Workshop - 10

Workshop Value: 10 marks
(Average mark of the Best Eight workshop will be 50% of your final grade,
Best Presentation's mark will be 5% of your final grade)

Learning Outcomes

Upon successful completion of this workshop, you will have demonstrated the abilities:

- to decipher and identify a problem
- to analyze and decompose a problem
- to identify the required detailed steps to solve a **larger problem** using **modularity**
- to communicate the solution to fellow peers and non-technical business persons

Workshop Grading and Promotion Policy

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

- Workshops must be completed during the class time to be graded
- You must successfully complete 9 workshops (if more than 9 are completed, the best 9 will be used)
- Each student is expected to be a presenter of the workshop solution at least once by the end of the term
- Workshop solutions and presentations will be evaluated using the published workshop rubrics

Workshop Overview

Vending machines now run without almost any human maintenance. They employ the "Internet of Things" (**IoT**), enablingreal-time updates on stock levels, payments, alerts, etc., 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 visit the machine physically is for restocking inventory and addressing any general mechanical maintenance (which would be infrequent).

But how should this all work?

Reminder

Always start by using **pseudocode** to develop the detailed step-by-step process. Once you are satisfied with your pseudocode solution, develop a flowchart.

Workshop Details

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

1. **The Hardware Components** (turn on/off)

2. **Software Logic** (main functional logic/controller)

<u>The states of the Software Logic</u> (finite state machines)

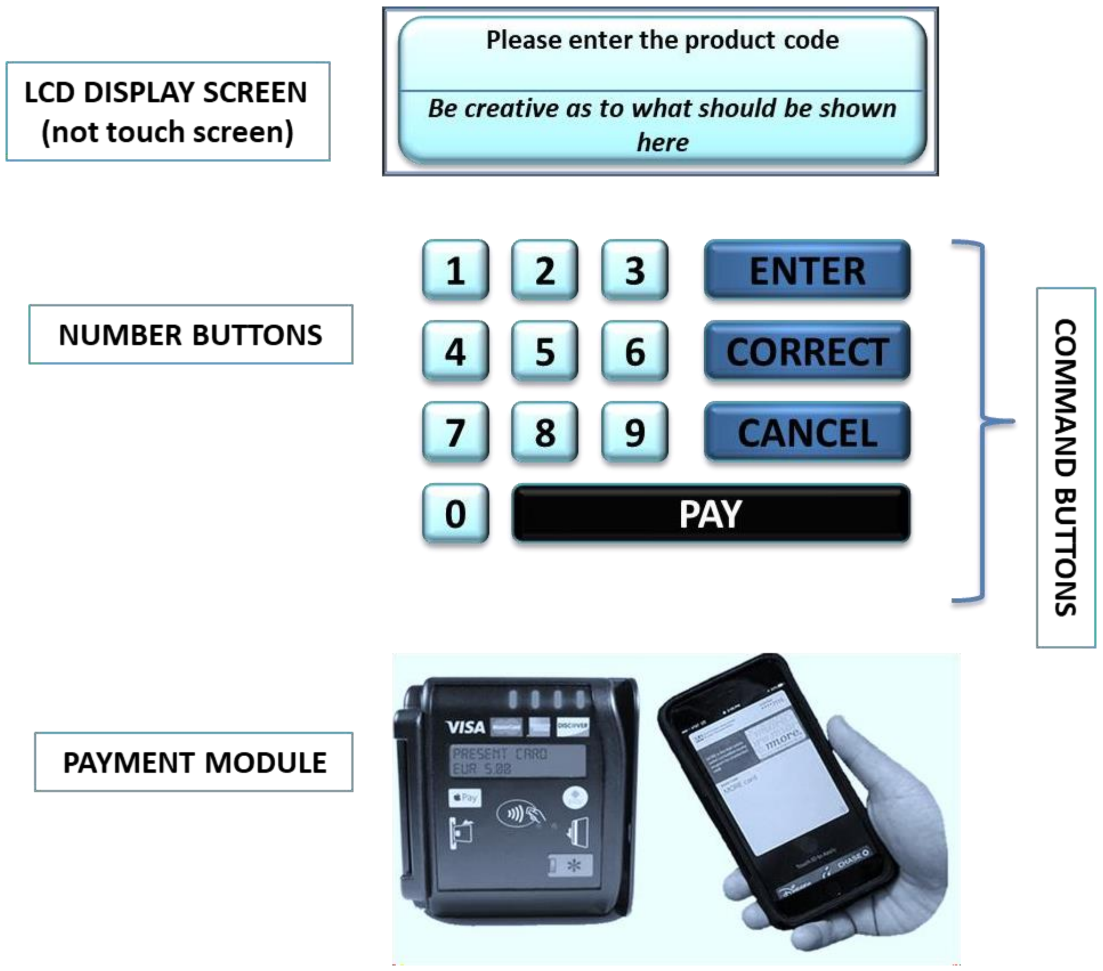
Hardware components are triggered by the software layer that implements the overall system process. The system has six central states (or modes):

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 will create a solution for these states.**

