1. Log in to your Sage/Cocalc account.

   (a) Start the Chrome browser.
   (b) Go to http://cocalc.com and sign in.
   (c) You should see an existing Project for our class. Click on that.
   (d) Click “New”, call it c15, then click “Sage Worksheet”.

**Timing**

For large programs or calculations that are at the edge of what’s possible. It is crucial to optimize and test the speed of your code. One simple first step is simply to **time** your program using Sage’s built-in `timeit()` function.

It is often intuitive to define a function recursively, but usually the same function can be defined without recursion.

2. The Fibonacci sequence $F_n$ is defined as follows $F_0 = 0$, $F_1 = 1$ and $F_n = F_{n-1} + F_{n-2}$ for $n > 1$. Here is a recursive function `fibonacci(n)` that computes the $n^{th}$ Fibonacci number.

```python
def fibonacci(n):
    if n==0:
        return 0
    elif n==1:
        return 1
    else:
        return fibonacci(n-1)+fibonacci(n-2)
```

Evaluate and write down what you get for `timeit("fibonacci(10)")`, `timeit("fibonacci(20)")`, and `timeit("fibonacci(25)")`.

3. Define a non-recursive (iterative) function `fibonacci2(n)` that computes the $n^{th}$ Fibonacci number.
4. Evaluate and write down what you get for `timeit("fibonacci2(10)")`, `timeit("fibonacci2(20)")`, and `timeit("fibonacci2(25)")`.

The recursive `fibonacci(n)` function we defined takes a very long time to respond for \( n = 30 \) and may never respond for \( n = 40 \). Now try `fibonacci2(40)` and `fibonacci2(400)`. Why does the iterative function work while the recursive function does not?

5. Solve the equation \( \frac{a+b}{a} = \frac{a}{b} \), for \( a \) and \( b \). Find \( \frac{a}{b} \). Get a 10-digit approximation for this quantity (this is the Golden Ratio).

6. Define a function `fib_ratio(n)` which returns the ratio of the \((n + 1)^{th}\) Fibonacci number to the \(n^{th}\). Find `fib_ratio(10)` and `fib_ratio(100)`. Compare this answer to your previous answer. What can you conjecture?