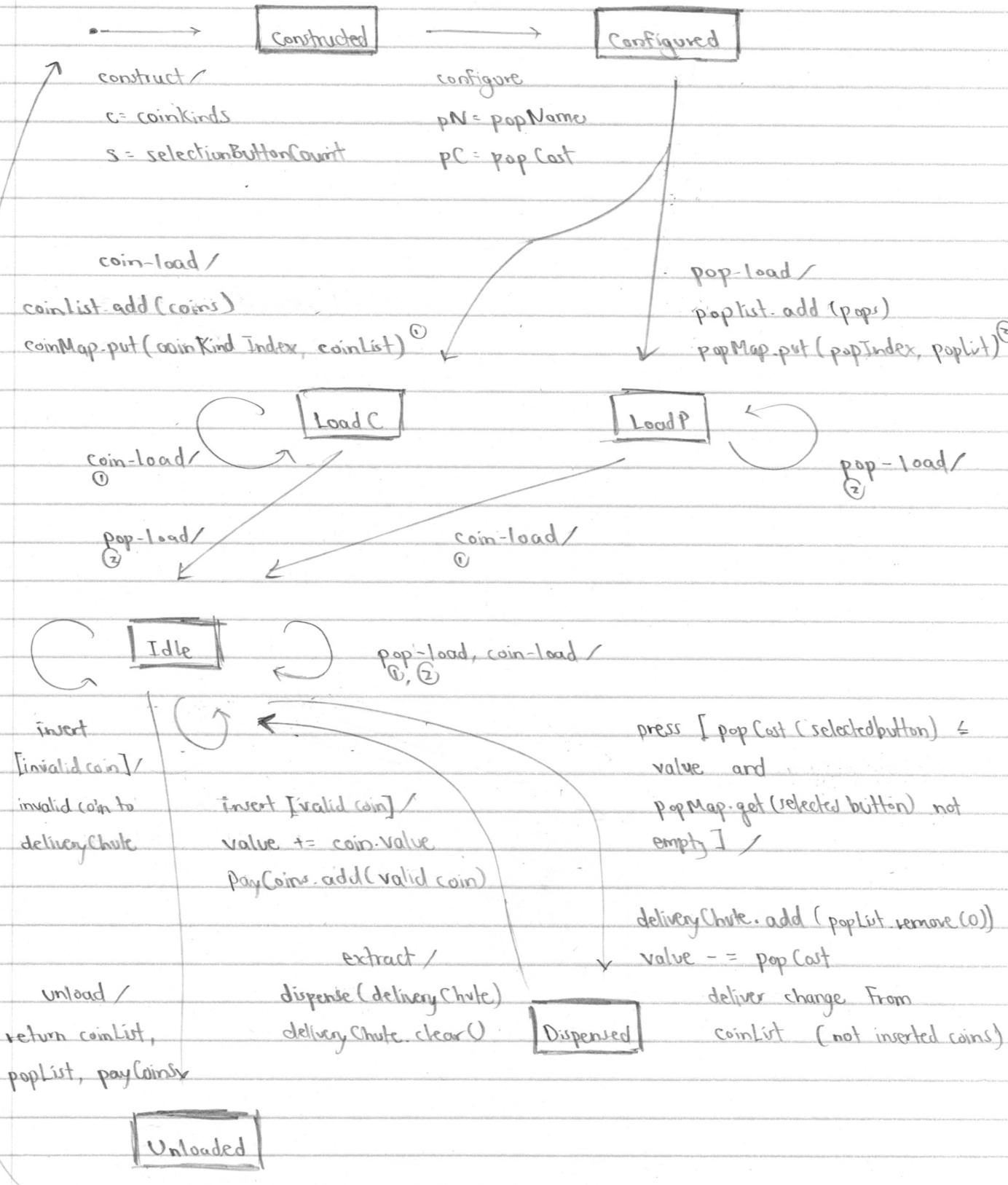
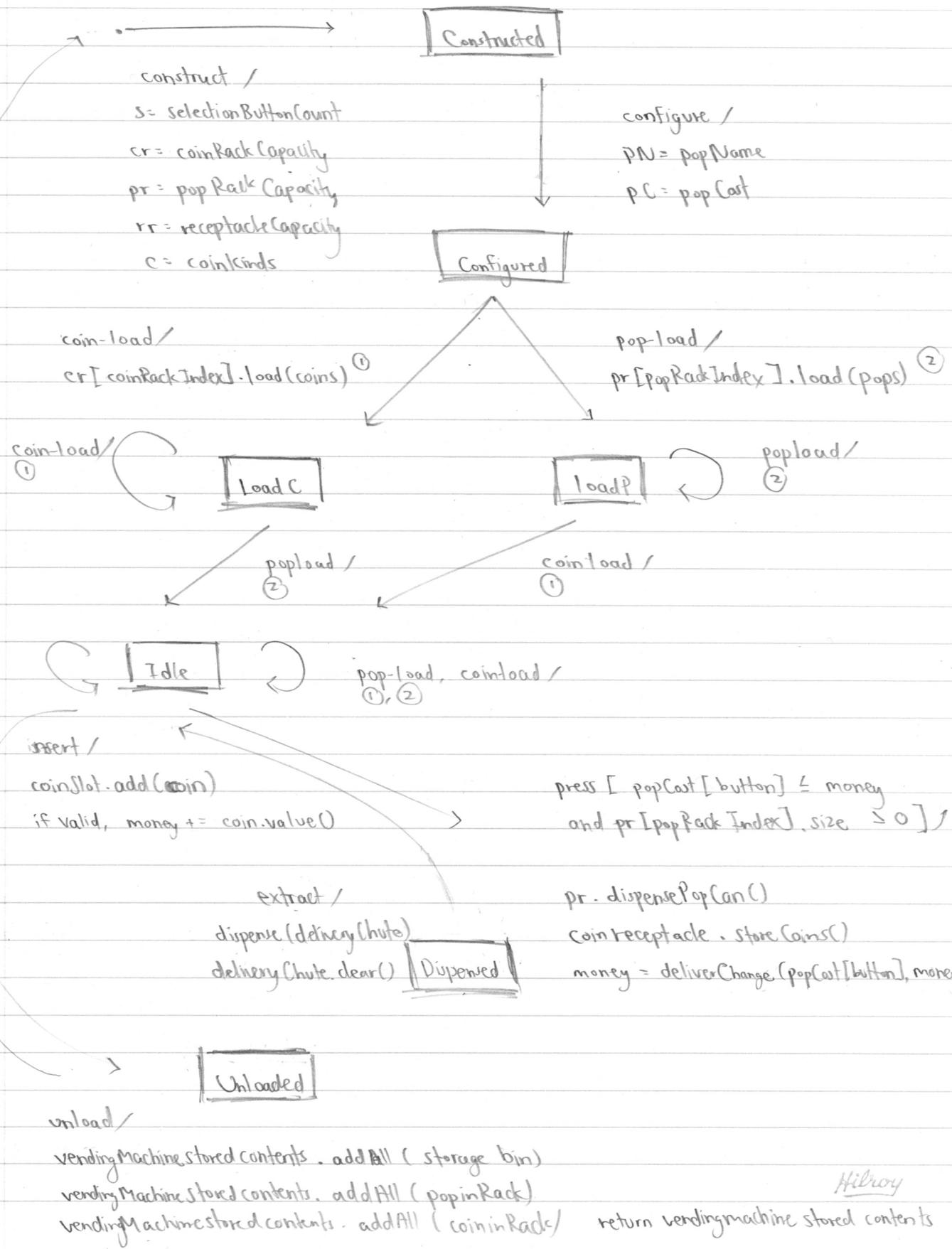


