

實作題

第 1 題 秘密差

問題描述

將一個十進位正整數的奇數位數的和稱為 A ，偶數位數的和稱為 B ，則 A 與 B 的絕對差值 $|A-B|$ 稱為這個正整數的秘密差。

例如：263541 的奇數位數的和 $A = 6+5+1 = 12$ ，偶數位數的和 $B = 2+3+4 = 9$ ，所以 263541 的秘密差是 $|12-9| = 3$ 。

給定一個十進位正整數 X ，請找出 X 的秘密差。

評分說明

輸入包含若干筆測試資料，每一筆測試資料的執行時間限制(time limit)均為 1 秒，依正確通過測資筆數給分。其中：

第 1 子題組 20 分： X 一定恰好四位數。

第 2 子題組 30 分： X 的位數不超過 9。

第 3 子題組 50 分： X 的位數不超過 1000。

輸入格式

輸入為一行含有一個十進位表示法的正整數 X ，之後是一個換行字元。

輸出格式

請輸出 X 的秘密差 Y (以十進位表示法輸出)，以換行字元結尾。

範例一：輸入

263541

範例一：正確輸出

3

(說明) 263541 的 $A = 6+5+1 = 12$ ， $B = 2+3+4 = 9$ ， $|A-B| = |12-9| = 3$ 。

範例二：輸入

131

範例二：正確輸出

1

(說明) 131 的 $A = 1+1 = 2$ ， $B = 3$ ， $|A-B| = |2-3| = 1$ 。

解題思路

- ◆ 讀取輸入數值字串 X
- ◆ 取得奇數位數值和 A 及取得偶數位數值和 B
 - ◇ 轉換後加總
- ◆ 計算秘密差並列印
 - ◇ $|A-B|$
 - ◇ `abs()` 取絕對值
 - ◇ 秘密差為絕對值，A B 順序不重要

2	6	3	5	4	1
-6	-5	-4	-3	-2	-1
-len(s)					

```
*10603P1.py - D:/APCS/APCS考古題/10603P1.py (3.8.1)*
File Edit Format Run Options Window Help
#讀取輸入數值字串 X
numStr = input()

#取得奇數位數值和 A 及取得偶數位數值和 B
A, B = 0, 0
numLen = len(numStr)
for i in range(-1, -numLen-1, -1):
    if(i%2==0):
        A = A + int(numStr[i])
    else:
        B = B + int(numStr[i])

# 計算秘密差並列印
x = abs(A-B)
print(x)
|
```

Ln: 16 Col: 0

實作題

第 2 題 小群體

問題描述

Q 同學正在學習程式，P 老師出了以下的題目讓他練習。

一群人在一起時經常會形成一個一個的小群體。假設有 N 個人，編號由 0 到 $N-1$ ，每個人都寫下他最好朋友的編號（最好朋友有可能是他自己的編號，如果他自己沒有其他好友），在本題中，每個人的好友編號絕對不會重複，也就是說 0 到 $N-1$ 每個數字都恰好出現一次。

這種好友的關係會形成一些小群體。例如 $N=10$ ，好友編號如下，

	0	1	2	3	4	5	6	7	8	9
好友編號	4	7	2	9	6	0	8	1	5	3

0 的好友是 4，4 的好友是 6，6 的好友是 8，8 的好友是 5，5 的好友是 0，所以 0、4、6、8、和 5 就形成了一個小群體。另外，1 的好友是 7 而且 7 的好友是 1，所以 1 和 7 形成另一個小群體，同理，3 和 9 是一個小群體，而 2 的好友是自己，因此他自己是一個小群體。總而言之，在這個例子裡有 4 個小群體： $\{0,4,6,8,5\}$ 、 $\{1,7\}$ 、 $\{3,9\}$ 、 $\{2\}$ 。本題的問題是：輸入每個人的好友編號，計算出總共有幾個小群體。

Q 同學想了想卻不知如何下手，和藹可親的 P 老師於是給了他以下的提示：如果你從任何一人 x 開始，追蹤他的好友，好友的好友，....，這樣一直下去，一定會形成一個圈回到 x ，這就是一個小群體。如果我們追蹤的過程中把追蹤過的加以標記，很容易知道哪些人已經追蹤過，因此，當一個小群體找到之後，我們再從任何一個還未追蹤過的開始繼續找下一個小群體，直到所有的人都追蹤完畢。

Q 同學聽完之後很順利的完成了作業。

在本題中，你的任務與 Q 同學一樣：給定一群人的好友，請計算出小群體個數。

輸入格式

第一行是一個正整數 N ，說明團體中人數。

第二行依序是 0 的好友編號、1 的好友編號、……、 $N-1$ 的好友編號。共有 N 個數字，包含 0 到 $N-1$ 的每個數字恰好出現一次，數字間會有一個空白隔開。

