1. Log in to your Sage Cloud account.
   
   (a) Start the Chrome browser.
   (b) Go to http://cloud.sagemath.com and sign in.
   (c) You should see an existing Project for our class. Click on that.
   (d) Click “New”, call it c19, then click “Sage Worksheet”.

Here are the Main Questions:

- When you flip a coin 100 times would you expect to see 6 heads or tails in a row at some point? We can investigate this question too by simulating coin flips and repeating our \textit{experiment} a number of times.
- If you flip a coin 100 times, you would expect about 50 heads. It’s possible that you could get 100 heads. But this would be rare. How rare? We can simulate flipping a coin a hundred times, write down how many heads we got, and then repeating this experiment. This will give us a \textit{distribution} of various possible outcomes.

2. Use \texttt{random()} to define a function \texttt{coin\_flip()} which randomly returns the string “H” (for heads) half the time and returns the string “T” (for tails) half the time. Check that it works.

3. Use your \texttt{coin\_flip()} to define a function \texttt{coin\_flips(n)} which returns a list of \texttt{n} random H’s or T’s (representing the result of \texttt{n} coin flips).

```python
def coin_flips(n):
    flip_results = []
    for i in [1..n]:
        flip_results.append(coin_flip())
    return flip_results
```

Check that it works.
4. Here is a function that counts and returns the number of heads you get after flipping a coin \( n \) times.

```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!

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

6. When you flip a coin a number of times you will get runs of one heads, two heads, three heads, etc, before getting a tails (that ends the run). Here is a function `longest_run_of_heads(n)` that returns the length of a longest run of heads after flipping a coin \( n \) times.

```python
def longest_run_of_heads(n):
    flip_results = coin_flips(n)
    count = 0
    longest = 0
    for i in range(n):
        if flip_results[i] == "H":
            count = count + 1
            if count > longest:
                longest = count
        else:
            count = 0
    return longest
```

Try `longest_run_of_heads(100)` a few times. The results should vary.

7. Add a `print` statement to `longest_run_of_heads(n)` to help you check that this code is doing what you expect. Then rerun the program a few times. When you are sure it is working properly, remove the print statement.