Pet Car Alarm: Challenge #3

In the Pet Car Alarm project, you will create a system that detects if the pet is in the car and measure the temperature The car will sound a warning with flashing lights and roll down the windows if your pet could be in danger

In this challenge, you will use the TI-Innovator, servo motor and temperature sensor to build a cooling system. This skill will be used to roll down the window of the car if it is too warm for the pet inside the car.

Goals:

1. Use the SET SERVO.CONTINUOUS command to turn a servo motor on and off.
2. Use the READ TEMPERATURE command to read ambient temperature using a sensor.
3. Use and If statement to make a selection.
4. Use a While loop to repeat code.

Background:

Have you ever wondered how heating and cooling systems work? Have you ever changed the theromstat setting in a room or a car? Some heating and cooling systems require the use to turn knobs to turn them on and off. The user must actively switch on and off the heating and cooling system. Other thermostats are temperature controlled. When the temperature reaches the predetermined level, the heating or cooling mechanisms is activated , and the user doesn't have to manually turn them on and off. In this activity, you'll program the calculator to use a sensor to measure the temperature and control a servo motor synthesizing turning on and off a fine based on the temperature .

Command	Example	Behavior
CONNECT <type> <number> TO <port>	`Send "CONNECT TEMPERATURE 1 TO IN1"`	Associates the first TEMPERATURE object with a temperature module plugged into port IN1 on the Hub.
SET <type> <number> TO <value>	`Send "SET SERVO.CONTINUOUS 1 CW 20 TIME 1"`	Turns on the first continuous servo motor object at power 20 (range 0-100) in the clockwise direction (CW vs. CCW) for a time of 1 second.
Wait <number>	`Wait 3`	Pauses the program for a specified number of seconds. In this case 3 seconds.
READ <type> <number>	`Send "READ TEMPERATURE 1"`	Reads a measurement from the first temperature object.
Get <variable>	`Get t`	Stores the temperature measurement into a variable named *t*. The value stored will contain the measurement from the immediately preceding READ command. Note: a Get command must immediately follow a READ command.
DispAt <line #> , <"text"> , <variable name>	`DispAt 3, "Temperature = ", t`	When variable *t* has a value of 26, the following line is displayed on the calculator: Temperature = 26 (temp readings are in °C by default)
While <Boolean expression> <statements>	`key:=" "` `While key≠"esc"`	The statements in the While loop are executed until the escape key is pressed. The While loop continues as long as the Boolean expression evaluates to "true". The

EndWhile	`READ "TEMPERATURE 1"`
	`Get t`
	`DispAt 3,"Temperature= ",t`
	`key:=getKey()`
	`Wait 1`
	`EndWhile`
		variable *key* is set to an initial value of an empty string using the assign function, ":=". getKey() is a function that returns a string with the name of the last key pressed while a programming is running. In this program the value of getKey() is stored to the variable *key*.

<Boolean expression> and <Boolean expression>	`If t>25 Then`
	`  Send "SET COLOR 0 255 0"`
	`EndIf`
		When the expression is "true" the statement is executed. Otherwise, the function returns false and the statement is skipped.

If <Boolean expression> Then
<statements 1>
Else
<statements 2>
End
	`If a < 75 Then`
	`DispAt 5,"Cool Enough"`
	`Else`
	`DispAt 5,"Too Hot- Fan please"`
	`EndIf`
		The example decision tree has a Boolean expression with corresponding statements to execute if true. It also has an Else condition that executes corresponding statements when the Boolean expression is false. This Else condition ensures that a set of statements will always be executed. When this decision tree executes, focus proceeds from top-down. If the value of *t* is less than 75 the commands after Then are executed. In this case the output "Cool Enough" is displayed on row 5. If the value of *t* is greater than or equal to 75 the commands after Else are executed. In this case the output "Too Hot- Fan please" is displayed on row 5.

Challenge:

Challenge 3a: Connect a continuous servo motor to the TI-Innovator Hub and cause it to rotate clockwise (CW) and then in the opposite direction, counterclockwise (CCW).

Challenge 3b: Connect a temperature sensor to the TI-Innovator Hub and display the temperature on the calculator.

Challenge 3c: Create a program that synthesizes the turning on and off of an air conditioning system. If the temperature is too warm turn the servo motor on for 2 seconds.