# CS 310: THEORETICAL COMPUTER SCIENCE

Pacific University, Fall 2018

Chris Lane, Price 211, lanec@pacificu.edu
Office Hours: M, F: 10:15-11:15 am, T: 12:00-1:00 pm
Website: zeus.cs.pacificu.edu/lanec



## Syllabus

*Introduction to the Theory of Computation* by Michael Sipser, (**Third Edition**)

- I will assign problems out of this book
- I expect you to do the readings

Policies:

- Class starts promptly at 1:00pm
- Assignments are due at the **beginning of class**. Late assignments will not be accepted.
- Programs that do not produce lose 70% (JFLAP)
- The cheating policy is defined in the Pacific Catalog
- Silence all electronic devices
- Participation can raise/lower your grade

# Syllabus

**Grade Distribution**

| | |
|---|---|
| Homework | 20% |
| Unannounced Quizzes | 5% |
| Exam 1 | 25% |
| Exam 2 | 25% |
| Final | 25% |

**Percent Breakdown**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | 92-100 | A | | 90-92 | A- |
| 88-90 | B+ | | 82-88 | B | | 80-82 | B- |
| 78-80 | C+ | | 72-78 | C | | 70-72 | C- |
| 68-70 | D+ | | 60-68 | D | | | |
| 0-60 | F | | | | | | |

- Tentative Exam Dates:
  - Midterm 1: Monday, October 1st, 2018
  - Midterm 2: Friday, November 9th, 2018
- Non-Tentative Final Exam Date and Time:
  - Tuesday, December 11, 8:30-11:00am



# Today

- Overview of class
- Mathematical Notation
- Proof by Induction

# Who are we?

- Is Computer Science a science
  - Is it a natural science?
- What do we study?



# Overview

- What are the fundamental capabilities and limitations of computers?
- How does theory related to programming?
- Complexity Theory
- Computability Theory
- Automata Theory

# Mathematical Notation (Chapter 0)

- Basic notations we will use in this class
  - Page 16 of your book has a partial list (no symbols!)
- Set
- Subset
- Proper Subset



# Sets

- Shorthand for describing a set: { n | rule about n}

# Set Operations

What can we do with sets?

- Union
- Intersection
- Complement
- Venn Diagrams, DeMorgan's Laws



# Sets

- Power Set:

  Ex: { 0, 1 }

- Cartesian Product (Cross Product)

  Ex: { 0, 1 } X { a, b }

# Sequences and Tuples

- Sequence

- Tuple, K-tuple



# Functions

- Object that takes input, produces output

  f (a) = b

- Domain and Range

  $f : D \rightarrow R$

- Onto

# Functions

- $f : A_1 \times A_2 \times \ldots \times A_k \rightarrow R$

  $(a_1, a_2, \ldots, a_k)$
  - k-ary
  - arity
  - unary (k=1), binary (k=2)
- Notation
  - Infix notation: a + b
  - Prefix notation: add(a,b)



# Relations

- Predicate (property)

  $f : D \rightarrow$ {TRUE, FALSE}
- Relation

  $f : A_1 \times A_2 \times \ldots \times A_n \rightarrow$ {TRUE, FALSE}
- Notation

  Table

  Set

# Equivalence Relations

- binary relation
- shows that two objects are equal

- must satisfy 3 conditions:

1. R is ***reflexive*** if for every x, xRx;
2. R is ***symmetric*** if for every x and y, xRy if and only if yRx;
3. R is ***transitive*** if for every x, y, and z, xRy and yRz implies xRz



# Proof by Contradiction

- Assume the concept is false
- Show this leads to a false consequence
  - tricky to pick exactly the false consequence

- Prove $\sqrt{2}$ is irrational
  - Assume it is rational: $\sqrt{2} = m/n$

# Proof by Induction

- Basis

  Prove P(1) is true
- Induction Step

  Prove that for each i≥1, if P(i) is true,

  then so is P(i+1); *assume P(i) is true*
- Basis + Induction Step

  P(1) is true, i = 1

  P( (i+1) ) is true

  P( (i+1) +1 ) is true …



# Proof by Induction

- Prove: $1 + 2 + \ldots + n = n(n+1) / 2$ for $n \geq 1$

Basis:

Induction:

# Proof by Induction

- $n^2 >= 3n$, n>=3

- $F_n$ is the *n*th Fibonacci number, n>=1.
  - $F_n = F_{n-1} + F_{n-2}$
  - $F_1=1$ ; $F_2=1$
- $F_{3n}$ is even, n>=1
- $F_{4n}$ is evenly divisible by 3, n>=1

# Graphs

- Graph

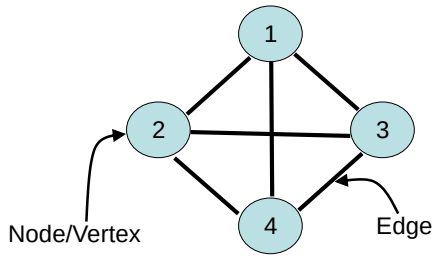
  Collection of nodes and edges

  G = (V,E)

  E =

  undirected

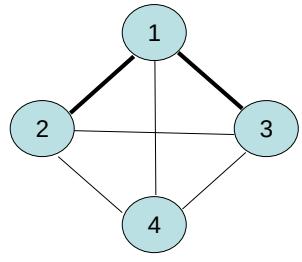
- Degree of a node





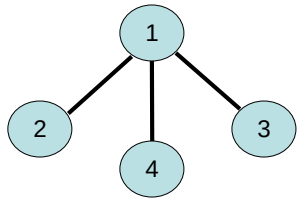

# Graphs

- Subgraph
- Path
- Simple Path
- Connected graph
- Cycle



# Trees

- Connected graph with no simple cycles




- Leaves

- Root



# Directed Graph

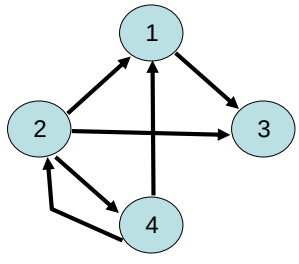
- Arrows denote which way an edge goes

Outdegree/indegree:

G=(V,E)
V=

E=

Directed path:




Strongly connected: