交換與排序

> https://leetcode-cn.com/problems/reshape-the-matrix/
>
> 複習 二維矩陣

實作 max() 變化版

輸入 一列數列

輸出 最大的整數,索引值

ex

1 9 8 67 32

67 3

-1 -9 -8 -67 -32

儲存最大值的變數 初始值?

(1) 設極值

(1) float("inf") 無限大 -float("inf")無限小

(2) 10**9 -10**9

(2) 設置串列的第一項

```
a = list(map(int,input().split()))
count = 0
b = 0
for i in range(len(a)):
    if a[i]>count:
        count = a[i]
        b = i
print(count,b)
```

> 最遙近的距離
>
> **判斷時間複雜度**
>
> (1) **暴力法**
>
> (2) **先 sort ， 再處理　O(nlogn)**
>
> 1. 串列.sort()　改動原本的串列
> 2. sorted(串列)　不會改動原本的串列
>
> 時間複雜度 ?
>
> > **ex.**
> >
> > n**2　n!
> >
> > 2 3
> >
> > n**2 > n!
> >
> > n >= 4
> >
> > n**2 < n!
>
> 不理會 常數次數
>
> 只在意 變數次數
>
> **為什麼要學會判斷時間複雜度?**
>
> (1) 時間越短越好，能知道如何改善程式碼
>
> (2) 由時間複雜來判斷，寫出的程式碼有沒有符合出題人的規定
>
> 由時間複雜來判斷，出題人要考的演算法 和 資料結構
>
> EX. len(a) == 1000 => 1000*1000 => 10**6
>
> len(a) == 10000 => 10000*10000 => 10**8
>
> 今天 oj 有時間上的規定， 原則上 c c++ : 1s， py : 3s
>
> c/c++/py : 10**6~7　基準判斷

(1) O(n**2) 一組測資

```
n=int(input())

for i in range(n):
    a=input().split()   #O(1)
    a=list(map(int,a))  #O(1)
    c=float("inf")      #O(1)

    for j in range(len(a)): #O(n)
        for i in range(j+1,len(a)): #O(n)   O(n**2)
            if abs(a[i]-a[j])<c: #O(1)
                c=a[i]-a[j] #O(1)

    print(c)
```

(2) O(n**2)

```
n = int(input())

for i in range(n):
    a = list(map(int,input().split()))
    c = 10**34000

    a.sort()

    for j in range(len(a)-1):   # O(n)
        if a[j+1]-a[j]<c:
            c = a[j+1]-a[j]
    print(c)
```

> Bubble sort
>
> 時間複雜度 O(n**2)
>
> 排序數列
>
> ex 產生隨機數列

```
import random   #匯入 模組
a = []
for i in range(5):
    a.append(random.randint(1,30))  #random.randint(1,30) 產生1 ~ 30 數字

print(a)
```

```
import random
a = []
for i in range(5):
    a.append(random.randint(1,30))
print(sorted(a))    #用sorted()來排序 跟 下面寫的程式碼 做比對
for k in range(len(a)-1): #len(a) * (len(a) - 1)  O(n*(n-1)) => O(n**2)
    for j in range(len(a)-1-k):
        if a[j]>a[j+1]:
            temp = a[j]
            a[j] = a[j+1]
            a[j+1]=temp
print(a)
```