6.100L Recitation 8 – Nov 4, 2022

Reminders:

- MQ8 on Monday 11/7
- PSET 4 halfway hand in due at 9PM on Wednesday 11/9

Lectures 15 & 16 - Recursion

What is Recursion?

- Programming method which is often used instead of iteration.
- Algorithmically a way to design solutions by “divide-and-conquer” (i.e. reduce a problem to a simpler version of the same problem).
- Semantically - programming technique where a function calls itself (but not infinitely).
- If you recognize solving the same problem repeatedly, it may be easier to use recursion.

Signs we should use recursion

- Recognize that we have a problem we are solving many times
- Number of iterations required is unknown
- We want a “nicer” looking solution to an iterative method.

Examples of tasks which can use recursion:

- Fibonacci
- Towers of Hanoi
- Sequences (e.g. Geometric, Arithmetic etc…)
- Multiplying/Adding/subtracting series

General Recursive structure

Goal: to keep reducing to a simpler problem until we know how to solve the simpler problem.

A general recursive program is comprised of two parts:

1. Base Case
   - When you reach a simple case that can be solved.
   - Must always reach this case otherwise you could have infinite recursion.

```
e.g.
if a == 1:
    return 1
elif a == 0:
    return 0
```

2. Recursive Case
   - Think: how can we reduce the problem at this step?
   - Typically call our function with inputs that make our problem simple / smaller

```
e.g.
recurse(a-1) # -2 is our modification to “simplify” the
problem.
```