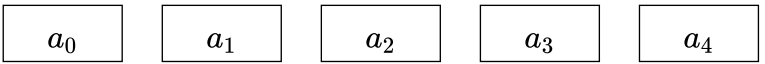
(一維)陣列; Array

資訊之芽語法班 2015 suhorng

陣列

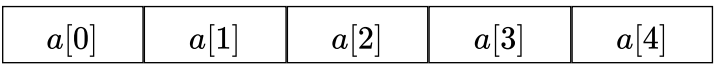
- 一群 **int** 變數

```
int a0, a1, a2, a3, a4;
```



- V.S. 一個 **int** 陣列

```
int a[5];  // 可以用的索引是 0, 1, 2, 3, 4
```

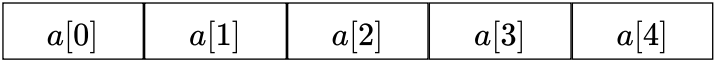


a

(只有一個, 不像上面有 5 個)

  - 符號 $a[i]$ 表示陣列 “a” 的第 “i” 的格子, **從 0 開始**

陣列 -- 有什麼差別?



- 可以宣告很多 (1000000 個!)

```
// help!!
int connect_0, connect_1, connect_2, ......
// ok
int connect_id[5000];
```

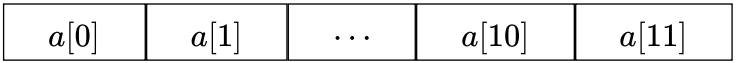
- 可以用**變數**存取, 甚至索引值可以用**算**的

  for (**int** $i = 0;\ i < 10;\ i = i + 1$) {

  - i 變數當索引: `std::cout << connect_id[i];`
  - 令 array 第 i 格數值為 i^2: `sqr[i] = i * i;`

  }

語法(宣告)



- 宣告陣列: $type\ \ name[size]$;

```
// 宣告一個 int 陣列, 長度是 2015
int my_array[2015];
```

- 宣告兼初始化: $type\ \ name[\ size\ ] = \{v_0, v_1, \ldots, v_k\}$;

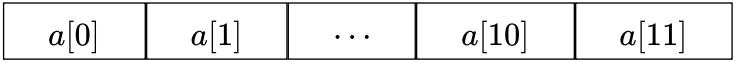
  (必須滿足 $k \leq size$)

```
// 初始化前三個元素, 後面會自動填 0
int arr_zero[100] = {1,2,3};
```

- 自動計算長度(為 n): $type\ \ name[\ ] = \{v_0, v_1, \ldots, v_{n-1}\}$;

```
// my_array 的長度將是 4, 由編譯器自動計算出來
int arr2015[] = {2,0,1,5};
```

語法(存取)



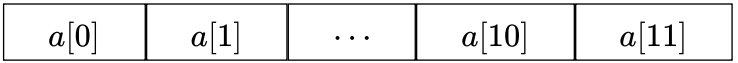
- 一個陣列 $\mathbf{int}\ a[12];$ 可以類比於 12 個**獨立**變數 $a_0, a_1, \ldots, a_{11}$ 的集合體
- 陣列 “a” 的第 “i” 的元素(~~太極~~**從 0 開始)**表達法為

$$a[i]\ (\Longleftrightarrow \text{意指 } a_i)$$

  - “i” 稱為「索引」(index)

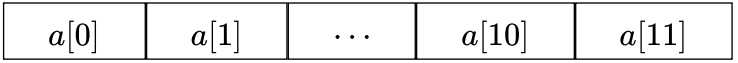
```
// 運算
3 * a[n] + 1

// 輸出
std::cout << a[50] << "\n";

// 寫入
a[i*2 + 1] = i;
```

語法 -- 中括弧?



- 需要注意**中括弧+數字**出現在兩種地方, 它們**不相關**、意思也**不一樣**
  - 宣告: **int** $sequence[57]$;
  - 存取: $sequence[i]$、$sequence[0]$、.....
- **宣告**時, 前面會有型別($type$例如 **int**、**char**), 中括弧內的數字 $[57]$ 代表陣列 $sequence$ 的**長度**
- **存取**時, 它是運算式的一部分, 中括弧內的數字 $[i]$、$[0]$ 是**索引**, 代表陣列 $sequence$ 這一排格子中的第幾個, 從 0 開始.

栗子




```cpp
#include <iostream>

int main() {
  // 紀錄每個月有幾天
  int days[12] = { 31, 28, 31, 30, 31, 30,  // Jan-Jun
                   31, 31, 30, 31, 30, 31}; // Jul-Dec

  // 讀入年份與月份
  int year, month;
  std::cin >> year >> month;

  if (year%400==0 || (year%4==0 && year%100!=0))
    days[1] = days[1] + 1;  // 閏年: 二月(索引為 1)多一天

  // 第 i 個月的索引是 i-1
  std::cout << "There are " << days[month-1] << " days.\n";
  return 0;
}
```

陣列的型別(type)

- 整數 ⇒ **int**
- 53 個只的陣列 ⇒ $\mathbf{int}[53]$

```cpp
using array_of_53_ints = int[53];

// 等同 int arr[53];
array_of_53_ints arr;
// ^型別, int[53]  ^變數名稱

arr[0] = 123;
```

- 同理: 可以把 **int** 換成其他型別, 如 **bool**、**double**、**char**

栗子二

- 輸入: 第一行是 n, 滿足 $1 \leq n \leq 1000$. 接下來有 n 個整數 $0 \leq a_0, a_1, \ldots, a_{n-1} < n$
- 輸出: 請輸出 n 行, 第 $i+1$ 輸出 a_{a_i}
- 舉個栗子來說, 若輸入

```
5
1 2 3 0 4
```

- 應該輸出

```
2
3
0
1
4
```

- 說明: 如以上的例子中 $a_0 = 1$, 因此第 1 行是 $a_{a_0} = a_1 = 2$.

栗子二

確定要看解法?

栗子二

確定要看解法?

```cpp
#include <iostream>

int main() {
  int n, a[1000];
  std::cin >> n;

  for (int i = 0; i < n; i = i+1)
    std::cin >> a[i];

  for (int i = 0; i < n; i = i+1)
    std::cout << a[a[i]] << "\n";

  return 0;
}
```

要注意的地方

- 長度為 n 的陣列, 可以用的格子索引是 $0, \ldots, n-1$. 使用這以外的索引是 undefined behavior (官方說法).
- 不要在 $main$ (或任何函式)內開太大的變數

```
// bad style
int main() {
  int big_array[5000000];
}
```

  - 以目前來說, 大陣列($10^6, 10^7$ 量級)只能先開成全域變數

- 以下這種在 C++ 中是非標準寫法, 不建議

```
int n;
std::cin >> n;

int variable_length_array[n];
```