由片語學習C程式設計

台灣大學資訊工程系劉邦鋒著

台灣大學劉邦鋒老師講授

August 19, 2016

第十單元

字元

片語 1: 字元變數的宣告

```
1 char c;
```

- 宣告一個字元變數。 一個字元就是一個 字。
- 字元在C 中是 char，就是 character 的前四個字母。
- C 使用一個位元組儲存一個字元， 所以一個字元能存一個 -128 到 127 之間的整數。

範例程式 2: (char-size.c) 一個 char 所佔的位元組數

```
1 #include <stdio.h>
2 int main(void)
3 {
4    char c;
5    printf("%d\n", sizeof(c));
6    return 0;
7 }
```

輸出

```
1 1
```

- 一個字元變數佔一個位元組。

學習要點

一個字元變數佔一個位元組。

片語 3: 字元的輸出

```
1 printf("%c", c);
```

- 字元最大的用處就是以ASCII 碼的形式將文字資訊顯示出來。
- ASCII 碼 (American Standard Code for Information Interchange)， 就是將 0 到 127 的整數對應到我們常用的英文大小寫字母， 0 到 9 的數字，以及標點符號等。
- 印出時要用 %c、 c 代表字元。 不在後面加換行字元因為一般我們希望字元是一個接一個輸出。

範例程式 4: (ascii.c) 印出部分 ASCII 碼

```
1  #include <stdio.h>
2  int main(void)
3  {
4     char c;
5     int i, j;
6     printf("0123456789abcdef\n");
7     for (i = 2; i <= 7; i++) {
8        for (j = 0; j <= 15; j++) {
9           c = i * 16 + j;
10          printf("%c", c);
11       }
12       printf("\n");
13    }
14    return 0;
15 }
```

- 印出由 32 到 127 的ASCII 碼。 對應到的 16進位數字是 20 到 7F， 我們選這個範圍是因為 32 之前的都是特殊字元。
- 輸出是 16 個字元一行，而且在第一行加上 0 到 f 的 16 進位。
- 小寫字母 a 的 16 進位為 61， 就是 10 進位的 $6 \times 16 + 1 = 97$。

輸出

```
1 0123456789abcdef
2  !"#$%&'()*+,-./
3 0123456789:;<=>?
4 @ABCDEFGHIJKLMNO
5 PQRSTUVWXYZ[\]^_
6 `abcdefghijklmno
7 pqrstuvwxyz{|}~-
```

字元常數

- 程式中用數字表示字元會很不方便，例如小寫字母 a， 雖然它在 ASCII 表中是 97，但是很難將 97 和 小寫字母 a 聯想起來。
- **字元常數** 將字元的 ASCII 值和它所代表的符號連接起來。
- C 中字元常數是用一對單引號將一個符號括起來，代表它的 ASCII 值。

特殊字元

C 中字元常數是用一對單引號，，括起來。

範例程式 5: (char-assignment.c) 將字元常數指定給字元變數

```
1  #include <stdio.h>
2  int main(void)
3  {
4     char c;
5     c = 'm';
6     printf("%c", c);
7     c = 'a';
8     printf("%c", c);
9     c = 'i';
10    printf("%c", c);
11    c = 'n';
12    printf("%c", c);
13    c = '(';
14    printf("%c", c);
15    c = ')';
16    printf("%c", c);
```

```
18    c = '\n';
19    printf("%c", c);
20    c = '{';
21    printf("%c", c);
22    c = '\n';
23    printf("%c", c);
24    c = '}';
25    printf("%c", c);
26    c = '\n';
27    printf("%c", c);
28    return 0;
29 }
```

輸出

```
1 main()
2 {
3 }
```

- 換行字元是用 \n 表示，所以以字元常數表示時就是 '\n'。

範例程式 6: (char-assignment-int.c) 將整數指定給字元變數並印出

```
1  #include <stdio.h>
2  int main(void)
3  {
4     char c;
5     c = 109;
6     printf("%c", c);
7     c = 97;
8     printf("%c", c);
9     c = 105;
10    printf("%c", c);
11    c = 110;
12    printf("%c", c);
13    c = 40;
14    printf("%c", c);
```

```
16      c = 41;
17      printf("%c", c);
18      c = 13;
19      printf("%c", c);
20      c = 123;
21      printf("%c", c);
22      c = 13;
23      printf("%c", c);
24      c = 125;
25      printf("%c", c);
26      c = 13;
27      printf("%c", c);
28      return 0;
29  }
```

