Announcements about PSet 7.

1. Q1 early turn-in option:
   Turn in before Tues 4pm.
   \[ \Rightarrow \text{graded before end of Sp Brk.} \]

2. Q2 has a 4820 and a 5820 version.

This will be last homework before Prelim 2.

Prelim 2 is not cumulative. Covers Ch 5, 7, 8.
(disk-cong, flow, NP-Complete problems)

**Definitions.**

1. **NP** is the class of decision problems for which there exists a polynomial-time algorithm to verify a “yes” answer, given a suitable hint.

   If \( T \) is a problem, we say \( T \in \text{NP} \) if there’s an algorithm \( V \) with two inputs:
   
   \[ V(x, y) \]

   denotes an instance of \( T \)

   \( y \) denotes a “hint” or “candidate solution”

   \( x \) runs in poly time in \( |x| + |y| \).

   \( y \) runs in poly time in \( |x| + |y| \).

   \( |x| \) length of \( x \)

   \( |y| \) length of \( y \)

   \( \text{poly} \) is polynomial

   \( D(\text{poly}(|x|)) \) is a polynomial function of \( |x| \)

   \( V(x, y) = \text{yes} \)

   if and only if

   \( \exists y \text{ s.t. } |y| \leq D(\text{poly}(|x|)) \) and \( V(x, y) = \text{yes} \).
Example: 3SAT.

\[ X = \text{specification of variables & clauses} \]
\[ Y = \text{specification of truth assignment} \]
\[ V(x, y) = \text{algorithm that checks } y \text{ satisfies every clause specified in } x. \]

1. \( P \subseteq \text{NP} \) because \( V(x, y) \) can ignore \( y \) and solve \( x \) in poly time.
2. \( P \supseteq \text{NP} \): most believe \( P \neq \text{NP} \).

2. \( T \) is \text{NP-hard} if any of the following equivalent statements hold:
   a. Every problem \( T' \in \text{NP} \) satisfies \( T' \leq_p T \).
   b. There exists an \text{NP-hard} \( T' \) st. \( T' \leq_p T \).
   c. 3SAT \( \leq_p T \).

3. \( T \) is \text{NP-complete} if \( T \in \text{NP} \) and is \text{NP-hard}.
   You can verify a solution of \( T \) in poly-time but can't find a solution unless \( P = \text{NP} \).

"Prove some problem ABC is \text{NP-Complete}.

1. Present a polytime verifier. (Show ABC \( \in \text{NP} \)).
2. Present a reduction from some other \text{NP-hard} problem, XYZ, to ABC.
3. Show reduction runs in poly time.
4. Reduction, applied to "yes" instance of XYZ, yields "yes" instance of ABC."
HAMILTONIAN CYCLE: input is a directed graph. Question: does the graph contain a simple cycle that visits all vertices?

(cycle with no repeated vertices)

1) Belongs to NP because a verifier given the graph, and a list of vertices in the cycle, checks that each vertex is in the cycle once and only once, and each edge of the cycle belongs to the graph.

2) Reduce from something NP-Hard

Textbook: 3SAT ≤p HAM CYCLE.
(Read it! It’s instructive)

Today: VERTEX COVER ≤p HAM CYCLE.

Given undirected G, k ∈ N.
Can we find k vertices in G such that each edge has at least one endpoint among the k vertices?
Will represent $G$ as a HAM CYCLE input with 3 types of gadgets.

**EDGE GADGET**

$u - v$

**VERTEX GADGET**

$u \rightarrow v_1 \rightarrow v_2 \rightarrow v_3$

**COUNTER GADGET**

To each vertex from each vertex
\[ k = 3 \]