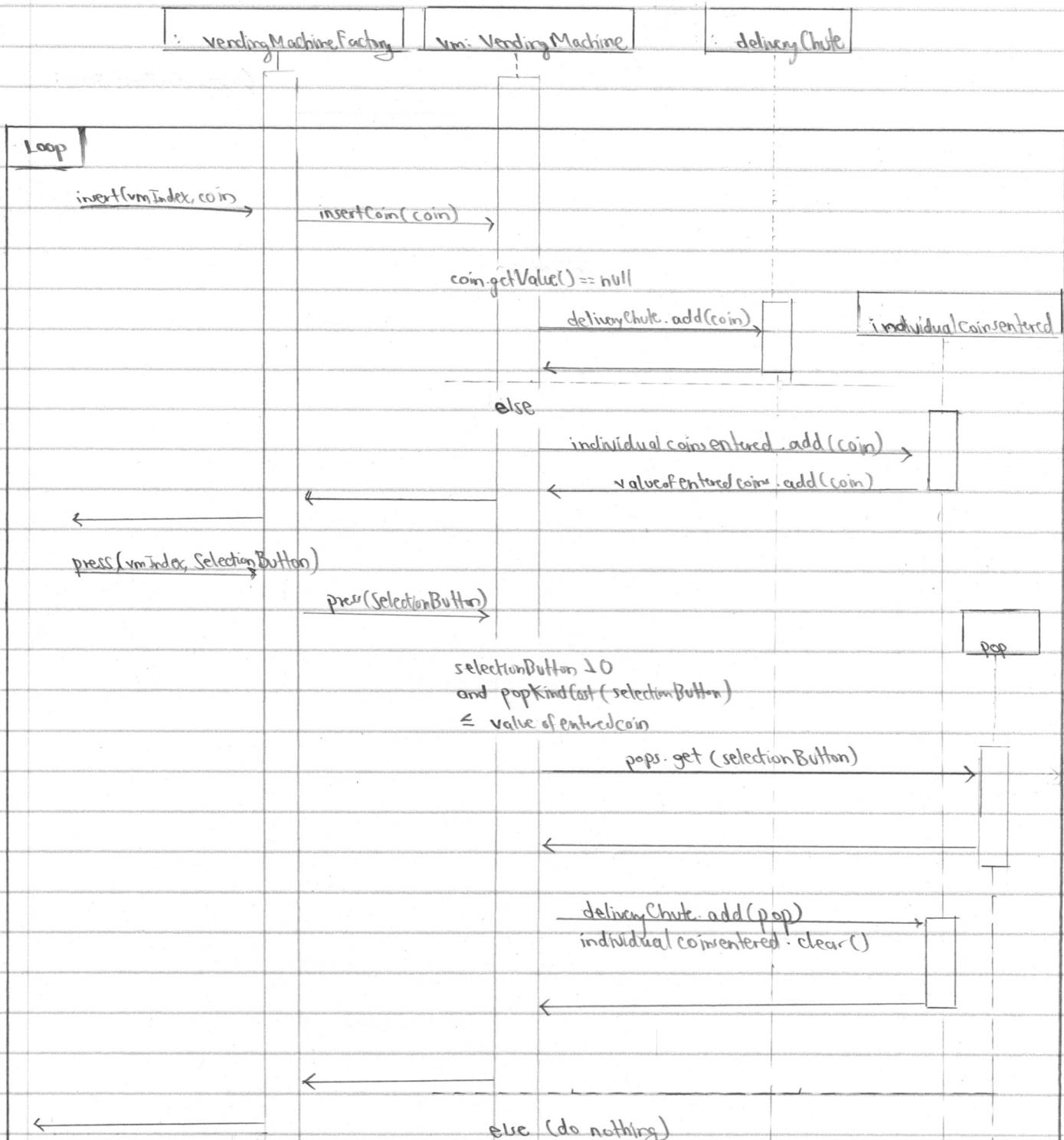
State Machine Diagram Assignment 1



State Machine Diagram Assignment 2.

