Write-Up for the Hiring Problem Simulation

1. Description of the Code

The provided Java and Python program simulates a variant of the hiring problem. In this version of the hiring problem, each candidate is assigned a suitability rating that is an integer chosen uniformly at random from 0 to (x), where (x) is defined differently under three cases:

- **Case A:** (x = ⌈log2n⌉)
- **Case B:** (x = n)
- **Case C:** (x = n^3)

Here, (n) represents the number of candidates, and we simulate the process for every (n) from 1 to 1000. For each (n) and each case, the program runs 10 independent simulation trials. During a trial, the program “hires” a candidate if the candidate’s rating is strictly greater than the current best rating.

The simulation for each (n) involves the following steps:

1. **Determine the Range of Ratings ((x))**:
    - For each candidate count (n), the value of (x) is computed for each case (A, B, and C). For example, in Case A, (x = \lceil \log_2 n \rceil).
2. **Simulation of the Hiring Process:**
    - The simulation starts with no candidate hired (using a `currentBest` initialized to -1).
    - For each candidate, a random rating is generated using Java’s `Random` class.
    - If the candidate’s rating is strictly greater than the `currentBest`, the candidate is hired, and `currentBest` is updated to that rating.
    - The number of hires is recorded for that trial.
3. **Computation of Statistics:**
    - After running the required number of trials (10 in this case), the program computes:
        - The **mean (average)** number of hires.
        - The **standard deviation** of the hires across the trials.
4. **Output to CSV:**
    - The code writes the output to a CSV file named `results.csv`.
    - Each row in the CSV file corresponds to a particular value of (n) and contains the following columns:
        - (n) (number of candidates)
        - For each case (A, B, and C): the value of (x), the computed mean number of hires, and the standard deviation.
    - This output is later imported into Python’s matplotlib for plotting and further analysis.

