

二分搜尋

黃 敦 紀

```
1  n = 8 # 總共有幾個數字
2  lookfor = int (input ())
3  sequence = list (map (int, input().split()))
4
5  i = 0
6  while i < n:
7      if sequence [i] == lookfor:
8          break
9      else:
10         i = i + 1
11
12  print (i)
13  # 這段程式還沒有處理 lookfor 不在 sequence 的狀況
```

二分搜尋效能分析

	一個一個依序找 要查幾次？	二分搜尋 要查幾次
8 個	8	8 → 4 → 2 → 1: 3 次 個候選項
16 個	16	16 → 8 → 4 → 2 → 1: 4 次
32 個	32	32 → 16 → 8 → 4 → 2 → 1: 5 次
n 個	n	? 次

```
1  n = 8  # 總共有幾個數字
2  lookfor = int (input ())
3  sequence = list (map (int, input ().split ()))
4
5
6
7
8
9
10
11
12
13
14
15
16
17  if ...:
18      print (...)
19  else: # 如果找不到
20      print (-1)
```

8. 下面哪組資料若依序存入陣列中，將無法直接使用二分搜尋法搜尋資料？

(A) a, e, i, o, u

(B) 3, 1, 4, 5, 9

(C) 10000, 0, -10000

(D) 1, 10, 10, 10, 100

3. 給定一整數陣列 $a[0]$ 、 $a[1]$ 、...、 $a[99]$ 且 $a[k]=3k+1$ ，以 $value=100$ 呼叫以下兩函式，假設函式 **f1** 及 **f2** 之 **while** 迴圈主體分別執行 $n1$ 與 $n2$ 次 (i.e, 計算 **if** 敘述執行次數，不包含 **else if** 敘述)，請問 $n1$ 與 $n2$ 之值為何？ 註： $(low + high)/2$ 只取整數部分。

```
int f1(int a[], int value) {
    int r_value = -1;
    int i = 0;
    while (i < 100) {
        if (a[i] == value) {
            r_value = i;
            break;
        }
        i = i + 1;
    }
    return r_value;
}
```

```
int f2(int a[], int value) {
    int r_value = -1;
    int low = 0, high = 99;
    int mid;
    while (low <= high) {
        mid = (low + high)/2;
        if (a[mid] == value) {
            r_value = mid;
            break;
        }
        else if (a[mid] < value) {
            low = mid + 1;
        }
        else {
            high = mid - 1;
        }
    }
    return r_value;
}
```

3. 給定一整數陣列 $a[0]$ 、 $a[1]$ 、...、 $a[99]$ 且 $a[k]=3k+1$ ，以 $value=100$ 呼叫以下兩函式，假設函式 **f1** 及 **f2** 之 **while** 迴圈主體分別執行 $n1$ 與 $n2$ 次 (i.e, 計算 **if** 敘述執行次數，不包含 **else if** 敘述)，請問 $n1$ 與 $n2$ 之值為何？ 註： $(low + high)/2$ 只取整數部分。

```
int f1(int a[], int value) {
    int r_value = -1;
    int i = 0;
    while (i < 100) {
        if (a[i] == value) {
            r_value = i;
            break;
        }
        i = i + 1;
    }
    return r_value;
}
```

```
int f2(int a[], int value) {
    int r_value = -1;
    int low = 0, high = 99;
    int mid;
    while (low <= high) {
        mid = (low + high)/2;
        if (a[mid] == value) {
            r_value = mid;
            break;
        }
        else if (a[mid] < value) {
            low = mid + 1;
        }
        else {
            high = mid - 1;
        }
    }
    return r_value;
}
```

