

PMR: This program was truly a challenge program. It was hard to compete. I think this is due to the fact that we didn’t do the 19.1 assignment. I enjoyed creating this program though. I look forward to future programs.

/\*

\* By Anika Jallipalli

\* Date: 4/24/2020

\* Purpose: Use assertions and exceptions to create a candy machine

\*

\* PMR in README.txt

\*/

public class CandyMachine

{

//main

public static void main(String[] args)

{

//create the dispenser and the cash register

Dispenser dispenser = new Dispenser(100, 50);

CashRegister cashRegister = new CashRegister();

//print starting stats

System.out.println("Dispenser: " + dispenser.getCount() + " items, " + " each costing " + dispenser.getProductionCost() + " cents");

System.out.println("Cash Register: " + cashRegister.getCashOnHand() + " cents");

System.out.println();

//sell some products

sellProduct(dispenser, cashRegister, 10);

System.out.println();

//print the stats

System.out.println("Dispenser: " + dispenser.getCount() + " items, " + " each costing " + dispenser.getProductionCost() + " cents");

System.out.println("Cash Register: " + cashRegister.getCashOnHand() + " cents");

}

//sell product

public static void sellProduct(Dispenser dispenser, CashRegister cashRegister, int amount)

{

System.out.println("Made sale: " + amount + " items");

//make sales from the dispenser

for(int i = 0; i < amount; i++)

{

dispenser.makeSale();

}

//accept the funds

cashRegister.acceptAmount(amount \* dispenser.getProductionCost());

}

}

/\*

\* By Anika Jallipalli

\* Date: 4/24/2020

\* Purpose: Use assertions and exceptions to create a candy machine

\*

\* PMR in README.txt

\*/

public class CashRegister

{

//instance variables

private int cashOnHand;

//getter

public int getCashOnHand()

{

return cashOnHand;

}

//default constructor

CashRegister()

{

cashOnHand = 500;

}

//alternate

CashRegister(int cashOnHand)

{

if(cashOnHand < 0)

{

throw new IllegalArgumentException("cash on hand can't be less than zero");

}

this.cashOnHand = cashOnHand;

}

//other methods

//set cash

public void setCashOnHand(int cashOnHand)

{

if(cashOnHand < 0)

{

throw new IllegalArgumentException("cash on hand can't be less than zero");

}

this.cashOnHand = cashOnHand;

}

//accept amount

public void acceptAmount(int amount)

{

if(amount < 0)

{

throw new IllegalArgumentException("We don't accept negative amounts here");

}

cashOnHand += amount;

}

}

/\*

\* By Anika Jallipalli

\* Date: 4/24/2020

\* Purpose: Use assertions and exceptions to create a candy machine

\*

\* PMR in README.txt

\*/

public class Dispenser

{

//instance variables

private int numberOfItems;

private int cost;

//getters

public int getCount()

{

return numberOfItems;

}

public int getProductionCost()

{

return cost;

}

//setters

public void setNumberOfItem(int numberOfItems)

{

this.numberOfItems = numberOfItems;

}

public void setCost(int cost)

{

this.cost = cost;

}

//default constructor

Dispenser()

{

this.numberOfItems = 50;

this.cost = 50;

}

//alternate

Dispenser(int numberOfItems, int cost)

{

if(numberOfItems < 0 || cost < 0)

{

throw new IllegalArgumentException("Arguments to a dispenser can't be less than zero");

}

this.numberOfItems = numberOfItems;

this.cost = cost;

}

//other methods

//makeSale

public void makeSale()

{

if(numberOfItems == 0)

{

System.out.println("Sorry, out of products :(");

return;

}

this.numberOfItems--;

}

}