

MC215: MATHEMATICAL REASONING  
AND DISCRETE STRUCTURES

- **Friday, 11/21/08**
  - From last time:
    - **Towers of Hanoi**
    - **Euclidean Algorithm**
  - Today:
    - **Recurrence Relations (also called Difference Equations)**
- **READING:**  
7.1-7.2
- **EXERCISES:**
  - pp. 342-345: 4-12
  - pp. 356-357: 11-12

Friday, 11/21/08, Slide #1  
Printed 11/21/2008 10:19 AM

---

---

---

---

---

---

---

---

Computing the Number of Moves in  
Towers of Hanoi Puzzle

- Let  $m_k$  = # of moves required to solve the Towers of Hanoi Puzzle. Using our algorithm:

$$m_k = \begin{cases} 0 & k = 0 \\ 2m_{k-1} + 1 & k > 0 \end{cases}$$

- **Theorem.** No algorithm to solve the Towers of Hanoi puzzle uses fewer moves than ours.
  - **Proof?**
- **Theorem.** The Towers of Hanoi puzzle with  $k$  disks requires  $2^k - 1$  moves.
  - **Proof by induction on  $k$ .**

Friday, 11/21/08, Slide #2  
Printed 11/21/2008 10:19 AM

---

---

---

---

---

---

---

---

Recurrence Relations (Warning:  
Different meaning for “relation!”)

- A **recurrence relation** (or **difference equation**) is a recursive definition for a **sequence of numbers**,  $a_0, a_1, a_2, \dots$ , i.e.,
  - A finite number of values are **explicitly defined**. These are called the **initial conditions**.
  - The other values of  $a_n$  are defined in terms of **earlier values**,  $a_0, a_1, \dots, a_{n-1}$ .
- **Example:** The Fibonacci Sequence is defined by the recurrence relation:

$$f_n = \begin{cases} 1 & n = 1, 2 \\ f_{n-1} + f_{n-2} & n \geq 3 \end{cases}$$

Friday, 11/21/08, Slide #3  
Printed 11/21/2008 10:19 AM

---

---

---

---

---

---

---

---

## Solving recurrence relations

- A **solution to a recurrence relation** for a sequence  $a_1, a_2, \dots, a_n$  is a "closed-form," non-recursive formula for  $a_n$  as a function of  $n$ .
  - **Closed-form** means the *length* of the formula doesn't change as  $n$  gets larger
    - $m_k = 2^k - 1$  is a closed-form formula for  $m_k$  in terms of  $k$
    - $n! = n \cdot (n-1) \dots 2 \cdot 1$  is *not* a closed-form formula for  $n!$  in terms of  $n$ .

Friday, 11/21/08, Slide #4  
Printed 11/21/2008 10:19 AM

---

---

---

---

---

---

---

---

## Example

- **Example:** 
$$a_n = \begin{cases} 4 & n=0 \\ a_{n-1} + 5 & n \geq 1 \end{cases}$$
  - Write down the terms  $a_0, \dots, a_4$
  - **Solve** this recurrence relation, by giving a closed-form formula for  $a_n$  in terms of  $n$ 
    - **Hint:** How many 5's are added to make  $a_n$ ?
  - **Prove** your solution is correct, using mathematical induction

Friday, 11/21/08, Slide #5  
Printed 11/21/2008 10:19 AM

---

---

---

---

---

---

---

---

## Solution by iteration

- Given  $a_n = a_{n-1} + 5$ , we can repeatedly apply this formula, until we get to  $a_0 = 4$ .

$$\begin{aligned} a_n &= a_{n-1} + 5 = a_{n-2} + 2 \cdot 5 = a_{n-3} + 3 \cdot 5 \\ &= a_{n-4} + 4 \cdot 5 = \dots = a_0 + n \cdot 5 = 4 + 5n \end{aligned}$$

- This method is called **iteration**, and it depends on you being able to discern a pattern that emerges as you iterate.

Friday, 11/21/08, Slide #6  
Printed 11/21/2008 10:19 AM

---

---

---

---

---

---

---

---

## Example: Compound Interest

- A person deposits **D dollars** at a simple yearly **interest rate R** (given as a decimal, e.g., 0.05 for 5%). What is the amount of money,  $M_n$ , **accumulated after  $n$  years**?
  - Write a **recurrence relation for  $M_n$** , with initial condition given for  $n = 0$ .
  - **Solve** this recurrence relation by iteration.
  - **Prove** your answer is correct by mathematical induction on  $n$ .

Friday, 11/21/08, Slide #7  
Printed 11/21/2008 10:19 AM

---

---

---

---

---

---

---

---