<u>Hardware Components:</u>

1. **<u>Payment Module</u>**: Physical hardware for reading credit cards, debit, NFC, Tap, etc.
2. **<u>LCD Colour Screen</u>**: Display
3. **<u>Number buttons</u>**: Number buttons (0-9)
4. **"<u>Enter</u>" <u>button</u>**: Adds a product selection to the transaction or applies entered quantity
5. **"<u>Correct</u>" <u>button</u>**: Used to backspace an entry/quantity, NOT remove an already added item
6. **"<u>Cancel</u>" <u>button</u>**: Cancels the entire session and resets
7. **"<u>Pay</u>" <u>button</u>**: Triggers the payment process and finalizes the transaction


LCD DISPLAY SCREEN (not touch screen)
Please enter the product code
Be creative as to what should be shown here
NUMBER BUTTONS
1
2
3
ENTER
4
5
6
CORRECT
7
8
9
CANCEL
0
PAY
COMMAND BUTTONS
PAYMENT MODULE
VISA
PRESENT CARD
EUR 5.00

1. **<u>POWER-ON STATE</u>**
2. **<u>POWER-OFF STATE</u>**

3. **<u>IDLE / READY STATE</u>**
   - LCD Display shows a welcome message to the customer until any number button is pressed.
   - LCD Display shows a 10-second marketing advertisement one after the other until the customer presses any number of buttons.

4. **<u>ACTIVE STATE</u>**

Product Selection

- Limited to double-digit/number buttons (ex: "1" button + "6" button).
- Products can only be added to a transaction if the requested quantity for the item is in stock (has inventory).

Product Inventory

- Following the 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 slot, more products should be ordered.)

5. PAYMENT STATE

Acceptable Payment types

- Credit card (tap/swipe)
- Bank card (tap/swipe)
- NFC phone payment
- Phone application payment
- **$$ NO CASH $$**

Black Box / External Functions:

Payment process (Input: **CardInfo, Transaction. Price**; Output: **PaymentReceived**)
This process sends the user's debit/credit card information and the price/cost to the banking systems to make the required financial transactions. This process returns (as an output) if the payment has been received successfully or not. The value of the **price received** could be True or False. Remember that the payment process should be retried if the operation fails.

Hints: This function should be called after the customer presses the "pay" button.

Usage in your pseudocode

```
    Call PaymentProcess (CardInfo, Transaction.Price)
```

(*Replace names with the variable name you use in your solution as required.)

RequestProduct (Input: **Product. Sku, RequestedQuantity;** Output: **InfoReceived)**

This process sends Product SKU information and the requested quantity of the product to the remote inventory system to request needed inventory for the vending machine. This process returns (as an output) if the inventory request has been processed successfully or not. The value of **InfoReceived** could be True or False.

Hints: This function should be called after the inventory is updated in the remote system.

Usage in your pseudocode

```
      Call RequestProduct (Product. Sku, RequestedQuantity)
```

(*Replace names with the variable name you use in your solution as required.)

<u>Hardware Components</u>

- Hardware components have only two possible modes: "**enabled**" or "**disabled**" based on the overall machine/logic state
- For each state (power-on, power-off, idle, active, payment, cancel), hardware componentswill have defined processes.

<u>Table: Hardware Control Modes</u>

		SOFTWARE LOGIC STATES			
HARDWARE COMPONENTS		IDLE / READY	ACTIVE	PAYMENT	CANCEL
	LCD Display	ENABLE			
	Payment Module	DISABLE			
	Number Buttons	DISABLE			
	"Enter" Button				
	"Correct" Button				
	The "Cancel" Button				
	"Pay" Button				

In your solution, please take into consideration any interruptions. (power-off interrupt)

Data Structures

Apply the following data structures in your solution:

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

SlotID	// Unique location slot ID (where it is physically placed in the machine ex: “34”)
Sku	// Unique product identifier
Quantity	// Actual quantity available (physically in the machine at the given slotID)
MaxQuantity	// Maximum machine quantity that can be stocked for the slotID
MinQuantity	// Re-order when this quantity is reached (based on: maxQuantity - quantity)
Price	// Vending machine price to charge customer per unit
Description	// Product name

“Transaction”

SlotID	// Unique location slot ID (where it is physically placed in the machine ex: “34”)
Quantity	// Requested quantity
Price	// Price per single unit quantity
Description	// Product name

Your Task

IMPORTANT: Divide the work among your group members to ensure you complete the requirements on time. You must periodically discuss the overall flow and overlapping logic to ensureyour solutions piece together seamlessly.

1. **Hardware**: For each state, define the necessary ***modes*** in the **“Hardware Control Modes” table**
- Essentially, define what controls need to be ***turned on or off*** per state
  (this should take no more than 15 minutes)

  Hint: the software logic will “call” these processes as required when changing states.

2. **Software**: For each state, define how the machine should work (***pseudocode*** and ***flowchart***)
- You should have four separately defined pseudocode processes
- You should have four separately illustrated flowcharts

3. **Present Solution**:
- The instructor will let you know at presentation time what states to present (there will not be time to cover the entire solution)
- Present the assigned state solutions to the instructor or class (single student)