複習REVIEW

日期：2014/11/26

講師：侯均靜

CH5: 陣列 & 字串

Array & String

日期：2014/11/26

講師：侯均靜

如果你的程式碼長這樣.....

```
int a, b, c, d, e, f, g, h, i, j, k, l, m,
    n, o, p, q, r, s, t ,u ,v, w, x ,y ,z,
    aa, ab, ac ,ad, ae, af, ag, ah, ai, aj,
    ......
```

一定有更好的方法.....

打死我也不想再看到那種程式碼第二次.....

陣列 Array

什麼是Array?

- **陣列**是電腦程式語言上，對於「Array」的中文稱呼。 -- Wikipedia (ZH)

- In computer science, **an array is a data structure consisting of a collection of elements** (values or variables), each identified by at least one array index or key. An array is stored so that the position of each element can be computed from its index tuple by a mathematical formula. The simplest type of data structure is a linear array, also called one-dimensional array. -- Wikipedia (EN)

什麼是Array?

- 當你有一堆相似或具有相同意義的資料要進行儲存，為了避免大量的變數宣告，我們可以使用陣列來做儲存。
- 「陣列」如同櫃子，依照你所定義的類型，放置你所想要放的東西，例如：「書櫃」會放置一堆「書籍」。

宣告

- 原本的變數宣告長這樣：

```
int a, b, c, d, e;
float f, g, h, i, j;
double k, l, m, n, o;
char p, q, r, s, t;
```

宣告

- 稍微動點手腳變成：

```
int num[5];
float fnum[5];
double dnum[5];
char ch[5];
```

宣告

- 陣列的宣告方式如下：
  - VariableType VariableName [ArraySize];

- 注意：ArraySize建議不要超過10000，雖然說超過理論上不會錯，但會有潛在的問題，如果要宣告超過10000個建議改用malloc，這部分我們之後會教。

存取

- 很有趣的是，陣列的存取跟一開始定義的大小有點不一樣。
- C語言的陣列定址從「 0 」開始，**換言之當陣列大小宣告為 n 的時候，可存取的範圍是 0~n-1**。

存取

- 各位可以透過VariableName[Index]來存取陣列元素，其中index必須在0到ArraySize -1 之間。

```
num[0] = 100;
fnum[1] = 200.0;
dnum[2] = 300.0;
ch[3] = 'A';
```

```c
scanf("%d", &num[0]);
scanf("%f", &fnum[0]);
scanf("%lf", &dnum[0]);
scanf("%c", &ch[0]);

printf("%d", num[0]);
printf("%f", fnum[0]);
printf("%lf", dnum[0]);
printf("%c", ch[0]);
```

存取

宣告時初始化

- 宣告時我們就可以給定初始值，宣告方式為：
- VariableType VariableName[ArraySize] = {Element1, Element2, …};

```
int num[5] = {0, 1, 2, 3, 4};
float fnum[5] = {0, 1.0, 2.0, 3.0, 4.0};
double dnum[5] = {1.2, 2.4, 3.6, 4.8, 6.0};
char ch[5] = {'A', 'B', 'C', 'D', 'E'};
```

宣告時初始化

- 那如果我都已經知道後面要給他什麼東西了，那我們可不可以省略 ArraySize？
- 答案是：可以。

宣告時初始化

```
int num[] = {0, 1, 2, 3, 4};
float fnum[] = {0, 1.0, 2.0, 3.0, 4.0};
double dnum[] = {1.2, 2.4, 3.6, 4.8, 6.0};
char ch[] = {'A', 'B', 'C', 'D', 'E'};
```

宣告時初始化

宣告時初始化

- 但是這樣寫都是錯的：

```
// It's wrong!!
int num;
float fnum;
double dnum;
char ch;
```

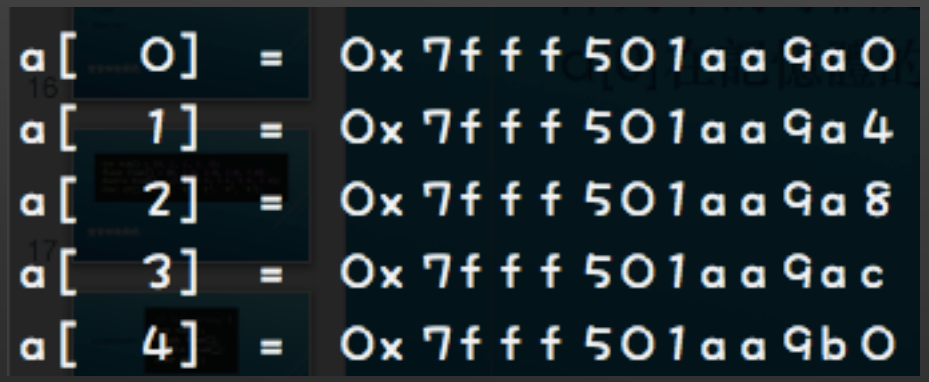
練習

- 宣告一個陣列，存放 0 ~ 10，並且分別印出來。

```
a[  0] =  0
a[  1] =  1
a[  2] =  2
a[  3] =  3
a[  4] =  4
a[  5] =  5
a[  6] =  6
a[  7] =  7
a[  8] =  8
a[  9] =  9
a[ 10] =  10
```

元素間的關係

- 陣列中的每個元素都有自己的地址，而這些地址是連續的，換言之，如果a[0]在記憶體的1位置，那a[1]就會在2位置。



- 這邊因為a陣列是int型態，所以a[1] - a[0]是4 byte。

如果你的程式碼長這樣.....

```
printf("a[0] = %d\n", num[0]);
printf("a[1] = %d\n", num[1]);
printf("a[2] = %d\n", num[2]);
printf("a[3] = %d\n", num[3]);
printf("a[4] = %d\n", num[4]);
printf("a[5] = %d\n", num[5]);
printf("a[6] = %d\n", num[6]);
printf("a[7] = %d\n", num[7]);
printf("a[8] = %d\n", num[8]);
printf("a[9] = %d\n", num[9]);
printf("a[10] = %d\n", num[10]);
```

一定有更好的方法.....

