

Chapter 11
String



Outline

- Fundamentals of Strings and Characters
- Character-Handling Library
- String-Conversion Functions
- Standard Input/Output Library Functions
- String-Handling Library
- Pointers and Strings



Fundamentals of Strings and Characters

Characters and Strings

- Characters
  - Character constant
    - An int value is represented as a character in single quote
    - ‘z’ represents the integer value of z (122)
- Strings
  - Series of characters
    - Can include letters, digits, and special characters (*, /, $, etc)
  - Written in double quote
    - “Hello”
  - A string is an array of characters
    - A string is a pointer to the first character.
    - Value of a string is the address of the first character.

String Definitions

- Format
  - char string_name[string_size]; /* no initialization */
  - char string_name[] = “string”; /* initialize an array; can ignore string size*/
  - char *string_name = “string”; /*initialize a pointer; can ignore string size*/
  - Remember that strings represented as character arrays end with '\0‘
- Example
  - char str[] = “hello”;
  - char *str2 = “hello world!”;

str: 6 elements

[0]	[1]	[2]	[3]	[4]	[5]
h	e	l	l	o	\0

Common Programming Error

- char str[6];

✗ str = “hello”;

Invalid assignment because str is an array

Correction:

```
char str[6];
str[0] = ‘h’;
str[1] = ‘e’;
str[2] = ‘l’;
str[3] = ‘l’;
str[4] = ‘o’;
str[5] = ‘\0’;
```

Printing a “string” that does not contain a terminating null character is an error.



Inputting string

- Use *scanf*
  - char str[string_size]; /* must specify string size */
  - scanf(“%s”, str);

Do not need &
(because a string is a pointer)

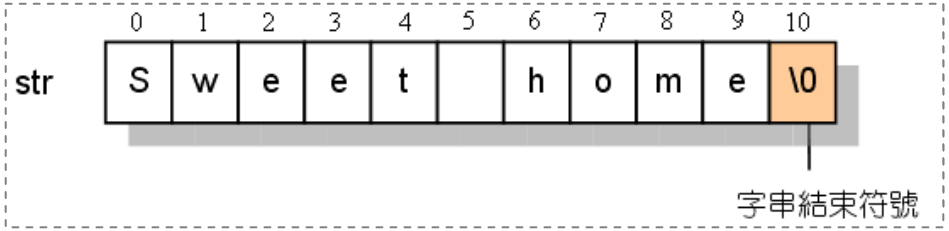
- Error usage
  - char *str; ✗
    scanf(“%s”, str);
    printf(“%s\n”, str); ; /* error because the size is unknown*/

Comparison Between Character and String

```
01  /* 印出字元及字串的長度 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06      char ch='a';                  /* 宣告字元變數 ch */
07      char str1[]="a";              /* 宣告字串變數 str1 */
08      char str2[]="Sweet home";     /* 宣告字串變數 str2 */
09
10      printf("字元 ch 佔了%d 個位元組\n",sizeof(ch));
11      printf("字串 str1 佔了%d 個位元組\n",sizeof(str1));
12      printf("字串 str2 佔了%d 個位元組\n",sizeof(str2));
13
14      system("pause");
15      return 0;
16  }
```

```
/* OUTPUT---

字元 ch 佔了 1 個位元組
字串 str1 佔了 2 個位元組
字串 str2 佔了 11 個位元組
--------------------*/
```





Character-Handling Library



Character-Handling Library

- #include <ctype.h>

- Functions that perform useful tests and manipulations of character data
- Each function receives a character (an int) or EOF as an argument



Character-Handling Library (Cont.)

Prototype	Description
int isdigit(int c)	Return 1 if c is a digit
int isalpha(int c)	Return 1 if c is a letter
int isalnum(int c)	Return 1 if c is a digit or letter
int isxdigit(int c)	Return 1 if c is a hex character
int islower(int c)	Return 1 if c is a lowercase letter
int isupper(int c)	Return 1 if c is an uppercase letter
int tolower(int c)	If c is an uppercase letter, return the lowercase letter of c
int toupper(int c)	If c is a lowercase letter, return the uppercase letter of c



Character-Handling Library (Cont.)

Prototype	Description
int isspace(int c)	Return 1 if c is a space
int iscntrl(int c)	Return 1 if c is a control character
int ispunct(int c)	Return 1 if c is a printing character other than a space, a digit, or a letter
int isprint(int c)	Return 1 if c is a printing character including a space ‘ ‘
int isgraph(int c)	Return 1 if c is a printing character other than a space ‘ ‘



Example

```
#include <stdio.h>
#include <ctype.h>

int main() {
        printf("Is \'8\' a digit? %s\n", isdigit('8')? "yes":"no");
        printf("Is \'a\' a digit? %s\n", isdigit('a')? "yes":"no");
        printf("Is \'A\' a letter? %s\n", isalpha('A')? "yes":"no");
        printf("Is \‘z\' a letter? %s\n", isalpha('z')? "yes":"no");
        printf("Is \'2\' a letter? %s\n", isalpha('2')? "yes":"no");
        printf("Is \'*\' a letter? %s\n", isalpha('*')? "yes":"no");
}
```

