Chpt.4

C 語言—陣列及字串

10/16 系程

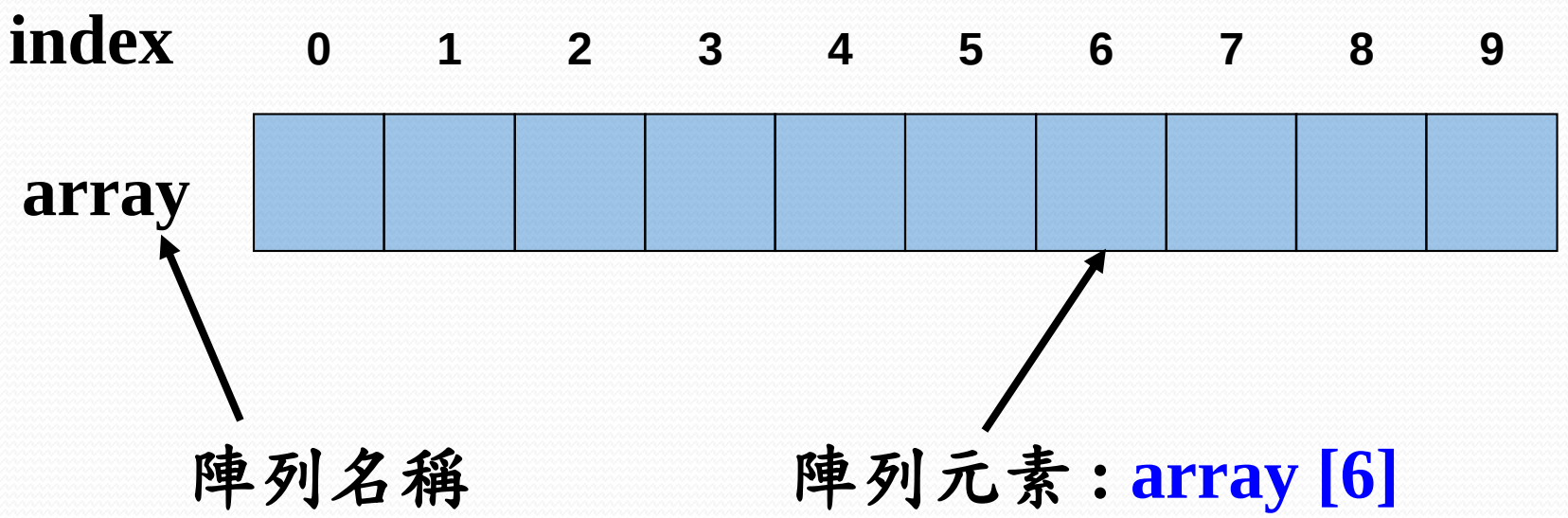
主講人:荊輔翔

陣列

- 概論

- 陣列是一個具有索引(index)性質的連續資料儲存空間集合。
- 陣列中每一個資料儲存空間稱之為陣列元素(array element)；它們都具有相同的資料名稱、資料型態、及空間大小；但存取它們時則須藉由索引(或稱註標)來區別辨識。
- 索引代表資料在陣列中的相對位址(其計數由0開始，其餘累加類推)，且須由中括號[ ]涵蓋之。

陣列



陣列—宣告/使用

- 一維陣列宣告格式：
  - <資料型態> <陣列名稱> [陣列大小] = {初始值設定}；

- **int array[100];**
- **float a[50];**

陣列—宣告/使用

- 為什要用陣列?

EX1. 請設計一個程式，讓使用者輸入10筆資料並印出。

EX2. 請設計一個程式，讓使用者輸入20筆資料並印出。

EX3. 請設計一個程式，讓使用者輸入100筆資料並印出。

陣列—宣告/使用

- EX 1.

```
main() {
        int x0, x1, x2, x3, x4, x5, x6, x7, x8, x9;

        scanf(“%d %d %d %d %d ”,&x0,&x1,&x2,&x3,&x4);
       scanf("%d %d %d %d %d", &x5,&x6,&x7,&x8,&x9);
        printf("%d %d %d %d %d ", x9, x8, x7, x6, x5);
        printf("%d %d %d %d %d ", x4, x3, x2, x1, x0);
}
```

陣列—宣告/使用

- EX 3.

```
main() {
        int x[100];
        int i;
        for ( i = 0; i < 100 ; i++ ){
                scanf(“ %d”, &x[i]);
        }
        for ( i = 100; i >= 0 ; i-- ){
                printf("%d ", x[i]);
        }
}
```

隨機

- 如果我有一個數叫做number ,現在我要給他一個亂數
  number=rand();
  這樣就能將一個亂數放入

- int number;
- number=rand();
- printf("%d\n",number);

隨機

- 你會發現結果都是41 !!!
而且不管幾次都一樣!!!