打死我也不想再看到那種程式碼第二次.....

用迴圈來存取

- 我們剛剛有提到每個元素都有一個Index，那我們就可以透過Index來做存取的動作。

```
for(index = 0; index < 11; index++){
    printf("a[%2d] = %2d\n", index, num[index]);
}
```

- 當然你的index可以簡寫成 i ，或 ..... 任何你想要取的名字。

陣列元素運算

- 陣列元素的運算等同變數，也就是說int陣列的元素的運算方式等同int變數。

```
a[0] = 10;
a[1] = a[0] + 1;
a[2] = a[1] - 1;
a[3] = a[2] * 2;
a[4] = a[3] / 2;
a[5] = a[0] % 5;
```

二維陣列

- 你剛剛看到的都是所謂的一維陣列，接下來是二維陣列。
- 一維陣列如果是一條線，那二維陣列就是一個面了。

宣告

- int num[2][3];

num	0	1	2
0	num[0][0]	num[0][1]	num[0][2]
1	num[1][0]	num[1][1]	num[1][2]

- 一元多次方程式：請用二維陣列儲存一個一元多次方程式並輸出~

方程式	*0*	*1*	*2*
0 (係數)	*1*	*2*	*1*
1 (次方)	*2*	*1*	*0*

練習

$$X^2+2x+1$$

多維陣列

- 一般來講我們很少很少用到二維以上的陣列。
- 宣告時後面有幾個[]就是幾維陣列。

字串 String

什麼是字串？

- “This is an apple.”
- “なつ　どらぐにる”
- “Shiba Tatsuya”
- “Kirito”
- “Asuna”
- “OXABC”

- 只要大於一個字以上就算是一個字串，儘管他可能無意義。

- 剛剛我們似乎有飄過一串東西。
- char ch[5];

- 如果我們把字串拆成一個字一個字去存的話？

如何儲存字串？

字元陣列

- 我們透過char 陣列來達成字串的儲存。
- 假設一個字串有十個字，那我們需要 10 + 1個空間來存。
- 每個字串的最後面會有一個不會顯示出來的字元 “ \0 “，負責告訴系統字串結束了。

宣告時初始化

- 方法有兩種：

```
char str1[] = {'T', 'a', 't', 's', 'u', 'y', 'a', '\0'};
char str2[] = "Tatsuya";
```

字串輸入輸出

- 記住，字串是一個字元陣列，因此再輸入跟輸出也會有點不一樣。

```
char str[256];
scanf("%s", str);
printf("%s", str);
```

- 你問我為什麼沒有“ & “？
- 因為字元陣列傳遞時給的是陣列開頭的「地址」，因此不需要再去取址了。

練習

- 請撰寫一個程式，不管我輸入什麼，都輸出Hello! + 那個字串。

```
C
Hello! C
Java
Hello! Java
Python
Hello! Python
Apple
Hello! Apple
Egg
Hello! Egg
Vongola
Hello! Vongola
Tatsuya
Hello! Tatsuya
```

字串其他運算

- 字串不可以使用 +-*/% 這幾個符號，那要怎麼進行字串的其他處理呢？
- ……
- 啊！<string.h>

字串函式庫

- #include <string.h>
- string.h裡面定義了很多跟字串有關的函式
  - strcpy
  - strcat
  - strcmp
  - ……等等

字串複製

- strcpy
  - char *strcpy(char *destination, const char *source);
  - 將source中的字串複製到destination裡面。
- strncpy
  - char *strncpy(char *destination, const char *source, size_t num);
  - 將source中的字串複製num個到destination。

字串串聯

- strcat
  - char *strcat(char *destination, const char *source);
  - 將source中的字串接到destination最後面。
- strncat
  - char *strncat(char *destination, const char *source, size_t num);
  - 將source中的字串接num個到destination的最後面。

字串比對

- strcmp
  - char *strcmp(char *str1, const char *str2);
  - 檢查str1跟str2兩個字串是否一致。
- strncmp
  - char *strncmp(char *str1, const char *str2, size_t num);
  - 檢查str1跟str2兩個字串前num個字是否一致。

字串搜尋

- strchr
  - char *strchr(char *str, int character);
  - 回傳character出現在str中的第一個位址。
- strrchr
  - char *strchr(char *str, int character);
  - 回傳character出現在str中的最後一個位址。
- strstr
  - char *strstr(char *str1, char *str2);
  - 回傳str2在str中出現的第一個位址。

字串長度

- strlen
  - size_t strlen(const char *str);
  - 計算字串中有幾個字。
  - 不包括 “ \0 “

http://www.cplusplus.com/reference/cstring/

其他字串函式

練習

- 很抱歉，沒有。