**ITECH1400 Assignment**

Aaron Medina

30422160

**PSEUDO CODES**

**TASK01 collatzSequence**

1. Start the function called **collatzSequence** and give it a number **n** as input.
2. Check if the number **n** is equal to 1.
3. If **n** is equal to 1, return a list that only contains 1.   
   This is because the sequence ends at 1.
4. If **n** is not equal to 1, check if it is an even number.
5. If **n** is even, divide **n** by 2 and use the result to call the **collatzSequence** function again. Then, add the original value of **n** to the beginning of the list that the function returns.
6. If **n** is odd, multiply it by 3 and add 1 to get a new number. Use this new number to call the **collatzSequence** function again. Then, add the original value of **n** to the beginning of the list that the function returns.
7. Keep repeating steps E or F until the function returns the list that ends with the number 1. H. End the function.

**TASK02 maxLengths**

1. Start the function called **maxLengths** and give it a number **m** as input.
2. Initialise two variables, **longest\_seq\_num** and **longest\_seq\_len**, both with the value of 0.
3. Loop through the numbers from **m** down to 1.
4. For each number in the loop, get its Collatz sequence using the **collatzSequence** function.
5. Find the length of the Collatz sequence obtained in step D.
6. Check if the length obtained in step E is longer than the current longest sequence length stored in the **longest\_seq\_len** variable.
7. If the length obtained in step E is longer than the current longest sequence length, update **longest\_seq\_len** to the length obtained in step E, and update **longest\_seq\_num** to the current number in the loop.
8. Repeat steps D-G for all numbers in the loop.
9. Return a list containing the longest sequence length and the number that produced it in that order.

**TASK03 maxValue**

1. Start the function called **maxValue** and give it a number **m** as input.
2. Initialise two variables, **max\_seq\_num** and **max\_seq\_val**, both with the value of 0.
3. Loop through the numbers from 1 to **m**.
4. For each number in the loop, get its Collatz sequence using the **collatzSequence** function.
5. Find the max value of the Collatz sequence obtained in step D.
6. Check if the max value obtained in step E is greater than the current max sequence value stored in the **max\_seq\_val** variable.
7. If the value obtained in step E is greater than the current max sequence value, update **max\_seq\_val** to the max value obtained in step E, and update **max\_seq\_num** to the current number in the loop.
8. Repeat steps D-G for all numbers in the loop.
9. Return a list containing the max sequence value and the number that produced it in that order.

**ACKNOWLEDGEMENT/TOOLS**

* + Python Docs
  + Lecture Slides
  + VisualStudio Code
  + Github CoPilot