輸出格式

請輸出小群體的個數。不要有任何多餘的字或空白，並以換行字元結尾。

範例一：輸入

10
4 7 2 9 6 0 8 1 5 3

範例一：正確輸出

4

(說明)

4 個小群體是 $\{0,4,6,8,5\}$, $\{1,7\}$, $\{3,9\}$ 和 $\{2\}$ 。

範例二：輸入

3
0 2 1

範例二：正確輸出

2

(說明)

2 個小群體分別是 $\{0\}$, $\{1,2\}$ 。

評分說明

輸入包含若干筆測試資料，每一筆測試資料的執行時間限制(time limit)均為 1 秒，依正確通過測資筆數給分。其中：

第 1 子題組 20 分， $1 \leq N \leq 100$ ，每一個小群體不超過 2 人。

第 2 子題組 30 分， $1 \leq N \leq 1,000$ ，無其他限制。

第 3 子題組 50 分， $1,001 \leq N \leq 50,000$ ，無其他限制。

解題思路

◆ 讀取

- ◆ 總群體人數 N 及最好朋友字串，切割轉型後存入集合 BF

◆ 小群體 G (二維List / List of Set) 搜尋處理

- ◆ 宣告列表 G ，包含多個集合 LG
- ◆ 索引 i 由 $0-N$ ，驗證 i 是否存在於已知 LG 中，存在則跳離迴圈
- ◆ 不存在則建立小群體 $newLG$ ，加入第一個人(索引 i)
 - 取得好友 $f = BF[i]$ ， f 不等於小群體中第一個人 $newLG[0]$ ，索引 f 加入 $newLG$
 - 取得下一個好友 $f = BF[f]$ ，重複執行直到 f 為小群體中第一個人 $newLG[0]$ 為止
- ◆ $newLG$ 加入列表 G

◆ 輸出 G 中 LG 的組數

```

1  N = int(input())          #讀取總群體人數N
2  BF = list(map(int, input().split())) #最好朋友字串，切割轉型後存入集合BF
3  #print(BF)
4
5  '''小群體G (二維List) 搜尋處理'''
6
7  G = []                    #宣告列表G，包含多個集合LG
8  for i in range(N):        #索引i由0-N，驗證i是否存在於已知LG中
9      for LG in G:
10         if(i in LG):       #i存在於其中一個小群體：跳離迴圈
11             break
12         else:               #i不存在於目前小群體：建立newLG，索引i加入newLG
13             newLG = []
14             newLG.append(i)
15             f = BF[i]       # 取得好友 f = BF[i]
16
17             while f != newLG[0]: # f 不等於newLG[0]，索引 f 加入newLG
18                 newLG.append(f)
19                 f = BF[f]     # 取得下一個好友f = BF[f]，重複執行
20                 #print(newLG)
21             G.append(newLG)   # newLG 加入列表G
22
23 print(len(G))               # 輸出G 中 LG 的組數
24 #print(G)
25

```


實作題

第 3 題 數字龍捲風

問題描述

給定一個 $N \times N$ 的二維陣列，其中 N 是奇數，我們可以從正中間的位置開始，以順時針旋轉的方式走訪每個陣列元素恰好一次。對於給定的陣列內容與起始方向，請輸出走訪順序之內容。下面的例子顯示了 $N=5$ 且第一步往左的走訪順序：

3	→ 4	→ 2	→ 1	→ 4
4	↑ 2	→ 3	→ 8	↓ 9
2	↑ 1	← 9	↓ 5	↓ 6
4	← 2	← 3	↓ 7	↓ 8
1	← 2	← 6	← 4	↓ 3

依此順序輸出陣列內容則可以得到「9123857324243421496834621」。
類似地，如果是第一步向上，則走訪順序如下：

3	↑ 4	→ 2	→ 1	→ 4
4	↑ 2	↑ 3	→ 8	↓ 9
2	↑ 1	↑ 9	↓ 5	↓ 6
4	↑ 2	← 3	↓ 7	↓ 8
1	← 2	← 6	← 4	↓ 3

依此順序輸出陣列內容則可以得到「9385732124214968346214243」。

輸入格式

輸入第一行是整數 N ， N 為奇數且不小於 3。第二行是一個 0~3 的整數代表起始方向，其中 0 代表左、1 代表上、2 代表右、3 代表下。第三行開始 N 行是陣列內容，順序是由上而下，由左至右，陣列的內容為 0~9 的整數，同一行數字中間以一個空白間隔。