Output

```
Is '8' a digit? yes
Is 'a' a digit? no
Is 'A' a letter? yes
Is ‘z' a letter? yes
Is '2' a letter? no
Is '*' a letter? no
```



Example (Cont.)

```
#include <stdio.h>
#include <ctype.h>

int main() {
        printf("Is \'B\' an uppercase letter? %s\n", isupper('B')? "yes":"no");
        printf("Is \'b\' an uppercase letter? %s\n", isupper('b')? "yes":"no");
        printf("Is \'5\' an uppercase letter? %s\n", isupper('5')? "yes":"no");
        printf("Is \'5\' a lowercase letter? %s\n", islower('5')? "yes":"no");
        printf("Is \'!\' an uppercase letter? %s\n", isupper('!')? "yes":"no");
        printf("Is \'!\' a lowercase letter? %s\n", islower('!')? "yes":"no");
}
```

Output

```
Is 'B' an uppercase letter? yes
Is 'b' an uppercase letter? no
Is '5' an uppercase letter? no
Is '5' a lowercase letter? no
Is '!' an uppercase letter? no
```



Example (Cont.)

```
#include <stdio.h>
#include <ctype.h>

int main() {
        printf("Is \'\\n\' a space? %s\n", isspace('\n')? "yes":"no");
        printf("Is \'\\t\' a space? %s\n", isspace('\t')? "yes":"no");
        printf("Is \' \' a space? %s\n", isspace(' ')? "yes":"no");
        printf("Is \'$\' a space? %s\n", isspace('$')? "yes":"no");
        printf("Is \'%\' a control character? %s\n", iscntrl('%')? "yes":"no");
}
```

Output

```
Is '\n' a space? yes
Is '\t' a space? yes
Is ' ' a space? yes
Is '$' a space? no
Is '' a control character? no
```



Example (Cont.)

```
/* OUTPUT----------
請輸入一個字串: Happy Birthday
轉換成大寫後: HAPPY BIRTHDAY
--------------------------*/
```

```
01  /* 將字串裡小寫字母轉換成大寫 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  void toUpper(char s[]);    /* 宣告函數 toUpper() 的原型 */
05  int main(void)
06  {
07     char str[15];           /* 宣告可容納 15 個字元的陣列 str */
08
09     printf("請輸入一個字串: ");
10     gets(str);              /* 輸入字串 */
11     toUpper(str);           /* 呼叫 toUpper() 函數 */
12     printf("轉換成大寫後: %s\n",str);      /* 印出 str 字串的內容 */
13
14     system("pause");
15     return 0;
16  }
17
18  void toUpper(char s[])
19  {
20     int i=0;
21     while(s[i]!='\0')       /* 如果 s[i] 不等於\0，則執行下面的敘述 */
22     {
23        if(s[i]>=97 && s[i]<=122) /* 如果是小寫字母 */
24           s[i]=s[i]-32;     /* 把小寫字母的 ASCII 碼減 32，變成大寫 */
25        i++;
26     }
27  }
```



String-Conversion Functions



String-Conversion Functions

- #include <stdlib.h>

- Convert strings of digits to integer and floating-point values
- Example
  - Convert integer to string
  - Convert string to float
  - Convert string to integer



String-Conversion Functions (Cont.)

Prototype	Description
double atof(const char *nPtr)	Convert the string nPtr to double
int atoi(const char *nPtr)	Convert the string nPtr to integer
long atol(const char *nPtr)	Convert the string nPtr to long int
char * itoa (int c, char *str, int base)	Convert integer to string using the specified base (Non-standard function; only supported by some compilers)



Example

```
#include <stdio.h>
#include <stdlib.h>

int main() {
        char *strf = "99.99";
        char *stri = "-100";
        double d = atof(strf);
        int i = atoi(stri);
        int j = 20;
        char strj[5];
        itoa(j, strj, 10);
        printf("d = %f; i = %d; strj = %s\n", d, i, strj);
}
```

Output

```
d = 99.990000; i = -100; strj = 20
```



Standard Input/Output Library Functions



Example (Cont.)