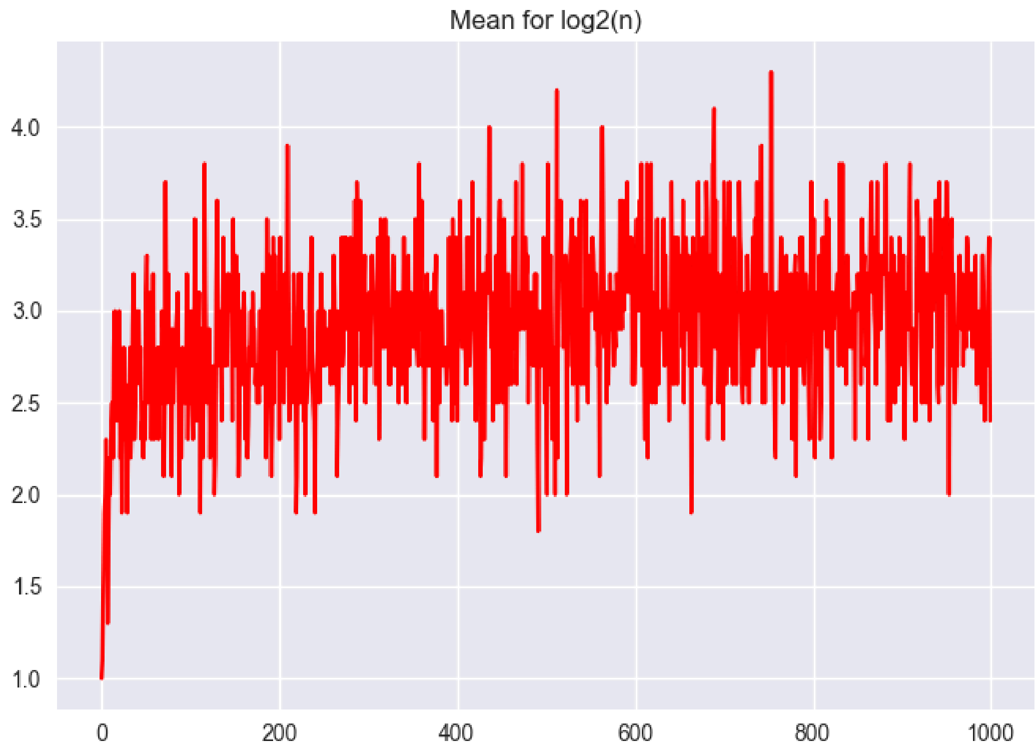
2. Analysis

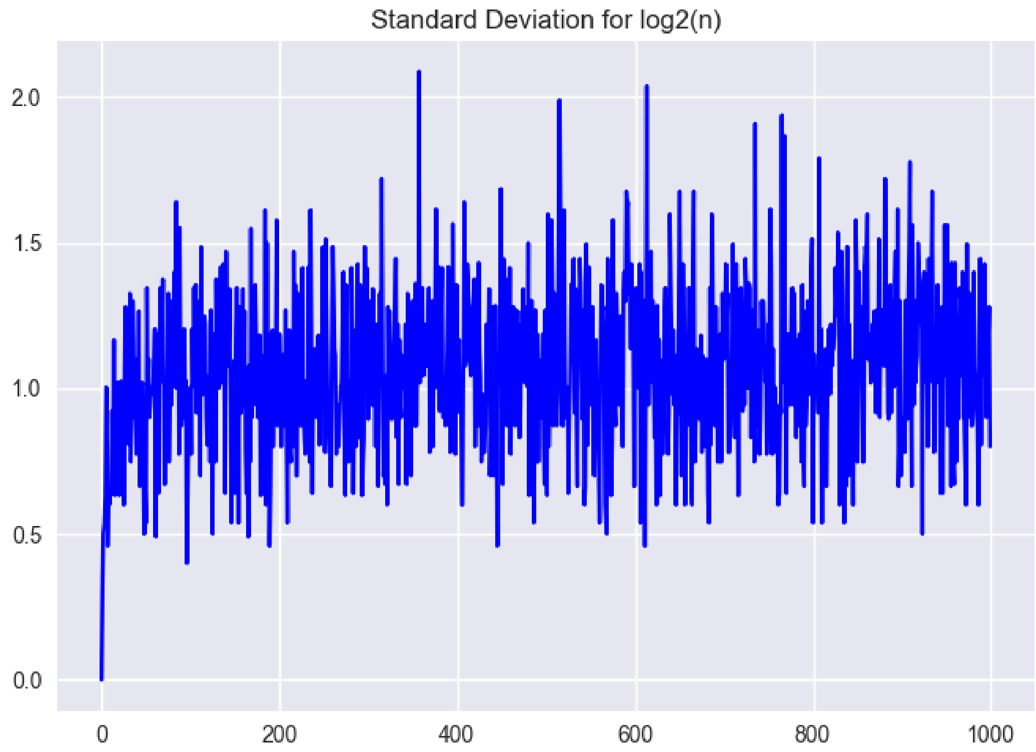
The simulation is designed to study the behavior of the hiring problem under different conditions of the candidate rating range:

1. **Case A ((x = ⌈log2n⌉))**:
   - Since (x) grows very slowly with (n) (logarithmically), the range of possible ratings is small.
   - This means that after a candidate with a high rating is hired, it becomes very unlikely to find a candidate with a strictly higher rating.
   - **Expected Behavior:** The number of hires will be relatively low and will tend to plateau as (n) increases.
2. **Case B ((x = n))**:
   - Here, the range of possible ratings increases linearly with the number of candidates.
   - This makes ties less frequent, and the process starts to resemble the classical hiring problem.
   - **Expected Behavior:** The expected number of hires should be similar to the harmonic series, approximately lnn. This means that as (n) increases, the number of hires grows logarithmically.
3. **Case C ((x = n^3))**:
   - With (x = n^3), the range is very large compared to the number of candidates.
   - Ties are highly unlikely, and the behavior should closely match the continuous case.
   - **Expected Behavior:** The number of hires is expected to be nearly identical to the classical hiring problem, again following roughly (lnn+O(1)).

