CCPROG1 Term 1, AY 2021 – 2022

Test Script Document

Name	DE LEON, FRANCIS ZACCHARIE B.
Section	S17B

Test Scripts

There should be at least 3 distinct test classes (as indicated in the description) per function. There is no need to test functions which are only for screen design.

Function Name: `displayArt`

Test #	Test Description	Sample Input	Expected Result	Actual Result	P/F
1	The files exist with the right directory and is a string.	filename =“AsciiArt/StartArt.txt”	Display Ascii Art	Display Ascii Art	P
2	File does not exist.	filename = “AsciiArt/hello.txt”	error opening AsciiArt/hello.txt	error opening AsciiArt/hello.txt	P
3	Directory is wrong	filename = “Merchant.txt”	error opening Merchant.txt	error opening Merchant.txt	P

Function Name: `start`

Test #	Test Description	Sample Input	Expected Result	Actual Result	P/F
1	User inputs the number corresponding to the city	dChoice = 1	Go to the city of Winterfell	Go to the city of Winterfell	P
2	User decides to change control method	dChoice = 7	Change control method and redraw the start screen.	Change control method and redraw the start screen.	P
3	User decides to quit the game	dChoice = 8	Display final user stats	Display final user stats	P
4	User inputs invalid numbers	dChoice = 0	The program redraws the start screen until there is a valid input	The program redraws the start screen until there is a valid input	P

Function Name: `randomNum`					
Test #	**Test Description**	**Sample Input**	**Expected Result**	**Actual Result**	**P/F**
1	The number range is [-1, 1]	dLowerBound = -1; dUpperBound = 1;	Generate a random number from -1 to 0	Generates a random number from -1 to 0	P
2	The number range is 0	dLowerBound = 0; dUpperBound = 0;	Generate a number of only 0	Generates a number of only 0	P
3	The number range is [5, 10]	dLowerBound = 5; dUpperBound = 10;	Generate a random number from 5 to 10	Generates a random number from 5 to 10	P

Function Name: `getPrice`					
Test #	**Test Description**	**Sample Input**	**Expected Result**	**Actual Result**	**P/F**
1	The city is Winterfell and the product that is to be printed is Timber.	dCity = 0; ProductNumber = 1;	Return a random number from 200 - 250	Returns a random number from 200 - 250	P
2	The city is Winterfell and the product that is to be printed is Sweet Beet	dCity = 0; ProductNumber = 0;	Return a random number from 180-200	Returns a random number from 180-200	P
3	The city is Volantis and the product that is to be printed is Sweet Beet	dCity = 5; ProductNumber = 0;	Return a random number from 100 - 150	Returns a random number from 100 - 150	P

Function Name: `displayDashBoard`					
Test #	**Test Description**	**Sample Input**	**Expected Result**	**Actual Result**	**P/F**
1	This displays the dashboard of products and their prices and their current status	dCity = 0, newCity = 0, dDays = 1, dGold = 100.00, dDebt = 100.00, dSavings = 100.00, dCapacity =	Displays the main game dashboard. It displays the city as Winterfell, the remaining days (1), Gold, Debt, Savings are	Displays the main game dashboard. It displays the city as Winterfell, the remaining days (1), Gold, Debt,	P

		20, dCapacity_Max = 50;	100.00, Capacity as 20 and max capacity as 50	Savings are 100.00, Capacity as 20 and max capacity as 50	
2	This displays the dashboard of products and their prices and their current status	dCity = 5; newCity = 1; dDays = 5; dGold = 100.00; dDebt = 100.00; dSavings = 100.00; dCapacity = 20; dCapacity_Max = 50;	Displays the main game dashboard. It displays the city as Volantis, the remaining days (5), Gold, Debt, Savings are 100.00, Capacity as 20 and max capacity as 50. It changes the newCity to 0	Displays the main game dashboard. It displays the city as Volantis, the remaining days (5), Gold, Debt, Savings are 100.00, Capacity as 20 and max capacity as 50. It changes the newCity to 0	P
3	This displays the dashboard of products and their prices and their current status	dCity = 2; newCity = 1; dDays = 3; dGold = 100.00; dDebt = 100.00; dSavings = 100.00; dCapacity = 20; dCapacity_Max = 100;	Displays the main game dashboard. It displays the city as Myr, the remaining days (3), Gold, Debt, Savings are 100.00, Capacity as 20 and max capacity as 50. It changes the newCity to 0	Displays the main game dashboard. It displays the city as Myr, the remaining days (3), Gold, Debt, Savings are 100.00, Capacity as 20 and max capacity as 50. It changes the newCity to 0	P