- #include <stdio.h>
  - int getchar();
  - int putchar(char);
  - char *gets(char *s);
  - int puts(const char *s);
  - int sprintf(char *s, const char *format, …);
  - int sscanf(char *s, const char *format, …);



getchar() / putchar()

- int getchar();
  - return the next character from the standard input
- int putchar(int char);
  - Write char to the standard output
  - If there are no errors, the same character that has been written is returned
  - If an error occurs, EOF is returned

```
#include <stdio.h>
int main () {
  char c;
  printf("Enter a dot ('.') in a sentence to exit:\n");
  do {
    c=getchar();
    putchar (c);
  } while (c != '.');
  return 0;
}
```



gets() / puts()

- char *gets(char *str);
  - Reads characters from stdin and stores them as a string into *str* until a newline character ('\n') or the End-of-File is reached
- int puts(const char *str);
  - Writes *str* to stdout and appends a newline character ('\n')

```
#include <stdio.h>
int main() {
  char string [256];
  printf("Input a line of string:");
  gets (string);
  puts(string);
  return 0;
}
```

Output

```
Input a line of string: This is a line of string
This is a line of string
```



gets() and puts()

```
/* OUTPUT---

What's your name?
David Young
Hi!
David Young
How are you?
----------------------*/
```

```
01  /* 輸入及印出字串 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06     char name[15];     /* 宣告字元陣列 name */
07
08     puts("What's your name?");
09     gets(name);        /* 利用 gets()讀入字串，並寫入字元陣列 name 裡 */
10     puts("Hi!");
11     puts(name);        /* 印出字元陣列 name 的內容 */
12     puts("How are you?");
13     system("pause");
14     return 0;
15  }
```



sscanf()

- int sscanf (const char * str, const char * format, ...);
  - Reads data from *str* and stores them according to the parameter *format* into the locations given by the additional arguments

```
#include <stdio.h>
int main() {
  char string [] = "Eric has 2 dogs.";
  char str[20];
  int i;
  sscanf (string,"%s %*s %d", str, &i);
  printf ("%s, %d\n", str, i);
  return 0;
}
```

Output

```
Eric, 2
```

*: An optional starting asterisk indicates that the data is to be retrieved from the string but ignored



sprintf()

- int sprintf (char * str, const char * format, ...);
  - Writes into the array pointed by *str* a C string consisting on a sequence of data formatted as the *format* argument specifies

```
#include <stdio.h>
int main() {
  char string[50] = "Original string.";
  int a = 10, b = 5;
  sprintf(string, "%s -> %s", string, "New string.");
  printf ("%s\n", string);
  sprintf(string, "%d + %d = %d", a, b, a + b);
  printf ("%s\n", string);
  return 0;
}
```

Output

```
Original string. -> New string.
10 + 5 = 15
```



String-Handling Library



String-Handling Library

- #include <string.h>
- char *strcpy(char *s1, const str *s2);
  - Copy string s2 to string s1. The value of s1 s returned.
- char *strncpy(char *s1, const str *s2, size_t n);
  - Copy at most n characters of string s2 to string s1. The value of s1 s returned.
- char *strcat(char *s1, const str *s2);
  - Append string s2 to string s1. The value of s1 is returned.
- char *strncat(char *s1, const str *s2, size_t n);
  - Append at most n characters of string s2 to string s1. The value of s1 is returned.



String-Handling Library (Cont.)

- int strcmp(const char *s1, const str *s2);
  - Compare string s1 with s2. The function returns 0, -1, or 1 if s1 is equal to, less than, or greater than s2, respectively.
- int strncmp(const char *s1, const str *s2, size_t n);
  - Compare up to n characters of string s1 with s2. The function returns 0, less than 0, or greater than 0 if s1 is equal to, less than, or greater than s2, respectively.
  - Check the details in http://www.cplusplus.com/reference/clibrary/cstring/



Example: strcmp()

```
#include <stdio.h>
int main(int argc, char **argv) {
  char *s1 = "Happy new year";
  char *s2 = "Happy new year";
  char *s3 = "happy birthday";

  if (strcmp(s1, s2) == 0)
                printf("s1 is equal to s2\n");
  else
                 printf("s1 is not equal to s2\n");

  if (!strcmp(s1, s3))
                 printf("s1 is equal to s3\n");
  else
                 printf("s1 is not equal to s3\n");
  return 0;
}
```

Output

```
s1 is equal to s2
s1 is not equal to s3
```

Array of Strings

Declaration of String

```
char StrName[NumOfStrings][StrengLength];
```

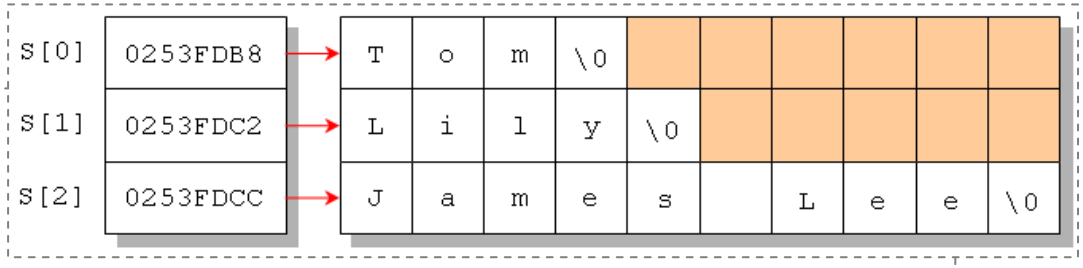
String Declaration and Initialization

```
char StrName[NumOfStrings][StringLength]=
        {“String1", “String2",..., “Stringn"};
```

```
char customer[6][15];
char S[3][10]={"Tom","Lily","James Lee"};
```