輸出

```
1 main()
2 {
3 }
```

- 直接用 ASCII 值指定給字元變數 c。 整個程式就變的莫名其妙，不知道在寫什麼。
- 例如 c = 109; 就看不出事實上是在指定小寫字母 m。

範例程式 7: (int-as-char.c) int 變數也可存字元

```
1  #include <stdio.h>
2  int main(void)
3  {
4    int c;
5    c = 'm';
6    printf("%c", c);
7    c = 'a';
8    printf("%c", c);
9    c = 'i';
10   printf("%c", c);
11   c = 'n';
12   printf("%c", c);
13   c = '(';
14   printf("%c", c);
```

```
16    c = ')';
17    printf("%c", c);
18    c = '\n';
19    printf("%c", c);
20    c = '{';
21    printf("%c", c);
22    c = '\n';
23    printf("%c", c);
24    c = '}';
25    printf("%c", c);
26    c = '\n';
27    printf("%c", c);
28    return 0;
29 }
```

int 及 char

- 字元 char 可視為一個 8 位元的整數， 而整數 int 則是一個 32 位元的整數。
- 如果一個 int 變數的值是 0 到 127 之間， 它也可以對應到一個有效的 ASCII 編碼值，

輸出

```
1 main()
2 {
3 }
```

學習要點

除了表示的範圍較小之外，`char` 的用法與 `int` 並無太大差別。

範例程式 8: (char-overflow.c) 字元變數溢位

```
1  #include <stdio.h>
2  int main(void)
3  {
4     char c;
5     int i;
6     scanf("%d", &i);
7     c = i;
8     c++;
9     i++;
10    printf("c = %d\n", c);
11    printf("i = %d\n", i);
12    return 0;
13 }
```

輸入

```
1 127
```

輸出

```
1 c = -128
2 i = 128
```

- char 能表示的範圍很小，所以很容易溢位。

片語 9: 字元的輸入

```
1 scanf("%c", &c);
```

- 從鍵盤讀入一個字元

範例程式 10: (char-io.c) 輸入一字元再分別用字元或整數輸出

```
1 #include <stdio.h>
2 int main(void)
3 {
4    char c;
5    scanf("%c", &c);
6    printf("%c\n", c);
7    printf("%d\n", c);
8    return 0;
9 }
```

輸入

```
1 a
```

輸出

```
1 a
2 97
```

- 在輸入一個字元後我們要再打入一個**換行鍵**程式才會開始執行。
- 輸入小寫字母 a，所以程式就輸出 a。 但用整數輸出時得到 97，

片語 11: 藉由 scanf 的回傳值判定是否還有資料。

```
1 while (scanf("%c", &c) != EOF) {
2   ...
3   process character c;
4 }
```

- 從鍵盤讀入所有字元直到 EOF。
- 利用 scanf 的回傳值判斷是否還有資料。

片語 12: 使用字元分類函式

```
1 #include <ctype.h>
2 ...
3 char c;
4 ...
5 if (isxxxxx(c))
```

- 系統的 <ctype.h> 中定義了一些好用的字元處理函式。
  - 字元分類函式
  - 字元轉換函式
- 使用時必須引入 <ctype.h> 標頭檔，

字元分類函式

- 字元分類函式決定傳進來字元是否屬於某一類字元。
  - `isalpha(c)` 會判斷 `c` 是否為英文字母。
- 若是則回傳 **真** 否則回傳 **偽**。

字元分類函式表

函式名稱	分類
isalnum	英文字母或數字
isalpha	英文字母
islower	小寫英文字母
isupper	大寫英文字母
isdigit	數字
isxdigit	16進位數字
isprint	可顯示字元 (包含空白)
isgraph	可顯示字元 (不包含空白)
isspace	空白
ispunct	標點符號
iscntrl	控制字元

