Practice with Fundamentals

Aidan Hunt, University of Washington

Learning Objectives

During this lesson, students will:

- Practice pseudocoding
- Combine functions, data structures, for loops, and logic into a single program
- Use logic to check if user inputs are valid.

Check-in / Announcements

- Homework 1 grades posted
- Homework 2 due Friday
- Office hours today (in person) and tomorrow (virtual)

An Example

Let's work through an example that combines everything we've learned, and introduces a few new tools, too!

Objective: create a calculator for a series circuit with an arbitrary number of resistances.

- User is prompted for supply voltage, number of resistors, and resistance value of each resistor
- Using these values, calculate the current in the circuit, voltages across each resistor, and print results

```
Welcome to the resistor voltage calculator.
Please enter the supply voltage (V): 5
Please enter the number of resistors: 3
Please enter the resistance (ohms) for resistor #1: 100
Please enter the resistance (ohms) for resistor #2: 20
Please enter the resistance (ohms) for resistor #3: 50
```

```
Supply voltage (V): 5.0
Current (A): 0.029411764705882353
Resistances (ohms): [100.0, 20.0, 50.0]
Voltage across each resistor (V): [2.941176470588235, 0.5882352941176471,
1.4705882352941175]
```

What can we assume about user input?

- Assume that the user provides input that can be converted to numerical values (e.g., they enter numbers, not letters).
- But, we cannot assume that the numbers they enter will be valid.

Other input cases to consider

- If the user enters an invalid number of resistors, print a message instead of performing calculation:
  - Negative number of resistors: print an invalid value message.
  - Zero resistors: print a short-circuit message.
- When providing the resistance values for each resistor (in ohms), if the user enters an invalid resistance (<= 0), prompt them again until they provide a valid resistance.

Example of invalid resistor:

```
Please enter the supply voltage (V): 10
Please enter the number of resistors: -4
Invalid number of resistors - no calculation performed.
```

Example of zero resistors:

```
Please enter the supply voltage (V): 10
Please enter the number of resistors: 0
No resistors specified - short circuit!
```

Example of reprompting if invalid resistance:

```
Please enter the supply voltage (V): 10
Please enter the number of resistors: 2
Please enter the resistance (ohms) for resistor #1: 0
Please enter the resistance (ohms) for resistor #1: -1
Please enter the resistance (ohms) for resistor #1: -2
Please enter the resistance (ohms) for resistor #1: -3
Please enter the resistance (ohms) for resistor #1: -4
Please enter the resistance (ohms) for resistor #1: 40
Please enter the resistance (ohms) for resistor #2: 10

Supply voltage (V): 10.0
Current (A): 0.2
Resistances (ohms): [40.0, 10.0]
Voltage across each resistor (V): [8.0, 2.0]
```

Pseudocode session

Develop a *pseudocode outline*

- Break down the problem into sub-tasks
- After that, if needed, break down the sub-tasks into sub-sub-tasks
- Outline the structure/flow of information through our program

Work for 1 min on your own, then chat with others for 1 min

In [1]:
```
# Print welcome message

# Get input voltage from user

# Get number of resistances from user

# Check if number of resistances is valid

    # If less than zero:
         # print invalid resistor message

    # If equal to zero: print short circuit message
        # print invalid resistor message

    # Otherwise, perform calculations:

        # Get values of resistances from user
            # Prompt user for each resistance
                # Reprompt if invalid input

        # Calculate current
            # I = V * R_total

        # Calculate voltage across each resistor
            # V = I * R[i]

        # Print a summary of the results
```

Filling in the pseudocode outline

Now we just need to write code for each subtask! We should use *functions* to represent each sub-task.

Start with the easiest, printing the welcome message:

In [2]:
```
# Print welcome message
def printStartup():
    '''
    Prints the startup message for the resistor calculator.
    '''
    print('Welcome to the resistor voltage calculator.')
    print()
```

Getting user input is also quite easy! We can use the built-in `input()` function.

- Put your prompt in the parentheses
- Outputs what the user entered, as as string.
- Use type conversion to convert output to proper type.

