100 囚犯抽签问题仿真报告

一.实验名称

100 囚犯抽签问题求解与成功率分析

二. 实验目的

通过计算机仿真验证 100 囚犯抽签问题中两种策略（随机策略与循环策略）在大量实验下的成功概率差异，并结合排列循环理论对循环策略的成功概率进行理论计算与验证。

三. 实验环境

操作系统：Windows 11

编程环境：Visual Studio Code

编程语言：Python 3.9.20

使用库：time、matplotlib（用于绘图）

四. 实验内容与方法

本实验基于著名的“100 囚犯抽签问题”，设计并实现两种囚犯搜索策略，通过大量仿真对比分析其成功概率差异，并验证理论推导结果的合理性。

具体方法如下：

a) 设置囚犯数量 N、每名囚犯尝试次数 K 和实验轮数 T。

b) 随机生成 1 到 N 的排列模拟盒内纸条编号。

c) 依次让囚犯根据指定策略尝试寻找自己的编号。

d) 记录每轮实验全员是否成功，并统计总体成功率。

e) 使用柱状图和直方图展示两种策略的成功概率与循环策略的分布特性。

f) 对比仿真结果与理论值，验证排列循环理论的有效性。

五. 核心算法说明

1) 随机策略

每名囚犯随机打开 K 个不同的盒子，寻找自己的编号。若在规定次数内找

到则视为成功，若全体囚犯均成功则本轮成功。

特点：算法实现简单。囚犯选择完全随机，策略无依赖。

成功概率随人数 N 增加迅速下降，理论概率近似为$(\frac{K}{N})^N$，极低。

2) 循环策略

每名囚犯从与自己编号相同的盒子开始，依次跳转至编号等于盒内纸条编号的盒子，直至找到自己的编号或达到 K 次尝试。

核心原理：基于排列循环结构，若所有循环长度均不超过 K，则本轮实验成功。

理论成功概率： $P_{success} \approx \prod_{i=K+1}^{N}(1-\frac{1}{i})$

优势：

成功概率显著高于随机策略。

当 N=100，K=50 时，理论成功概率约为 31.18%，接近仿真结果。

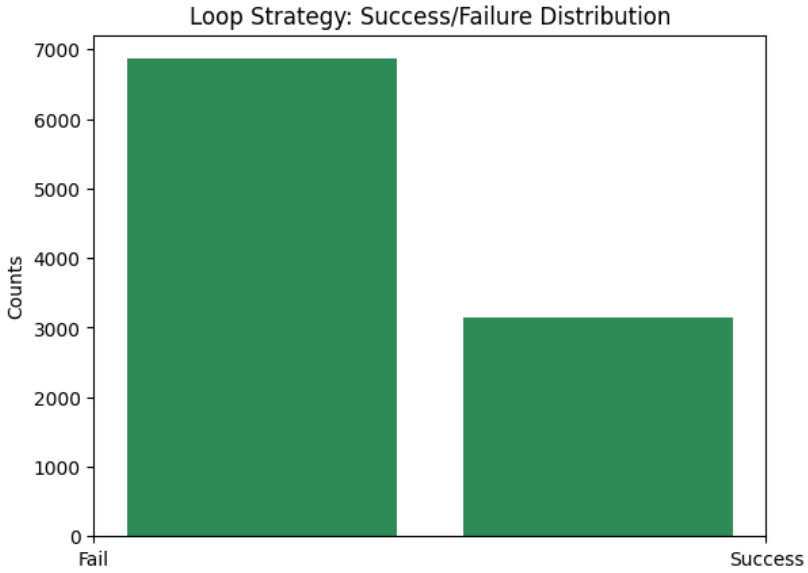
六.实验结果

(1) 成功概率统计

策略	成功概率
随机策略	约 0.000
循环策略	约 0.311

(2) 策略成功率对比柱状图

(3) 循环策略成功分布直方图



七.算法复杂度分析

1) 随机策略

在每轮仿真中，每名囚犯随机选择 K 个不同的盒子，最多尝试 K 次，判断是否找到自己的编号。

单次仿真时间复杂度：$O(N \times K)$

T 轮仿真总时间复杂度：$O(T \times N \times K)$

由于每次抽取随机盒子，且无依赖关系，故复杂度与囚犯人数 N 和尝试次数 K 线性相关。

2) 循环策略

在每轮仿真中，每名囚犯从固定编号盒子出发，根据盒内纸条编号依次跳转，最多 K 次，判断是否找到自己的编号。

单次仿真时间复杂度：$O(N \times K)$

T 轮仿真总时间复杂度：$O(T \times N \times K)$

尽管跳转过程受排列结构影响，但每名囚犯最多执行 K 次操作，复杂度同样为$O(N \times K)$。不过该策略避免了随机抽取，实际运行效率略优。

八.优化思路

1) 向量化优化：利用 NumPy 数组批量操作，减少循环次数，提高仿真效率。
2) 并行计算：多线程或多进程方式并行执行仿真轮次，进一步提升性能。
3) 参数化设计：将囚犯数量、尝试次数等设为可配置变量，方便扩展其他场景测试。
4) 结果可视化增强：增加分布曲线拟合、误差条等展示方式，提升结果分析严谨性。

九.实验结论

循环策略基于排列循环结构理论，显著提高了囚犯获释的成功率。实验仿真结果与理论计算值相符，验证了循环策略作为最优策略的有效性。随机策略在该问题中的成功概率接近于零，无法满足实际需求。