(A) $n1=33, n2=4$

(B) $n1=33, n2=5$

(C) $n1=34, n2=4$

(D) $n1=34, n2=5$

解 答

3. 給定一整數陣列 $a[0]$ 、 $a[1]$ 、...、 $a[99]$ 且 $a[k]=3k+1$ ，以 $value=100$ 呼叫以下兩函式，假設函式 **f1** 及 **f2** 之 **while** 迴圈主體分別執行 $n1$ 與 $n2$ 次 (i.e, 計算 **if** 敘述執行次數，不包含 **else if** 敘述)，請問 $n1$ 與 $n2$ 之值為何？ 註： $(low + high)/2$ 只取整數部分。

```
int f1(int a[], int value) {
    int r_value = -1;
    int i = 0;
    while (i < 100) {
        if (a[i] == value) {
            r_value = i;
            break;
        }
        i = i + 1;
    }
    return r_value;
}
```

```
int f2(int a[], int value) {
    int r_value = -1;
    int low = 0, high = 99;
    int mid;
    while (low <= high) {
        mid = (low + high)/2;
        if (a[mid] == value) {
            r_value = mid;
            break;
        }
        else if (a[mid] < value) {
            low = mid + 1;
        }
        else {
            high = mid - 1;
        }
    }
    return r_value;
}
```

- (A) $n1=33, n2=4$
- (B) $n1=33, n2=5$
- (C) $n1=34, n2=4$
- (D) $n1=34, n2=5$

「資料結構」

- 資料在記憶體空間怎麼根據個別的意義和彼此的關係安放

「演、運) 算法」

- 程式怎麼設計

什麼事先做、什麼後做、如何抓取、改變、存放資料，
根據什麼樣的情況做幾次

這一連串的流程設計

算法的分析

$O(\dots)$

台中女中程式解題系統

Green Judge, An Online Judge System for TCGS

| [Login](#) | [Register](#)

[HOME](#) | [Problems](#) | [Status](#) | [Rank](#) | [Forum](#) | [Contest](#)

a045: 質數判斷

對於一個正整數 N 來說，如果它的因數只有 1 和 N 本身，沒有其他的因數，則我們稱它為「質數」。現在給你一個正整數，請你判斷它是不是質數。

Input :

輸入一個正整數 N 。

Output :

若 N 為質數，則輸出 YES，否則輸出 NO。

Sample Input :

輸入1:

5

輸入2:

9

Sample Output

輸出1:

YES

輸出2:

NO

Hint :

可用 `sqrt(n)` 求 \sqrt{n} 的值。

請 `#include <cmath>` 或 `#include <math.h>`

因數檢查從 2 到根號 n 就可以了

如果 n 有一個因數比根號 n 大，那這因數一定會是某個小於根號 n 的數 m 的倍數，即 m 也是 n 的因數，而 m 在根號 n 之前就檢查過了

例如：100 的因數 50, 20, 10, 5, 2

```
6 // Copyright © 2020 Elton Huang. All rights reserved.
7 //
8
9 #include <iostream>
10 using namespace std;
11
12 bool prime (int n) {
13     // 特例
14     if (n <= 1)
15         return false;
16     else if (n <= 3)
17         return true;
18
19     // 檢查  $6k + 2$ ,  $6k + 4$ ,  $6k + 3$ ,  $6k + 0$ 
20     else if (n % 2 == 0 || n % 3 == 0)
21         return false;
22
23     else {
24         // 檢查  $6k + 5$ ,  $6k + 1$ 
25         for (int i = 5; i*i <= n; i = i + 6) // i <= sqrt (n)
26             if (n % i == 0 || n % (i + 2) == 0)
27                 return false;
28         // 也沒有  $6k + 5$ ,  $6k + 1$  形式的因數
29         return true;
30     }
31 }
32
33 int main () {
34     int n;
35     while (cin >> n) // 處理連續測資
36         cout << (prime (n) ? "YES" : "NO") << endl; // if (prime (n))
37                                                     // cout << "YES" << endl;
38                                                     // else
39                                                     // cout << "NO" << endl;
40     return 0;
41 }
```

問題：判斷質數	時間複雜度
解法：檢查因數 從 2 到 $n - 1$	$O(n)$
解法：檢查因數從 2 到 $\text{sqrt}(n)$	$O(\text{sqrt}(n))$