In [3]:
```
# Get input voltage from user
def getSupplyVoltage():
    '''
    Prompts the user for the supply voltage and returns the input as a float.
    '''

    vSupply = input('Please enter the supply voltage (V): ')
    return float(vSupply)
```

Group work session

Work with those around you to define functions for other tasks, ignoring error checking for now.

(Note: don't use `numpy` instead, let's practice using our basic built-in lists)

Continuing to fill in our program

Getting the number of resistances is similar to getting the supply voltage:

In [4]:
```
# Get number of resistances from user
def getNumberOfResistors():
    '''
    Prompts the user for the number of resistors in the circuit and returns
    the result as an integer.
    '''

    nRes = input('Please enter the number of resistors: ')
    return int(nRes)
```

Calculating the current and the voltage across each resistor are also fairly straightforward.

In [5]:
```
# Calculate current
def calcCurrent(vSupply, totalRes):
    '''
    Given the supply voltage and total resistance in a series circuit,
    calculates and returns the current in the circuit (in Amperes).
    '''

    I = vSupply / totalRes
    return I

# Calculate voltage across each resistor
def calcResistorVoltages(resOhms, I):
    '''
    Given the supply voltage and total resistance in a series circuit,
    calculates and returns the current in the circuit (in Amperes).
    '''

    resVolts = [ohms * I for ohms in resOhms]
    return resVolts
```

As is printing the results summary

In [6]:
```
# Print a summary of the results
def printSummary(vSupply, I, resOhms, resVolts):
    '''
    Given the supply voltage of a series circuit (in Volts), the current in the
    circuit (in Amps), a list of the resistances in the circuit (in ohms),
    and a list of the voltages across each resistor (in Volts), prints a
    summary of the circuit properties to the console.
    '''

    print()
    print('Supply voltage (V):', vSupply)
    print('Current (A):', I)
    print('Resistances (ohms):', resOhms)
    print('Voltage across each resistor (V):', resVolts)
```

The trickiest sub-task is getting the resistances:

- Want to prompt *for* each resistance, so probably want a for loop!

- Create a prompt using the index of each resistor.
- Can either add to an empty list, or update a list of a certain size

In [7]:
```
# Get values of resistors from user
def getResistances(nRes):
    '''
    Given the number of resistors in a series circuit, prompts the user for the
    resistance in ohms of each resistor. Returns the resistances (as a list of
    floats), as well as the total resistance in the circuit (as a float).

    It is assumed that the number of resistors provided is valid (an integer
    greater than 0.)
    '''

    resOhms = [None] * nRes
    totalRes = 0

    # Prompt user for each resistance
    for i in range(nRes):
        promptStr = 'Please enter the resistance (ohms) for resistor #' + str(i+1) + ':
        currOhms = float(input(promptStr))

        # Add to list and total
        resOhms[i] = currOhms
        totalRes += currOhms

    return resOhms, totalRes
```

Our pseudo-code outline becomes the main function that uses the sub-functions to perform the task!

In [8]:
```
def resistorCalculator():
    '''
    Prompts the user for properties of a series circuit. Computes the current in
    the circuit as a function of the specified resistances, as well as the
    voltage drop across each resistor. Prints a summary to the console.
    '''

    printStartup()

    # Get input voltage and number of resistors
    vSupply = getSupplyVoltage()
    nRes = getNumberOfResistors()

    # Calculate resistances
    resOhms, totalRes = getResistances(nRes)

    # Calculate current and resistor voltages
    I = calcCurrent(vSupply, totalRes)
    resVolts = calcResistorVoltages(resOhms, I)

    # Print results
    printSummary(vSupply, I, resOhms, resVolts)
```

In [9]:
```
# Example output
resistorCalculator()
```

```
Welcome to the resistor voltage calculator.

Please enter the supply voltage (V): 5
Please enter the number of resistors: 3
Please enter the resistance (ohms) for resistor #1: 10
Please enter the resistance (ohms) for resistor #2: 20
Please enter the resistance (ohms) for resistor #3: 30
```

```
Supply voltage (V): 5.0
Current (A): 0.08333333333333333
Resistances (ohms): [10.0, 20.0, 30.0]
Voltage across each resistor (V): [0.8333333333333333, 1.6666666666666665, 2.5]
```

Key takeaways:

- Pseudocoding breaks down the problem and leads to good program structure.
- All functions should have docstrings!