Element Access




```
01  /* 字串陣列 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06     char S[3][10]= {"Tom","Lily","James Lee"};
07     int i;
08     for(i=0;i<3;i++)
09        printf("S[%d]=%s\n",i,S[i]);    /* 印出字串陣列內容 */
10     printf("\n");
11     for(i=0;i<3;i++)   /* 印出字串陣列元素的位址 */
12     {
13        printf("S[%d]=%p\n",i,S[i]);
14        printf("address of S[%d][0]=%p\n\n",i,&S[i][0]);
15     }
16     system("pause");
17     return 0;
18  }
```



String Copy

```
/*  OUTPUT---

arr2[0]=Tom
arr2[1]=Lily
arr2[2]=James Lee
----------------------*/
```

```
01  /*  字串陣列的複製 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  #define MAX 3
05  #define LENGTH 10
06  int main(void)
07  {
08     char arr1[MAX][LENGTH]={"Tom","Lily","James Lee"};
09     char arr2[MAX][LENGTH];
10     int i,j;
11     for(i=0;i<MAX;i++)      /* 將 arr1 的內容複製到 arr2 中 */
12     {
13        for(j=0;j<LENGTH;j++)
14           if(arr1[i][j]=='\0')   /* 如果遇到「\0」,代表讀到字串結束 */
15              break;          /* 此行的 break 敘述會跳到第 19 行執行 */
16           else
17              arr2[i][j]=arr1[i][j];
18        arr2[i][j]='\0';
19     }
20     for(i=0;i<MAX;i++)
21        printf("arr2[%d]=%s\n",i,arr2[i]); /* 印出陣列 arr2 的內容 */
22     system("pause");
23     return 0;
24  }
```



Arguments in main()

- int main(int argc, char **argv)
  - argc: Number of input arguments
  - argv: An array of input arguments (strings)

- Use command line to specify arguments



Example

```
#include <stdio.h>
int main(int argc, char **argv) {
        int i;
        printf("There are %d arguments\n", argc);
        for (i = 0; i < argc; i++)
                printf("arg %d: %s\n", i, argv[i]);
    return 0;
}
```

Output

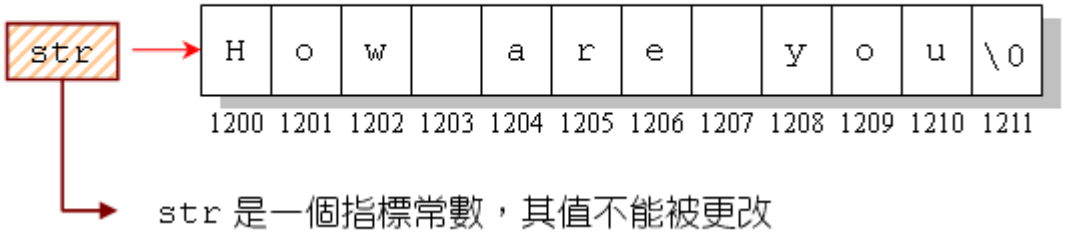
```
D:\tmp\Debug>tmp.exe hw 4 number 50
There are 5 arguments
arg 0: tmp.exe
arg 1: hw
arg 2: 4
arg 3: number
arg 4: 50
```



Pointers and Strings

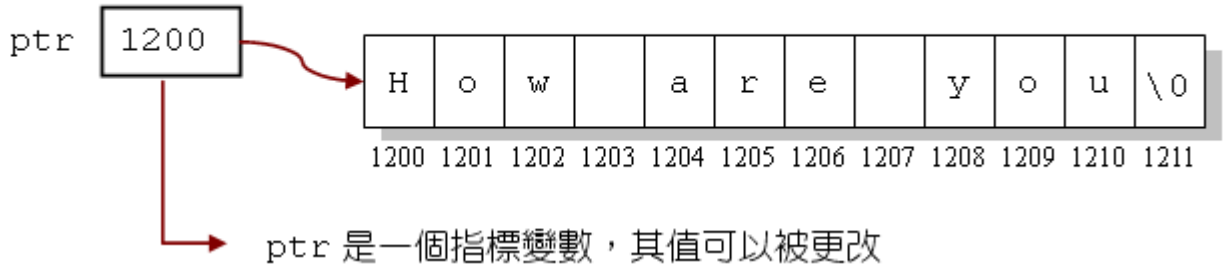
Pointers to Arrays

```
char str[]="How are you?";
```



