屏東縣第 60 屆國中小學科學展覽會
作品說明書

科　　別：生活與應用科學科(一)

組　　別：機電與資訊

作品名稱：尼姆(Nim)遊戲程式設計

關 鍵 詞：Nim 、＿＿＿＿＿＿、＿＿＿＿＿＿（最多三個）

編號：

製作說明：

1.說明書封面僅寫科別、組別、作品名稱及關鍵詞。

2.編號：由承辦學校統一編列。

3.封面編排由參展作者自行設計。

摘要

撿石頭（Nim）的遊戲源自中國，經由被販賣到美洲的奴工外傳。所以這個小遊戲先在工人間流行，他們就地取材撿小石子來玩。後來流傳到上流人士，改以銅板在酒吧櫃檯上玩，遊戲的規則很簡單：兩人輪流取銅板，每次需在某一列取一枚或一枚以上的銅板，但不能同時在兩列取銅板，最後將銅板拿光的人輸。

壹、 研究動機

有天同學在班上玩起撿石頭（Nim）遊戲，而我們玩了一陣子後思考這遊戲看似有某種規則，試著找出贏的方式。現在是一個講求自動化的時代，機械和電子產品的廣泛運用於各個層面，甚至於在我們的日常生活裏都用得到，使得我們行事更加方便，生活品質大為提升。希望結合數學與程式開發一遊戲程式，讓玩家一個人的時候也能跟電腦挑戰。

貳、 研究目的

一、透過數學計算撿拾石頭遊戲（Nim）的必勝方程式。

二、運用電腦程式人工智慧開發撿拾石頭遊戲（Nim），玩家可以自行輸入石頭的堆數、每堆石頭的數量、每次撿取石頭的最大數量，只要玩家發生一次失誤，電腦一定贏。

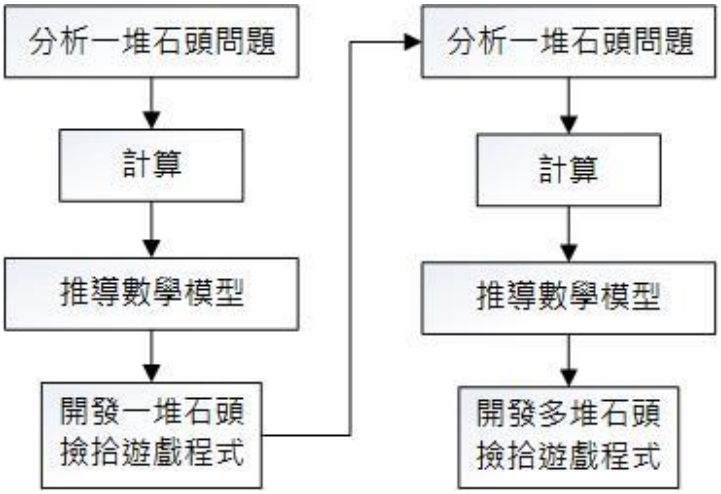
三、運用歸納、分類、統整的科學能力，培養科學素養。

參、 研究設備及器材

本研究使用 Python 語言開發模擬程式，Python 是廣泛使用的高階程式語言，作為一種直譯語言，擁有簡潔易讀和快速學習的特性讓程式的結構清晰明了。

肆、 研究過程或方法

整體研究分成五大步驟：分析問題、計算、推導數學模型、開發程式，如下圖一。



圖一、研究流程圖

一、 分析與計算一堆石頭撿拾問題

一堆石頭在輪流撿拾的過程中，因為遊戲規則為最後將銅板拿光的人輸，所以只要取到倒數第二個銅板的人必贏，所以要計算如何才能取到倒數第二個銅板。以 8 顆石頭為例，我們若以最多取 2、3、4、5、6、7 等五種情況，每次都取相同的數量來計算，計算結果如表一。

石頭總數量	倒數第二顆石頭	每次取的石頭數量	算式	先取輸贏判斷
8	$(8-1)$	2	$(8-1)\div 2=3\ldots 1$	輸
8	$(8-1)$	3	$(8-1)\div 3=2\ldots 1$	贏
8	$(8-1)$	4	$(8-1)\div 4=1\ldots 3$	輸
8	$(8-1)$	5	$(8-1)\div 5=1\ldots 2$	輸
8	$(8-1)$	6	$(8-1)\div 6=1\ldots 1$	輸
8	$(8-1)$	7	$(8-1)\div 7=1\ldots 0$	贏

