**Shake matching** - jobs, applicants, random process
- Add applicants to jobs if one is our their match
- Del. Shake matching - perfect matching
  - Del. Rare cases: not possible (fail)

- Theorem: There exists a shake matching whenever the job and applicant graph is bipartite.

**Algorithm**
1. Each applicant will choose their favorite job.
2. If a job has more than one applicant, apply at all the applicants, favorite first. (not optimal)
3. Applicants who are rejected can choose another job.
4. Repeat, until every job has at most one applicant.

**Proof**
1. Each job gets their most preferred applicant.
2. Each applicant gets a job.
3. No more couples.

**Converse:** This yields a shake matching.

**Proof:**
1. Each applicant has a list of jobs.
2. Each job has a list of applicants.

- Let's assume that someone is unmatched.
- Then, it must be an applicant.
- If it is an applicant, it is unmatched.
- For each job, if there is an applicant, then they could be matched.

- For each job, if there is an applicant, then they could be matched.
- If there is no applicant, then there is no match.

**Shake Matching**
- Del. Rare cases: not possible if
- Then, exists a shake matching.
- Del. Matching if job j is possible.
- Then, each job gets their most preferred applicant.
- Repeat, until every job has at most one applicant.

**Proof:**
1. Each job gets their most preferred applicant.
2. Each applicant gets a job.
3. No more couples.

**Converse:**
1. Each applicant has a list of jobs.
2. Each job has a list of applicants.

- Let's assume that someone is unmatched.
- Then, it must be an applicant.
- If it is an applicant, it is unmatched.
- For each job, if there is an applicant, then they could be matched.

- For each job, if there is an applicant, then they could be matched.
- If there is no applicant, then there is no match.