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 c18, then click “Sage Worksheet”.

**Debugging**

2. Type in the function `leftpoint_riemann(f,a,b,n)` which computes the leftpoint Riemann sums for \(n\) equal intervals.

   ```python
   def leftpoint_riemann(f,a,b,n):
       area=0
       Delta=(b-a)/n
       for i in [0..n]:
           leftpoint=a+i*Delta
           newarea=f(leftpoint)*Delta
           area=area+newarea
       return area
   ```

3. Type in the function `rightpoint_riemann(f,a,b,n)` which computes the rightpoint Riemann sums for \(n\) equal intervals.

   ```python
   def rightpoint_riemann(f,a,b,n):
       area=0
       Delta=(b-a)/n
       for i in [0..n]:
           rightpoint=b-i*Delta
           newarea=f(rightpoint)*Delta
           area=area+newarea
       return area
   ```

4. Find the value of `leftpoint_riemann(f,a,b,n)` for \(f(x)=x^2\) on \([0,3]\) with \(n=2, n=5, n=10\) and \(n=100\). Here you are making the intervals smaller and smaller, giving a better and better approximation.

5. Find the values of `rightpoint_riemann(f,a,b,n)` for \(f(x)=x^2\) on \([0,3]\) with \(n=2, n=5, n=10\) and \(n=100\). Compare with your results for `leftpoint_riemann(f,a,b,n)`. What do you notice???
6. Now let's add print statements to get an idea of what these programs are doing.

```python
def leftpoint_riemann(f, a, b, n):
    area = 0
    Delta = (b - a) / n
    for i in range(n):
        leftpoint = a + i * Delta
        newarea = f(leftpoint) * Delta
        area = area + newarea
        print "i, leftpoint newarea are {},{},{}".format(i, leftpoint, newarea)
    return area

def rightpoint_riemann(f, a, b, n):
    area = 0
    Delta = (b - a) / n
    for i in range(n):
        rightpoint = b - i * Delta
        newarea = f(rightpoint) * Delta
        area = area + newarea
        print "i, rightpoint newarea are {},{},{}".format(i, rightpoint, newarea)
    return area
```

Can you figure out why these two programs give the same answers? If not, can you think of any more print statements to add that would be helpful?

7. After you figure out why these programs give the same answer, adjust your code so we are calculating what we want. Then recalculate the values of `leftpoint_riemann()` and `rightpoint_riemann()` for \( f(x) = x^2 \) on \([0,3]\) with \( n = 2, n = 5, n = 10 \) and \( n = 100 \).

8. **Coin Flips.** Here is a function that counts and returns the number of heads you get after flipping a coin \( n \) times. (It uses the `coin_flips(n)` we defined last class.)

```python
def number_of_heads(n):
    flip_results = coin_flips(n)
    heads = 0
    for i in flip_results:
        if i == "H":
            heads = heads + 1
    return heads
```

Evaluate `number_of_heads(100)` a few times. You should get different results!

9. Write a function `flip_data(n)` which prints the numbers of both heads and tails you get after flipping a coin \( n \) times.