Dishan's Thanksgiving Problem The final results are back, and the 381 GTAs need to grade a whopping n student submissions! They want to spend as little time as possible grading (they want to play GTA 6). They can choose between two methods for grading any given submission:

- **Real grading:** This always takes a couple of painstaking minutes per submission, but it guarantees an accurate grade. The solution type will also be cached into the GTAs brain.
- **Skimming:** This method takes negligible time. However, the grader will skim so quickly that they only have a 50% chance of giving an accurate grade! That is, of course, unless they've seen the solution type before (in which case it is guaranteed with 100% chance to be accurate).

Unfortunately, the students have many different solution types. A preliminary ChatGPT analysis showed that every solution type had at least k instances among the n submissions.

Give a randomized algorithm to minimize the total number of submissions that were *real graded* such that the probability of grading any submission inaccurately is at most $1/n$ (the GTAs can handle a couple regrade requests, after all). Express the asymptotic bound of real graded submissions in terms of n and k.

Solution. Our algorithm will be to cache r real grades and grade the $(n-r)$ remaining submissions by skimming. More specifically, we choose r random submissions (chosen uniformly) to grade for real.

We show that the probability that any i type of solution is graded incorrectly is at most $1/n$.

$$\begin{aligned}
\Pr[\exists \text{ type i graded incorrectly}]] &\leq (n/k) \times \Pr[\text{ fixed i graded incorrectly}] \\
&= (n/k) \times \Pr[\text{did not see type i in r sub.}] \times \Pr[\geq 1 \text{ skim grade incorrect}] \\
&\leq (n/k) \times \left(\frac{n-k}{n}\right)^r \times (1/2) \\
&\leq (n/2k) \times \exp(-kr/n)
\end{aligned}$$

For $r = (2n/k) \times \log(n^2/2k)$, we have

$$\begin{aligned}
(n/2k) \times \exp(-kr/n) &= (n/2k) \times \exp(-\log(n^2/2k)) \\
&= (n/2k) \times (2k/n^2) = 1/n.
\end{aligned}$$

Asymptotically, we require $r = O\left(\frac{n}{k} \log\left(\frac{n}{k}\right)\right)$ submissions

Turing Machine. Recall that a turing machine $\mathcal{M}$ is represented by the 7-tuple $(Q, \Sigma, \tau, \delta, q_0, q_{acc}, q_{rej})$. The language of $\mathcal{M}$ is all the strings which lead to acceptance, $L(\mathcal{M}) = \{x \in \Sigma^* : q_{accept} \leftarrow \mathcal{M}(x)\}$. Note that a turing machine may not halt on strings x which it does not accept.

Let $\mathcal{M}_1$ and $\mathcal{M}_2$ be two turing machines. Show how we can construct:

1. $L(\mathcal{M}_1 \cap \mathcal{M}_2) := L(\mathcal{M}_1) \cap L(\mathcal{M}_2)$;
2. For any fixed $k \in \mathbb{N}$: $L_k(\mathcal{M}_1) := \{x \in L(\mathcal{M}_1) : \mathcal{M}_1(x)$ accepts in k steps$\}$
3. $L(\mathcal{M}_1 \cup \mathcal{M}_2) := L(\mathcal{M}_1) \cup L(\mathcal{M}_2)$;