輸出格式

請輸出走訪順序的陣列內容，該答案會是一連串的數字，數字之間不要輸出空白，結尾有換行符號。

範例一：輸入

```
5
0
3 4 2 1 4
4 2 3 8 9
2 1 9 5 6
4 2 3 7 8
1 2 6 4 3
```

範例一：正確輸出

```
9123857324243421496834621
```

範例二：輸入

```
3
1
4 1 2
3 0 5
6 7 8
```

範例二：正確輸出

```
012587634
```

評分說明

輸入包含若干筆測試資料，每一筆測試資料的執行時間限制(time limit)均為 1 秒，依正確通過測資筆數給分。其中：

第 1 子題組 20 分， $3 \leq N \leq 5$ ，且起始方向均為向左。

第 2 子題組 80 分， $3 \leq N \leq 49$ ，起始方向無限定。

提示：本題有多種處理方式，其中之一是觀察每次轉向與走的步數。例如，起始方向是向左時，前幾步的走法是：左 1、上 1、右 2、下 2、左 3、上 3、.....一直到出界為止。

解題思路

◆ 建立 $N \times N$ 的二維陣列

◆ 初始行進方向：0>左、1>上、2>右、3>下

◆ 觀察：走的方向及走的步數

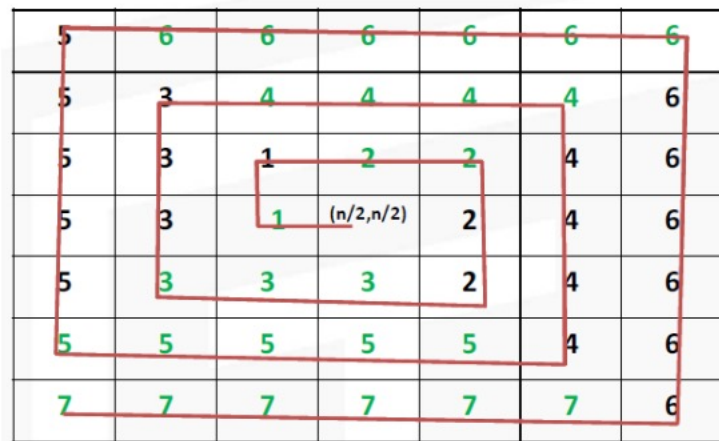
◆ 由 $[N//2, N//2]$ 位置開始

◆ 左1>上1>右2>下2>左3>上3>右4>下4

◆ 步數為由1步開始，每2次改變後遞增

◆ 方向順序為[左,上,右,下]


● $[[0,-1], [-1,0], [0,1], [1,0]]$



```

1 direction=[[-1,-1],[-1,0],[0,1],[1,0]] #方向順序[左、上、右、下]
2 n = int(input())
3 d = int(input()) #初始方向: 0代表左、1代表上、2代表右、3代表下
4
5 data=[]
6 for i in range(n):
7     temp=[]
8     temp=input().split()
9     data.append(temp)
10
11 step = 1
12 stepcounter = 0
13 counter = 1
14 row = n // 2
15 col = n // 2
16 print("%d" %int(data[row][col]),end='')
17 while counter < n * n:
18     for i in range(step):
19         row += direction[d][0]
20         col += direction[d][1]
21         print("%d" %int(data[row][col]),end='')
22         counter=counter+1
23         if counter==n*n: # 超過最大數量跳出迴圈
24             break
25     if counter==n*n:# 超過最大數量跳出迴圈
26         break
27     stepcounter=stepcounter+1#步驟數量+1
28     if stepcounter % 2 == 0: #步驟數量 如是偶數就改變動作次數
29         step=step+1
30     d += 1 #切換到下一個動作
31     d %= 4 #每四筆換到0

```

[illegible]

```
IDLE Shell 3.9.7
File Edit Shell Debug Options Window Help

>>>
= RESTART: D:/APCSClclass3/Solutions/10603/10603P3.py
5
0
3 4 2 1 4
4 2 3 8 9
2 1 9 5 6
4 2 3 7 8
1 2 6 4 3
9123857324243421496834621
>>>
```

