100 囚徒问题仿真报告

算法说明

1. **问题背景**

   100 个囚徒每天可进入房间，打开最多 50 个盒子寻找自己的编号。若所有囚徒均在某一天成功找到编号，则全体释放；否则游戏继续，盒子重新随机排列。

2. **策略设计**

   - **随机策略（Random）**：每个囚徒每天随机选择 50 个盒子检查。
   - **链式策略（Wise）**：每个囚徒从自己的编号开始，跟踪盒子中的编号形成链，直到找到自己或超过 50 次。

3. **模拟逻辑**

   - 每天生成新的随机盒子排列。
   - 囚徒按策略选择盒子，成功则被释放。
   - 若所有囚徒在某一天均成功，游戏结束；否则持续至最大天数（默认 10 天）。

实验结果

1. **单次模拟结果（随机策略）**

   - 10 天内未达成全体释放。
   - 失败原因为每天需所有囚徒在同一日成功，概率极低。

2. **多次模拟统计（链式策略，10,000 次）**

天数	成功次数	占比	累计占比
1	3149	31.49%	31.49%
2	2146	21.46%	52.95%
3	6765	14.70%	67.65%
4	7790	10.25%	77.90%
5	8522	7.32%	85.22%
6	8936	4.14%	89.36%
7	9273	3.37%	92.73%
8	9508	2.35%	95.08%
9	9651	1.43%	96.51%
10	9755	1.04%	97.55%

- **关键结论**：10 天内成功率约 97.55%。

理论分析

1. **随机策略**
   - 每个囚徒单日成功概率：$q = \frac{k}{n} = 0.5$。
   - 全体单日成功概率：$p = (0.5)^{100} \approx 7.8 \times 10^{-31}$。
   - 10 天累积成功概率：$p_d = 1 - (1 - p)^{10} \approx 10 \times 10^{-31}$，几乎为零。
2. **链式策略**
   - 全体单日成功概率：$p = 1 - \sum_{i=k+1}^{n} q_i$。$qi$为存在长度为 i的环的概率：$qi = \frac{C_n^i \cdot (i-1)! \cdot A_{n-i}^{n-i}}{A_n^n} = \frac{1}{i}$
   - 当 $n = 100, k = 50$ 时, $p \approx 0.3118$
   - 10 天累积成功概率：$p_d = 1 - (1 - p)^{10} \approx 0.9762$，概率非常高。

优化方向

1. 盒子配置的批量预生成

问题： 逐天生成随机盒子导致重复调用 random.sample，效率低下。

优化方案： 预生成所有天数的盒子序列，存储为三维张量（形状：[max_days, n, n]），其中 max_days 为最大天数，n=100。

通过 numpy 的 permutation 函数一次性生成所有随机排列，利用广播机制加速。

2. 囚徒查询的并行化

问题： 逐囚徒循环调用 choose_wisely 或 choose_randomly，无法利用硬件并行性。

优化方案：

链式策略向量化：

构建囚徒-盒子追踪矩阵（形状：[n, max_days, k]），记录每个囚徒每天每一步的盒子索引。

通过矩阵运算一次性计算所有路径结果（如 boxes[追踪矩阵] 直接映射路径）。

随机策略向量化：

预生成所有囚徒每天随机选择的 k 个盒子索引（形状：[n, max_days, k]），通过 numpy 索引直接提取目标值。