- 利用指標指向字串：

```
char *ptr="How are you?";
```





Pointers to Arrays (Cont.)

```
/* OUTPUT---

what's your name? Wien
Hi, Wien, How are you?
-----------------------*/
```

```
01  /* 以指標變數指向字串 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06     char name[20];
07     char *ptr="How are you?";     /* 將指標指向字串"How are you?" */
08     printf("what's your name? ");
09     gets(name);                   /* 由鍵盤讀入字串 */
10     printf("Hi, %s, ",name);      /* 印出字串陣列 name 的內容 */
11     puts(ptr);                    /* 印出由 ptr 所指向的字串 */
12
13     system("pause");
14     return 0;
15  }
```



Pointer Array

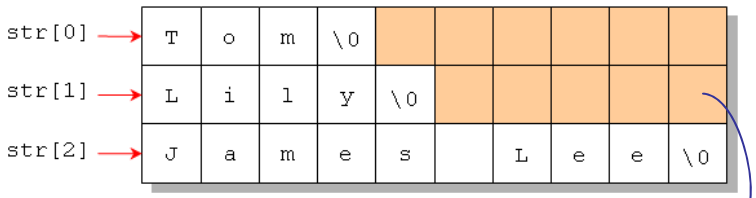
Pointer array declaration

```
DataType *ArrayName[NumOfElements];
```

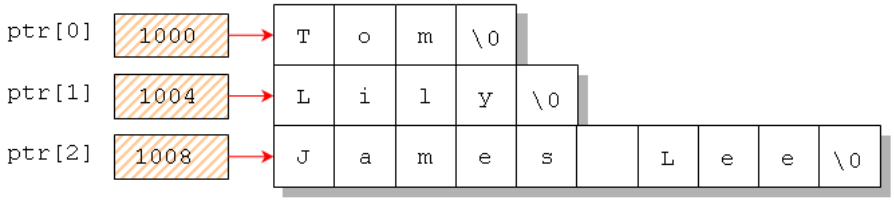
- Two dimensional array
- Use pointer

```
char str[3][10]={"Tom", "Lily", "James Lee"};
```

```
char *ptr[3]={"Tom", "Lily", "James Lee"};
```



Waste space



No space wasting



Pointer Array (Cont.)

```c
/* 指標陣列 */
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
   int i;
   char *ptr[3]={"Tom", "Lily", "James Lee"};
   for(i=0;i<3;i++)
      puts(ptr[i]);   /* 印出指標 ptr[i]所指向的字串 */

   system("pause");
   return 0;
}
```

```c
/* OUTPUT--

Tom
Lily
James Lee
----------------------*/
```



Pointer to Pointer (雙重指標)


```
DataType **PointerName;
```

– Declaration：
  - int **ptri;
  - char **ptrc;

Pointer to Pointer (雙重指標) (Cont.)

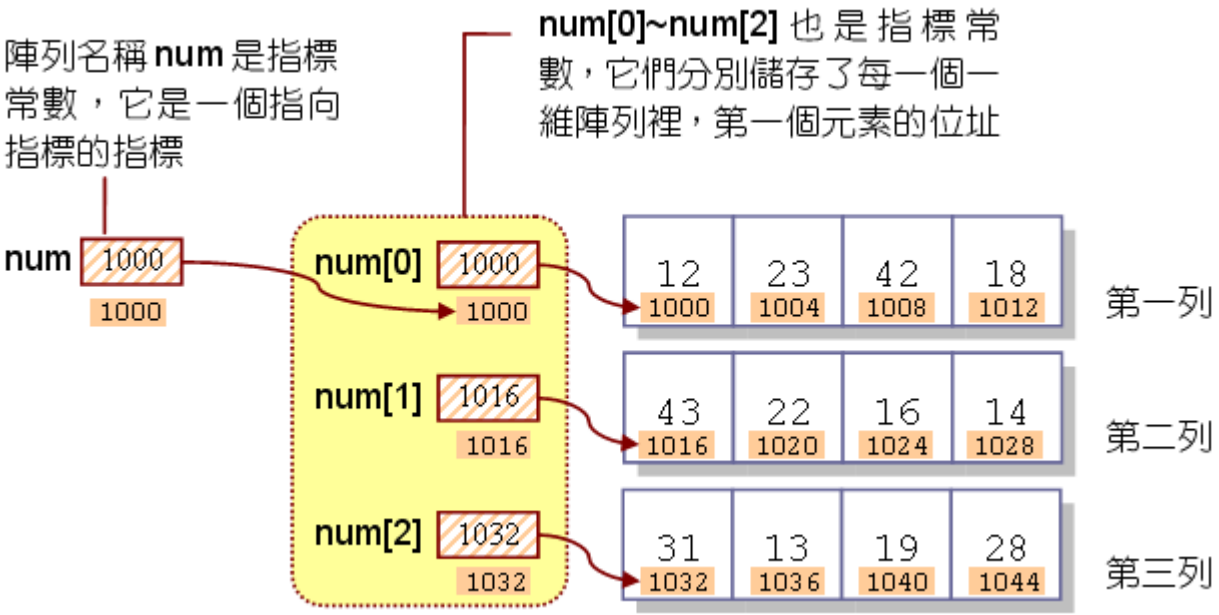

```
01  /* 雙重指標的範例 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06      int n=20,*p,**pp;
07      p=&n;
08      pp=&p;
09      printf("n=%d,&n=%p,*p=%d,p=%p,&p=%p\n",n,&n,*p,p,&p);
10      printf("**pp=%d,*pp=%p,pp=%p,&pp=%p\n",**pp,*pp,pp,&pp);
11
12      system("pause");
13      return 0;
14  }
```

```
/*  OUTPUT----------------------
n=20,&n=0022FF6C,*p=20,p=0022FF6C,&p=0022FF68
**pp=20,*pp=0022FF6C,pp=0022FF68,&pp=0022FF64
----------------------------------------*/
```