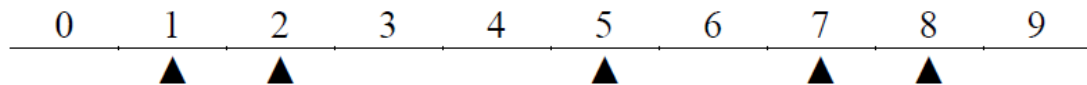
實作題

第 4 題 基地台

問題描述

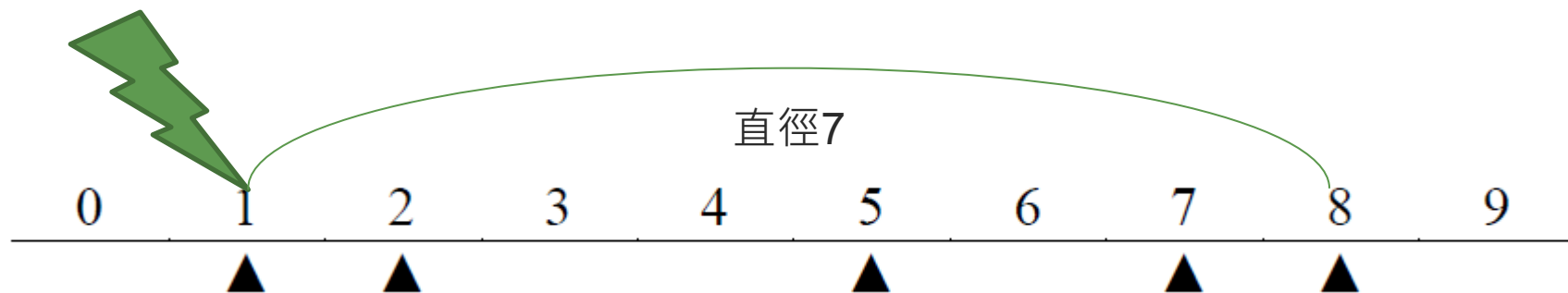
為因應資訊化與數位化的發展趨勢，某市長想要在城市的一些服務點上提供無線網路服務，因此他委託電信公司架設無線基地台。某電信公司負責其中 N 個服務點，這 N 個服務點位在一條筆直的大道上，它們的位置(座標)係以與該大道一端的距離 $P[i]$ 來表示，其中 $i=0\sim N-1$ 。由於設備訂製與維護的因素，每個基地台的服務範圍必須都一樣，當基地台架設後，與此基地台距離不超過 R (稱為基地台的半徑)的服務點都可以使用無線網路服務，也就是說每一個基地台可以服務的範圍是 $D=2R$ (稱為基地台的直徑)。現在電信公司想要計算，如果要架設 K 個基地台，那麼基地台的最小直徑是多少才能使每個服務點都可以得到服務。

基地台架設的地點不一定要在服務點上，最佳的架設地點也不唯一，但本題只要求最小直徑即可。以下是一個 $N=5$ 的例子，五個服務點的座標分別是 1、2、5、7、8。

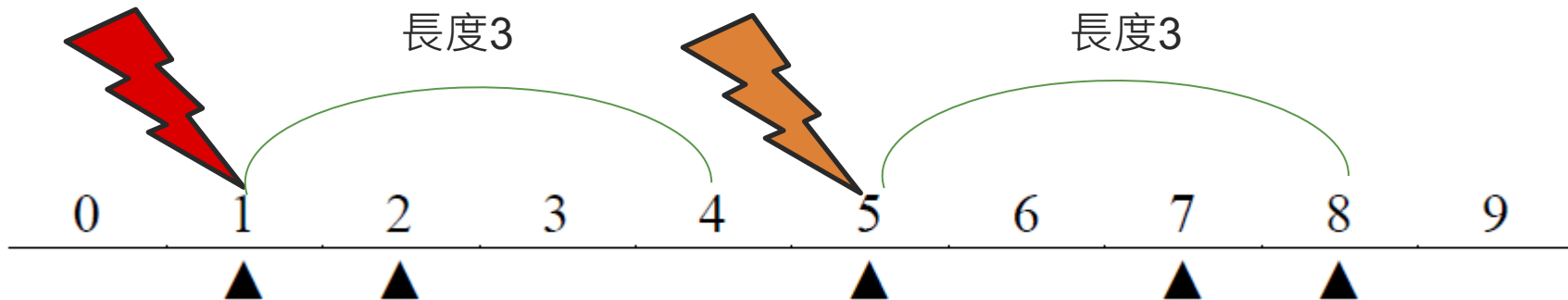


假設 $K=1$ ，最小的直徑是 7，基地台架設在座標 4.5 的位置，所有點與基地台的距離都在半徑 3.5 以內。假設 $K=2$ ，最小的直徑是 3，一個基地台服務座標 1 與 2 的點，另一個基地台服務另外三點。在 $K=3$ 時，直徑只要 1 就足夠了。

