Section 2 Overview

Agenda

- Inductive Proof
  - Inductive Proof Steps
  - Recurrence Relation
- Recursive Python
- Optional Problem
  - Pascal’s Triangle

Inductive Proof

Inductive Proof Steps

Steps of Induction:
1. Problem Statement (kind of optional but nice)
2. Base Case
3. Inductive Hypothesis
4. Inductive Leap of Faith (Inductive Step)
5. Conclusion

Inductive Proof

1 Money

1.1 Problem

Prove that \( \sum_{i=1}^{n} [(i) \times (i + 1)] = \frac{(n)(n+1)(n+2)}{3} \) where \( n \geq 1 \) using a beautiful inductive proof.
1.2 Solution

Base Case:

Let $n = 1$.

$1 \times 2 = 2$

$\frac{1 \times 2 \times 3}{3} = 2$

Inductive Hypothesis:

Assume, for $n=k$, that $\sum_{i=1}^{k} [(i) \times (i + 1)] = \frac{(k)(k + 1)(k + 2)}{3}$

Inductive Step:

By the Inductive Hypothesis,

$\sum_{i=1}^{k} [(i) \times (i + 1)] = \frac{(k)(k + 1)(k + 2)}{3}$

Add $((k+1) \times (k+2))$ to both sides of the equation.

$\sum_{i=1}^{k} [(i) \times (i + 1)] + ((k+1) \times (k+2)) = \frac{(k)(k + 1)(k + 2)}{3} + ((k+1) \times (k+2))$

Simplify the left side of the equation.

$\sum_{i=1}^{k+1} [(i) \times (i + 1)] = \frac{(k)(k + 1)(k + 2)}{3} + ((k+1) \times (k+2))$

Simplify the right side of the equation.

$\sum_{i=1}^{k+1} [(i) \times (i + 1)] = \frac{(k)(k + 1)(k + 2)}{3} + \frac{3((k+1)(k + 2))}{3}$

$\sum_{i=1}^{k+1} [(i) \times (i + 1)] = \frac{(k + 1)(k + 2)(k + 3)}{3}$

$\sum_{i=1}^{k+1} [(i) \times (i + 1)] = \frac{(k + 1)((k + 1)(k + 3) + (k + 1)(k + 2))}{3}$
Recurrence Relation Solution

RECURRENCE

The game of Hanoi Tower is to play with a set of disks of graduated size with holes in their centers and a playing board having three spokes for holding the disks.

The object of the game is to transfer all the disks from spoke A to spoke C by moving one disk at a time without placing a larger disk on top of a smaller one. The minimal number of moves required to solve the problem with $n$ disks can be modeled by the following recurrence relation:

\[ a_n = 2a_{n-1} + 1, \quad n \geq 1 \]

\[ a_1 = 1 \]

Plug and Chug Solution:

\[ a_1 = 1 \]
\[ a_2 = 2a_1 + 1 = 2\times1 + 1 = 3 \]
\[ a_3 = 2a_2 + 1 = 2\times3 + 1 = 7 \]
\[ a_4 = 2a_3 + 1 = 2\times7 + 1 = 15 \]
\[ a_n = 2a_{n-1} + 1 = 2^n - 1, \quad n \geq 1 \]

Recursive Python Example

**Sum All**

Given a positive integer $x$, recursively find the sum of all numbers from 1 to $x$.

```python
def sumAll(x):
    '''Function that recursively sums positive integers less than or equal to x'''
    if x==0:
        return 0
    else:
        return sumAll(x-1)+x
```
Optional Problems

Pascal’s Triangle

Write an algorithm that takes an integer n and returns the nth row of Pascal’s Triangle (see here for an explanation: https://en.wikipedia.org/wiki/Pascal%27s_triangle)

```python
def pascal(n):
    """
    input: an int, n, the line of pascal's triangle that you want to return
    output: a list representing the nth line of pascal's triangle. If n is less than 2, then the function should return [1]
    """
    if n < 2:
        return [1]

    prev_line = pascal(n-1)
    line = [prev_line[i]+prev_line[i+1] for i in range(len(prev_line)-1)]
    line.insert(0,1)
    line.append(1)

    return line
```