CCPROG3 - S11

MCO - Convenience Store Test Script

James Benedict C. Benito

Margaret Patrice M. Ramos

Class: Transaction						
Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P / F
calculateTotals ()	1	Correctly removes VAT and applies 20% discount for Senior/PWD	subtotal=112.00, isSeniorOrPWD=true	vatAmount=0 seniorDiscountAmount=20.00, finalTotal=80.00	vatAmount=0 seniorDiscountAmount=20.00, finalTotal=80.00	p
	2	Keeps VAT included for regular customer without discount	subtotal=112.00, isSeniorOrPWD=false	vatAmount≈12.00, seniorDiscountAmount=0, finalTotal=112.00	vatAmount=12.00, seniorDiscountAmount=0, finalTotal=112.00	p
	3	Handles edge case when subtotal is exactly P100 for Senior/PWD	subtotal=100.00, isSeniorOrPWD=true	vatableSale≈89.29, seniorDiscountAmount≈17.86, finalTotal≈71.43	vatableSale≈89.29, seniorDiscountAmount≈17.86, finalTotal≈71.43	p
redeemPoints()	1	Does not Accept >0 inputs	Subtotal=100 Points used=-9	Not accepted	Not accepted	p
	2	Does nothing when	finalTotal=200.00,	pointsDis	pointsDis	p

		customer has no membership	pointsToUse=75	countAmount=75.00, finalTotal=125.00, pointsRedeemed=75	countAmount=75.00, finalTotal=125.00, pointsRedeemed=75	
	3	Applies full points discount when less than total	hasMembership()=false, pointsToUse=50	pointsDiscountAmount=0, finalTotal unchanged, pointsRedeemed=0	pointsDiscountAmount=0, finalTotal unchanged, pointsRedeemed=0	p
processPayment()	1	Calculates correct change when payment exceeds total	finalTotal=250.00, amount=300.00, method="Cash"	change=50.00	change=50.00	p
	2	Pop up appears when payment<total	finalTotal=250.00, amount=200.00, method="Cash"	Not enough funds	Not enough funds	p
	3	Handles exact payment amount with zero change	finalTotal=250.00, amount=250.00, method="Card"	change=0.00	change=	p

Class: MembershipCard						
Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P / F
accumulatePoints()	1	Accumulates 4 points when amount is P249	initialPoints=0, amount=249.00	earnedPoints=4, totalPoints	earnedPoints=4, totalPoints=4	P

				=4		
	2	Handles large amounts correctly	initialPoints=0, amount=1234.56	earnedPoints=24 (1234/50), totalPoints=24	earnedPoints=24, totalPoints=24	P
	3	Accumulates zero points when amount is exactly P49.99	initialPoints=10, amount=49.99	earnedPoints=0, totalPoints=10	earnedPoints=0, totalPoints=10	P
usePoints()	1	Uses exact number of requested points when sufficient	points=100, pointsToUse=75	returns 75.0, remaining points=25	returned 75.0, points=25	P
	2	Uses all available points when request exceeds balance	points=20, pointsToUse=100	returns 20.0, remaining points=0	returned 20.0, points=0	P
	3	Returns zero and keeps points unchanged when pointsToUse is zero	points=50, pointsToUse=0	returns 0.0, remaining points=50	returned 0.0, points=50	P
refundPoints()	1	Correctly adds refunded points back to balance	points=50, refund=20	totalPoints=70	totalPoints=70	p
	2	Rejects refund when amount is zero or negative	points=50, refund=-10	points=50, refund=-10	totalPoints=50	p
	3	Handles multiple refund operations	points=50, refund1=10refund2=5	totalPoints=65	totalPoints=65	p

		cumulatively				

Class: PerishableProduct						
Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P / F
isExpired()	1	Checks if product expired yesterday	expirationDate="2025-11-25", currentDate="2025-11-26"	T	T	P
	2	Checks if product expires today	expirationDate="2025-11-26", currentDate="2025-11-26"	F	F	P
	3	Checks if product expires tomorrow	expirationDate="2025-11-27", currentDate="2025-11-26"	F	F	P
	4	Checks if product expired last month	expirationDate="2025-10-15", currentDate="2025-11-26"	T	T	P
isFlaggedForRemoval()	1	Expires within 7 days (equal)	expirationDate = now.plusDays(7)	T	T	P
	2	Already expired	now.minusDays(2)	T	T	P
	3	Far future	now.plusDays(30)	F	F	P