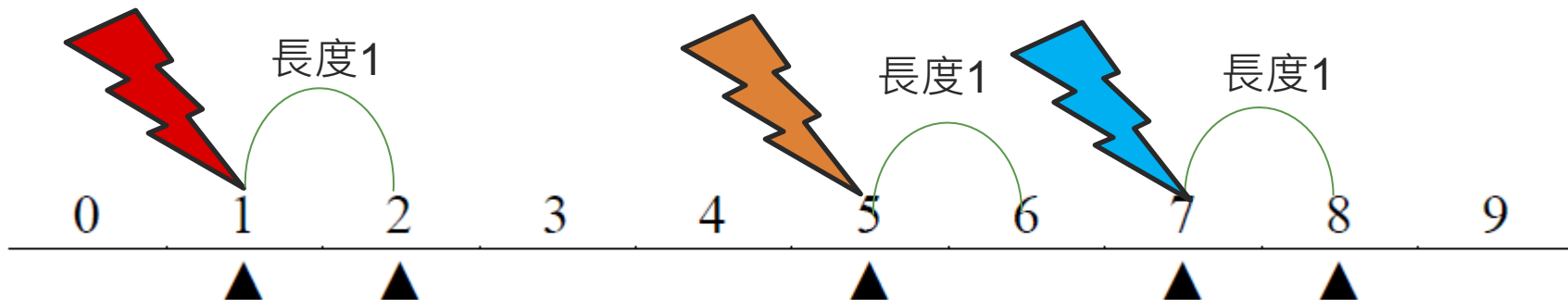
一個基地台



兩個基地台



三個基地台



輸入格式

輸入有兩行。第一行是兩個正整數 N 與 K ，以一個空白間格。第二行 N 個非負整數 $P[0], P[1], \dots, P[N-1]$ 表示 N 個服務點的位置，這些位置彼此之間以一個空白間格。請注意，這 N 個位置並不保證相異也未經過排序。本題中， $K < N$ 且所有座標是整數，因此，所求最小直徑必然是不小於 1 的整數。

輸出格式

輸出最小直徑，不要有任何多餘的字或空白並以換行結尾。

範例一：輸入

5 2
5 1 2 8 7

範例一：正確輸出

3
(說明) 如題目中之說明。

範例二：輸入

5 1
7 5 1 2 8

範例二：正確輸出

7
(說明) 如題目中之說明。

評分說明

輸入包含若干筆測試資料，每一筆測試資料的執行時間限制(time limit)均為 2 秒，依正確通過測資筆數給分。其中：

第 1 子題組 10 分，座標範圍不超過 100， $1 \leq K \leq 2$ ， $K < N \leq 10$ 。

第 2 子題組 20 分，座標範圍不超過 1,000， $1 \leq K < N \leq 100$ 。

第 3 子題組 20 分，座標範圍不超過 1,000,000,000， $1 \leq K < N \leq 500$ 。

第 4 子題組 50 分，座標範圍不超過 1,000,000,000， $1 \leq K < N \leq 50,000$ 。

解題思路 算出基地台最大直徑

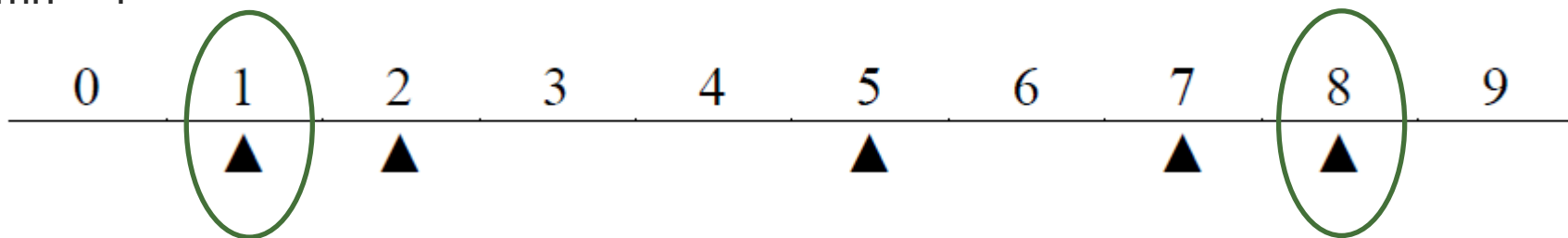
假設基地台數量為2

算出基地台最大直徑

$mx = (\text{最後一個服務位置} - \text{第一個服務位置}) // k + 1$

$mx = (8-1)/2 + 1 = 4$

$mn = 1$



解題思路 用二元搜尋找出第一個最小測試直徑

$mn = 1$

$mx = 4$

$mid = (mn + mx) // 2$

$mid = 2$



解題思路 測試直徑是否包含所有服務

$mn = 1$

$mx = 4$

$mid = (mn + mx) // 2$

- 當isCovered()為True， $mx == mid$ 找更小的直徑
- 當isCovered()為False， $mn == mid$ 找更大的直徑
- 當 $mn == mx$ 時終止 最大值與最小值一樣肯定是最小直徑