表一、8 顆石頭撿拾狀況分析圖

從上表算式與輸贏判斷我們可以得知，算式中的商數代表一共可以取幾次，歸納為下列 3 點：

（一） 若商數為奇數且餘數為 0，則先取的會贏

（二） 若商數為奇數且餘數不為 0，則先取的會輸

（三） 若商數為偶數且餘數為 0，則先取的會輸

（四） 若商數為偶數且餘數不為 0，則先取的會贏

由上述條件可知商數為奇數且餘數為 0 和商數為偶數且餘數不為 0 兩種狀況下，則先取的才會贏，商數為奇數代表下一輪換後取的拿石頭，因此只有在奇數，並且取到第 6 顆石頭，下一輪換後取的只能拿的 7 顆石頭，先取的必贏。商數的奇數表示先取的取完結束，偶數表示後取的取完結束，若把先取和後取兩個取的石頭數量加起來當作一組，那就無需判斷奇偶數的差別。餘數代表多餘的石頭，只要有餘數就代表先取的一定會輸，因為後取的把剩下的取完就會獲勝，所以先取的人要先取餘數數量的石頭，剩下的數量就一定可以整除，才代表有贏的機會。

二、 推導數學模型

由於遊戲規則每次最少取 1 顆，最多可取的數量由玩家輸入，假設最少取 1 顆，最多能取 3 顆石頭，雖然兩個人取的數量不一定，但兩個人取的數量加起來一定大於等於 4（3+1）顆。假設石頭總數量為 n，倒數第二顆石頭為 n-1，規則中每次取的數量是玩家自行規定且有限制，因此每次取的數量也是變因之一，每人每次最多可以取 p 個石頭，則兩個人取的數量為 p + 1，假設餘數為 r，歸納下列條件推導出數學模型：

（一） 取到倒數第二個銅板（n-1）

（二） 先取的要取餘數數量 r 的石頭

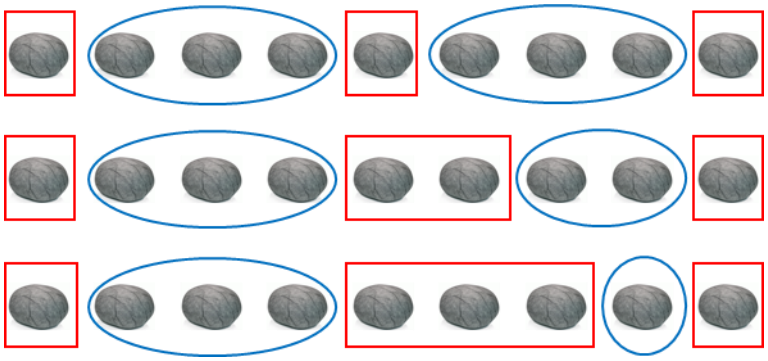
（三） 先取和後取兩個取的石頭數量加起來當作一組（p+1）

（四） 取完餘數數量的石頭後要整除（p+1）

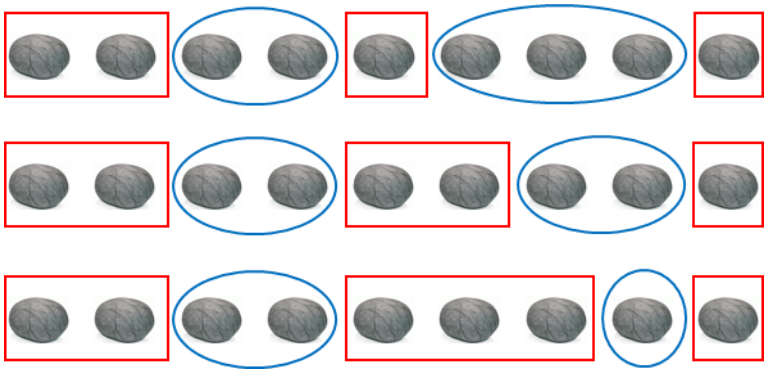
推導出數學模型為 $[(\mathrm{n}-1)-\mathrm{r]} \div (p+1)=\mathrm{y} \ldots 0$ => $\mathrm{r}=(\mathrm{n}-1)-\mathrm{y}(p+1)$，若 r = 0 後取的會贏，若 r > 0 先取的會贏。利用此數學模型來模擬 9 顆石頭，最多可取

3 顆的遊戲情形，$r = (9-1) - y(3+1)$，可以得到 y=2、r=0，後取的必贏，由於 r=0 所以先取的玩家第一手可以取 1 顆、2 顆或 3 顆石頭，圖二為各種取法的過程模擬圖：