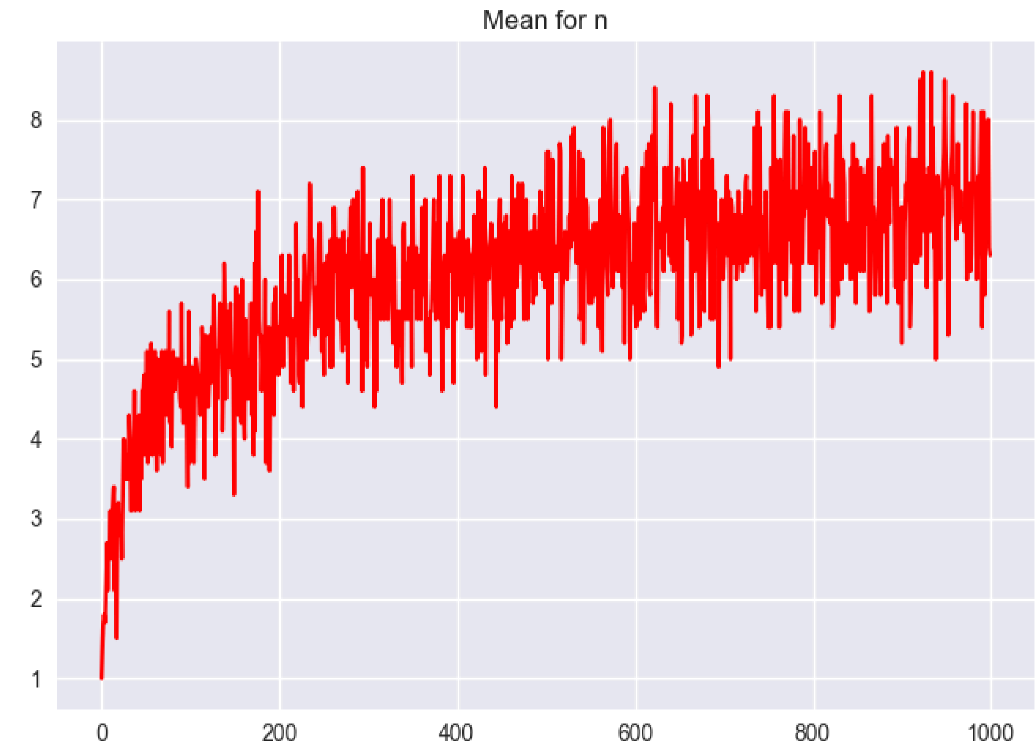
3. Plots

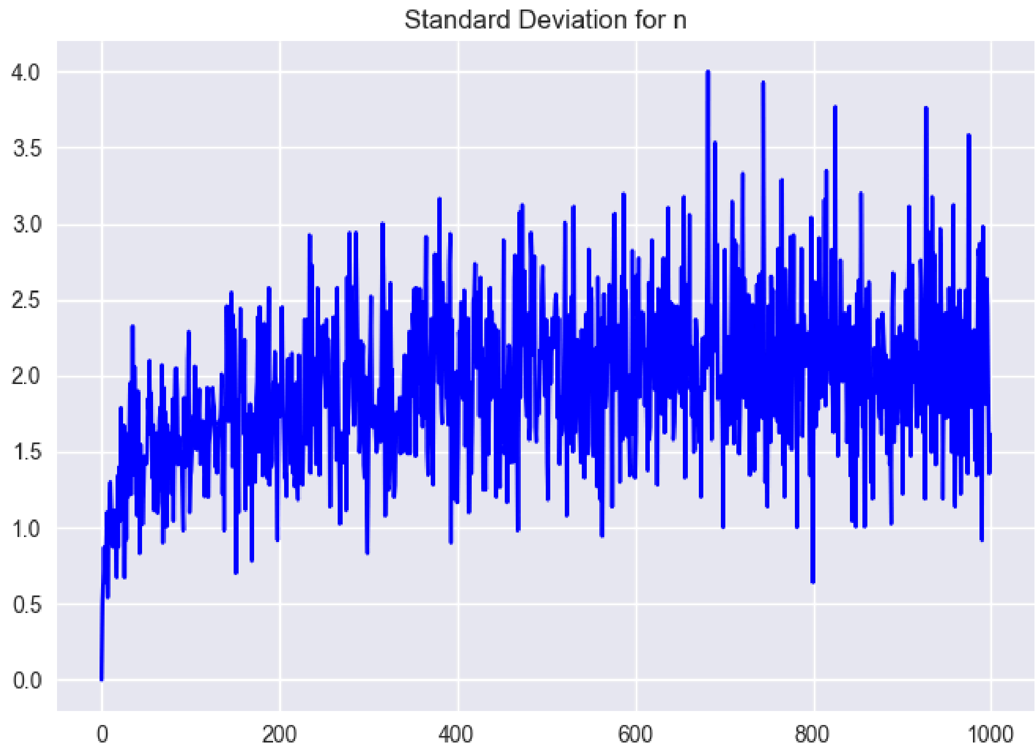
1. **Mean and Standard Deviation graph for x = ⌈log2n⌉:**