2D Array and Pointer to Pointer

```
int num[3][4]
```

- 由 3 個一維陣列所組成，每個一維陣列裡各有4個元素
- num[0]、num[1] 與 num[2] 為指標常數，它們分別指向這3個一維陣列

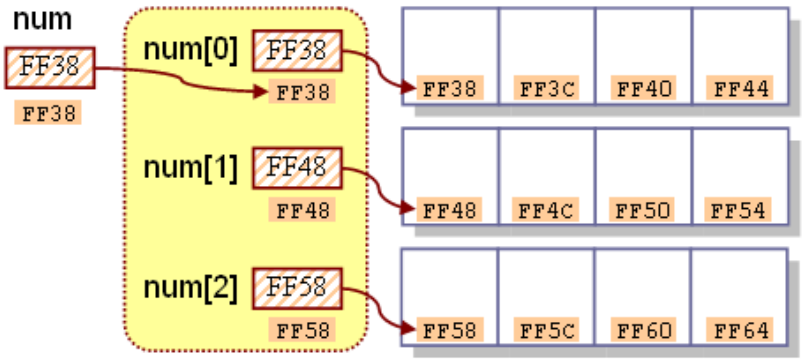




Example

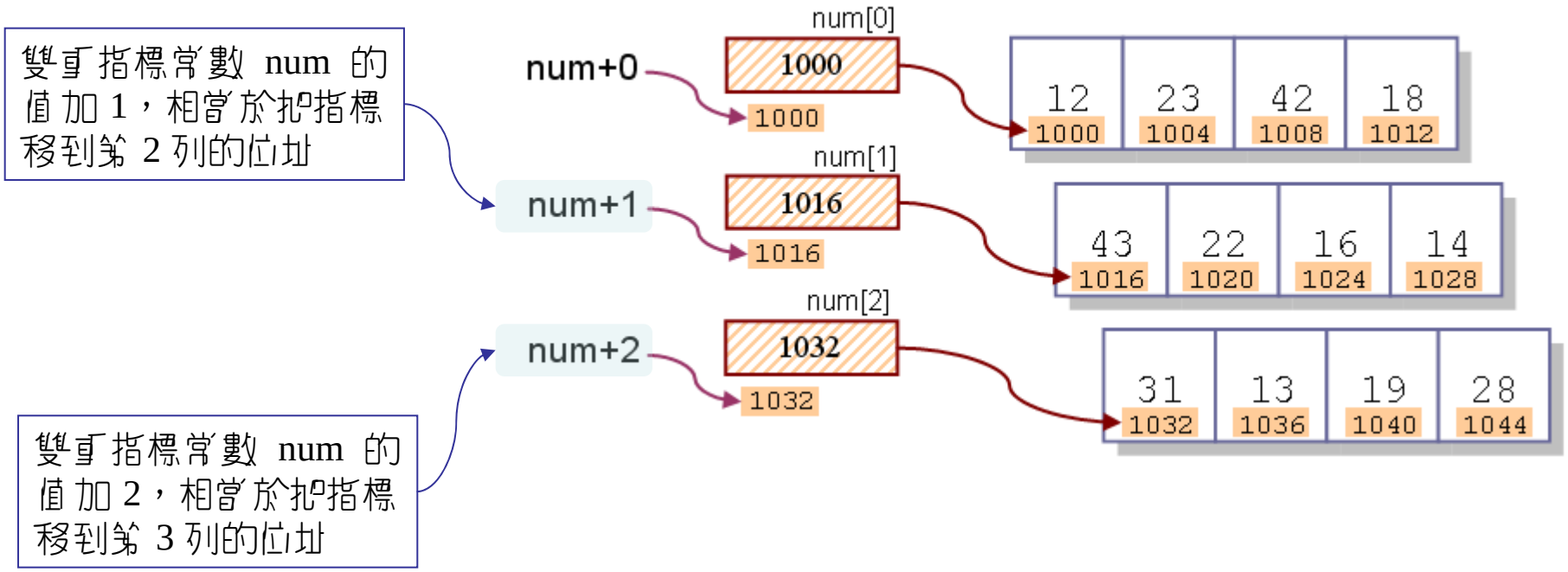
```
01  /* 印出陣列的位址 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06      int num[3][4];
07
08      printf("num=%p\n",num);
09      printf("&num=%p\n",&num);
10      printf("*num=%p\n",*num);
11
12      printf("num[0]=%p\n",num[0]);
13      printf("num[1]=%p\n",num[1]);
14      printf("num[2]=%p\n",num[2]);
15
16      printf("&num[0]=%p\n",&num[0]);
17      printf("&num[1]=%p\n",&num[1]);
18      printf("&num[2]=%p\n",&num[2]);
19      system("pause");
20      return 0;
21  }
```

```
/*  OUTPUT--
num=0022FF38
&num=0022FF38
*num=0022FF38
num[0]=0022FF38
num[1]=0022FF48        } 指標常數的值
num[2]=0022FF58
&num[0]=0022FF38
&num[1]=0022FF48        } 指標常數的位址
&num[2]=0022FF58
----------------------*/
```