隨機

需要以下兩樣東西

#include<time.h>　引入標頭檔

srand(time(NULL)); 取隨機種子

隨機

範圍限制

我們用“取餘數”的方法做範圍限制~

假設我們現在需要一個0~20的亂數
(共有20-0+1=21個數字)
就要寫成number=rand()%21;
因為任何數除以21的餘數皆落於0~20之間

陣列—宣告/使用

- 練習一.
- 輸入10筆資料(使用陣列)，然後依序印出。
- 練習二.
- 自動產生10筆隨機資料，然後依序印出。

陣列

- 二維陣列

  需要兩層迴圈

  可視為以平面方式組織的資料集合，故具二個索引註標(x與y軸方向)。

  <資料型態> <陣列名稱> [列數][行數];
  int array[50][20];
  float array[10][10];

陣列

- int data[3][2];

邏輯觀念

data[0][0]	**data[0][1]**
data[1][0]	**data[1][1]**
data[2][0]	**data[2][1]**

記憶體實際儲存樣式

data[0][0]
data[0][1]
data[1][0]
data[1][1]
data[2][0]
data[2][1]

練習—九九乘法表

j=1

j=5

i=1

i=4

```
1x1= 1  2x1= 2  3x1= 3  4x1= 4  5x1= 5  6x1= 6  7x1= 7  8x1= 8  9x1= 9
1x2= 2  2x2= 4  3x2= 6  4x2= 8  5x2=10  6x2=12  7x2=14  8x2=16  9x2=18
1x3= 3  2x3= 6  3x3= 9  4x3=12  5x3=15  6x3=18  7x3=21  8x3=24  9x3=27
1x4= 4  2x4= 8  3x4=12  4x4=16  5x4=20  6x4=24  7x4=28  8x4=32  9x4=36
1x5= 5  2x5=10  3x5=15  4x5=20  5x5=25  6x5=30  7x5=35  8x5=40  9x5=45
1x6= 6  2x6=12  3x6=18  4x6=24  5x6=30  6x6=36  7x6=42  8x6=48  9x6=54
1x7= 7  2x7=14  3x7=21  4x7=28  5x7=35  6x7=42  7x7=49  8x7=56  9x7=63
1x8= 8  2x8=16  3x8=24  4x8=32  5x8=40  6x8=48  7x8=56  8x8=64  9x8=72
1x9= 9  2x9=18  3x9=27  4x9=36  5x9=45  6x9=54  7x9=63  8x9=72  9x9=81
```

練習

```
#include<stdio.h>
#include<stdlib.h>
int main(){
    int i , j;
    for(  i=1; i<=9; I++ ){
        for( j=1 ; j<=9 ; j++){
            printf( “ %d x %d=%2d ” ,j , i ,  i *j);
        }
        printf("\n");
    }
    system("pause");
    return 0;
}
```

陣列的初值設定

- int array[4] = { 100, 200, 300, 400 };
- int array[] = { 100, 200, 300, 400 };
- int array[4][3] = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12 };
- int array[][3] = { { 1, 2, 3 } , { 4, 5, 6 } , { 7, 8, 9 } , { 10, 11, 12 } };

陣列的初值設定

- int array[5] = {0}; (將所有元素均設定為0值)
- int arr[5]={3, 5}; (除前兩個元素值分別為3及5之外，其他均清除為0)

陣列資料之複製

```
int arr1[5], arr2[]={1, 2, 3, 4, 5};


arr1[4] = arr2[4]   // 只有一個元素會被拷貝
arr1=arr2;          //錯誤寫法
arr1[]=arr2[];      //錯誤寫法
**********************************
for (int i=0; i < 5; i++)
    arr1[i] = arr2[i];   // 正確寫法 (利用迴圈)
```

