

- ▶ IPv4 的 IP 格式為『A.B.C.D』, 其中 A, B, C 與 D 為 0 至 255 間的整數。
要檢測一字串是否符合此規則, 可分成兩個步驟來處理。
 - ▷ 首先確認字串的格式為『A.B.C.D』, 其中 A, B, C, D 各為 1 至 3 位數。
此部分可利用正規式來處理。
 - ▷ 接著檢查 A, B, C, 與 D 的值是否在 0 與 255 之間, 可利用 shell 之條件式來進行數值比較。
- ▶ 『字串格式為「A.B.C.D」, 其中 A, B, C, D 各為 1 至 3 位數』之正規式有數種寫法:
 - ▷ 『`^[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}$`』
此為最直覺的寫法, 其中
 - 『`{1,3}`』與『`[0-9]`』結合, 代表 1 至 3 位數。
 - 『`\.`』代表句點(period)這個字元,
若只寫『`.`』則被視為正規式中的特殊字元, 用以符合任意字元。
 - 執行『`egrep '^[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}$'`』進行測試,
因未指名檔名, `egrep` 從標準輸入讀取資料, 此時可輸入幾行測試, 若輸入符合格式的資料行, `egrep` 會馬上印出來。
 - ▷ 這個正規式中有重覆出現 3 次的部分, 即『`[0-9]{1,3}\.`』, 可利用『`()`』把重覆出現的部分設定為一群組(group), 再搭配『`{}`』來限制重覆的次數, 即
『`([0-9]{1,3}\.){3}`』, 整個正規式可簡寫成『`^([0-9]{1,3}\.){3}[0-9]{1,3}$`』,
請執行『`egrep '^([0-9]{1,3}\.){3}[0-9]{1,3}$'`』進行測試。
 - ▷ 以下 script 將一直執行直到使用者輸入符合格式的字串後結束:

```
#!/bin/bash
read -p "Give me an IP..." IP
while [ 1 ]
do echo "$IP" | egrep -q '^([0-9]{1,3}\.){3}[0-9]{1,3}$'
  if [ $? -eq 0 ]
  then break
  fi
  read -p "Incorrect format, please input again..." IP
done
echo "IP=$IP"
```

(註: 為易於閱讀, 上述 script 及正規式並非最精簡的版本; 若您有興趣可參閱本文最後兩段)

- ▶ 接著要檢查字串中的四個整數是否在 0 與 255 的範圍內. 問題是, 要如何取出這四個整數?

```
valid=1
for number in `echo $IP | tr '.' ' '`
do if [ $number -gt 255 ]
  then valid=0
  break
fi
done
```

- 此例中使用變數 `valid` 做為旗號(flag), 初始值為 1。
一旦在檢測過程中發現有任何數值超過 255 時, 則將 `valid` 設定為 0, 並結束迴圈。
- `tr` 則用以將句點換成空格。

- ▷ Shell 內建的參數替換(parameter expansion)亦可用於將句點換成空格,
例如 shell 會將『`${IP//./ }`』換成『變數 IP 的值內所有句點換成空格後之結果』,
因此前例可改寫如下:

```

valid=1
for number in ${IP//./}
do if [ $number -gt 255 ]
then valid=0
break
fi
done

```

► 整合兩部分並稍加修改, 得到

```

#!/bin/bash
read -p "Give me an IP..." IP
while [ 1 ]
do echo "$IP" | egrep -q '^([0-9]{1,3}\.){3}[0-9]{1,3}$'
if [ $? -eq 0 ]
then valid=1
for number in ${IP//./}
do if [ $number -gt 255 ]
then valid=0
break
fi
done
if [ $valid -eq 1 ]
then break
fi
fi
read -p "Incorrect format, please input again..." IP
done
echo "IP=$IP..."

```

► 接著嘗試撰寫一函式(function)用以檢測字串是否為 IPv4 格式,

- ▷ 若字串為 IPv4 的 IP, 則傳回 0,
- ▷ 若引數個數不為 1, 則傳回錯誤碼 1,
- ▷ 若字串不符合『A.B.C.D』的格式則傳回 2,
- ▷ 若 A, B, C, D 不在 0 至 255 的範圍則傳回 3.
- ▷ 範例

```

function check_ip
{ if [ $# -ne 1 ]; then return 1; fi
if `echo $1 | egrep -q '^([0-9]{1,3}\.){3}[0-9]{1,3}$'`
then for number in ${1//./}
do if [ $number -gt 255 ]; then return 3; fi
done
return 0
fi
return 2
}

```

► 稍早我們使用『`^([0-9]{1,3}\.){3}[0-9]{1,3}$`』這個正規式, 但它並不是最精簡的。

其實『`[0-9]{1,3}\.`』及『`[0-9]{1,3}$`』差別僅在於前者續接『`.`』, 後者續接『行尾』(字串結尾), 若利用『`|`』(代表『或』的意思)進行整合, 可得『`[0-9]{1,3}(\.|$)`』, 代表 1 至 3 位數之後, 可以是『`.`』, 或是行尾

整個正規式可再次簡寫成『`^([0-9]{1,3}(\.|$)){4}$`』。

測試後發現, 『`1.2.3.4.`』也能符合, 但這並不符合我們的期待. 那該怎麼辦?

► 在正規式中, 『字』(word)由字母, 數字字元, 或底線所組成。

- ▷ 觀察『192.168.0.1』這字串中,一共有『192』,『168』,『0』與『1』四個字,字串內『.』之後均緊接另一個『字』,『.』本身並非字,在『非字』與『字』間(或『字』與『非字』間)可視為存在著『字的邊界』(word boundary),它不佔空間,長度為0,正規式中的寫法為『\b』.
- ▷ 而『1.2.3.4.』結尾的『.』,其後已是『行尾』(字串結尾),並非另一個字,因此這個『.』之後並非『字的邊界』.
- ▷ 了解兩者的差異性後,把正規式修改成『^[0-9]{1,3}(\.\b|\$){4}\$』即可.

▶ 終日奔波忙碌的你,需要來個心靈深呼吸.

想換個口味,過個不一樣的聖誕夜嗎?

歡迎參加**台北真理堂** 2008年聖誕夜崇拜,有戲劇演出,見證與信息分享.

時間: 2008年12月24日(三),晚上 6:30, 8:30 與 10:30

地點: 台北市新生南路三段86號,台灣大學側門對面

網址: <http://www.tlc.org.tw>

凡接待祂(耶穌)的,就是信祂名的人,
祂就賜他們權柄,作 神的兒女。 約翰福音 1:12

給耶穌一個機會,就能得著以下七重祝福喔:

病得醫治.心靈平安.家庭和樂.

凡事富足.罪得赦免.得著永生.聖靈充滿

方法就在**滿福寶** <http://www.sblog.com.tw/fullblessings>