RegularMachine.java

Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P/F
restockItem	1	Assuming that there are already 5 stocks of a certain product	1	6	6	P
	2	Assuming that there are already 5 stocks of a certain product	2	7	7	P
	3	Assuming that there are already 5 stocks of a certain product	10	15	15	P

Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P/F
updateLastRestockDate	1	Changes the lastRestockDate attribute to the current date	2023-06-30	2023-06-30	2023-06-30	P
	2	Changes the lastRestockDate attribute to the current date	2023-07-1	2023-07-1	2023-07-1	P
	3	Changes the lastRestockDate attribute to the current date	2023-01-01	2023-01-01	2023-01-01	P

Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P/F
restockItem	1	Assuming that there is the price of the called item is P150	20	P170	P170	P
	2	Assuming that there is the price of the called item is P150	30	P180	P180	P
	3	Assuming that there is the price of the	50	P200	P200	P

		called item is P150				

Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P/F
replenishMachineBal	1	Replenish the balance per denomination, with the following current balance: Php 1s: 5 Php 5s: 1 Php 10s: 3 Php 20s: 6 Phps 50s: 2 Php 100s: 3 Php 200s: 5 Php 500s: 1 Php 1000s: 1	1 2 5 3 8 9 7 7 5	6 3 8 9 10 12 12 8 6	6 3 8 9 10 12 12 8 6	P
	2	Replenish the balance per denomination, with the following current balance: Php 1s: 1 Php 5s: 2 Php 10s: 3 Php 20s: 5 Phps 50s: 4 Php 100s: 7 Php 200s: 6 Php 500s: 0 Php 1000s: 5	2 3 4 5 6 1 2 5 2	3 5 7 10 10 8 8 5 7	3 5 7 10 10 8 8 5 7	P
	3	Replenish the balance per denomination, with the following current balance: Php 1s: 0	2 3 7 7	2 3 7 7	2 3 7 7	P

		Php 5s: 0 Php 10s: 0 Php 20s: 0 Phps 50s: 0 Php 100s: 0 Php 200s: 0 Php 500s: 0 Php 1000s: 0	5 5 10 9 2	5 5 10 9 2	5 5 10 9 2	

Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P/F
collectMachineBalance	1	Replenish the balance per denomination, with the following current balance: Php 1s: 5 Php 5s: 1 Php 10s: 3 Php 20s: 6 Phps 50s: 2 Php 100s: 3 Php 200s: 5 Php 500s: 1 Php 1000s: 1	N/A	Php 1s: 0 Php 5s: 0 Php 10s: 0 Php 20s: 0 Phps 50s: 0 Php 100s: 0 Php 200s: 0 Php 500s: 0 Php 1000s: 0	Php 1s: 0 Php 5s: 0 Php 10s: 0 Php 20s: 0 Phps 50s: 0 Php 100s: 0 Php 200s: 0 Php 500s: 0 Php 1000s: 0	P
	2	Replenish the balance per denomination, with the following current balance: Php 1s: 1 Php 5s: 2 Php 10s: 3 Php 20s: 5 Phps 50s: 4 Php 100s: 7 Php 200s: 6	N/A	Php 1s: 0 Php 5s: 0 Php 10s: 0 Php 20s: 0 Phps 50s: 0 Php 100s: 0 Php 200s: 0 Php 500s: 0 Php 1000s: 0	Php 1s: 0 Php 5s: 0 Php 10s: 0 Php 20s: 0 Phps 50s: 0 Php 100s: 0 Php 200s: 0 Php 500s: 0 Php 1000s: 0	P

		Php 500s: 0 Php 1000s: 5				
	3	Replenish the balance per denomination, with the following current balance: Php 1s: 0 Php 5s: 0 Php 10s: 0 Php 20s: 0 Phps 50s: 0 Php 100s: 0 Php 200s: 0 Php 500s: 0 Php 1000s: 0	N/A	Php 1s: 0 Php 5s: 0 Php 10s: 0 Php 20s: 0 Phps 50s: 0 Php 100s: 0 Php 200s: 0 Php 500s: 0 Php 1000s: 0	Php 1s: 0 Php 5s: 0 Php 10s: 0 Php 20s: 0 Phps 50s: 0 Php 100s: 0 Php 200s: 0 Php 500s: 0 Php 1000s: 0	P

Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P/F
countItems	1	There are 12 items available	N/A	12	12	P
	2	There are 6 items available	N/A	6	6	P
	3	There are 13 items available	N/A	13	13	P

Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P/F
checkQuantity	1	Returns the quantity of item number 6.	6	10	10	P
	2	Returns the quantity of item number 5.	5	3	3	P
	3	Returns the quantity of item number 2.	2	1	1	P

Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P/F
checkDenom	1	Checks if the input data is a valid denomination	20	true	true	P
	2	Checks if the input data is a valid denomination	6	false	false	P
	3	Checks if the input data is a valid denomination	-5	false	false	P

Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P/F
processPayment	1	Checks if the entered payment is equal or greater than the total payable of 50 pesos.	5 5 10 10 20	true	true	P
	2	Checks if the entered payment is equal or greater than the total payable of 100 pesos.	20 20 0	false (user canceled the transaction)	false (user canceled the transaction)	P
	3	Checks if the entered payment is equal or greater than the total payable of 20 pesos.	50	true (proceed to produceChange)	true (proceed to produceChange)	P

Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P/F
addQuantityToDenom	1	Add 1 to denomination stock. Assume Php 20s have 5 stocks.	20	Php 20s: 6	Php 20s: 6	P

	2	Add 1 to denomination stock. Assume Php 100s have 10 stocks.	100	Php 100s: 11	Php 100s: 11	P
	3	Add 1 to denomination stock. Assume Php 1000s have 2 stocks.	1000	Php 1000s: 3	Php 1000s: 3	P

Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P/F
deductQuantityToDenom	1	Deduct 1 to denomination stock. Assume Php 20s have 5 stocks.	20	Php 20s: 4	Php 20s: 4	P
	2	Deduct 1 to denomination stock. Assume Php 100s have 10 stocks.	100	Php 100s: 9	Php 100s: 9	P
	3	Deduct 1 to denomination stock. Assume Php 1000s have 2 stocks.	1000	Php 1000s: 2	Php 1000s: 2	P

Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P/F
deductQuantityToDenom	1	Deduct 1 to denomination stock. Assume Php 20s have 5 stocks.	20	Php 20s: 4	Php 20s: 4	P
	2	Deduct 1 to denomination stock. Assume Php 100s have 10 stocks.	100	Php 100s: 9	Php 100s: 9	P
	3	Deduct 1 to denomination stock. Assume Php 1000s have 2 stocks.	1000	Php 1000s: 2	Php 1000s: 2	P

Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P/F

hasDenomStock	1	Checks if the given denomination has more than 0 stock.	5	true	true	P
	2	Checks if the given denomination has more than 0 stock.	10	true	true	P
	3	Checks if the given denomination has more than 0 stock.	200	true	true	P

Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P/F
produceChange	1	Return true if the producing change is successful and if it is equal to the change amount. Assume the change amount is Php15.	5 5 5	true	true	P
	2	Return true if the producing change is successful and if it is equal to the change amount. Assume the change amount is Php50.	10 20 20	true	true	P
	3	Return true if the producing change is successful and if it is equal to the change amount. Assume the change amount is Php35.	5 5 10	false	false	P

Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P/F
produceChange	1	Return true if the producing change is	5	true	true	P

		successful. Assume the change amount is Php15.	5 5			
	2	Return true if the producing change is successful. Assume the change amount is Php50.	10 20 20	true	true	P
	3	Return true if the producing change is successful. Assume the change amount is Php35.	5 5 10	false, there are not enough denominations to accomplish the transaction	false, there are not enough denominations to accomplish the transaction	P

Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P/F
isChangeComplete	1	Return true if the total of the elements inside change stack is equal to the change amount. Assume the change amount is Php15.	5 5 5	true	true	P
	2	Return true if the total of the elements inside change stack is equal to the change amount. Assume the change amount is Php50.	10 20 20	true	true	P
	3	Return true if the total of the elements inside change stack is equal to the change amount. Assume the change amount is Php35.	5 5 10	false	false	P

Driver.java

Method	#	Test Description	Sample Input Data	Expected Output	Actual output	P/F
CreateRegularMachine	1	Creates a RegularMachine object. All the inputs are accepted and valid.	Arraylist<Items> balance	Created a new RegularMachine object	Created a new RegularMachine object	P
	2	Creates a RegularMachine object but the balance is 0	Arryalist <Items> Balance =0	Created the new RegularMachine with a balance of zero	Created the new RegularMachine with a balance of zero	P
	3	Creates a Regular Machine object but the imputed values are not valid	Arraylist<Items> balance	A loop asking for proper input will only stop until the user enters valid inputs	A loop asking for proper input will only stop until the user enters valid inputs	P

Method	#	Test Description	Sample Input Data	Expected Output	Actual output	P/F

testVendingFeatures	1	Buys an Item with all valid inputs and gave the exact change	1(Selection to buy)
			
			2(Item Index)
			
			50(item price that is payed)
				The item is dispensed and the function will prompt again the menu to buy / test the machine again	The item is dispensed and the function will prompt again the menu to buy / test the machine again	P

	2	Buys an Item but the imputed payment is negative	1(Selection to buy)
			
			2(Item Index)
			
			-50(item price that is payed)
				The payment prompt will not accept the payment and loop until the user inputs a valid denomination until the item price is met	The payment prompt will not accept the payment and loop until the user inputs a valid denomination until the item price is met	P

	3	Buys an Item but the imputed payment is not in the denomination the the machine accepts	1(Selection to buy)
			
			2(Item Index)
			
			(item price that is payed)
				The payment prompt will not accept the payment and loop until the user inputs a valid denomination until the item price is met	The payment prompt will not accept the payment and loop until the user inputs a valid denomination until the item price is met	P

Method	#	Test Description	Sample Input Data	Expected Output	Actual output	P/F
askItemName	1	Returns the imputed string	Chicken	Chicken	Chicken	P
askItemPrice	1	The user inputs a valid price	150	150	150	P
	2	The user inputs a negative number	-150	Th price must be non-negative and non-zero value (printed)	The price must be non-negative and non-zero value (printed)	P

Method	#	Test Description	Sample Input Data	Expected Output	Actual output	P/F
askItemCalories	1	The user inputs a valid price	300	300	300	P
	2	The user inputs a negative integer	-300	The calories must be a non-negative and non-zer0 value (printed)	The calories must be a non-negative and non-zer0 value (printed)	P
	3	The user inputs a float	4.0	4	4	P

Method	#	Test Description	Sample Input Data	Expected Output	Actual output	P/F
askItemQuantity	1	The user enter a valid input that is greater then 10	15	15	15	P
	2	Th user inputs a valid input but it is less than 10	6	Minimun quantity fo a product is 10 items (printed)	Minimun quantity fo a product is 10 items (printed)	P
	3	The user inputs a negative number	-10000	Minimun quantity fo a product is 10 items (printed)	Minimun quantity fo a product is 10 items (printed)	P

Method	#	Test Description	Sample Input Data	Expected Output	Actual output	P/F
askMachineBal	1	The inputs are all valid and with zero	1 2 3 4 5 6 7 8 0	1 2 3 4 5 6 7 8 0	1 2 3 4 5 6 7 8 0	P
	2	The inputs are negative	-1 -2 -3 -43 -23 -43 -432	Negative values are not accepted. The machine only accepts 0 quantity or more.(printed gets a new input)	Negative values are not accepted. The machine only accepts 0 quantity or more.(printed gets a new input)	P

			-4324 -423			
	3	The inputs are all zero	0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0	P