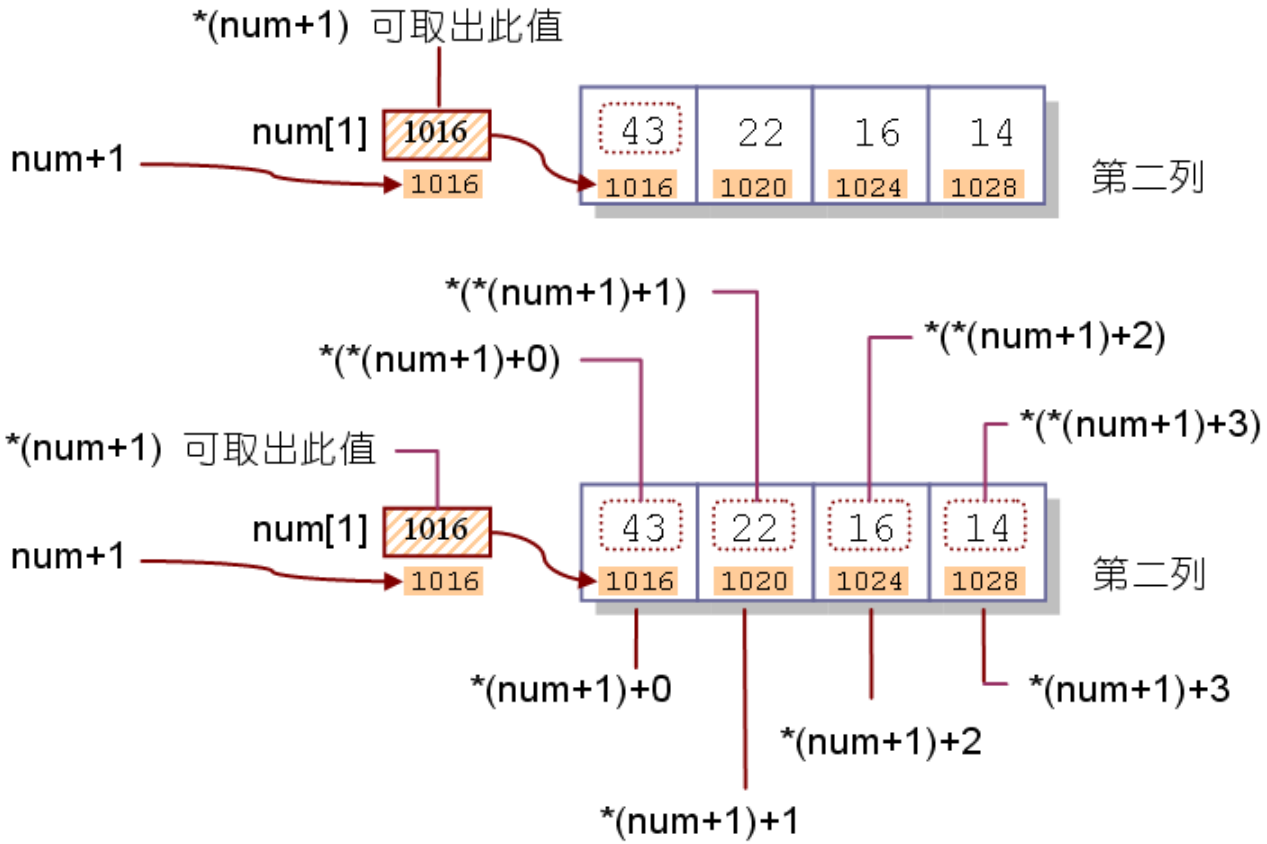


Presentation of Pointer to Pointer



– num+i equals to num[i]

Array Element Access



- 取出第 m+1 列，第 n+1 行的地址：*(num+m)+n
- 取出第 m+1 列，第 n+1 行的元素： *(*(num+m)+n)

