程式設計與實習(二)

BY 孫茂勛

EMAIL:JOHN85051232@GMAIL.COM

一個程式語言的基本概念

- 開專案
- Hello World
- 資料型態(Data type)
- 變數(Vairable)
- 輸入輸出(Input/Output)
- 流程控制(If…else…)
- 迴圈(Loop)
- 函式(Function)
- 陣列(Array)
- 指標(Pointer)
- 字串(String)
- 讀寫檔(File I/O)

其實這堂課在兩年前只有一學期

所以這學期要做什麼

- 單次兩題(以上)
- 累積四題
- 這學期的CPE都請報名，報名會加分
  有答對也可以加分

加分就從現在開始，趕緊報名3/28的CPE

然後之後白天的課也是由助教上

可是我禮拜二早上有必修

....

所以還有另外一位助教

經歷：CPE 5題、ITSA佳作、桂冠杯程式競賽佳作

EMAIL：wu810517@gmail.com

- 上課練習題目
- 出作業
- 考試

- 講新的內容

- 作業
  - 每週數題UVA題目
  - 期中前的作業DEADLINE：期中考週
  - 期末前的作業DEADLINE：期末考週
- 考試
  - 可以Open Book，但一樣不能上網

這學期晚上我們會做什麼

- 一點點的資料結構
- 一點點的C應用
- 想到什麼講什麼

邏輯運算

以下的執行結果是什麼？

```
printf("%d \n",1 && 1);
printf("%d \n",0 && 1);
printf("%d \n",1 && 0);
printf("%d \n",0 && 0);
```

```
printf("%d \n",1 || 1);
printf("%d \n",0 || 1);
printf("%d \n",1 || 0);
printf("%d \n",0 || 0);
```

位元運算

把&跟|各拿掉一個，再看一次結果

```
printf("%d \n",1 & 1);
printf("%d \n",0 & 1);
printf("%d \n",1 & 0);
printf("%d \n",0 & 0);
```

```
printf("%d \n",1 | 1);
printf("%d \n",0 | 1);
printf("%d \n",1 | 0);
printf("%d \n",0 | 0);
```

位元運算

再換一組數字試試看

```
printf("%d \n",2 && -3);
printf("%d \n",2 & -3);
```

```
1
0
請按任意鍵繼續 . . .
```

位元運算

邏輯運算：只有true(非0的數字)和false(0)

位元運算：一個bit一個bit做判斷

```
printf("%d \n",2 && -3);
printf("%d \n",2 & -3);
```

```
  00000010
AND 11111101
  --------
  00000000
```

true && true == true

位元運算

邏輯運算：只有true(非0的數字)和false(0)

位元運算：一個bit一個bit做判斷

```
printf("%d \n",2 || 0);
printf("%d \n",2 | 0);
```

```
00000010
OR 00000000
---------
00000010
```

true || false == true

```
1
2
請按任意鍵繼續 . . .
```

Q：想把11001001前4個bits保留，後4個bits清空(變成0)該怎麼辦？

1.變成字串，一個一個字元改

```
str[0] = '0';
str[1] = '0';
str[2] = '0';
str[3] = '0';
...
```

2.利用位元遮罩

位元遮罩(Mask)

利用AND和OR的特性，保留和清空特定的bit

```
        11001001          11001001
AND     11110000    OR    11110000
--------------    --------------
        11000000          11111001
```

