Design a Vending Machine (will accept coins of different denominations and vends out candies, soda etc)

Requirements:

1. Machine accepts payment and vends out snack
2. Customer selects snack using keypad
3. LCD displays price of snack
4. Customer pays using cash/credit card
5. Customer picks up snack

Class VendingMachine
Data: List<Snack> snacks, KeyPad keypad, LCDDisplay display, CoinCashCollector ccc, List<String> validSnackIds, List<Float> snackPrices, List<int> snackAvailability,

Behavior: initializeVendingMachine () {

```
        snacks = new Snacks[Number of Snacks to be loaded];
        keypad = new KeyPad();
        display = new LCDDisplay();
        ccc =  new CoinCashCollector();
        foreach (snack in Snacks) {
                validSnackIds.add(snackID of snack);
                snackAvailability[snackID] = Initial Number of snacks loaded;
                snackPrices[snackID] = price of Snack;
        }
}
```

Behavior: processInput (inputValue) {

```
    If (inputValue belongs to validSnackIds) {
       If (this.snackAvailability[inputValue] > 0 ) {
          display.DisplayToCustomer("Snack Available");
          display.DisplayToCustomer("Price of snack :"+ this.snackPrices[inputValue] );
          display.DisplayToCustomer("Insert Cash/Coins");
           if ( ccc.CollectCoinCash(this.snackPrices[inputValue]) == TRUE) {
                   // Vend out snack;
                   this.snackAvailability[inputValue] = this.snackAvailability[inputValue – 1;
           } else {
                   display.DisplayToCustomer("Transaction Cancelled");
           }

       } else {
          display.DisplayToCustomer("Snack Not Available");
       }
```

```
        } else {
            display.DisplayToCustomer("Not a valid SnackId");
        }
    }
```

```
Class Customer
Data: name, cashInPocket, snackDesired
Behavior: enterInput(){
            //press keypad buttons to enter snackId
            Keypad. AcceptInput();
        }
Behavior: lookUpLCDDispaly(){
            //check the display for the price
        }

Behavior: insertCoinCash(){
            this.lookUpLCDDisplay();
            CoinCashCollector.CollectCoinCash(amount on display);
        }
Behavior: pickUpSnack(){
            //pick up snack
        }
```

```
Class Keypad
Data: alphaNumericButtons, oKbutton, deleteButton
Behavior: acceptInput() {
            While (OKButton is pressed) {
                    //Accept input through alphaNumericButton
                    If (DeleteButton is pressed) {
                            // Delete last inputted alphaNumericValue;
                    }
            }
            If (validInput) {
                    VendingMachine.ProcessInput(inputValue);
            }
    }
```

```
Class LCDDisplay
Data: displayModule,
Behavior: displayToCustomer (String s) {
```

```
            System.Out.Println(s);
        }
```

```
Class CoinCashCollector
Data: List<Denomination> acceptedDenominations
Behavior: initialzeCoinCashCollector {
            // Populate the acceptedDenominations list
            // Eg: $1, $5
        }

Behavior: collectCoinCash(float amountToBeCollected) {
            while (amountToBeCollected > 0.0) {
                //CollectCash
                If (denomination is in acceptedDenominations ) {
                    If (Cash/CoinInserted > amountToBeCollected) {
                        // Give back change
                    } else {
                        amountToBeCollected = amountToBeCollected – Cash/CoinInserted;
                        VendingMachine.display.DisplayToCustomer(amountToBeCollected);
                    }
                }
                // If cash collection times out, give back amount inserted till now
                // return false
            }
            return true;
        }
```