Space complexity

- So far, have only talked about time complexity
- Space complexity: amount of space that program uses
- Example with $O(n^2)$ space complexity (and $O(n^2)$ runtime):

```
A = [ ]
for i in range(n):
    for j in range(n):
        A.append(1)
```

Desirable runtime and space complexities

- Want program to finish while we are alive
- Find the 100th Fibonacci number (assume each operation is 1 microsecond):
  - Naive recursive program is $O(2^n)$, so will run in time proportional to 2^{100}, so around 10^{16} years (Earth has only existed for around 4.5×10^9 years)
  - Iterative program is O(n), so will run in time proportional to 100, so around 100 microseconds
- Typically want O(n); if possible, want even lower, like O(log(n)).
- At worst, $O(n^2)$ may be acceptable. Exponential time or worse is typically bad.