七日世界模组词条出现概率计算工具

这是一个用于计算七日世界游戏中模组词条出现概率的Python工具。

游戏机制

词条类型

游戏中共有10种不同的词条：

1. **异常伤害** - 提升异常状态伤害
2. **弹匣容量** - 增加武器弹匣容量
3. **换弹速度加成** - 提升换弹速度
4. **对普通敌人伤害** - 对普通敌人造成额外伤害
5. **对精英敌人伤害** - 对精英敌人造成额外伤害
6. **对上位者伤害** - 对上位者敌人造成额外伤害
7. **最大生命值** - 增加角色最大生命值
8. **头部受伤减免** - 减少头部受到的伤害
9. **枪械伤害减免** - 减少枪械造成的伤害
10. **异常伤害减免** - 减少异常状态伤害

随机规则

- 同一个模组中，**相同词条不会重复出现**
- 每次随机都是从剩余的词条池中选择
- 这是一个无重复的组合问题

功能特性

- ✅ 计算指定词条数量和范围的出现概率
- ✅ 显示所有满足条件的具体组合
- ✅ 支持自定义词条数量（1-10个）
- ✅ 支持自定义目标词条范围
- ✅ 详细的数学计算结果
- ✅ 交互式命令行界面

使用方法

方法1: 交互式运行

```
python3 mod_affix_probability.py
```

按照提示输入：

1. 参与随机的词条数量（1-10）
2. 目标词条范围（用逗号分隔的数字）
3. 是否显示具体组合（y/n）

方法2: 查看示例

```
python3 affix_examples.py
```

方法3: 编程调用

```
from mod_affix_probability import ModAffixProbabilityCalculator

calculator = ModAffixProbabilityCalculator()

# 计算概率
result = calculator.calculate_probability(
    n_slots=3,           # 词条数量
    target_range=[1,4,5,6]  # 目标范围
)

print(f"概率: {result['probability_percent']:.2f}%")
```

计算原理

这是一个组合数学问题，使用以下数学公式：

数学公式

设：

- N = 参与随机的词条数量
- T = 目标词条范围集合，$|T|$ = 目标范围大小
- A = 所有词条集合，$|A| = 10$

概率公式：

$$P = \frac{\binom{|T|}{N}}{\binom{|A|}{N}} = \frac{\binom{|T|}{N}}{\binom{10}{N}}$$

其中：

- $\binom{|T|}{N}$ = 从目标范围中选择N个词条的组合数（满足条件的组合数）
- $\binom{10}{N}$ = 从所有10个词条中选择N个的组合数（总组合数）

组合数计算

组合数的计算公式为：

$$\binom{n}{r} = \frac{n!}{r!(n-r)!}$$

简化说明

- **总组合数** = $\binom{10}{N}$ = 从10个词条中选择N个的组合数
- **满足条件的组合数** = $\binom{|T|}{N}$ = 从目标范围中选择N个的组合数
- **概率** = 满足条件的组合数 ÷ 总组合数

例子：N=3，目标范围=[1,4,5,6]

当 $N = 3$，$T = {1,4,5,6}$ 时：

$$P = \frac{\binom{4}{3}}{\binom{10}{3}} = \frac{4}{120} = \frac{1}{30} ≈ 3.33%$$

计算过程：

- 总组合数 = $\binom{10}{3} = \frac{10!}{3!(10-3)!} = \frac{10 \times 9 \times 8}{3 \times 2 \times 1} = 120$
- 目标范围大小 = $|T| = 4$
- 满足条件的组合数 = $\binom{4}{3} = \frac{4!}{3!(4-3)!} = \frac{4}{1} = 4$
- 概率 = $\frac{4}{120} = \frac{1}{30} ≈ 3.33%$

满足条件的4种组合：

1. [1,4,5] - 异常伤害+对普通敌人伤害+对精英敌人伤害
2. [1,4,6] - 异常伤害+对普通敌人伤害+对上位者伤害
3. [1,5,6] - 异常伤害+对精英敌人伤害+对上位者伤害
4. [4,5,6] - 对普通敌人伤害+对精英敌人伤害+对上位者伤害

示例场景

场景1: 想要伤害类词条

目标：3个词条都是伤害类型（1,4,5,6）

- 概率：3.33%
- 适合追求高输出的玩家

场景2: 想要防御类词条

目标：2个词条都是防御类型（7,8,9,10）

- 概率：10.00%
- 适合追求生存能力的玩家

场景3: 混合搭配

目标：4个词条在前5种中（1,2,3,4,5）

- 概率：1.19%
- 平衡输出和实用性

概率规律

1. **词条数量越多，概率越低**
    - N=1: 几乎必中
    - N=4: 概率显著下降
2. **目标范围越大，概率越高**

    - 范围=3个词条: 概率较低
    - 范围=7个词条: 概率较高

3. **极端情况**

    - 目标范围=所有词条: 概率=100%
    - 需要词条数 > 目标范围大小: 概率=0%

文件说明

- `mod_affix_probability.py`: 主要的计算工具
- `affix_examples.py`: 详细的使用示例
- `AFFIX_README.md`: 本说明文档

数学复杂度

时间复杂度

- **计算组合数**：$O(1)$ （使用内置函数）
- **总体时间复杂度**：$O(1)$ （常数时间计算）

空间复杂度

- **存储组合列表**（可选）：$O(\binom{|T|}{N})$
- **不显示组合时**：$O(1)$

与强化概率计算的对比

特性	词条出现概率	强化成功概率
数学模型	静态组合问题	马尔可夫链
公式复杂度	$O(1)$	$O(\text{指数级})$
计算方式	直接公式计算	穷举所有路径
时间复杂度	$O(1)$	$O(k^n)$

其中 k 是平均可选择数，n 是强化次数。

系统要求

- Python 3.6 或更高版本
- 无需额外依赖库（仅使用标准库）

适用场景

这个工具适合：

- 想要了解特定词条组合概率的玩家
- 制定模组收集策略
- 评估不同目标的实现难度

- 优化游戏资源分配