解題思路 isCovered

由前面算出最長直徑是4

使用二元搜尋找到第一個直徑是2

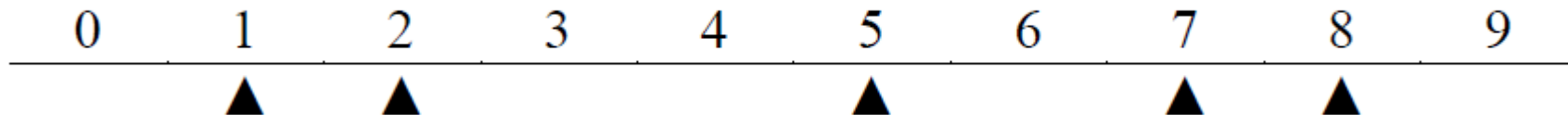
$mn = 1$

$mx = 4$

$mid = (mn + mx) // 2$

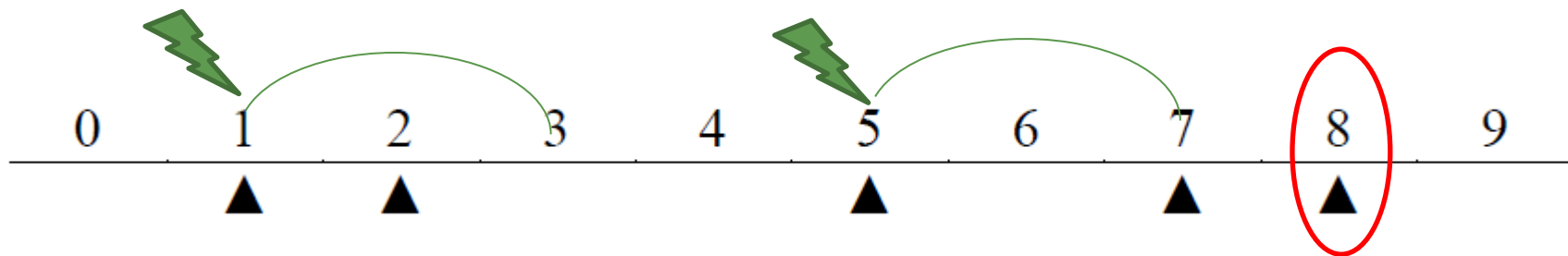
$mid = (1 + 4) / 2$

開始測試直徑2且基地台數量為2是否可以包含所有的服務台



解題思路 isCovered

開始測試直徑2且基地台數量為2是否可以包含所有的服務台



上圖看出來直徑2無法涵蓋到位置為8的基地台所以直徑要放大

解題思路 isCovered

由前面算出最長直徑是4 之前的mid為2

將mn設為 3 (mid+1)

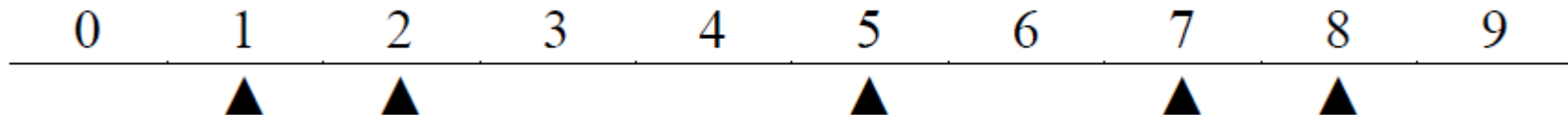
$mx = 4$

$mn = 3$

$mid = (mn + mx) // 2$

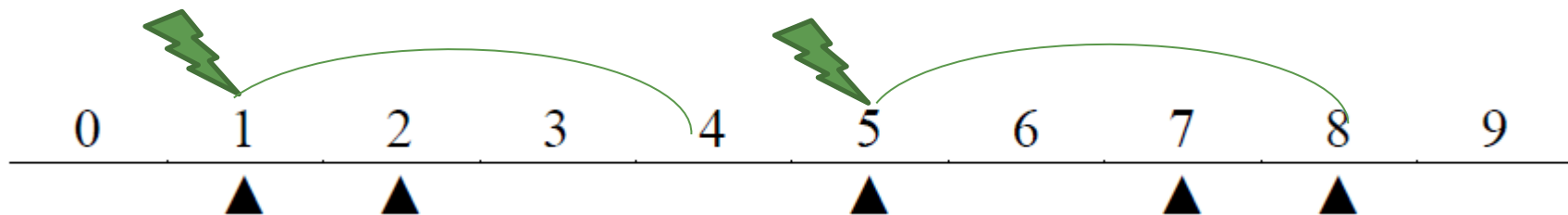
$mid = (3 + 4) // 2$

開始測試直徑3且基地台數量為2是否可以包含所有的服務台



解題思路 isCovered

開始測試直徑3且基地台數量為2是否可以包含所有的服務台



上圖看出來直徑3可涵蓋到所有服務

二元搜尋程式碼

```
N, K = map(int, input().split())
P = list(map(int, input().split()))
P.sort()
```

```
mn = 1
mx = (P[-1]-P[0])//K+1
```

```
while mn <= mx:
    mid = (mn + mx) // 2
    if(isCovered(mid)):
        mx = mid
    else:
        mn = mid + 1
    if mn == mx:
        break
print("%d" %mx)
```