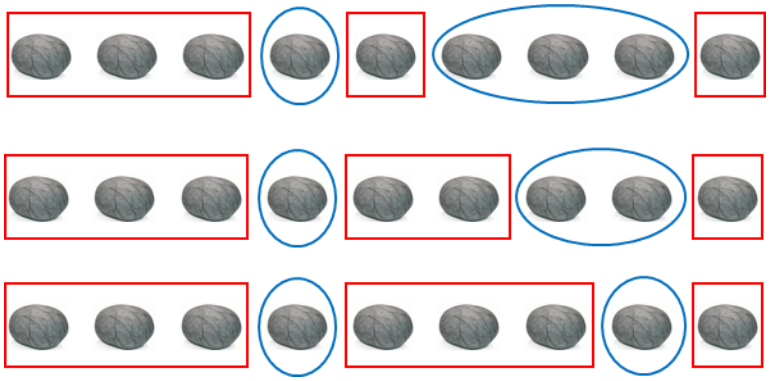
1. 第一手取 1 顆石頭：



2. 第一手取 2 顆石頭：

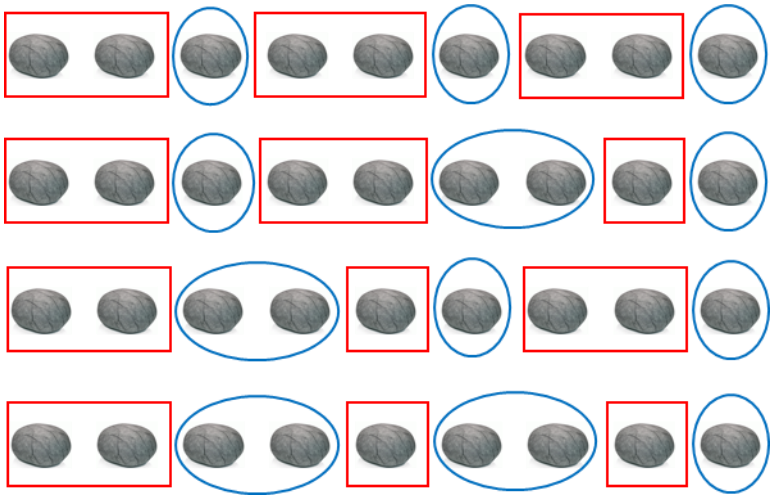


3. 第一手取 3 顆石頭：



圖二、 9 顆石頭最多取 3 顆撿拾情況

從圖二可看出無論先取的玩家取幾顆石頭，只要後取的玩家取(4 - 前一手撿的數量) 顆，後取的玩家必贏。再模擬 9 顆石頭，最多可取 32 顆的遊戲情形，$r = (9-1) - y(2+1)$，可以得到 y=2、r=2，由於 r=2 所以先取的玩家第一手取 2 顆石頭就必贏，圖三為各種取法的過程模擬圖：