Function Name: `displayCity`					
Test #	**Test Description**	**Sample Input**	**Expected Result**	**Actual Result**	**P/F**
1	The user decides to buy sweet beet	OptAct = 1 Action = 0	Return the product	Returns the product	P

			number and change the OptAct Variable value	number and changes the OptAct Variable value	
2	The user decides to sell timber	OptAct = 0 Action = 1	Return the product number and change the OptAct Variable value	Returns the product number and changes the OptAct Variable value	P
3	The user decides to change control method	Action = 67 or 'C'	Change control method	Changes control method	P
4	The user inputs an invalid number.	Action = 8	The program will ask again what input the user wants	The program asks again what input the user wants	P
5	The user inputs an invalid character.	Action = 90 or 'Z'	The program will ask again what input the user wants	The program asks again what input the user wants	P
6	The user dices to go to the Wheelhouse	Action = 87 or 'W'	This function will return the number 87 or 'W'	This function returned the number 87 or 'W'	P
7	The user decides to go to the IronBank.	Action = 73 or 'I'	This function will return 73 or 'I'	This function returned 73 or 'I'	P

Function Name: `opBuySell`					
Test #	**Test Description**	**Sample Input**	**Expected Result**	**Actual Result**	**P/F**
1	The operation is buy, and the user chooses product 0 (Sweet Beet). The player also has enough gold.	OptAct = 1 dAction = 0 dGold = 2000.00	The price of the product is subtracted to the player's gold and the capacity is added and inventory is updated.	The price of the product is subtracted to the player's gold and the capacity is added and inventory is updated	P

2	The operation is buy and the user chooses the product 0 (Sweet Beet). The player has insufficient gold.	OptAct = 1 dAction = 0 dGold = 50.9	The program will not deduct gold and will not update inventory and loop back to displayCity and ask which product the user wants to buy.	The program did not deduct gold and did not update inventory and loop back to displayCity and ask which product the user wants to buy.	P
3	The operation is sell and the user chooses the product 1 (Timber). The player has the available product for sale.	OptAct = 0 dAction = 1 dInventory[dAction] = 1	The price of the product is added to the player's current gold. The capacity is subtracted, and inventory is updated.	The price of the product is added to the player's current gold. The capacity is subtracted, and inventory is updated.	P
4	The operation is sell and the user chooses the product 1 (Timber). The player has insufficient product for sale.	OptAct = 0 dAction = 1 dInventory[dAction] = 0	The program will not add gold and will not update inventory and loop back to displayCity and ask which product the user wants to sell.	The program did not add gold and did not update inventory and loop back to displayCity and ask which product the user wants to sell.	P

Function Name: `displayBankDetails`					
Test #	**Test Description**	**Sample Input**	**Expected Result**	**Actual Result**	**P/F**
1	This function is used for the IronBank. It displays the user's gold, debt, savings.	dGold = 100.00; dDebt = 0.00; dSavings = 150.5	It displays the user's details. GD = 100.00; Debt = 0.00 Savings = 150.50	It displays the user's details. GD = 100.00; Debt = 0.00 Savings = 150.50	P

2	This function is used for the IronBank. It displays the user's gold, debt, savings.	dGold = 200.50; dDebt = 15.30; dSavings = 100.05	Display: Gold = 200.50; Debt = 15.30; Savings = 100.05	Display: Gold = 200.50; Debt = 15.30; Savings = 100.05	P
3	This function is used for the IronBank. It displays the user's gold, debt, savings.	dGold = 150.2 dDebt = 0.00 dSavings = 100.05	Display: Gold = 150.2 Debt = 0.00 Savings = 100.05	Display: Gold = 150.2 Debt = 0.00 Savings = 100.05	P

Function Name: `absVal`					
Test #	**Test Description**	**Sample Input**	**Expected Result**	**Actual Result**	**P/F**
1	Input is a float number	num = 0.0	num = 0.0	num = 0.0	P
2	Input is a float number	num = 15.3	num = 15.3	num = 15.3	P
3	Input is a float number	num = -20.5	num = 20.5	num = 20.5	P

