# Assignment 04 – Sum of Fibonacci Numbers (loops)

THE PROBLEM:

Write a program that computes the sum of the Fibonacci numbers between two inputs, X and Y.

What's a Fibonacci number? A Fibonacci number is any number that appears in a Fibonacci sequence.

The first six Fibonacci numbers are: 0, 1, 1, 2, 3, 5

* The Fibonacci sequence begins with 0 and 1.
* Then, after that, the next Fibonacci number is the sum of the last two Fibonacci numbers. So, 0 + 1 = 1. *One* is the third Fibonacci number.
* Then, after that, the next Fibonacci number is the sum of the last two Fibonacci numbers. So, 1 + 1 = 2. *Two* is the fourth Fibonacci number.
* Then, after that, the next Fibonacci Number is the sum of the last two Fibonacci numbers. So, 1 + 2 = 3. *Three* is the fifth Fibonacci number.
* Then, after that, the next Fibonacci Number is the sum of the last two Fibonacci numbers. So, 2 + 3 = 5. *Five* is the sixth Fibonacci number.

The first ten Fibonacci numbers are: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34. See the pattern?

We can express that mathematically like this:



Where:



ERROR TRAPPING:

* As in previous assignments, if a user provides any input that is not a natural number, then the function should return the error message: "Bad data. Try again.”
* You must reuse your isNaturalNumber() function from a prior assignment to help you with your error trapping.
* Also, if X > Y then report “Bad data. Try again.”

**FUN FACT:** Fibonacci numbers appear in many real-world applications, including:

* Biology & Nature – Found in plant growth, flower petal arrangements, and DNA structures.
* Finance – Used in Fibonacci retracement, a stock market analysis tool.
* Computer Science – Applied in dynamic programming, sorting algorithms, and recursion problems.
* Art & Design – Related to the golden ratio, used in photography and architecture.

## Instructions

1. Retrieve assignment04.zip provided by your instructor.
2. Extract the code into your mis2402workspace and open the fibonacci.html file in Visual Studio Code.
3. Bring in the isNaturalNumber() function from a previous assignment. Put it in portion of your code designated for “supporting functions”.
4. Use the web developer tools in Chrome to test your isNaturalNumber() function.
5. Now work on the sumOfFibonacciNumbers function.
	1. Start your function with error trapping. You need to test the inputs numx and numy. If either input is bad, you need to return “Bad data. Try again.”
	2. Think ahead. In addition to numx, and numy, you will need to create four variables. You will need variables to hold:
* Fn-1
* Fn
* The next Fibonacci number (that is, the sum of Fn-1 and Fn )
* The grand total
	1. After your error trapping, write a loop that goes from one to 100.
	2. As noted in the last step, you are expected to solve this problem using a loop. Other approaches will not be accepted.
	3. In each iteration of the loop, calculate the next Fibonacci number.
	4. If the next Fibonacci number is greater than your ***upper*** bound, ***return*** the grand total.
	5. If the next Fibonacci number is greater than or equal to your ***lower*** bound, add it to the grand total.
	6. Before the iteration completes, be sure to assign Fn-1 and Fn to their new, correct values.
	7. Test your work.
		1. The sum of the Fibonacci numbers between 4 and 9 is 13.
		2. The sum of the Fibonacci numbers between 4 and 25 is 47.
		3. The sum of the Fibonacci numbers between 1 and 3 is 6. (Notice that here the “1” is only counted once, even though it appears twice in the sequence. The Fibonacci numbers between 1 and 3 are 1, 2, and 3. And, you guessed it, 1 + 2 + 3 = 6.)
	8. Once you are satisfied that small tests are being passed successfully, then alter your loop so that it runs infinitely. It no longer stops at 100 like you did back on step c. Typically, an infinite loop is a bad thing, but in this case, it is OK because your loop does quit (because of what you did back on step f).

If you ignore these instructions and just try to write the code perfectly the first time, the odds of you getting frustrated are high. You will probably lock up your browser, getting it stuck in an infinite loop. The right way to tackle this problem is to start small, start simple, test as you go and add complexity as you go.

1. Upload your work to the class server and turn in the URL to your work using the corresponding canvas assignment.

How will this assignment be graded?

This assignment will be evaluated by an automated process.

* If your work generates **all output** correctly, you will get a score of 100%.
* If your work generates **almost all output** correctly (**only one** bad output), you will get a score of 80%
* If your work generates **some output** correctly (some right output, some wrong output), you will get a score of 60%
* If your work generates **only one output** correctly, you will get a score of 40%
* If your work does not generate any correct output, you will get a score of **zero**.