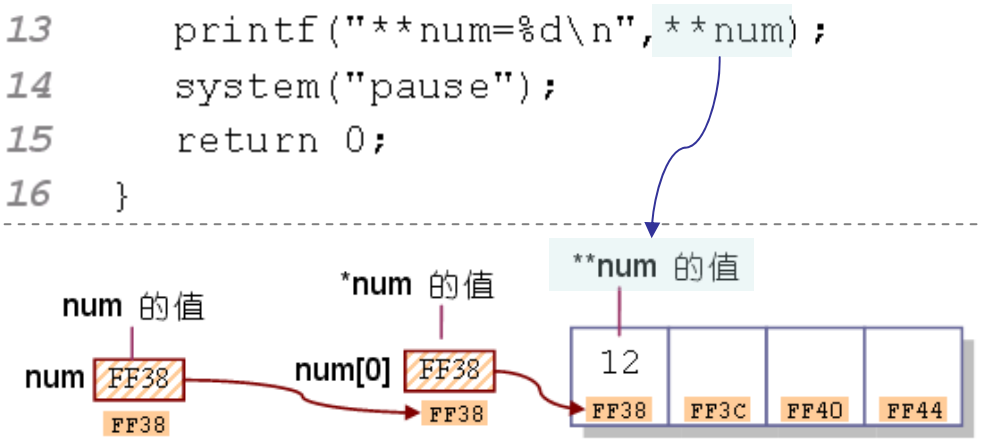


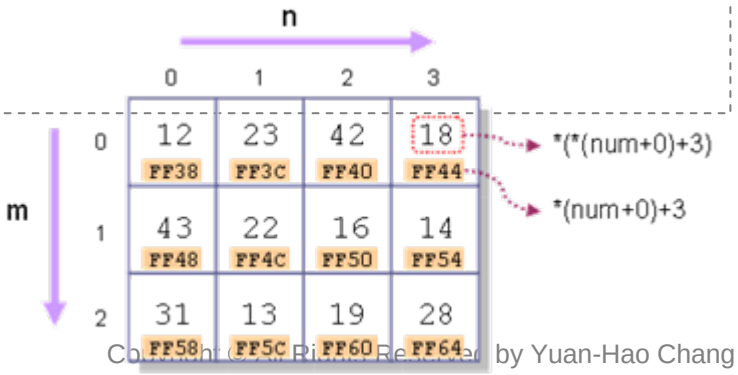
Example

```
01  /* 印出陣列的位址 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06     int num[3][4]={{12,23,42,18},
07                    {43,22,16,14},
08                    {31,13,19,28}};
09     int m,n;
10     for(m=0;m<3;m++)
11      for(n=0;n<4;n++)
12        printf("num[%d][%d]=%d, 位址=%p\n",m,n,*(*(num+m)+n),*(num+m)+n);
13     printf("**num=%d\n",**num);
14     system("pause");
15     return 0;
16  }
```

```
/* OUTPUT------
num[0][0]=12, 位址=0022FF38
num[0][1]=23, 位址=0022FF3C
num[0][2]=42, 位址=0022FF40
num[0][3]=18, 位址=0022FF44
num[1][0]=43, 位址=0022FF48
num[1][1]=22, 位址=0022FF4C
num[1][2]=16, 位址=0022FF50
num[1][3]=14, 位址=0022FF54
num[2][0]=31, 位址=0022FF58
num[2][1]=13, 位址=0022FF5C
num[2][2]=19, 位址=0022FF60
num[2][3]=28, 位址=0022FF64
**num=12
------------------------*/
```

num[m][n] 的值　　num[m][n] 的位址







Another Example

```
01  /* 利用指標將大於 40 的陣列元素設值為 40 */
02  #include <stdio.h>
03  #include <stdlib.h>
04  int main(void)
05  {
06     int num[3][4]={{12,23,42,18},
07                    {43,22,16,14},
08                    {31,13,19,28}};
09     int m,n;
10     for(m=0;m<3;m++)
11     {
12        for(n=0;n<4;n++)
13        {
14           if(*(*(num+m)+n)>40)   /* 判別 num[m][n]的值是否大於 40 */
15              *(*(num+m)+n)=40;   /* 如果是，則將元素值設為 40 */
16           printf("%3d",*(*(num+m)+n));  /* 印出元素 num[m][n]的值 */
17        }
18        printf("\n");
19     }
20     system("pause");
21     return 0;
22  }
```

```
/* OUTPUT---

 12 23 40 18
 40 22 16 14
 31 13 19 28
----------------------*/
```



Lab 11

- Write a program to output the size of char *arr1="Johnson" and the size of char arr2[]="Johnson";
- Let the user input a character,
  - Convert it to the lower case, and print it on the screen if it is a upper case.
  - Convert it to the upper case, and print it on the screen if it is a lower case.
  - Otherwise, terminate your program.
- Let the user input a string and print the string on the screen in the reverse order.
- Write a for loop; let the user input the names of 5 students and store the names as an array of strings (e.g., char name[5][50]). Then output the student's names.