字元與字串

- 字元(Character)：一個文、數字或符號，以單引號涵括之。
- 字串(Character string)：一串文、數字或符號組成，以雙引號涵蓋，或可視為字元陣列(Character array)。

字元的宣告與輸出入

- 字元的宣告
  - **char ch;**
- 字元的輸入(一)：讀入輸入緩衝區內第一個字元 (含任何字元及空白)
  - **scanf (“%c”, &ch);**
- 字元的輸入(二)：讀入輸入緩衝區內第一個非空白或非控制性的字元
  - **scanf (“□%c”, &ch);**

字元的宣告與輸出入(續)

- 字元的輸出：將變數ch內的字元輸出至螢幕上
  - **printf (“%c”, ch);**

字元輸入的相關函式

函式名稱	函式意義	引入標頭檔
getchar()	從鍵盤輸入一個字元；須按 enter 鍵完成	<stdio.h>
getche()	從鍵盤輸入一個字元，並顯示至畫面上；不必按 enter 鍵	<conio.h>
getch()	類似getche()函式，但輸入字元不會顯示在螢幕畫面上	<conio.h>

字元輸出的相關函式

函式名稱	函式意義	引入標頭檔
putchar()	輸出字元至螢幕上	<stdio.h>
putch()	輸出字元至螢幕上	<conio.h>

格式： putchar (字元資料);

範例：

```
putchar(ch);
putchar("\n");
```

字元輸入範例程式

```
#....

int main(void)
{
    int num;
    char ch;

    printf("請輸入一個整數: ");
    scanf("%d", &num);
```

字元輸入範例程式 (續)

```
        printf("請輸入一個字元: ");
        scanf ("%c", &ch);

/* 印出num與ch的ASCII碼 */
        printf("num=%d, ASCII of ch=%d\n",
                num, ch);
    .......
}
```

此程式將會印出何種結果？為什麼？

字元輸入範例程式 (續)

上述程式將輸出如下結果：

請輸入一個整數:22
請輸入一個字元: num=22, ASCII of ch=10

請注意：第二個輸入敘述將無任何字元被讀入，即印出結果了。

字元輸入範例程式的解決技巧

```
#.....

int main(void)
{
    int num;
    char ch;

    printf("請輸入一個整數: ");
    scanf("%d", &num);
```

字元輸入範例程式的解決技巧 (續)

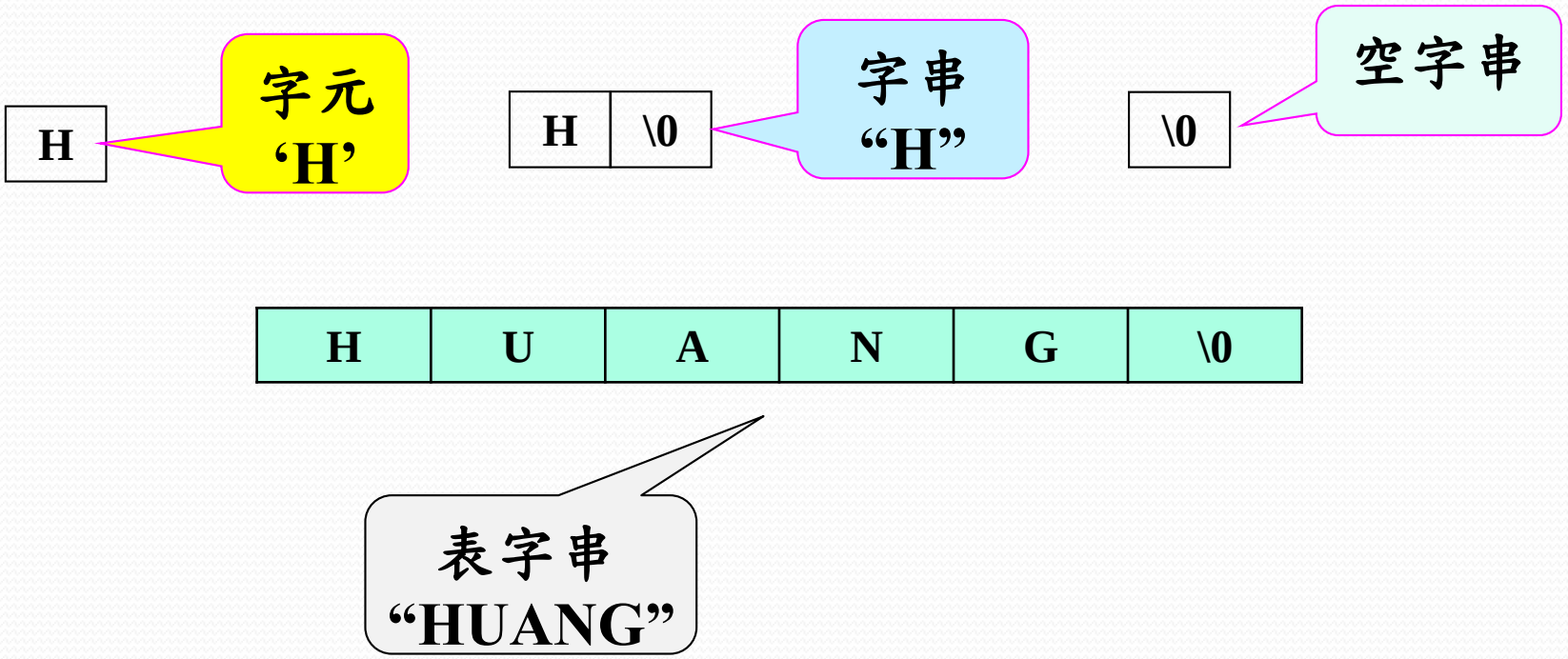
```
        printf ("請輸入一個字元: ");
        scanf (“ %c”, &ch);  // 在%c之前加空白

        printf("num=%d, ASCII of ch=%d\n",
                num, ch);
  ……
}
```

