- Welcome!

- OH tomorrow (Fri): 10-11am
  - Zoom link will be available on Canvas & on Piatta

- Hws 0 and 1 will be released today.

- Course policies

Stable Matching

Input: - n people, P

   - n pets, T

   - each person in P has a ranking of all pets in T (no ties)

   - each pet in T has a ranking of all people in P (no ties)

Objective: To design an algorithm that
outputs a **stable matching**.

Matching in a graph $G$ is a set of edges, no two of which are adjacent.

**Example:**

$$(t_2, t_1, t_3) \in P_1 \quad (t_1, P_2, P_3)$$

$$(t_3, t_2, t_1) \in P_2 \quad (P_2, t_1, P_3)$$

$$(t_3, t_1, t_2) \in P_3 \quad (P_1, P_2, P_3)$$

**Defn:** A matching $M$ in the given instance is stable iff there is no $(p, t)$ pair, where $p \in P$ and $t \in T$ s.t. $p$ and $t$ prefer each other over their existing (current) partner.

Q: Does every input have a stable matching?

Ans: Yes?
Q: Can an input have more than one stable matching?

Ans: Yes?

Alg:
1. Fix an arrangement \( I \) of all people in \( P \) (make all people stand in a row)
2. for each permutation \( f \) of the pets in \( T \) do

match the 1\textsuperscript{st} pet with the 1\textsuperscript{st} perm

... 2\textsuperscript{nd} ... 2\textsuperscript{nd} ...

... 2\textsuperscript{nd} ...

\( n \textsuperscript{th} \) pet \( n \textsuperscript{th} \) perm.

check for any instabilities

if no instability then

\( O/P \) the matching.

3. \( O/P \) no stable matching.

\[ p_1, p_2, \ldots, p_n, \]

\[ t_1, t_2, \ldots, t_n \]

\( t_1 < t_2 < t_3 \ldots t_n \)

The above algorithm takes \( n! \) iterations in
the worst case. For \( n = 30 \), this alg. takes \( \geq 10^{25} \) years to finish.

\[
\begin{align*}
  p_1 &\rightarrow t_1 \\
  p_2 &\rightarrow t_2 \\
  p_3 &\rightarrow t_3 (p_5, p_3) \\
  (t_4, t_5, \ldots) &\rightarrow \min \rightarrow t_4 \\
  (t_3, t_5, \ldots) &\rightarrow 0 t_5 \\
  p_5 &\rightarrow \ldots
\end{align*}
\]

Gale-Shapley

1. Initially all people & all pets are free.
2. While there is a free person \( p \) who has not yet proposed to all pets in \( T \) do
3. \[ t \leftarrow \text{highest ranked pet on p's list whom p has not yet proposed to.} \]

4. if \( t \) is free then

5. \((p,t)\) becomes a pair

6. else if \((p',t)\) exists then

7. if \( t \) prefers \( p \) over \( p' \) then

8. \((p,t)\) forms a pair

9. \( p' \) becomes free.

10. output all pairs.

Q: Can this alg. go into infinite loop?
A: No. At most \( n^2 \) iterations of the loop.

Lemma 1: Once a pet receives their first
propose they always remain "engaged" & as the algorithm progresses, their partners can only get better.

Lemma: GS alg. returns a perfect matching (matching in which every person & every pet is matched).

Proof Sketch: Assume for contradiction that the alg. has ended & there is a free person, say p.

- p must have proposed to all pets.
- all pets are paired.
- By PHP, some person must be paired with $\geq 2$ pets, which we know is not possible in our alg.
Similar proof for when a partner is free.

Lemma: GS alg. ops a stable matching.

Proof: Assume for contradiction that there is an instability in the op GS alg.

Let (p, t') be the unstable pair.

\((\ldots, t', \ldots, t)\) must be t

\(p, \ldots, p', \ldots, t'\) (\(\ldots, p, \ldots, p', \ldots\))

- p must have proposed to \(t'\) before proposing to \(t\).

- \(t'\) is paired with \(p'\) who is lower.
than \( p \) on \( t \)'s preference list, contradicting Lemma 1.

**Defn:**

\[
\text{valid}(p) = \{ t \in T \mid \text{there is a stable matching containing } (p, t) \text{ as a pair} \}
\]

\[
\text{Best}(p) = t \iff
\begin{align*}
- & t \in \text{valid}(p) \\
- & \text{no } t' \text{ ranked higher than } t \text{ on } p \text{'s preference list belongs to } \text{valid}(p)
\end{align*}
\]

\[
(\ldots \left[ t \right], \ldots) \quad \text{and} \quad x < \ldots
\]
$S^* = \{ (p, \text{Best}(p)) \}$

**Then**: GS alg. always outputs $S^*$.

**Proof**: Assume for contradiction that during some execution $E$ of the alg, some people get rejected by their Best valid partners. Among these people, let $t$ have the honor of being the first person to be rejected by their Best $(p)$. Let $t = \text{Best}(p)$.

Why did $t$ reject $p$?

$p'$.

Since $t \in \text{valid}(p)$, by def'., then
is a stable matching $S$ that contains $(p, t)$ as a pair.

\[ \rightarrow p \quad \text{unmatch} \quad (\ldots, p', \ldots, p \ldots) \]

\[ S : \]

\[ \rightarrow p' \quad \text{unmatch} \quad t' \]

\[ (\ldots, t', \ldots, t \ldots) \quad (t' \in \text{valid}(p')) \]

\[ \rightarrow \quad \text{p' proposed} \quad \text{p' gets rejected} \]

\[ \rightarrow \quad \text{by t'} \quad \text{by t becaun of p'} \]

This is a contradiction since p is not the first person to be rejected by their valid partner.
Define: valid(t) = \{ p \in P \mid (p, t) \text{ is a pair} \}

\text{Worst}(t) = p \text{ s.t.}
- p \in \text{valid}(t)
- no p' ranked lower than p on t's list belongs to \text{valid}(t).

Thus: In the o/p of the GS alg, each pet t is paired with worst(t).