

Workshop Grading and Promotion Policy

Workshops for this course will be assessed using the following criteria:

- Workshops are graded based on two components:
 1. Individual Logic Assignment (40%)
 - Individual work is due **2 days after the assigned date** (class) by **end of day 23:59 EST**
 - Individual logic assignments are to be done individually
 - Members who do not submit work on-time, will receive a zero grade for the workshop
 - Members who receive a zero grade for the individual part, will not be eligible to receive grades for the group solution part
 2. Sub-Group Overall Solution (60%)
 - Group solution is due **4 days after the assigned date** (class) by **end of day 23:59 EST**
 - **Name and ID of all contributing members must be stated at the top of all file submissions**
 - If not submitted on-time, a zero grade will be applied for the group portion of the workshop
 - If the submitted solution is essentially a copy of the individual parts thrown together containing no effort to properly integrate as a seamless overall solution, a zero grade will be applied for the group portion of the workshop
- A zero grade on a workshop will not be counted towards the minimum necessary number of completed workshops
- Video presentations are due **1 day after your next class by end of day 23:59 EST**
 - Each student must do a video presentation **at least once** by the end of the term and should minimally consist of the following:
 - Description of the problem and its solution in non-technical terms. You should assume your audience is non-technical and interested in using your application solution.
 - Market your application solution by providing sample screenshots of how you envision your application to look which should include a sample workflow demonstrating how easy it is to use
- You must **successfully complete 9 workshops** (if > 9 are completed, the best 9 will be used)
- Workshop solutions and presentations will be evaluated using the published workshop rubrics

Group Breakdown

Each group has **two sub-groups** determined by the assigned **member number**:

Sub-Group 1: Members 1-3

- **Member-1:** Responsible for doing workshop **Logic 1**
- **Member-2:** Responsible for doing workshop **Logic 2**
- **Member-3:** Responsible for doing workshop **Logic 3**

Sub-Group 2: Members 4-6

- **Member-4:** Responsible for doing workshop **Logic 1**
- **Member-5:** Responsible for doing workshop **Logic 2**
- **Member-6:** Responsible for doing workshop **Logic 3**

Sub-Group Solution

- Each sub-group is a team and **must work together** creating the overall group solution
- The group solution is not to be done by an individual. The group solution is expected to be a seamless solution (looking as though one person has done it) and has undergone refinement and testing to ensure the logic properly addresses the workshop problem.
- If the submitted work amounts to essentially copying and pasting everyone's logic part together, a zero grade will be applied for the group work portion.

Work Submission

All work must be emailed to your instructor. You must follow the email guidelines described below.

- All work submitted (applied to both individual and group submissions) **requires all contributing members names to be stated at the top of all files being submitted**

Email Subject Line

- Highlighted parts indicate your specific information
- There are no spaces
- **APS145-[SECTION]-WS[#]:Group[#]**
 - Example: APS145-NAA-WS1:Group3

File Attachment

Individual Work Submissions

Attach a file containing your work (**pseudo code** OR **flowchart**)

- Highlighted parts indicate your specific information
- **Pseudo code**: **logic[#].fullname.pseudocode.txt**
 - Example: logic2.Cameron Gray.pseudocode.txt
- **Flowchart**: **logic[#].fullname.flowchart.jpg** (*Note: .jpg or .png*)
 - Example: logic3.Cameron Gray.flowchart.png





Sub-Group Solution Submission

Attach a file containing your group work (**pseudo code** OR **flowchart**)

- Highlighted parts indicate your specific information
- There are no spaces
- **Pseudo code**: **ws[#].group.pseudocode.txt**
 - Example: ws1.group.pseudocode.txt
- **Flowchart**: **ws[#].group.flowchart.jpg** (*Note: .jpg or .png*)
 - Example: ws3.group.flowchart.png

Presentation Submission

Video files can be quite large and will most likely be rejected by Seneca's email services. Therefore, you will have to **SHARE** your video file using your Seneca account Microsoft **ONE drive**.

- **Video file name**: **WS[#].fullname.video.mp4**
 - Example: WS4.Cameron Gray.video.mp4
- Go to <https://myseneca.ca>, click on (top left corner)  and select the One Drive application option
- Upload your video file: 
- Share  the file with your instructor: **Copy the shared link** 
- Paste the shared link into your email

Workshop - 6

Workshop Value: 10 marks (5% of your final grade)

Workshop Overview

Vending machines now almost run without any human maintenance. They employ the “internet of things” (**IoT**) enabling real-time updates on stock levels, payments, and alerts for machine maintenance. Electronic payments use both swipe and “tap” technology for debit and credit cards and even accept cell phone payments using “near field communications” (**NFC**). No physical money is stored! Maintenance costs are drastically reduced with these enhancements over the older models as routine inventory and money pickups are eliminated. The only time a service provider needs to physically visit the machine, is for restocking inventory and addressing any general mechanical maintenance (which would be infrequent). **But how should this all work?**



Workshop Details

Applications of vending machines essentially have two main logical components to define:

1. **Hardware States** (physical interface)
2. **Software Logic** (main functional logic/controller)

States

Hardware states are triggered by the software layer that implement the overall system process. The system has 6 main states (or modes):

1. **Power-On**
2. **Power-Off**
3. **Idle/Ready** (stand-by: waiting for customer)
4. **Active** (customer interaction building an order)
5. **Payment** (final step in customer transaction)
6. **Cancel**

You define these

For each of these states, both the **hardware** and **software** components will have their own set of defined processes.