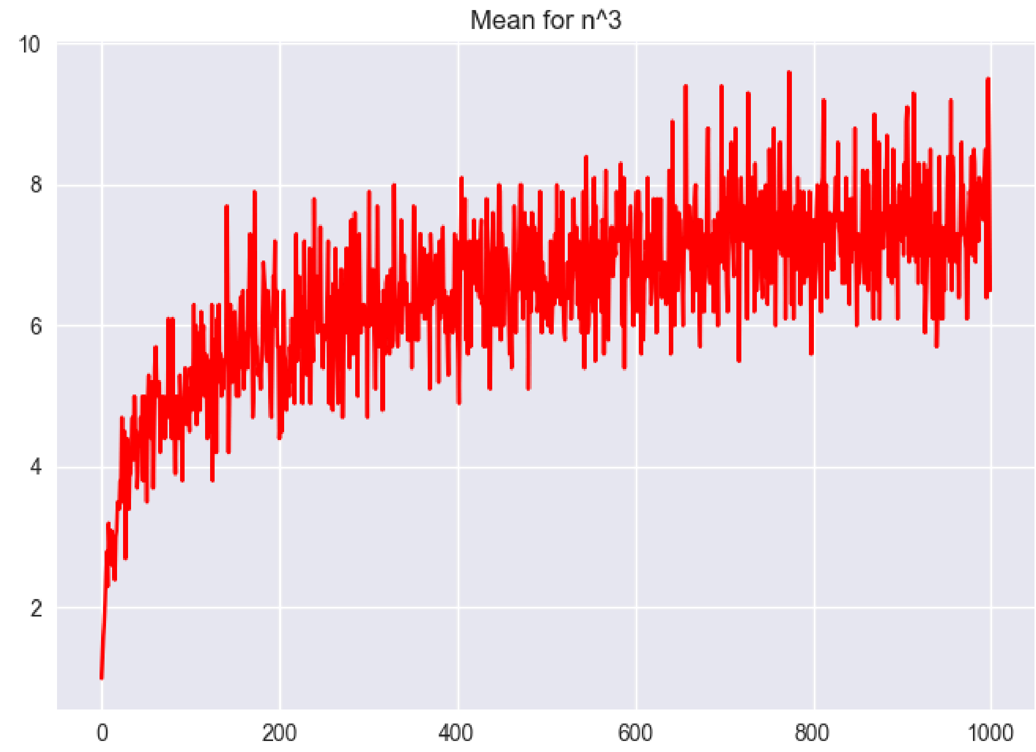


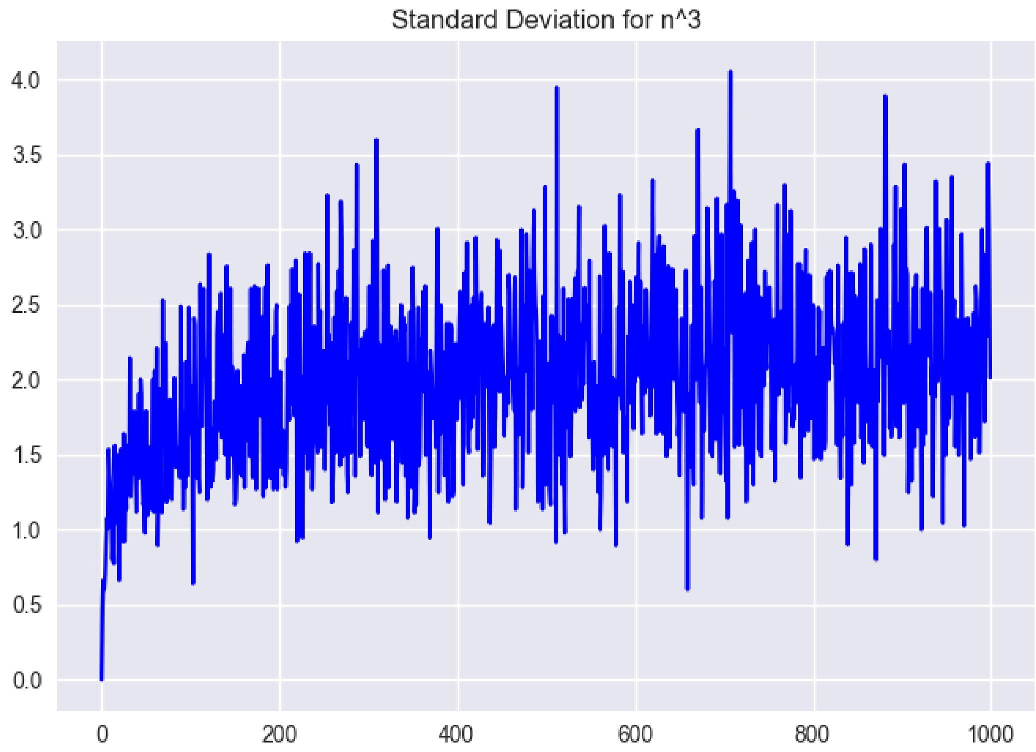
2. **Mean and Standard Deviation graph for x = n:**