#服務點數目N, 基地台數目K
#服務點位置序列P
#服務點位置排序

#最小直徑
#最大直徑(最小與最大服務站距離)/基地台個數+1
#答案介於這兩數之間, 使用二元搜尋找出答案

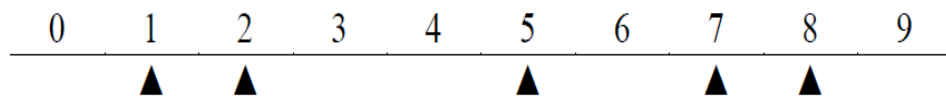
#二分搜尋中間直徑mid
#K 個基地台, 能覆蓋 N 個服務點
#最大直徑改為mid

#最小直徑改為mid+1

isCovered程式碼

'''測試K個基地台服務範圍為d時，
是否可覆蓋所有服務點'''

```
def isCovered(d):  
    global N, K, P  
    nP = 0  
    count = 0  
    pos = 0  
    while pos < N:  
        nP = P[pos] + d;  
        count += 1  
        if count > K:  
            return False;  
        if (P[N-1] <= nP) and (count <= K):  
            return True;  
        pos = pos + 1  
        while P[pos] <= nP:  
            pos += 1
```



#服務點數目N，基地台數目K

#從最前面開始取得服務點

#基地台的下一個涵蓋範圍

#記錄基地台數目

#如果基地台數量已大於K，無法涵蓋，傳回False

#如果涵蓋範圍包含全部，則傳回True

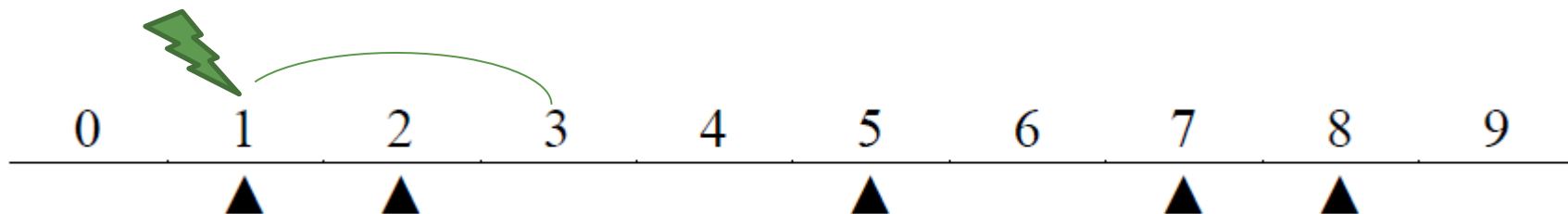
#在基地台數量未大於K時，最後一個基地台被涵蓋

#尋找超過涵蓋範圍的基地台

#基地台仍在涵蓋範圍，pos遞增

解題思路 isCovered 細部分析

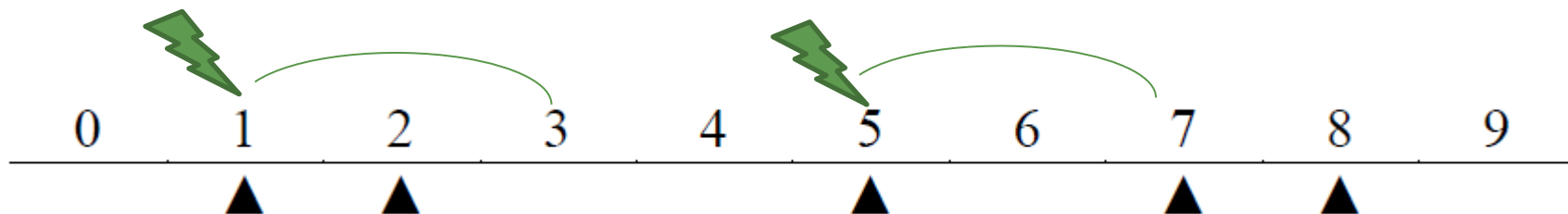
服務數量(N)為5 基地台數量(K)為2 直徑(d)為2 ((4+1) / 2) 第一台加入



1. 目前位置是否超過服務數量
2. 基地台位於位置1 可服務範圍到3
3. 加入基地台數量為1
4. 基地台數量總數為2 加入1台沒超過2(K)
5. 目前基地台是否包含所有服務位置 為false
6. 找下一個涵蓋點找到了2是否小於等於3 true 往下個點找
7. 找下一個涵蓋點找到了5是否小於等於3 false離開迴圈加入新基地台