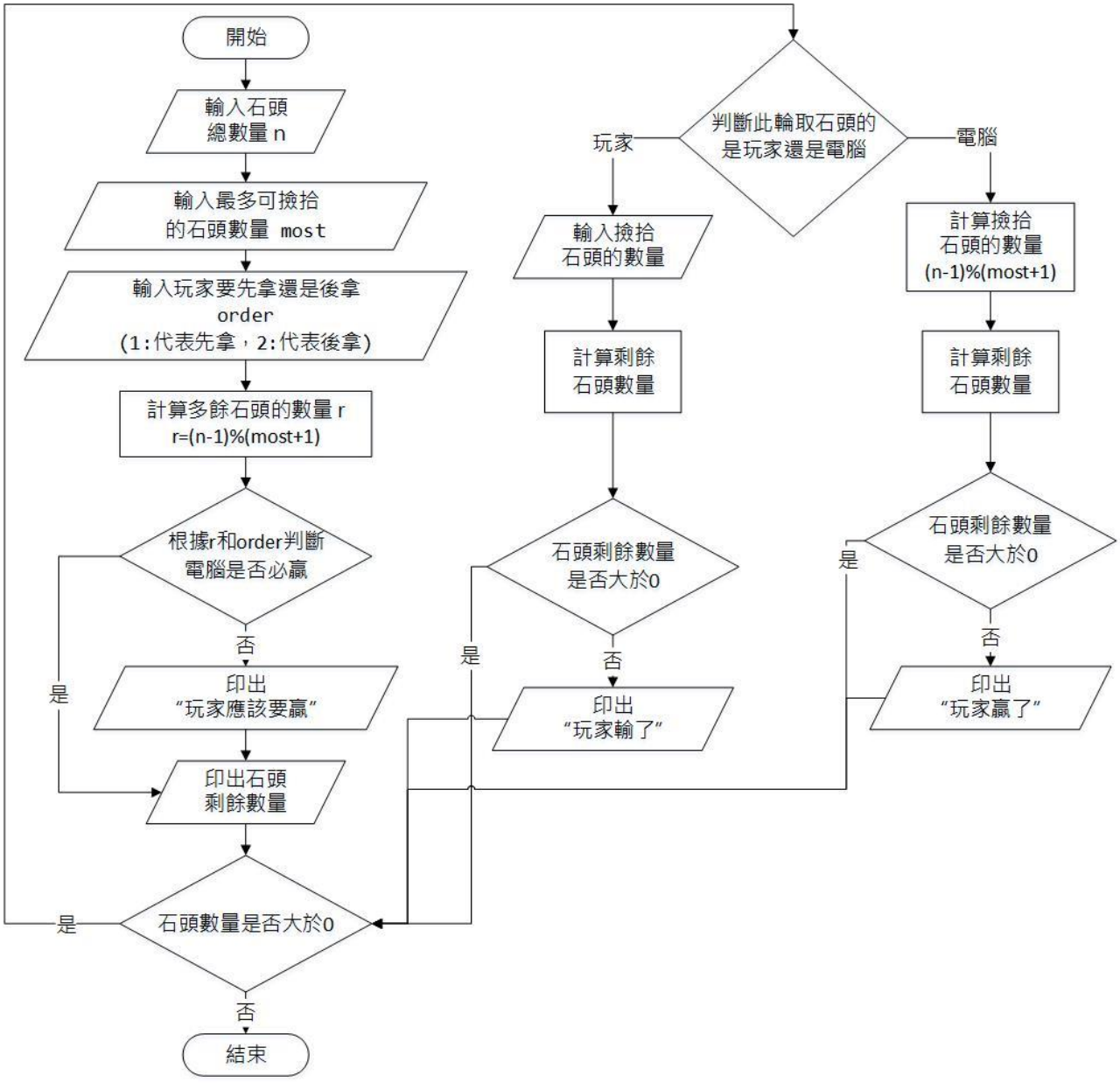


圖三、 9 顆石頭最多取 2 顆撿拾情況

從圖三看出只要先取的玩家第一手取 2 顆石頭，之後只要取 (3 - 前一手撿的數量) 顆，先取的玩家必贏，也驗證推論出的數學模型正確。

三、 開發一堆石頭撿拾遊戲程式

根據統整的必贏條件和推論出來的數學模型，我們可以繪出一堆石頭撿拾遊戲的程式流程圖，如圖四。



圖四、一堆石頭撿拾遊戲程式流程圖

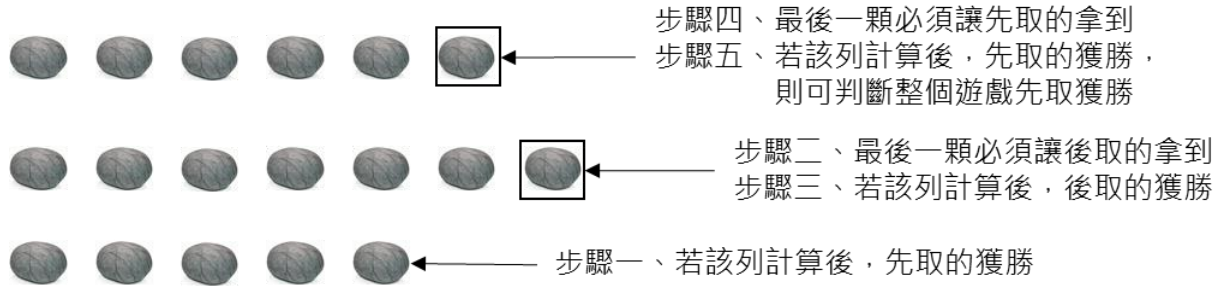
利用 Python 開發一堆石頭撿拾遊戲程式，由玩家輸入石頭總數量、每次最多可撿石的數量和選擇先取還是後取等三個變數，計算餘數 r，並根據玩家選擇先取還後取判斷電腦是否有贏的機會，並進行遊戲，程式碼如下：

```
n = int(input('請輸入石頭總數量:'))
most = int(input('請輸入最多可撿拾的石頭數量:'))
order = int(input('請問您要先拿還是後拿(1:代表先拿，2:代表後拿):'))
r = (n-1)%(most+1)
#先判斷電腦是否有贏的機會
if (order==2):
    if r==0 or r>most:
        print('玩家應該要贏')
if (order==1):
    if r>0:
        print('玩家應該要贏')
remaining = n #剩下的石頭數量
# 印出石頭數量
for i in range(0,remaining):
    print('o',end='')
print()
#遊戲開始
while(remaining > 0):
    p = 0 #玩家撿石頭的數量
    print()
    if order == 1:
        while (p == 0 or p > most): #判斷玩家輸入撿拾的數量是否合理
            print('請輸入撿拾的石頭數量(1~',most,'):')
            p = int(input())
        remaining = remaining - p
        order = 2 #下一手換電腦撿石頭
        for i in range(0,remaining):
            print('o',end='')
        if (remaining <= 0):
            print('玩家輸了')
    else:
        comp = (remaining-1)%(most+1) #計算電腦撿石頭的數量
        if (comp == 0): #若計算後電腦撿的數量為 0，則讓電腦撿 1 顆石頭
            comp = 1
        #若計算後電腦撿的數量大於 1，且會撿到最後 1 顆石頭，則讓電腦少撿 1 顆
        if (remaining - comp == 0 and comp > 1):
            comp = comp - 1
        remaining = remaining - comp
```

```
        order = 1 #下一手換玩家揀拾石頭
        print('電腦撿取',comp,'顆石頭')
        for i in range(0,remaining):
            print('o',end='')
        print()
        if (remaining <= 0):
            print('玩家贏了')
```