字串 (Character String)

- C語言中無字串基本資料型態，它可以字元陣列的方式來表示。
- 字串(字元陣列)宣告格式：
  - **char str [10];**
- 由於字串是一種資料結構，長度不定，故必須有方式來告知字串之結束時機。其採用之符號為空字元(null character，'\0')

字串(續)

- 一個字元與一個字的字串其在記憶體內的儲存表現方式是不同的。



字串之輸入方法

- 利用 **scanf (“%s”, 陣列名稱);** 之敘述。
- **注意：陣列名稱前不需加地址運算子 “&”**
- scanf()由輸入緩衝區內第一個非空白的字元開始讀取，直到碰觸到另一個空白為止。
- 範例：

```
char  str [20];
scanf(“%s”, str);
```

字串輸入的相關函式

函式名稱	函式意義	引入標頭檔
gets()	由緩衝區讀入所有字元 直到碰到 enter 為止	<stdio.h>

範例：

```
char str[20];
gets (str);
```

字串輸出的相關函式

函式名稱	函式意義	引入標頭檔
puts()	將字串輸出至螢幕上	<stdio.h>

範例：

```
char str[20];
 …..
puts (str);
```

字串輸入範例程式

```
#....
int main()
{
        char ch[10];
        printf("請輸入一個字串: ");
        scanf("%s", ch);
        printf("輸入字串為：%s\n", ch);
    .....
}
```

字串輸入範例程式(續)

執行結果:

請輸入一個字串: abc def <enter>
輸入字串為：abc

字串輸入範例程式(一)

```
#....
int main()
{
   char month[10];
   scanf("%3s", month);  // 指定最大輸入長度
   printf("字串為：%s\n", month);
    ....
}
```

字串輸入範例程式(一)(續)

執行結果:

若輸入: January

則印出:

字串為： Jan　　(%3s　只取前3個字)

字串輸入範例程式(二)

```
#.....
int main()
{
   char month[10], val[10];
   scanf ("%4s", month);
   printf("月份=%s\n", month);
   scanf ("%7s", val);
   printf("值=%s\n", val);
}
```

字串輸入範例程式(二)(續)

執行結果:

若輸入: January

印出: 月份=Janu (%4s)

值=ary (前述輸入殘留下來的字串被後續輸入指令立刻讀取)

請問：如何解決?

