Implementing Pseudocode

Code-Along Activity: Thermostat

NOTE: This activity is meant to be fully instructor-led. Directions are provided in case someone wants to refer to the process later, repeat a step, or move ahead to do the bonus.

Some people prefer to keep it a little cooler in their homes and others want to feel like they're in Hawaii. People on opposite sides of this issue may never come to an agreement; however, one thing we can all agree on is that it's ideal to be able to program thermostats to our personal preferences!

Part One: Starter Code

For this short code-along activity, we'll be working in a REPL. You can access that starter code here. Go ahead and fork the code to your own account.

As always, take a look at the starter code you already have before getting started with anything new!

Part Two: Test the Existing Code

Take a moment to review the lines of existing code. We have a variable, `currentTemp`, that holds the current temperature. This number is randomized every time we re-run the code.

```
// Note: We'll be using the Fahrenheit scale for this assignment.

// This code sets a random starting current temperature between 0 and 100 degrees.
let currentTemp = Math.round(Math.random() * 100)

// Give the thermostat user a status message.
console.log(`The current temperature is ${currentTemp}°F` .)
```

Click here to copy

Directions

Run the code in the REPL at least three times. You should get a different value every time. Make sure this works!

Part Three: Make Pseudocode

Now we want to write out the logic for our thermostat. We'll need to determine which variables we need to track, what conditions may exist, and what we should do for each of those conditions.

Directions

Take a moment to think about each of the questions below. Discuss with the class.

1. Write out what variables you think your thermostat needs to track. Remember, we already have one-`currentTemp`.
2. Given your variables, what are the possible states at the beginning of the program? Too hot, too cold, and just right? What do your variable's values look like in each of those cases?
3. In each of the conditions from the last step, what is the proper action to take? (Assume you can only increment or decrement by one degree at a time.)

4. Does any of your code need to be repeated? If so, which parts?

Now that we've done a little thinking about what our code needs to do, let's try writing out some pseudocode. Here are some tips!

- Start with variable declaration.
- Write code that checks the conditions.
- Focus on a single condition first. For example, don't worry about the "too hot" and "just right" conditions, just start with the "too cold" logic.
- For code that may need to happen more than once, it should go inside of your loop.

At the end of this process, you should have some pseudocode that looks like this. Go ahead and check your work. Remember, all pseudocode doesn't look the same and it's completely okay if your style or ordering choice varies a bit from our example.

Part Four: Write It Out!

Now it's time to implement our code. Your instructor will be leading you through this part. Pay close attention to what your instructor does. What parts of the code are written first? Does your instructor make the opening and closing curly braces together? Why do you think that is? How does your instructor approach debugging when an issue comes up?

All Done?

Check out the solution code!

Bonus

What if, instead of just writing down an unchangeable preferred temperature, we actually had the ability to take input from the user? That would be neat! Let's think about what we'll need.

- A default preferred temperature, in case the thermostat user sets an invalid value or hasn't gotten around to setting a preference yet
- A way to take user input. We can do this with the `prompt` function. See the example below.
- Some error checking in case the user tries to enter an invalid value. This could include anything that's not a number or any number outside of a reasonable range.
- A way to change the value the user types from a string to a number.

A Prompt Example

```
let userInput = prompt('Would you like to set a preferred temperature? (Type degrees in Fahrenheit.)')
```

Click here to copy

Converting to a Number

```
let numValue = Number(userInput)
```

Click here to copy

Error Checking

You can check whether a value evaluates to a number or not by using the `isNaN` function. This function returns `true` if the value is NOT a number, and `false` if the value is a valid number. In this case, we want to execute our code within the conditional if the `isNaN` function returns `false`. Therefore, we'll be using the negation operator, `!`.

Your Starter Code

Here is the starter code that you'll need. You can put it right below the variable declarations. Implement each portion with the word *TODO*.

```
const DEFAULT_PREFERRED_TEMP = 72
// TODO: Set preferredTemp to default temp (this involves rewriting a line from your original solution).

let userInput = prompt('Would you like to set a preferred temperature? (Type degrees in Fahrenheit.)')

if (!isNaN(userInput)) {
 let numValue = Number(userInput)
 // Your code here
 // TODO: Check that the numValue entered is between 0 and 100.
 // TODO: Set the preferred temperature to the numValue entered.
}
```

Click here to copy

Remember, if the user enters an invalid or outrageous value, the proper action is to do nothing. This is why we set the default preferred temperature at the beginning!

When you're all through, take a look at the solution to the bonus. Were you able to get close? Congratulations on making it this far!