四、 分析與計算多堆石頭撿拾問題

完成一堆石頭的遊戲數學模型推論和程式開發後，我們利用一堆石頭的數學模型來推論多堆石頭的情況，多堆的遊戲規則如下：石頭由第一列依序往下一列取，每人每次需在某一列取一枚或一枚以上的銅板，但不能同時在兩列取銅板，直到最後，將銅板拿光的人輸。我們由最後一堆石頭往前推算，若最後一堆先取的獲勝，則代表倒數第二堆的最後一顆必須讓後取的取到，反之，若最後一堆後取的獲勝，則代表倒數第二堆的最後一顆必須讓先取的取到，推導模式如下圖五。

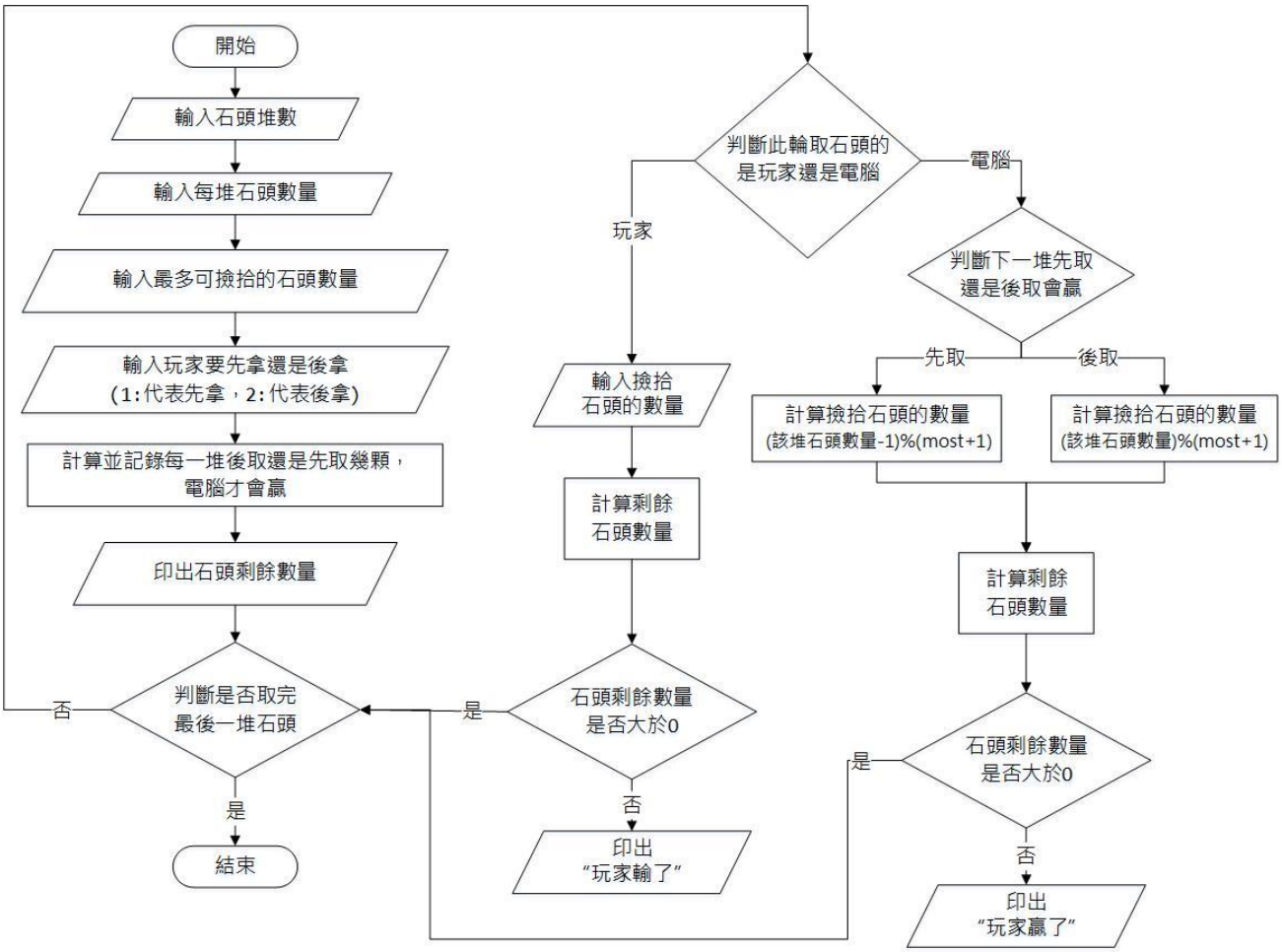


圖五、多堆石頭推導步驟圖

從最後一堆往前推算並記錄各堆需要先取還是後取，推論至第一堆石頭即可知道獲勝模式，若下一堆必須先取，則該堆最後一顆要給玩家取模型為 r = (該堆石頭數量-1)%(most+1)，若下一堆必須後取，則該堆最後一顆必須電腦取模型為 r = (該堆石頭數量)%(most+1)。

五、 開發多堆石頭撿拾遊戲程式

根據統整多堆石頭的必贏條件和推論出來的數學模型，我們可以繪出一堆石頭撿拾遊戲的程式流程圖，如圖六。



圖六、多堆石頭撿拾遊戲程式流程圖

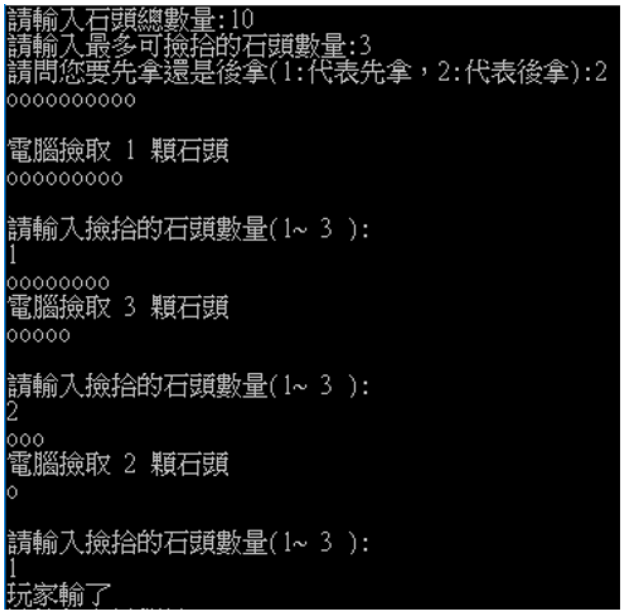
利用 Python 開發多堆石頭撿拾遊戲程式，由玩家輸入石頭堆數、每堆石頭數量、每次最多可撿石的數量和選擇先取還是後取等四個變數，電腦根據玩家所輸入的變數，由最後一堆推算至第一堆，判斷電腦是否有贏的機會，並進行遊戲，程式碼如下：

```
#多堆石頭(每堆石頭數量不同)的遊戲
import os
n = int(input('請輸入堆數:'))
stoneStr=input('請輸入各堆石頭數量，以空白區隔:')
most = int(input('請輸入最多可撿拾的石頭數量:'))
stone = [int(i) for i in stoneStr.split(' ')]
stoneori = [int(i) for i in stoneStr.split(' ')]
order = [0] * n
comOrder = 0
#由最後一堆往前推算每一堆要先取還是後取，並記錄到 order 陣列
#0 代表後取，其餘數字代表先取的數量
for i in range(n,0,-1):
    if i == n or comOrder == 1:
        r = (stone[i-1]-1)%(most+1)
    if comOrder == 2:
        r = (stone[i-1])%(most+1)
    if r==0 or r>most:
        comOrder = 2
        print('後取',end='，')
        order[i-1] = 0
    else:
        comOrder = 1
        print('先取',r,end='，')
        order[i-1] = r
print()
playerorder = int(input('請問您要先拿還是後拿(1:代表先拿，2:代表後拿):'))
#先判斷電腦是否有贏的機會
if (playerorder==comOrder):
    print('玩家應該要贏')
else:
    print('玩家會輸')
for i in range(0,n):
    #判斷第 i 堆石頭剩餘數量是否大於 0，若還有石頭則繼續取，否則換下一堆
    while(stone[i] > 0):
        p = 0 #玩家撿石頭的數量
        for j in range(i,n):
            for k in range(0,stone[j]):
                print('o',end='')
```

