

資料結構



演算法

列舉

暴力法

列舉，暴力法

[十八豆教育科技](#)

雞兔同籠(1)

- 還記得你是怎麼回答這題目的嗎？
 - 一元一次方程式？
 - 抬腳法？
 - 公式解？
 - 畫圖？



雞兔同籠，共有 8 個頭，20 隻腳，
請問籠子裡面雞兔各有幾隻？



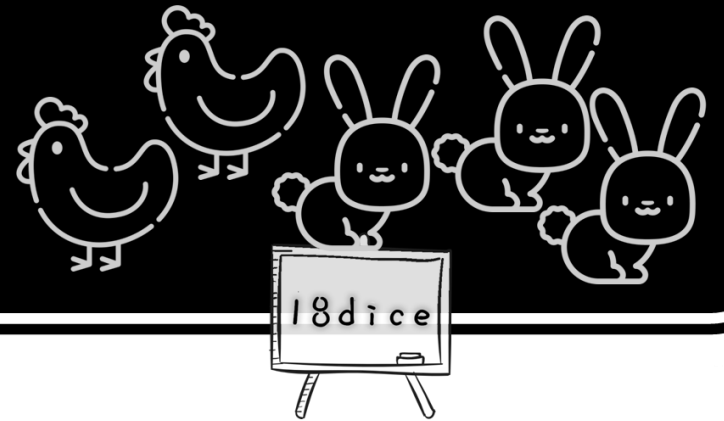
雞兔同籠(2)

- 如果記得，你還記得**第一次**是怎麼回答的嗎？
 - 一元一次方程式？
 - 抬腳法？
 - 公式解？
 - 畫圖？

相信很多第一次遇到這
題目的小學生，使用的
方法都是最直接單純的
畫圖！












雞兔同籠，共有 8 個頭，20 隻腳，
請問籠子裡面雞兔各有幾隻？



雞兔同籠(3)

最簡單直覺得方法就是
把所有可能列出來！

	8隻兔子，0隻雞	32隻腳
	7隻兔子，1隻雞	30隻腳
	6隻兔子，2隻雞	28隻腳
	5隻兔子，3隻雞	26隻腳
	4隻兔子，4隻雞	24隻腳
	3隻兔子，5隻雞	22隻腳
	2隻兔子，6隻雞	20隻腳
	1隻兔子，7隻雞	18隻腳
	0隻兔子，8隻雞	16隻腳

這就是『**列舉**』

1 一
2 二
3 三

列出所有可能的答案

就是『**列舉**』



列舉

- 概念：
 - 列出所有的可能答案，逐一檢視是否符合需求。
- 限制：
 - **答案範圍**必須是有**限制**的，當答案的範圍可以被無限擴展時，就不適用列舉法。
- 列舉法又稱：
 - 窮舉法
 - 暴力法

雞兔同籠 - 程式答題

```
#include<stdio.h>
int main()
{
    int lifeAllCount = 8, legAllCount = 20;
    int liftChickenCount = 0, liftRabbitCount = 0;
    int count = 0;

    for (liftChickenCount=0; liftChickenCount<=lifeAllCount; liftChickenCount++) {
        liftRabbitCount = lifeAllCount - liftChickenCount;
        count = liftRabbitCount * 4 + liftChickenCount * 2;

        if (count == legAllCount) {
            printf("Chicken: %d, Rabbit: %d\n", liftChickenCount, liftRabbitCount);
        }
    }

    return 0;
}
```

列舉法會使用迴圈列出
所有可能答案

再搭配 if 判斷式確認可
能的答案是否有完全符
合題目需求

