or not the front door is open. When the door is shut it operates in vending mode. Customers may enter 5¢, 10¢, 25¢, $1.00, and $2.00 coins[1]. They may also press buttons labelled "A" through "J" to select the row, and "1" through "8" to select the column of the item they wish to purchase. They may also push the coin return button at any time.

In vending mode, the machine normally responds to coin entries by displaying the current total entered on a small display. When the machine's customers push one of the buttons their choice is displayed instead. When a customer completes a choice one of three things happens. Either the machine dispenses the requested item and releases the customer's change through the coin return; or it displays a message saying that the requested selection isn't available; or it displays a message saying that the customer hasn't entered enough money to purchase the item. When a customer presses the coin return button any unspent money in the machine is returned through the coin return.

When the door is open, the machine operates in restock mode and none of the normal vending functions are available. In this mode it is possible:

- to set the price for each of the spring coils;
- to refill or partially refill each of the spring coils;
- to empty the cash box;
- to refill or partially refill the change box; and
- to close the door.

The coin-handling mechanism of the vending machine consists of a coin entry slot; a coin return slot; a cash box; and a change box that has a column for each kind of coin. When a coin is put in the entry slot the machine can control whether it goes back out the coin return slot or gets dropped into the cash box. The machine can also direct the change box to drop a coin from a particular column into the coin return slot. Note that once coins enter the cash box the machine cannot move them elsewhere. When the change box is close to empty the machine can display a message asking the user to enter exact change only.

Your simulation should allow the simulation user either to directly control the events that happen or enter an automatic mode where customers and machine restockers arrive randomly and use/restock the machine. It should be able to help answer questions such as how much change needs to be in the change box, so that most of the time the machine runs out of stock before it runs out of change. In order to do this your simulation needs to keeps statistics as it runs and print them out on request, and needs to use random number generators to simulate customer behaviour.

[1] the vending machine rejects 50¢ coins