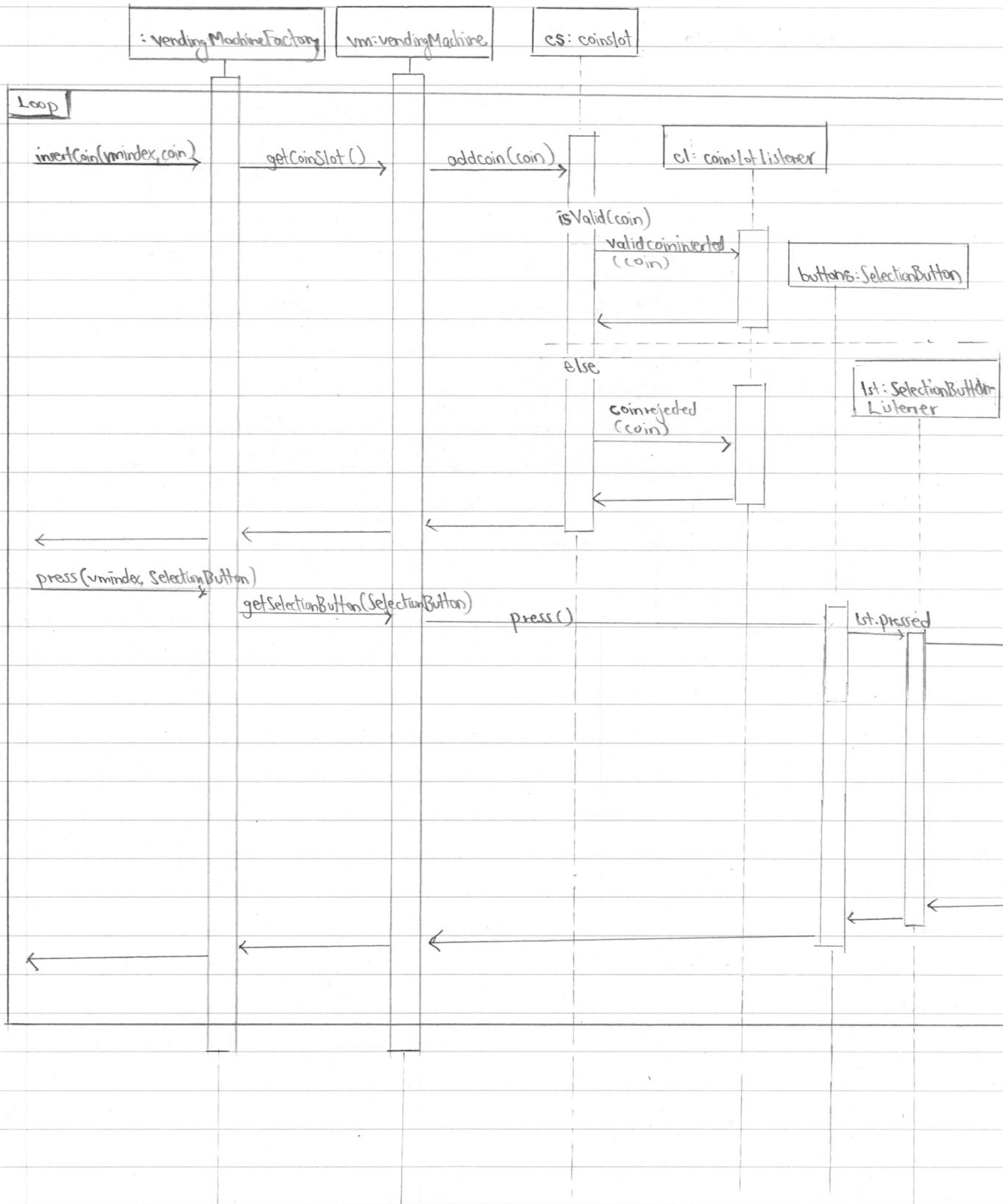


Sequence Diagram Assignment 1



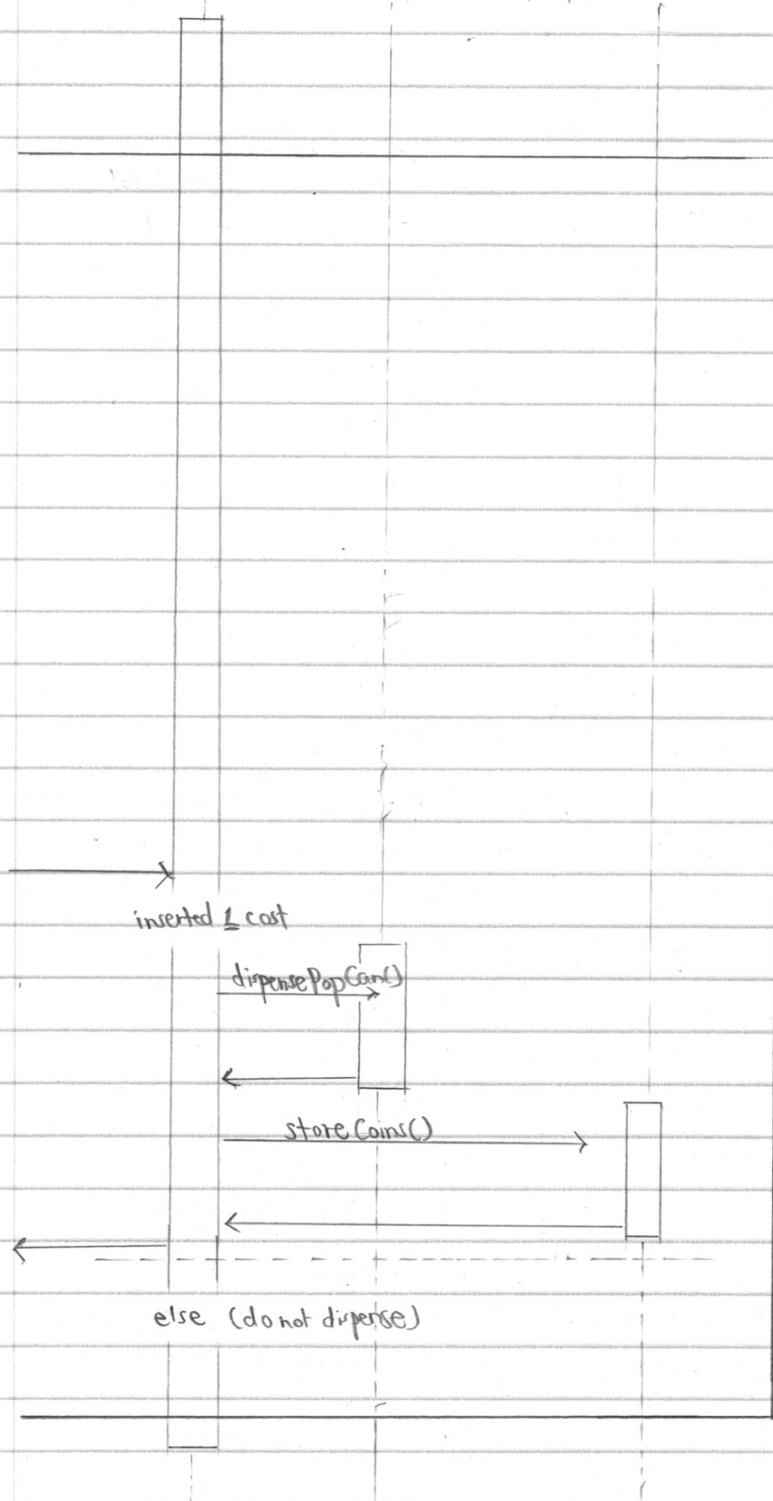
Hilary

Sequence Diagram Assignment 2 (Continued on next page)



Hilroy

log:Logic rock:PopCanBack : coinceptorade



Mitroy

Structural Diagram Assignment 1

VendingMachineFactory

- List <Vending Machine> vmList
- + VendingMachineFactory () : ctor
- + extractFromDeliveryChute (int vmIndex) : List <Deliverable>
- + insertCoin (int vmIndex, Coin coin) : void
- + pressButton (int vmIndex, int selectionButtonIndex) : void
- + constructNewVendingMachine (List <Integer> coinKinds, int selectionButtonCount) : int
- + configureVendingMachine (int vmIndex, List <String> popNames, List <Integer> popCosts) : void
- + loadCoins (int vmIndex, int coinKindIndex, Coin... coins) : void
- + loadPops (int vmIndex, int popKindIndex, Pop... pops) : void
- + unloadVendingMachine (int vmIndex) : List <List <?>>