解題思路 isCovered 細部分析

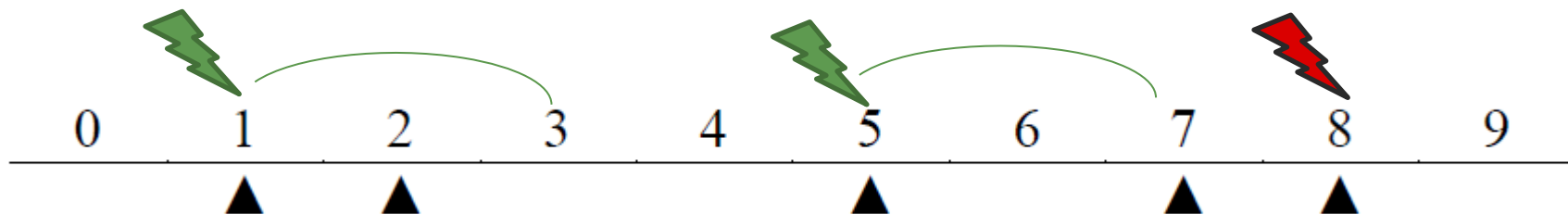
服務數量(N)為5 基地台數量(K)為2 直徑(d)為2 第二台加入



1. 目前位置是否超過服務數量 False
2. 基地台位於位置5 可服務範圍到7
3. 加入基地台數量為2
4. 基地台數量總數為2 加入2台沒超過2(K)
5. 目前基地台是否包含所有服務位置 為false
6. 找下一個涵蓋點找到了7是否小於等於7 true 往下個點找
7. 找下一個涵蓋點找到了8是否小於7 false離開迴圈加入新基地台

解題思路 isCovered 細部分析

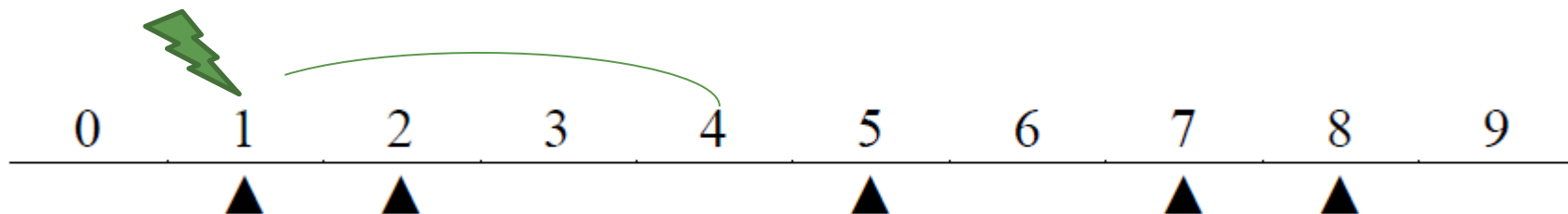
服務數量(N)為5 基地台數量(K)為2 直徑(d)為2 第三台加入



1. 目前位置是否超過服務數量 False
2. 基地台位於位置8 可服務範圍到10
3. 加入基地台數量為3
4. 基地台數量總數為2 加入3基地台超過2 回傳false

解題思路 isCovered 細部分析

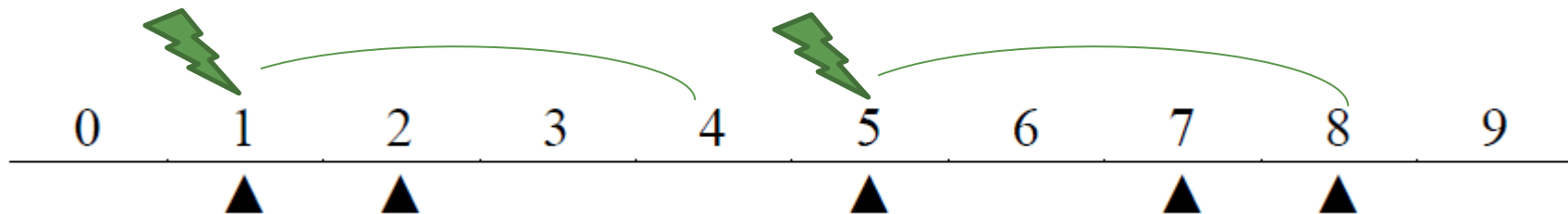
服務數量(N)為5 基地台數量(K)為2 直徑(d)為3 (3 + 4) // 2 第一台加入



1. 目前位置是否超過服務數量 False
2. 基地台位於位置1 可服務範圍到4
3. 加入基地台數量為1
4. 基地台數量總數為2 加入1台沒超過2(K)
5. 目前基地台是否包含所有服務位置 為false
6. 找下一個涵蓋點找到了2是否小於等於4 true 往下個點找
7. 找下一個涵蓋點找到了5是否小於等於4 false離開迴圈加入新基地台

解題思路 isCovered 細部分析

服務數量(N)為5 基地台數量(K)為2 直徑(d)為3 第二台加入



1. 目前位置是否超過服務數量 False
2. 基地台位於位置5 可服務範圍到8
3. 加入基地台數量為2
4. 基地台數量總數為2 加入2台沒超過2(K)
5. 目前基地台是否包含所有服務位置 為True 回傳True

解題思路 二元搜尋部分

1. $\max = 3$ $\min = 3$
2. 所以最小值直徑是3