- **[Logic 1] Hardware States**
 - Hardware components have only two possible states: “**enabled**” or “**disabled**”
 - As the machine changes from one state to another, the various hardware components need to be initialized to the new state and should be set to complement the initial state of the software logic (example: disable controls that don’t apply to the current state and only enable those that should be)
 - The software logic will modify/change the hardware components as needed after the initialization of the hardware is set to the new state

- **[Logic 2] Product Selection** (define **Idle** and **Active** states)
 - Limited to a single letter (column) button followed by a single number (row) button sequence (ex: “A” button + “9” button)
 - After selecting a product, you can enter a quantity from 1-9 followed by the **Enter** key. The **Correct** button can be used to erase the quantity before the **Enter** key is pressed.
 - Products can only be added to a transaction if the requested quantity for the item is in-stock (has inventory)
 - Multiple products can be added to a single transaction by entering one selection after another
 - Using the **Pay** button will indicate the end of the session

- **[Logic 3] Acceptable Payments & Product Inventory** (define **Payment** and **Cancel** states)
 - Credit card (tap/swipe)
 - Bank card (tap/swipe)
 - NFC phone payment
 - Phone application payment
 - **\$\$ NO CASH \$\$**
 - Following a successful payment, inventory must be adjusted in real-time (hint: local and remote data)
 - Inventory minimum stock levels must be enforced (if the quantity in-stock reaches the minimum stock level set for that product’s row & column location, more products should be ordered)

Hardware Controls/Interface (*see last page for illustrations*)

- **Payment Module:** Physical hardware for reading credit cards, debit, NFC, Tap etc.
- **LCD Colour Screen:** 10” wide X 6” high (10 cm X 15 cm)
- **Column/Row buttons:** Column-letter buttons (A-E), row-number buttons (1 – 10)
- **“Enter” button:** Adds a product selection to the session (transaction), or applies entered quantity
- **“Correct” button:** Used to backspace an entry/quantity, NOT remove an already added item
- **“Cancel” button:** Cancels the entire session and resets
- **“Pay” button:** Triggers the payment process and finalizes the session

Data Structures

Use the following data structures in your solution:

“Product” (*“Inventory” is simply a **collection/array** of “Product” data*)

slotID	// Unique location slot ID (physical placement in the machine ex: “D8”)
sku	// Unique product identifier
quantity	// Actual quantity available (physically in machine at the given slotID)
maxQuantity	// Maximum machine qty that can be stocked for the slotID
minQuantity	// Re-order when this qty is reached (based on: maxQuantity - quantity)
price	// Vending machine price to charge customer per unit
description	// Product name

“Transaction” (*“Session” is simply a **collection/array** of “Transaction” data*)

slotID	// Unique location slot ID (physical placement in the machine ex: “D8”)
quantity	// Requested quantity
price	// Price per single unit quantity
description	// Product name

Payment Module



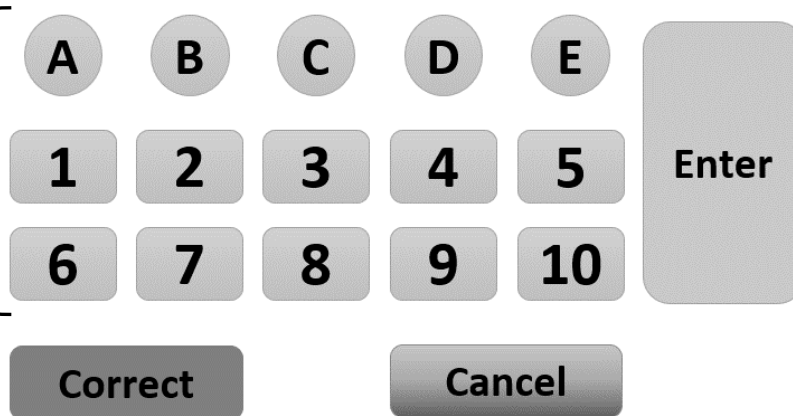
LED - Screen (NOT touch screen)

Vending Machine

LCD Display Screen

Be Creative As To What Should Be Shown Here!

Column/Row Buttons



Command Buttons

Your Tasks

1. [Logic 1] Hardware

- For each state, create the necessary **FLOWCHARTS** (pseudo code not required – even members who are in the pseudo code group will do a flowchart for this one)
 - Essentially, define what controls need to be **enabled** or **disabled** per state
 - Each control should be in its own process
 - You should have 4-flowcharts in total (one for each hardware state)

Hint: The software logic layer will “call” the hardware processes when initializing to a new state

2. [Logic 2] Software

- Define 2 separately defined processes (**Idle** and **Active** states which involve the **product selection** logic)

3. [Logic 3] Software

- Define 2 separately defined processes (**Payment** and **Cancel** states which involve **payment** and **inventory adjustments**)

4. Group Solution

- Create a "main" process that will apply the overall vending machine logic
- Include the hardware initialization flowcharts
- Include all sub-processes that are needed to process product selections, payments, and inventory adjustments (with cancel logic)
- Hint: the vending machine should work continuously until it is manually shutdown

5. Create a video presentation to market your envisioned application

Individual and sub-group assignments

Sub-Group 1 (pseudo code)				
Task	Subtask	Member(s)	Marks	Comments
Pseudocode	Logic 1	4	40%	Members are graded <u>individually</u>
	Logic 2	5	40%	
	Logic 3	6	40%	
	Group Solution	4-6	60%	Eligible members get <u>same mark</u>
Sub-Group 2 (flowchart)				
Task	Subtask	Member(s)	Marks	Comments
FlowChart	Logic 1	1	40%	Members are graded <u>individually</u>
	Logic 2	2	40%	
	Logic 3	3	40%	
	Group Solution	1-3	60%	Eligible members get <u>same mark</u>
Video	Presentation *	3 or 6	100%	Members rotate weekly

* **Presentation:** Decide among yourselves which member among the entire group will be doing the presentation. Priority should be given to those who have not yet done one.