↓

Vending Machine

- deliveryChute : ArrayList <Deliverable>
- valueOfEnteredCoins : int
- individualCoinsEntered : List <Coin>
- paymentCoins : List <Coin>
- coinKinds : int []
- valueToIndexMap : Map <Integer, Integer>
- coins : Map <Integer, ArrayList <Coin>>
- popKindName : String []
- popKindCosts : int []
- pops : Map <Integer, ArrayList <Pop>>
- + vendingMachine (List <Integer> coinKinds, int buttonCount) : ctor
- + configure (List <String> names, List <Integer> costs) : void
- + loadCoins (int coinKindIndex, Coin... coins) : void
- + press (int selectionbuttonIndex) : void
- + changeHelper (ArrayList <Integer> values, int index, int changeDue) : Map <Integer, List <Integer>>
- + deliverChange (int cost, int entered) : int
- + loadPops (int popKindIndex, Pop... pop) : void
- + insert (Coin coin) : void
- + unload () : List <List <?>>
- + extract () : List <Deliverable>

Hilroy

Structural Diagram Assignment 2 (Continued on next page)

VendingMachineFactory

- vend: ArrayList<VendingMachine>
- log: Logic
- vmi: VendingMachine

+ main (String [] args) : static void

+ vendingMachineFactory () : ctor

+ extractFromDeliveryChute (int vmiIndex) : List<Deliverable>

+ insert (Coin int vmiIndex, Coin coin) : void

+ pressButton (int vmiIndex, int value) : void

+ constructNewVendingMachine (List<Integer> coinCounts, int selectionButtonCount, int coinRackCapacity, int popRackCapacity, int receptacleCapacity) : int

+ configureVendingMachine (int vmiIndex, List<String> popNames, List<Integer> popCounts) : void

+ loadCoins (int vmiIndex, int coinRackIndex, Coin coins) : void

+ loadPopCans (int vmiIndex, int popCanRackIndex, PopCan... popCans) : void

+ unloadVendingMachine (int vmiIndex) : VendingMachineStoredContents



Logic

- vendIMech: vendingMachine
- money: int
- buttonsToIndex: Map<SelectionButton, Integer>
- valueToIndexMap: Map<Integer, Integer>

+ Logic (VendingMachine vmi) : ctor

+ validCoinInserted (CoinSlot slot, Coin coin) : void

+ press (SelectionButton button) : void

+ changeHelper (ArrayList<Integer> values, int index, int changeDue) : Map<Integer, List<Integer>>

+ deliverChange (int count, int entered) : int

VendingMachine

+ getDeliveryChute () : DeliveryChute

+ getSelectionButton (int value) : SelectionButton

+ getCoinSlot () : CoinSlot

+ VendingMachine (List<Integer> coin, int button, int coinR, int popR, int ReceptacleR) : ctor

+ configure (List<String> popCanNames, List<Integer> popCanCounts) : void

+ getCoinRack (int coinRackIndex) : CoinRack

+ getPopCanRack (int popCanRackIndex) : PopCanRack

+ getStorageBin () : StorageBin

+ getCoinKindForCoinRack (int value) : CoinKind

+ getPopKind (int index) : int

+ getCoinReceptacle () : CoinReceptacle



CoinSlot

+ addCoin (Coin coin) : void



PopCanRack

+ load (PopCan... popCans) : void

+ unload () : List<PopCan>

CoinRack

+ load (Coin... coins) : void

+ unload () : List<Coins>

DeliveryChute

+ removeItem () : Deliverable []

Hilary

SelectionButton Listener

+ pressed (SelectionButton button) : void

CoinSlot Listener

+ validCoinInserted (CoinSlot slot, Coin coin) : void
+ coinRejected (CoinSlot slot, Coin coin) : void

CoinReceptacle

+ storeCoins () : void

Selection Button

+ press () : void