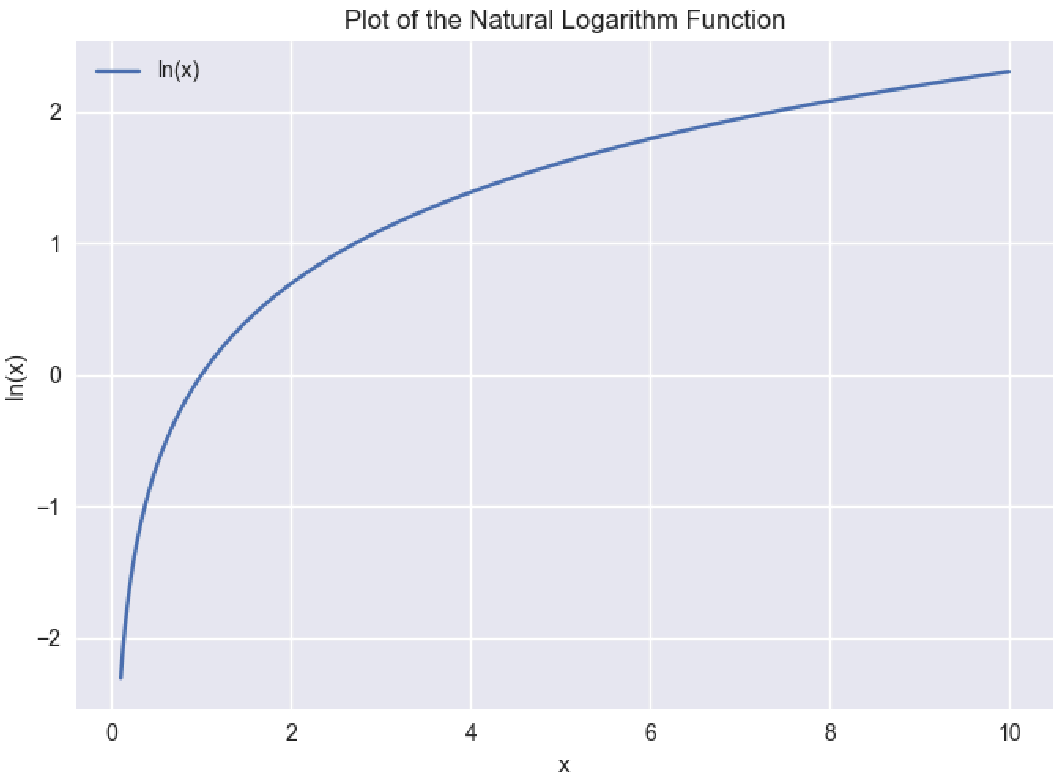


3. **Mean and Standard Deviation graph for x = n^3:**





4. **Plot of log(n) for comparison:**



4. Conclusions

From the simulation and subsequent analysis, we can conclude the following:

- **Impact of the Rating Range (x) on Hiring:**
  - When the rating range is small (Case A), the number of hires is limited. This is because once the maximum possible rating is achieved, no further candidates can be hired. The simulation shows that the hiring process saturates quickly.
  - When (x) increases (Cases B and C), ties become rare, and the process more closely resembles the classical hiring problem. The number of hires grows in a logarithmic fashion with the number of candidates.

In summary, the simulation demonstrates that the discrete nature of candidate ratings and the choice of (x) significantly influence the number of hires. For large ranges of ratings, the hiring process follows the logarithmic behavior predicted by theory, while small ranges quickly lead to saturation in the number of hires. This insight is valuable for understanding both randomized algorithms and practical scenarios where candidate selection is based on discrete ratings.

HW1

Q1 The number of permutations of 'n' numbers is $n!$. As the input of the program is 'n'; and it outputs a permutations of those numbers (random permutation).

$\therefore$

Total Permutation of 'n' numbers =

$$n!$$

Probability of each permutation of 'n' number; where each outcome is equally likely

$$\frac{1}{n!}$$

This problem is similar to Coupon collector's problem. The problem reduces to determining the expected number of trials (permutations generated) to collect $n!$ distinct permutations.

Let event $X \Rightarrow$

$X =$ probability of generating a permutation that is distinct

$$P(X) = \frac{1}{k}$$

$$X_i = \sum_{i=1}^{n!} \frac{1}{i} = \sum_{i=1}^{n!} P(X) \quad \text{—} \ (1)$$

random indicator variable

$$E[X] = E\left[\sum_{i=1}^{n!} P(x)\right]$$

$$= \sum_{i=1}^{n!} E[P(x)]$$

$$= \sum_{i=1}^{n!} \frac{1}{i}$$ From 1

$$= \frac{1}{1} + \frac{1}{2} + \frac{1}{3} \cdots\cdots \frac{1}{(n-1)!} + \frac{1}{n!}$$

$$\approx \ln(n!) + c \quad \text{—— } 2$$

In order to collect all distinct permutations we would have to perform above operation $n!$ times.

$\therefore$ Let T be expected number of trials to generated all distinct permutations.

$\therefore$

$$n! \times E[X]$$

$$= n! \times \ln(n!) + c \quad \text{From } (2)$$

$$T = n! \times \ln(n!) + c$$

Q2 The program simulates hiring process for n candidates and each candidate has a skill level randomly generated between 0 to x.