範例程式 13: (ctype-class.c) 字元分類測試

```
1  #include <stdio.h>
2  #include <ctype.h>
3  void check(char c)
4  {
5    printf("c = %c\n", c);
6    if (isalnum(c))
7      printf("isalnum\n");
8    if (isalpha(c))
9      printf("isalpha\n");
10   if (islower(c))
11     printf("islower\n");
12   if (isupper(c))
13     printf("isupper\n");
14   if (isdigit(c))
15     printf("isdigit\n");
```

```
17    if (isxdigit(c))
18      printf("isxdigit\n");
19    if (isprint(c))
20      printf("isprint\n");
21    if (isgraph(c))
22      printf("isgraph\n");
23    if (isspace(c))
24      printf("isspace\n");
25    if (ispunct(c))
26      printf("ispunct\n");
27    if (iscntrl(c))
28      printf("iscntrl\n");
29    return;
30  }
```

```
32 int main(void)
33 {
34   char c;
35   while (scanf("%c", &c) != EOF)
36     check(c);
37   return 0;
38 }
```

輸入

```
1 aZ3  .
2 =\*
```

輸出

```
c = a
isalnum
isalpha
islower
isxdigit
isprint
isgraph
c = Z
isalnum
isalpha
isupper
isprint
isgraph
```

```
14 c = 3
15 isalnum
16 isdigit
17 isxdigit
18 isprint
19 isgraph
20 c =
21 isprint
22 isspace
23 c = .
24 isprint
25 isgraph
26 ispunct
```

```
27 c =
28 isspace
29 iscntrl
30 c =
31
32 isspace
33 iscntrl
34 c = =
35 isprint
36 isgraph
37 ispunct
38 c = \
```

```
39 isprint
40 isgraph
41 ispunct
42 c = *
43 isprint
44 isgraph
45 ispunct
```

範例程式 14: (ascii-dec.c) 用十進位印 ASCII 表

```
1  #include <stdio.h>
2  int main(void)
3  {
4    int c;
5    printf("   0123456789\n");
6    for (c = 30; c <= 127; c++) {
7      if (c % 10 == 0)
8        printf("%2d ", c / 10);
9      if (isprint(c))
10       printf("%c", c);
11     else
12       printf(" ");
13     if (c % 10 == 9)
14       printf("\n");
15   }
16   return 0;
17 }
```

ASCII

- 想輸出的範圍是 32 到 127 可印出的部分，
- 以用 `isprint()` 檢查是否列印，或是用空白代替。
- 每一行的開始會印出十進位的前兩位。 為了對齊，這前兩位指定用兩位數字印出。 方法是在 `%` 及 `d` 中放一個 `2`。

輸出

```
1     0123456789
2   3    !"#$%&'
3   4 ()*+,-./01
4   5 23456789:;
5   6 <=>?@ABCDE
6   7 FGHIJKLMNO
7   8 PQRSTUVWXY
8   9 Z[\]^_`abc
9  10 defghijklm
10 11 nopqrstuvw
11 12 xyz{|}~
```

字元轉換函式

函式名稱	動作
tolower	轉成小寫英文字母
toupper	轉成大寫英文字母

- 式將傳進來字元參數由小寫轉大寫，或是由大寫轉小寫。轉換的結果當作回傳值。

範例程式 15: (toupper.c) 用 toupper 轉換所有字元

```
1  #include <stdio.h>
2  int main(void)
3  {
4    char c;
5    c = toupper('m');
6    printf("%c", c);
7    c = toupper('a');
8    printf("%c", c);
9    c = toupper('i');
10   printf("%c", c);
11   c = toupper('n');
12   printf("%c", c);
13   c = toupper('(');
14   printf("%c", c);
```

```
16    c = toupper(')');
17    printf("%c", c);
18    c = toupper('\n');
19    printf("%c", c);
20    c = toupper('{');
21    printf("%c", c);
22    c = toupper('\n');
23    printf("%c", c);
24    c = toupper('}');
25    printf("%c", c);
26    c = toupper('\n');
27    printf("%c", c);
28    return 0;
29  }
```

輸出

```
1 MAIN()
2 {
3 }
```

- 將所有字元用 toupper 轉換後再印出。
- 如果參數不是字母，則 toupper 回傳原來的參數值。