字串輸入範例程式(三)

```
#....
#define flush while (getchar() != '\n')
int main()
{
   char month[10], val[10];
   scanf("%4s", month);
   flush;   // 亦可用 fflush(stdin) + <stdlib.h>
   printf("月份=%s\n", month);
   .....
```

字串輸入範例程式(三)(續)

執行結果:

January　　（輸入資料）
月份=Janu
February　　（輸入資料）
值=Februar

字串範例程式(四)

```
#....
int main()
{
    printf("%c\n", "HELLO"[1] );
    ….
}
```

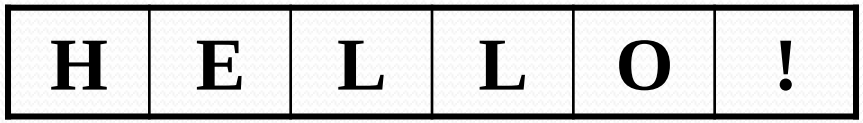
印出結果：**E**

一維的字元陣列

- 範例：

```
char string[] = {‘H’,’E’,’L’,’L’,’O’,’!’};
```

string陣列的內容：



一維的字元陣列範例程式(一)

```
#....
int main()
{
    char ch[10]={'H', 'E', 'L', 'L', 'O'};
    printf("string=%s\n", ch);
    ……
}
```

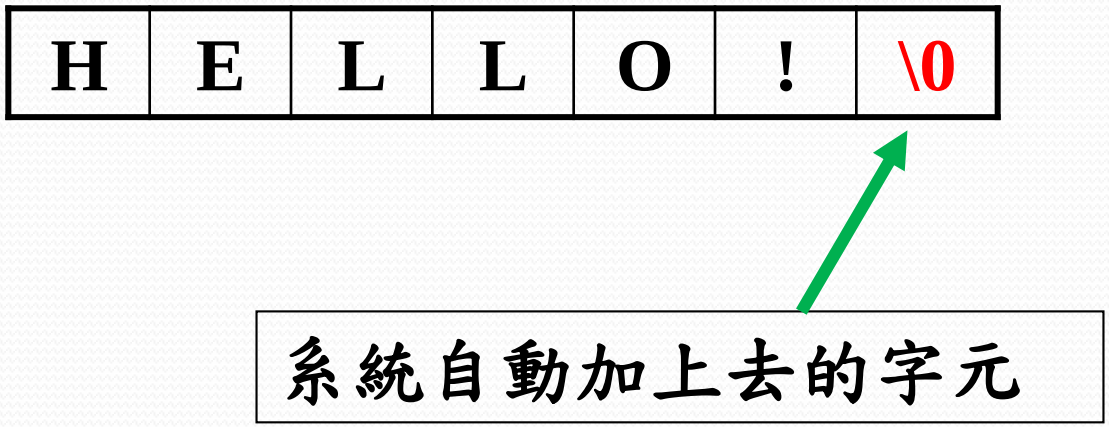
印出結果：**string=HELLO**

一維的字元陣列(二)

- 範例：

  char string[] = “HELLO!”;

string陣列的內容：



一維的字元陣列範例程式(二)

```
#....
int main()
{
      char ch[]= "HELLO!";
      printf("string=%s \n", ch);
       printf("string=%c \n", ch[1]);
    ……
}
```

印出結果：**string=HELLO!**
string=E

一維的字元陣列範例程式(三)

```
#....
int main()
{
   char ch[15] =  {‘H’, ‘E’, ‘L’, ‘L’, ‘O’, ‘!’, ‘\0’, ‘W',
   'O', 'R', 'L', 'D'};
    printf("string=%s\n", ch);
  …..
}
```

印出結果：**string=HELLO!**　為什麼？

二維的字元陣列

- 範例：

```
char Stringarray[][5] = {“abcd”,
                         “efgh”, ”ijkl”, ”mnop”};
```

	[0]	[1]	[2]	[3]	[4]
Stringarray[0]	a	b	c	d	\0
Stringarray[1]	e	f	g	h	\0
Stringarray[2]	i	j	k	l	\0
Stringarray[3]	m	n	o	p	\0

二維的字元陣列範例程式(一)

```
#....
int main()
{
   char ch[][10]= {"abcd", "efgh", "ijk", "mnop"};
   printf("string=%s \n", ch[2] );
  .....
}
```

印出結果：**string=ijk**