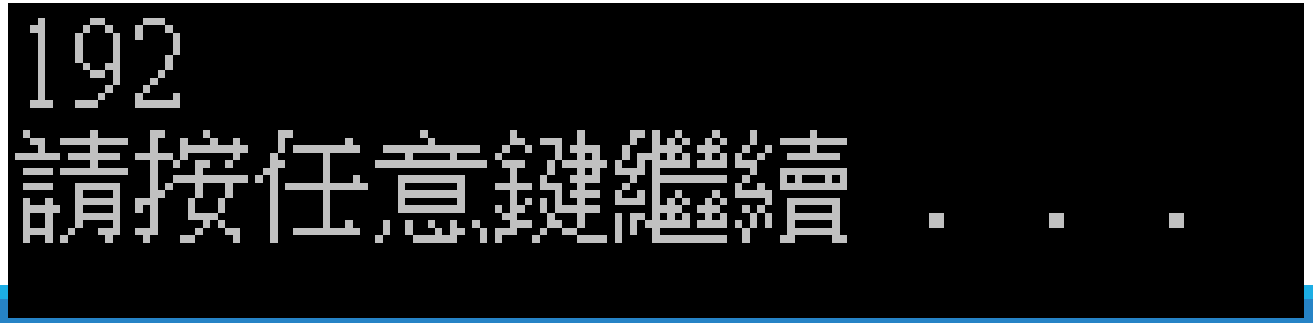
AND的1是保留，0是清空(變成0)
OR的1是設定(變成1)，0是保留

Q：想把11001001前4個bits保留，後4個bits清空(變成0)該怎麼辦？

1.變成字串，一個一個字元改

2.利用位元遮罩

```
Printf("%d \n",201 & 240); //201(11001001) & 240(11110000)
```



Q：判斷某個數字是奇數或偶數？

1.對2取餘數

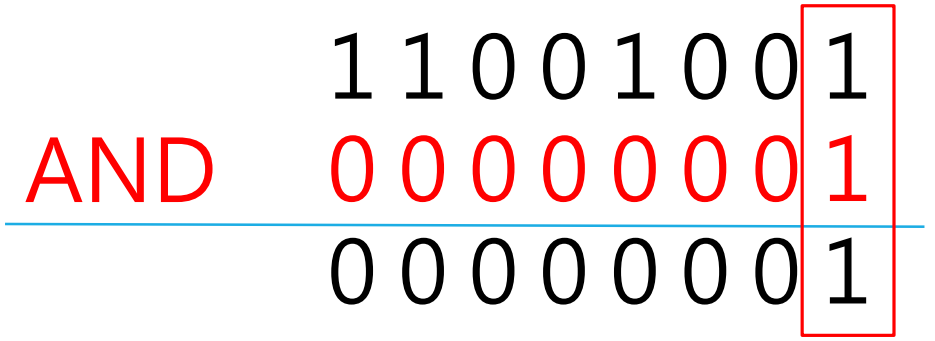
```
if(x % 2 == 0)
{
    printf("even");
}
else
…
```

2. 位元運算

Q：判斷某個數字是奇數或偶數？

1. 對2取餘數
2. 位元運算

```
if(x & 1 == 0)
{
    printf("even");
}
else
…
```



位元運算

在二進位中：

左移一個bit = 原數字 * 2

右移一個bit = 原數字 / 2

2 * 2 :

```
  00000010
x       10
----------
  00000100
```

```
printf("%d \n",2 <<1);
```

8 / 4 :

```
  00001000
/      100
----------
  00000010
```

```
printf("%d \n",8 >>2);
```

Q：取得33在二進位中從右數來的第2個bit?

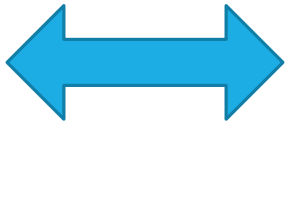
```
printf("%d \n", (33 >>2) & 1);
```

應用

- 網路IP子遮罩
- 加速運算速度
- 加解密

簡易加解密

將檔案加密/解密的原理是什麼？



把可辨識的內容變成別人看不懂的內容就是加密。
將被加密過後的內容變回原本的內容稱做解密。

簡易加解密

Q：這兩串文字間有什麼關係？

GWKKI ⟷ HELLO

簡易加解密

只要知道是如何加密的，就有可能反向進行解密
簡單加密的方法有很多：XOR、鍵盤上的關係、ASCII、Hash...

接下來用ASCII表上的關係寫兩個程式，一個將檔案加密，一個將檔案解密

ASCII ?

簡易加解密

複習一下複製檔案怎麼寫

```
FILE *fptr = fopen("test.txt","r");
FILE *fptr1 = fopen("test_out.txt","w");
char c = 0;
while( (c = fgetc(fptr)) != EOF )
{
    fputc(c,fptr1);
}
fclose(fptr);
fclose(fptr1);
```

簡易加解密

1.將檔案加密

```
FILE *fptr = fopen("test.txt","r");
FILE *fptr1 = fopen("test_out.txt","w");
char c = 0;
while( (c = fgetc(fptr)) != EOF )
{
    fputc(c+1,fptr1);
}
fclose(fptr);
fclose(fptr1);
```

簡易加解密

2.看一下產生的txt內容長怎樣？

簡易加解密

3.將檔案解密

```
FILE *fptr = fopen(" test_out.txt ","r");
FILE *fptr1 = fopen("test_out2.txt","w");
char c = 0;
while( (c = fgetc(fptr)) != EOF )
{
    fputc(c-1,fptr1);
}
fclose(fptr);
fclose(fptr1);
```

簡易加解密

不只txt，也可以用在其他格式的檔案(image、exe.....)