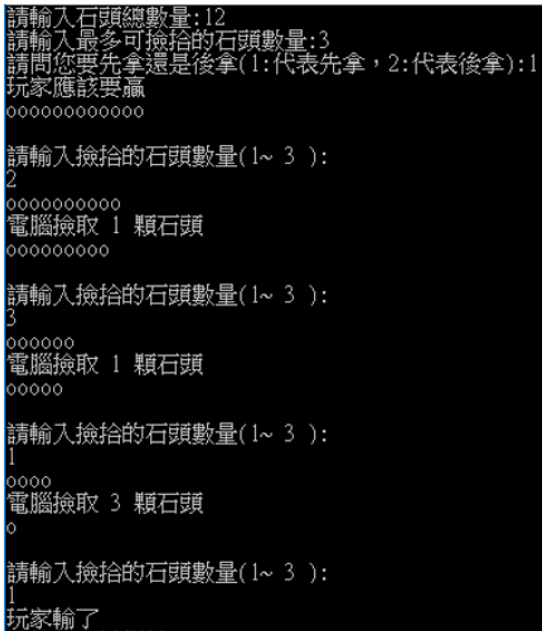
```
            print()
        print()
        #判斷這手是玩家下，還是電腦下
        if playerorder == 1:
            while (p == 0 or p > most):
                print('請輸入撿拾的石頭數量(1~',most,'):')
                p = int(input())
            if stone[i] <= p:
                stone[i] = 0
            else:
                stone[i] = stone[i] - p
            playerorder = 2
            if (stone[n-1] <= 0):
                print('玩家輸了')
        else: #電腦取石頭
            if i < n-1: #判斷是否不是最後一堆石頭
                #每堆石頭取的公式要根據後一堆先取還是後取而有不同
                #若後一堆必須先取，則代表此堆最後一顆必須給玩家撿
                if order[i+1] > 0:
                    comp = (stone[i]-1)%(most+1)
                #若後一堆必須後取，則代表此堆最後一顆必須給電腦撿
                if order[i+1] == 0:
                    comp = (stone[i])%(most+1)
            else:#判斷是否為最後一堆石頭，最後一堆石頭的最後一顆必須給玩家撿
                comp = (stone[i]-1)%(most+1)
            if (comp == 0):#若計算後電腦撿的數量為 0，則讓電腦撿 1 顆石頭
                comp = 1
            #若計算後電腦撿的數量大於 1，且會撿到最後 1 顆石頭，
              則讓電腦少撿 1 顆
            if (stone[i] - comp == 0 and comp > 1):
                comp = comp - 1
            stone[i] = stone[i] - comp
            playerorder = 1
            print('電腦撿取',comp,'顆石頭')
            if (stone[j] <= 0):
                print('玩家贏了')
    print()
os.system("pause")
```

