TEST PLAN (MANUAL TESTING)

PLANT WATERING SCHEDULE TERMINAL APPLICATION

Test Date: 4th May, 2022

PROCESS	NO.	TEST CASE	STEP	DESCRIPTION	STATUS	EXPECTED RESULT	ACTUAL RESULT	COMMENT
Check/create data.json file	1	File handling						
			1	Check to see if .json file exists	passed	Uses try to check if file exists		
			2	If it exists, just read file	passed	Reads file and data correctly		
			3	If it does not exist, create new .json file	passed	Creates file if not found		
			4	Write empty list into file	passed	Creates empty list []		
			5	data = []	passed			
			6	File should be named data.json	passed	file name is data.json		
				Enter some sample data to test input is added correctly to data.json				
				"Name" = "Monstera"	passed	Added to data.json		
				"Frequency" = "fortnightly"	passed	Added to data.json		
				"Last_Watered" = "2023-05-02"	passed	Added to data.json		
				"Water_Needed" = "100"	passed	Added to data.json		
Add plant	2	Function to Add						
			1	Ask for user input (Name of plant)	passed	Allows user to enter a string		
			2	Ask for user input (Frequency)	passed	Accepts "weekly", "fortnightly" and "monthly" only		
			3	If user does not enter weekly, fortnightly or monthly, raise Value Error	passed	Raise exception		
			4	Keep raising Value Error until user enters valid input	passed	Keeps looping through exception		
			5	Allow user to quit loop ("q" or "quit") and go back to main menu	passed	Exits the entire function		
			6	Ask for user input (When the plant was last watered in YYYY-MM-DD)	passed	Allows user to enter date		
			7	Raise Value Error if not in the correct date format	passed	Raise exception		
			8	Allow user to quit loop ("q" or "quit") and go back to main menu	passed	Exits function		
			9	Ask for user input (How much water is needed in mL)	passed	Allows user to enter data in mL		
			10	If input can not be converted to integer, raise Value Error	passed	If input can be converted to integer, proceed		
			11	If input is not greater than or equal to 0, raise Value Error	passed	Raise exception if number is less than 0		
			12	Once all inputs entered correctly, read data.json and create dictionary with inputs	passed	Open and read data.json, create new dictionary		
			13	Append to empty list and write to data.json	passed	Append to empty list and write to data.json		
			14	Adding a new plant should not overwrite any existing data in data.json	passed	Adds new plant to list and keeps exisitng data intact		
			15	Format should be consistent	passed	Same format across all plants		
			16	If user quits midway through adding plant, no infromation gets added to data.json	passed	No input is stored and no data is added to file		
Remove plant	3	Function to Remove						
			1	Ask for user input (Name of plant)	passed	Allows user to enter input (Name of plant)		
			2	Opens and reads data.json	passed	Open data.json and read contents		
			3	If the string (from user input) is not equal to the key value under "Name"	passed	Conditional statement		
			4	Append to empty list variable (plant_list)	passed	Adds all the plants that do not have the same string as user input		
			5	Write data to data.json	passed	Writes the correct data to data.json		
			6	data.json should contain all existing plants minus the one the user entered	passed	Whatever the user entered is now omitted from data.json		
			7	If user enters a plant not in data.json, does nothing	passed	Does not omit anything if user enters a string not found in file		
			8	Test to see if function removes desired input from data.json and writes the rest	passed	Does function work as intended		
View plants that need watering today	4	Function to View which Plants need Watering						
			1	The variable now, is a time date object and is equal to today's date	passed	Variable is equal to today's date		
			2	Open and read data.json	passed	Opens file		
			3	Loop through every iteration (dictionary) of data.json	passed	Loops through every iteration in list		
			4	Calculate the difference in days between variable now and "Last_Watered" date	passed	Calculates the difference in .days		
			5	if "Frequency" is equal to "weekly" and difference in days is >= 7, display	passed	Display correctly with sample data		
			6	if "Frequency" is equal to "fortnightly" and difference in days is >= 14, display	passed	Display correctly with sample data		
			7	if "Frequency" is equal to "monthly" and difference in days is >= 30, display	passed	Display correctly with sample data		
				Enter some sample data to test if program is displaying plants correctly				
			8	Plant1, "weekly", last watered 10 days ago	passed	Printed in display		
			9	Plant2, "weekly", last watered 2 days ago	passed	NOT printed in display		
			10	Plant3, "fortnightly", last watered 40 days ago	passed	Printed in display		
			11	Plant4, "fortnightly", last watered 6 days ago	passed	NOT printed in display		
			12	Plant5, "monthly", last watered 34 days ago	passed	Printed in display		
			13	Plant6, "monthly", last watered 10 days ago	passed	NOT printed in display		
Mark plant as watered	5	Function to Mark Plant as Watered						
			1	Ask for user input (Name of plant)	passed	Allow user to enter input (Name of Plant)		
			2	Get today's date as a variable	passed	Fetch today's date as a variable		
			3	Convert today's date into a string	passed	Convert today's date into a string datatype		
			4	Open and read data.json	passed	Open file		
			5	If user input (name) is equal to key value of "Name" in data.json	passed	Loop through data.json file		
			6	Change "Last_Watered" to today's date	passed	If name matches "Name", change "Last_Watered" to today's date as a string		
			7	Write changes to data.json	passed	Write changes into data.json		
				Test to see whether function changes date in data.json				
			8	Does "Last_Watered" change to today's date in data.json	passed	Manually check whether this function works		
Update amount of water needed	6	Function to Change Amount of Water Needed						

			1	Ask for user input (Name of plant)	passed	Allow user to enter input (Name of Plant)		
			2	Open and read data.json	passed	Open file		
			3	Allow user to exit back to main menu by typing "q" or "quit"	passed	Typing "q" or "quit" will exit program and not store/change any data in data.json		
			4	Ask for user input (change amount of water needed for a plant in mL)	passed	Allow user to enter input (Amount of water needed mL)		
			5	Use try, to change data type into an integer	passed	Can the input be converted into an integer data type?		
			6	Is the integer greater than or equal to 0? Then proceed	passed	Is the integer >= 0 ?		
			7	Otherwise, raise Value Error	passed	Does the function raise a Value Error if the two above conditions are not True?		
			8	Loop through data, if all conditions are true. Change value of "Water_Needed" to new input	passed	Does function change the value we want, if the plant names are equal?		
			9	Write data to data.json	passed	Writes new data to data.json		
				Test to see whether function changes date in data.json				
			10	Does "Water_Needed" change to user input entered in data.json	passed	Manually check		
View entire plant list	7	Function to View Entire Plant List						
			1	Open and read data.json	passed	Open file		
			2	Print off plant names (i.e. "Name" key)	passed	Loop through data.json to print plant names		
			3	Print off other data that the programmer wishes the user to see	passed	Displays other information as desired		