Here $x = \lceil \log_2(n) \rceil$

i.e. all the skill levels are discrete and bounded by $\log_2(n)$

Expected number of hires for standard hiring problem is given by

$$E[X] = \ln(n) + O(1)$$

In this case, each hire corresponds to a new maximum skill level

The probability of a new maximum skill level occuring at each step is $\frac{1}{i}$, where i is the number of candidates seen so far.

$$Pr(X_i) = \frac{1}{i}$$

$\because$ skill levels are bounded by $\log_2(n)$

$$\therefore \; E[X] = \sum_{i=1}^{X} Pr(X_i)$$

$$= \sum_{i=1}^{\log_2 n} \frac{1}{i}$$

$$\underline{E[X] = \ln(\log_2(n))}$$

Q3 Let $E_{i,j}$ be an event where dice i and dice j show different outcomes.

$\therefore$

S = Sample Space of one dice = $\{1,2,3,4,5,6\}$

Let A be an event where dice ~~A~~ shows x.

$\therefore x \in S$

~~$P(x) = \frac{1}{8}$~~

$P(x=1) = \frac{1}{6}$

$P(x=2) = 1/6$

$\vdots$

$P(x=6) = \frac{1}{6}$

Let compute $P(E_{i,j})$

$\boxed{X_1}$ $\boxed{X_2}$

1^{st} dice 2^{nd} dice

Let ~~A~~ x_1 be outcome on dice 1.

$\therefore$ ~~S~~

$P(x_1) = \frac{6}{6}$ where $x_1 \in S$.

$\therefore$ All possible outcomes for dice 2 are $S - x_1$

$\therefore \quad \|S - x_1\| = 6 - 1 = 5$

Hence let X_2 be outcome of dice 2.
Since X_1 cannot be equal to X_2 (for $E_{i,j}$)

$P(x_2 | x_1) = \frac{5}{6}$ As x_1 is removed from S.

Let

$P(X_1 \cap X_2) = P(X_1) \cdot P(X_2)$ — ①
where X_1 & X_2 are mutallly independent events

But since X_2 the outcome of dice 2 (X_2) depends on the outcome of dice 1 (X_1).
They are $\boxed{\text{not}}$ mutally independent.

$\therefore \quad P(X_1 \cap X_2) \neq P(X_1) \cdot P(X_2)$

for mutually dependent events.

Now lets prove $E_{i,j}$ lack mutual independence ~~and~~ but are pairwise independent.

Let us consider events E_{ij} & E_{kl}.

$$P(E_{ij} \cap E_{kl}) = P(E_{ij}) \cdot P(E_{kl})$$

case ① if $i \neq j \neq k \neq l$:

$$P(E_{ij} \cap E_{kl}) = \left(\frac{5}{6}\right) \times \left(\frac{5}{6}\right) = \left(\frac{5}{6}\right)^2$$

case ② for $i \neq j = k \neq l$

$$P(E_{ij} \cap E_{jl}) = \left(\frac{5}{6}\right) \times \left(\frac{5}{6}\right) = \left(\frac{5}{6}\right)^2 \quad —②$$

$\therefore$ E_{ij} is pairwise independent ; since all pairs of Events satisfy the condition for pairwise independence.

Now lets prove that E_{ij} is ~~ot~~ not mutually independent. Recall ①

To show that the collection $\{E_{ij}\}$ is not mutually independent, we need to find a subset of events whose joint probability does not equal the product of individual opportunities.

$$P(E_{12} \cap E_{23} \cap E_{13}) = P(E_{12}) \cdot P(E_{23}) \cdot P(E_{13})$$

$$= \left(\frac{5}{6}\right)^3$$

However events that require:

$$- P(E_{12} \cap E_{23} \cap E_{13}) = \cancel{\frac{6}{6}} \cdot \frac{5}{\underset{3}{\cancel{6}}} \cdot \frac{\overset{2}{\cancel{4}}}{\cancel{6}\,3} = \frac{5}{9}$$

$$= \frac{6}{6} \cdot \frac{5}{6} \cdot \frac{4}{6} = \frac{5}{9}$$

Since $P(E_{12} \cap E_{23} \cap E_{13}) \neq P(E_{12}) \cdot P(E_{23}) \cdot P(E_{13})$ the event E_{ij} is pairwise independent, but not mutually independent.