Error Checking

Now let's add the error checking behavior:

The easiest one to address are the different number of resistors cases.

- Main function is delegating to subfunctions, then making decisions with the results.
- Remember to update docstring

In [9]:
```
def resistorCalculator():
    '''
    Prompts the user for properties of a series circuit. Computes the current in
    the circuit as a function of the specified resistances, as well as the
    voltage drop across each resistor. Prints a summary to the console.

    If the user specifies an invalid number of resistors (less than or equal
    to zero), no calculation is performed.
    '''

    printStartup()

    # Get input voltage and number of resistors
    vSupply = getSupplyVoltage()
    nRes = getNumberOfResistors()

    if nRes < 0: #  A negative number of resistors was specified
        print('Invalid number of resistors - no calculation performed.')
    elif nRes == 0: # Zero resistors specified
        print('No resistors specified - short circuit!')
    else: # A positive number of resistors specified
        # Calculate resistances
        resOhms, totalRes = getResistances(nRes)

        # Calculate current and resistor voltages
        I = calcCurrent(vSupply, totalRes)
        resVolts = calcResistorVoltages(resOhms, I)

        # Print results
        printSummary(vSupply, I, resOhms, resVolts)
```

In [10]:
```
# Example output
resistorCalculator()
```

```
Welcome to the resistor voltage calculator.

Please enter the supply voltage (V): 5
Please enter the number of resistors: 0
No resistors specified - short circuit!
```

In [11]:
```
# Example output
```

```
resistorCalculator()
```

```
Welcome to the resistor voltage calculator.

Please enter the supply voltage (V): 10
Please enter the number of resistors: -12
Invalid number of resistors - no calculation performed.
```

Next, we need to add the logic for reprompting. To do this, let's use a `while` loop.

- A while loop executes the code in its block as long as the specified condition is true.
- This loop should go in the getResistances function, since it is that function's job to collect resistance values from the user.
- We want to loop *while* the input is invalid, and then exit the loop when it is valid.
- Again, update the docstring!

In [13]:
```
def getResistances(nRes):
    '''
    Given the number of resistors in a series circuit, prompts the user for the
    resistance in ohms of each resistor. Returns the resistances (as a list of
    floats), as well as the total resistance in the circuit (as a float)

    If the user enters a resistance less than or equal to zero for any
    resistor, they are prompted again until they enter a positive value.
    '''

    resOhms = [None] * nRes
    totalRes = 0
    for i in range(nRes):
        promptStr = 'Please enter the resistance (ohms) for resistor #' + str(i+1) + ':
        currOhms = float(input(promptStr))

        while currOhms <= 0: # While input resistance is invalid
            currOhms = float(input(promptStr)) # Reprompt the user

        # Add to list and total
        resOhms[i] = currOhms
        totalRes += currOhms

    return resOhms, totalRes
```

In [14]:
```
# Example output
resistorCalculator()
```

```
Welcome to the resistor voltage calculator.

Please enter the supply voltage (V): 10
Please enter the number of resistors: 2
Please enter the resistance (ohms) for resistor #1: -1
Please enter the resistance (ohms) for resistor #1: -1
Please enter the resistance (ohms) for resistor #1: -1
Please enter the resistance (ohms) for resistor #1: 0
Please enter the resistance (ohms) for resistor #1: -1000
Please enter the resistance (ohms) for resistor #1: 42.5
Please enter the resistance (ohms) for resistor #2: 55

Supply voltage (V): 10.0
Current (A): 0.10256410256410256
Resistances (ohms): [42.5, 55.0]
Voltage across each resistor (V): [4.358974358974359, 5.6410256410256405]
```