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

earnPoints()	1	Earns points only when customer has valid membership card	hasMembership()=true, amount=250.00	points increase by 5 (250/50)	points increased by 5	P
	2	Does not earn points when membership card is expired	membershipCardID="DLSUCS-12345678", cardExpiryDate="2025-11-25" (expired), amount=250.00	points remain unchanged (0)	points remain 0	P
	3	Earns zero points when purchase is less than P50	hasMembership()=true, amount=49.00	points=0 (no points earned)	points=0	P
usePoints()	1	Uses partial points and returns correct discount amount	points=100, pointsToUse=50	returns 50.0, remaining points=50	returned 50.0, points=50	P
	2	Attempts to use more points than available, uses all available instead	points=30, pointsToUse=50	returns 30.0, remaining points=0	returned 30.0, points=0	P
	3	Attempting to use negative or zero points returns zero discount	points=100, pointsToUse=-10	returns 0.0, points unchanged=100	returned 0.0, points=100	P
isInactiveForTwoYears()	1	Returns true when lastActivityDate is exactly 2 years ago	lastActivityDate="2023-11-26", currentDate="2025-11-26"	returns true	t	P
	2	Returns false when lastActivityDate is	lastActivityDate="2023-11-27",	returns false	f	P

		1 year and 364 days ago	currentDate="2025-11-26"			
	3	Returns true when lastActivityDate is more than 2 years ago	lastActivityDate="2023-01-01", currentDate="2025-11-26"	returns true	t	P

Class: Product						
Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P / F
isLowStock()	1	Normal: stock above threshold	quantityInStock=10	F	F	P
	2	Edge: stock equals threshold	quantityInStock=5	T	T	P
	3	Boundary: zero stock	quantityInStock=0	T	T	P
	4	Negative stock	quantityInStock=-1	T	T	P
reduceStock()	1	Normal: sufficient stock	quantityInStock=10 quantity=3	T quantityInStock=7	T quantityInStock=7	P
	2	Edge: reduce exactly all stock	quantityInStock=3 quantity=3	T quantityInStock=0	T quantityInStock=0	P
	3	Error: insufficient stock	quantityInStock=2 quantity=5	F quantityInStock unchanged	F quantityInStock unchanged	P
	4	Zero quantity	quantity=0	T quantityInStock unchange	T quantityInStock unchange	P

				d	d	
	5	Negative quantity	quantity=-3	F	F	P
displayDetails()	1	Normal: brand+variant present	brand="Coke", variant="Zero", price=50, stock=10, id="B-001"	"B-001", "Coke (Zero)", "P50.00", "Stock: 10"	"B-001", "Coke (Zero)", "P50.00", "Stock: 10"	P
	2	Edge: no brand or variant	brand="", variant=""	string contains no parentheses for brand/variant	string contains no parentheses for brand/variant	P
getSubcategoriesString()	1	Normal: multiple subcats	["Food","Snack"]	"Food, Snack"	"Food, Snack"	P
	2	Single subcat	["Beverage"]	"Beverage"	"Beverage"	P
	3	Null list	null	(empty string)	(empty string)	P
getBrandAndVariant()	1	Both present	brand="Dove", variant="100ml"	"Dove (100ml)"	"Dove (100ml)"	
	2	Only brand present	brand="Dove", variant=""	"Dove"	"Dove"	
	3	Only variant present	brand="", variant="500ml"	"500ml"	"500ml"	
	4	Neither present	brand="", variant=""	""	""	

Class: SaleManager

Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P / F
getDiscountedPrice(Product)	1	Applies percentage	price=100, percentage=15%	85.00	85.00	P
	2	Applies fixed discount	price=50, fixed 20	30.00	30.00	P
	3	Discount clamps to zero	price=10, fixed 20	0.00	0.00	P

Class: ProductSale						
Method	#	Test Description	Sample Input Data	Expected Output	Actual Output	P / F
calculateDiscountedPrice()	1	Percentage active	price=100 type=PERCENTAGE value=25 active	75.00	75.00	P
	2	Fixed active	price=80 type=FIXED value=30 active	50.00	50.00	P
	3	Fixed discount >= price	price=20 fixed=30	0.00	0.00	P