伍、 研究結果

一堆石頭撿拾遊戲執行結果如下：



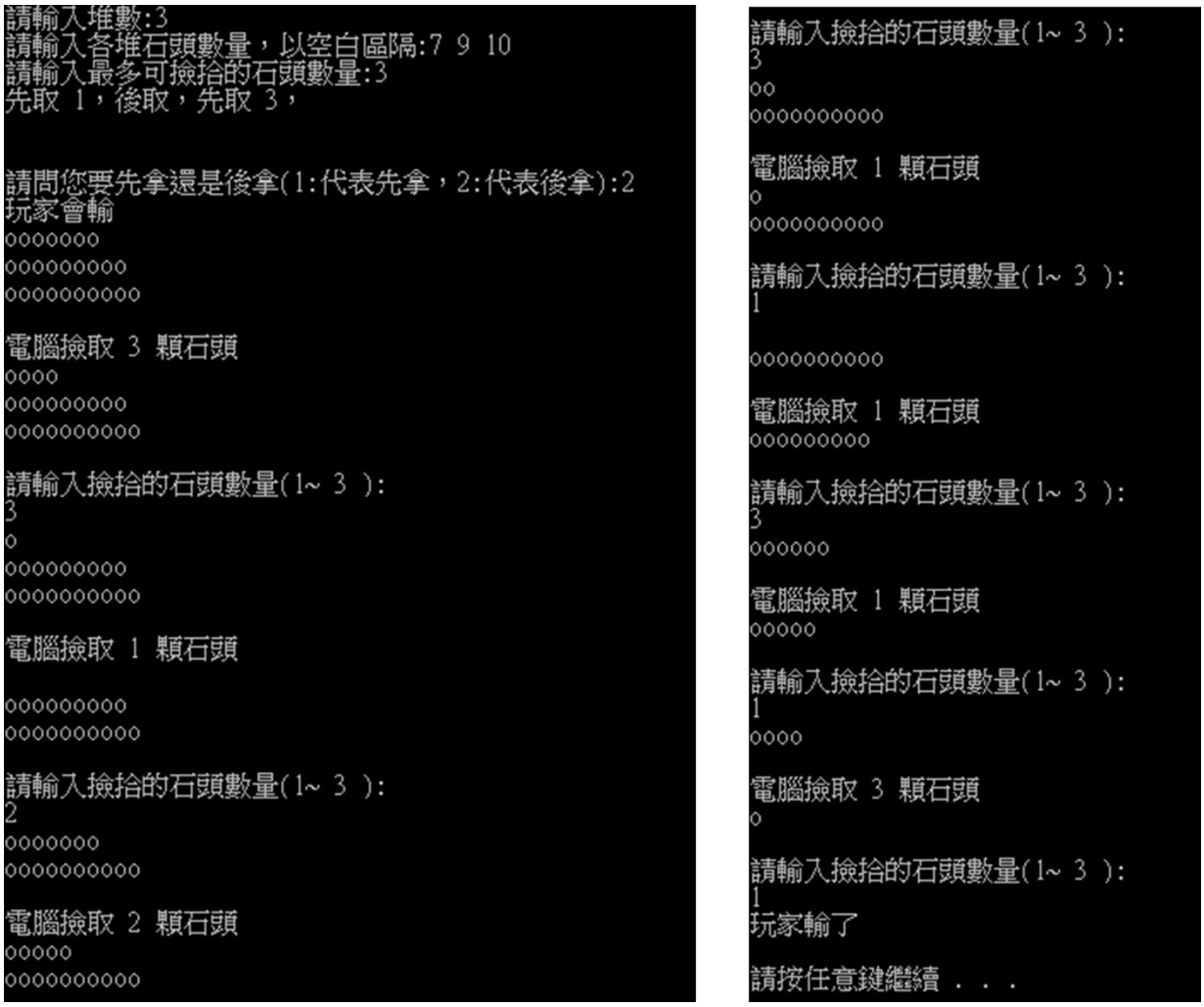
圖七、10 顆石頭遊戲模擬



圖八、12 顆石頭遊戲模擬

經由執行結果圖七可看出電腦一開始沒有印出＂玩家應該要贏＂，代表電腦會贏，最後執行結果也是電腦獲勝，圖八電腦一開始判斷玩家應該會贏，但在取石頭的過程中玩家取的數量錯誤，電腦經過計算取正確的數量後獲勝。

多堆石頭撿拾遊戲執行結果如下：



圖九、多堆石頭遊戲模擬

由執行結果圖九可看出電腦一開始印出＂玩家會輸＂，代表電腦一定會贏，最後執行結果也是玩家輸了，電腦獲勝，程式開發完成。

過半年的推算和程式開發，最終順利開發出可與玩家對戰的智慧 Nim 遊戲，整個遊戲過程清楚的顯示介面，並且每一手撿拾的數量都會根據不同的狀況，算出最有利的撿拾數量。

陸、 結論

撿石頭遊戲在已知的條件下是有一必勝的模式，根據堆數、石頭數量、先取和後取等已知條件，可以推導出先取的人第一手要取幾顆必贏，或是後取才會贏，在一堆石頭的遊戲中玩家可以迅速透過數學模型推算出必贏的條件，但多堆石頭的情況下並不容易計算，但透過程式，電腦在遊戲開始前就可以非常迅速推算出玩家會贏或是電腦會贏，若是電腦會贏則玩家贏的機率為 0%，若是一開始推算玩家會贏，但在遊戲過程中只要其中一次玩家撿的數量錯誤，那就會變成電腦必贏。