UVA10222 – Decode the Mad man

點我

就從資料結構開始吧

其實上學期教過了幾個資料結構

- array
- struct
- union
- enum

所以資料結構是什麼？？？？

資料結構 [編輯]

維基百科，自由的百科全書

在電腦科學中，**資料結構**（英語：data structure）是電腦中儲存、組織資料的方式。

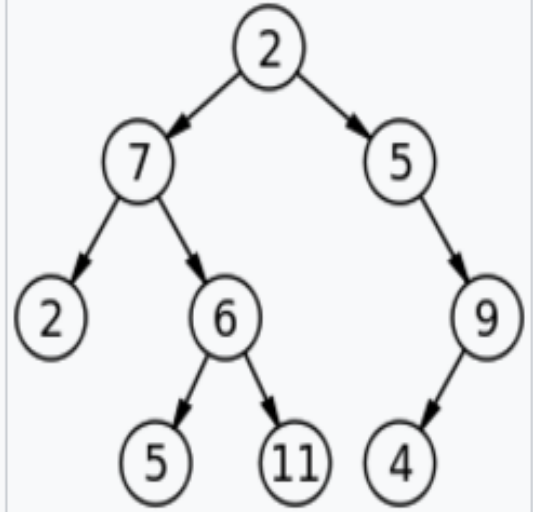
資料結構意味著介面或封裝：一個資料結構可被視為兩個函式之間的介面，或者是由資料類型聯合組成的儲存內容的存取方法封裝。

大多數資料結構都由數列、記錄、可辨識聯合、參照等基本類型構成。舉例而言，可為空的參照（nullable reference）是參照與可辨識聯合的結合體，而最簡單的鏈式結構連結串列則是由記錄與可空參照構成。

資料結構可透過程式語言所提供的資料類型、參照及其他操作加以實現。一個設計良好的資料結構，應該在儘可能使用較少的時間與空間資源的前提下，支援各種程式執行。

不同種類的資料結構適合不同種類的應用，部分資料結構甚至是為了解決特定問題而設計出來的。例如B樹即為加快樹狀結構存取速度而設計的資料結構，常被應用在資料庫和檔案系統上。

正確的資料結構選擇可以提高演算法的效率（請參考演算法效率）。在電腦程式設計的過程裡，選擇適當的資料結構是一項重要工作。許多大型系統的編寫經驗顯示，程式設計的困難程度與最終成果的品質與表現，取決於是否選擇了最適合的資料結構。



二元樹是資料結構的一種類型

系統架構的關鍵因素是資料結構而非演算法的見解，導致了多種形式化的設計方法與程式語言的出現。絕大多數的語言都帶有某種程度上的模組化思想，透過將資料現封裝隱藏於使用者介面之後的方法，來讓不同的應用程式能夠安全地重用這些資料結構。C++、Java、Python等物件導向的程式語言可使用類別來達到這個目的。

因為資料結構概念的普及，現代程式語言及其API中都包含了多種預設的資料結構，例如 C++ 標準模板庫中的容器、Java集合框架以及微軟的.NET Framework。

資料結構

- 資料儲存的方式
- 能夠讓你的程式執行的更有效率，花的時間更少
  (同樣的程式你花1秒別人要100秒，why?)
- 對於不同種類的資料會有不同的資料結構能夠使得在操作上有著較好的效率

題外話 - 關於效能這件事

程式能正確執行就好了，效能很重要嗎？

網路上某家公司的面試題簡化再簡化後大概長這樣：
Q：請設計一個程式，輸入數字N(N可能會到很大很大，假設不會overflow)，然後輸出1+2+3+4+5+...N的結果

題外話 - 關於效能這件事

A1：
```
for(int I = 0 ; I < N ; ++I)
{
    ans += I;
}
```

A2：
```
ans = (1 + N ) * N / 2;
```

假設N有10億，A1的for迴圈就要跑10億次，A2卻只需要執行一次。

結論：好的程式設計能夠有效地幫你降低程式執行的時間

文章網址：http://www.vixual.net/blog/archives/99