Function Name: `IronBank`					
Test #	**Test Description**	**Sample Input**	**Expected Result**	**Actual Result**	**P/F**
1	The user is not in any operation and inputs an invalid input.	Operation = -1; dChoice = 65 or 'A';	The program will loop back to displayBankDetails and ask again for input	The program looped back to displayBank Details and ask again for input	P
2	The user is in deposit mode and has entered an amount higher than their gold	Operation = 0 dGold = 150.7 dAmount = 250.5	The program will ask again how much the user wants to deposit and will not deduct or add anything.	The program asked again how much the user wants to deposit and did not deduct or add anything.	P
3	The user is in deposit mode and has entered an amount lower or equal to their gold	Operation = 0 dGold = 150.05 dAmount = 150.05	dGold will be subtracted by the amount and savings will be added by the amount.	dGold is subtracted by the amount and savings is added by the amount.	P

4	The user is in withdraw mode and has entered an amount lower or equal to their savings	Operation = 1 dSavings = 150.5 dAmount = 140.5	dGold will be added by the amount and savings will be subtracted by the amount.	dGold is added by the amount and savings is subtracted by the amount.	P
5	The user is in withdraw mode and has entered an amount higher than their savings	Operation = 1 dSavings = 10.5 dAmount = 2000.00	The program will ask again how much the user wants to withdraw and will not deduct or add anything.	The program asked again how much the user wants to deposit and did not deduct or add anything.	P
6	The user is in get a loan mode and has entered an amount higher than the debt or player's gold	Operation = 2 dAmount = 500.50 dDebt = 200.07 dGold = 400.1	The program will ask again how much the user wants to withdraw and will not deduct or add anything.	The program asked again how much the user wants to withdraw and did not deduct or add anything.	P
7	The user want to go back to the main menu while in any operation.	Operation >= 1 and Operation <= 4; dAmount < 0	Operation = -1 The program loops back this function.	Operation = -1 The program loops back this function.	P

Function Name: `WheelHouseChoice`					
Test #	**Test Description**	**Sample Input**	**Expected Result**	**Actual Result**	**P/F**
1	This function is very similar to the start() function. It displays choices of city and returns what the user chooses.	dChoice = 0;	Return 0	Returns 0	P
2	This time, the user decides to change control method	dChoice = 6;	loop back to this function, but with a different control method.	loops back to this function, but with a different control method.	P

3	The user decides to enter an invalid digit or character	dChoice = 65 or 'A'	Loop this function until the user inputs a valid input.	Loops this function until the user inputs a valid input.	P

Function Name: `newCityDay`					
Test #	**Test Description**	**Sample Input**	**Expected Result**	**Actual Result**	**P/F**
1	This function is called whenever the user decides to go to another city. The dCity is changed and newCity is turned intro 1 (true)	dCityTemp = 1; newCity = 0;	dCity is assigned by the value of dCityTemp newCity = 1;	dCity is assigned by the value of dCityTemp newCity = 1;	P
2	dDay is subtracted by 1	dDay = 15;	dDay--;	dDay--;	P
3	dSavings is compounded by 10% and dDebt is compounded by 5%	dSavings = 50.00; dDebt = 100.00;	dSavings = 55; dDebt = 105;	dSavings = 55; dDebt = 105;	P

Function Name: `capMerchent`					
Test #	**Test Description**	**Sample Input**	**Expected Result**	**Actual Result**	**P/F**
1	This function is called before the whole program loops into the main dash board and the user enters a new city. dEncounter and rand() % 10 is equal	dEncounter = 0; rand() % 10 = 0	The function will execute.	The function will execute.	P
2	This function is called before the whole program loops into the main dash board and the user enters a new city. dEncounter and rand() % 10 is not equal	dEncounter = 0; rand() % 10 = 1;	The function will not execute.	The function did not execute.	P
3	The user decides to buy the merchant's extra capacity and has enough gold	dFinalAns = 1; dGold = 500;	dCapacity_Max += 50; dGold -= 200;	dCapacity_Max += 50; dGold -= 200;	P
4	The user decides to buy the merchant's extra capacity and has not enough gold	dFinalAns = 1; dGold = 199.6;	Will not change any variable.	Did not change any variable.	P

5	The user does not want to do business with the merchant.	dFinalAns = 0;	Will not change any variable.	Did not change any variable.	P