問題：在排好序的陣列搜尋元素	時間複雜度
解法：循序搜尋	$O(n)$
解法：二分搜尋	$O(\log(n))$

不過，要能二分搜尋之前要先將資料排序，
排序的時間複雜度是 $O(n \times \log_2 n)$

	一個一個依序找 要查幾次？	二分搜尋 要查幾次
n 個	n	$\log_2 n$ (次)
一次	<p>假設有 1024 筆資料 $n = 1024 = 2^{10}$</p> <p>而查找的操作需要做 4096 次 即 2^{12} 次</p>	
8 個		
1024 個		
	1024 時間單位	100 時間單位

$$n = 1024 = 2^{10}$$

$$4096 = 2^{12} \text{ 次操作}$$

排序的時間複雜度是 $O(n \times \log_2 n)$

	循序搜尋	二分搜尋
前置作業	0 (不需前置作業)	排序一次 $1024 \times \log_2 1024 == 2^{10} \times 10 < 2^{14}$ 假設一次排序需要 12 個時間單位 總共需要 $< 2^{14} \times 12$ 個時間單位
對於 1024 筆資料， 最糟比對幾次？ 假設比對 1 次需要 1024 筆	$1024 == 2^{10}$ 1 時間單位 2^{10} 時間單位	$\log_2 1024 == 10 < 2^4$ 5 個時間單位 $< 2^4 \times 5$ 時間單位
但是搜尋的操作共 4096 = 2^{12} 次	共 4096×2^{10} 時間單位 == 2^{22} 時間單位	共 $< 4096 \times 2^4 \times 5$ 時間單位 == $2^{16} \times 5$ 時間單位
加上前置作業	共需 2^{22} 時間單位	共需 $< 2^{14} \times 12 + 2^{16} \times 5$ 時間單位

$$n = 1024 = 2^{10}$$

$$4096 = 2^{12} \text{ 次操作}$$

排序的時間複雜度是 $O(n \times \log_2 n)$

	循序搜尋	二分搜尋
前置作業	0 (不需前置作業)	排序一次 $1024 \times \log_2 1024 = 2^{10} \times 10 < 2^{14}$ 假設一次排序需要 12 個時間單位 總共需要 $< 2^{14} \times 12$ 個時間單位
對於 1024 筆資料， 最糟比對幾次？ 假設比對 1 次需要 1024 次	$1024 = 2^{10}$ 1 時間單位 2^{10} 時間單位	$\log_2 1024 = 10 < 2^4$ 5 個時間單位 $< 2^4 \times 5$ 時間單位
但是搜尋的操作共 4096 = 2^{12} 次	共 4096×2^{10} 時間單位 == 2^{22} 時間單位	共 $< 4096 \times 2^4 \times 5$ 時間單位 == $2^{16} \times 5$ 時間單位
加上前置作業	共需 2^{22} 時間單位	共需 $< 2^{14} \times 12 + 2^{16} \times 5$ 時間單位

7. 若 n 為正整數，右側程式三個迴圈執行完畢後 a 值將為何？

- (A) $n(n+1)/2$
- (B) $n^3/2$
- (C) $n(n-1)/2$
- (D) $n^2(n+1)/2$

```
int a=0, n;  
...  
for (int i=1; i<=n; i=i+1)  
    for (int j=i; j<=n; j=j+1)  
        for (int k=1; k<=n; k=k+1)  
            a = a + 1;
```

17. 給定右側函式 $F()$ ， $F()$ 執行完所回傳的 x 值為何？

- (A) $n(n+1) \sqrt{\lceil \log_2 n \rceil}$
- (B) $n^2(n+1)/2$
- (C) $n(n+1) \lceil \log_2 n + 1 \rceil / 2$
- (D) $n(n+1)/2$

```
int F (int n) {  
    int x = 0;  
    for (int i=1; i<=n; i=i+1)  
        for (int j=i; j<=n; j=j+1)  
            for (int k=1; k<=n; k=k*2)  
                x = x + 1;  
    return x;  
}
```