

0~999 對應的中文讀法

實習目標：

丁培毅

1. 條件判斷 if, switch
2. 簡易函式設計
3. 分析問題、解決問題與邏輯設計
4. 流程圖運用
5. 常數字串陣列與 printf("%s", name[i])

數字的中文讀法

1. 程式輸入一個十進位整數 n , $0 \leq n \leq 999$, 例如 123
程式輸出數字的中文讀法: 一百二十三
2. 其它範例

零	十	二十	一百	一百一十
一	十一	二十一	一百零一	一百一十一
二	十二	二十二	一百零二	一百一十二
三	十三	二十三	一百零三	一百一十三
四	十四	二十四	一百零四	一百一十四
五	十五	二十五	一百零五	一百一十五
六	十六	二十六	一百零六	一百一十六
七	十七	二十七	一百零七	一百一十七
八	十八	二十八	一百零八	一百一十八
九	十九	二十九	一百零九	一百一十九

分析

1. 標準讀法:

1	2	3
一	百	二
	十	三

2. 列表檢查:

0~9	100 101~109	200 201~209
10 11~19	110 111~119	210 211~219
20 21~29	120 121~129	220 221~229
30 31~39	130 131~139	230 231~239
⋮	⋮	⋮
90 91~99	190 191~199	290 291~299

3. $h==0$ 時, 零百不唸; $h==0, t==0$ 時, 零百零十不唸
4. 除了 0 之外, 個位數 o 為 0 時不唸
5. $h==0, t==1$: 10 ~ 19: 十位數只唸 十 (不唸 一十)
6. $h>0, t==0, o>0$: 十位數唸零 (不唸 零十)
7. $h>0, t==0, o==0$: 十位數不唸, 個位數不唸

8. $0 \leq n \leq 999, \quad n ==> h * 100 + t * 10 + o$

```

h = n / 100;
t = (n % 100) / 10;
o = n % 10;
switch (o) {
case 0: printf("零"); break;
case 1: printf("一"); break;
...
case 9: printf("九"); break;
}

```

9. 輸出 零, 一, 二, ..., 九

10. 程式裡很多地方都需要輸出 零, 一, 二, ..., 九

```

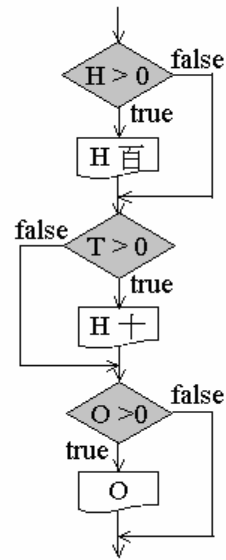
void printMandarin(int o) {
switch (o) {
case 0: printf("零"); break;
...
case 9: printf("九"); break;
}
}

```

流程圖運用

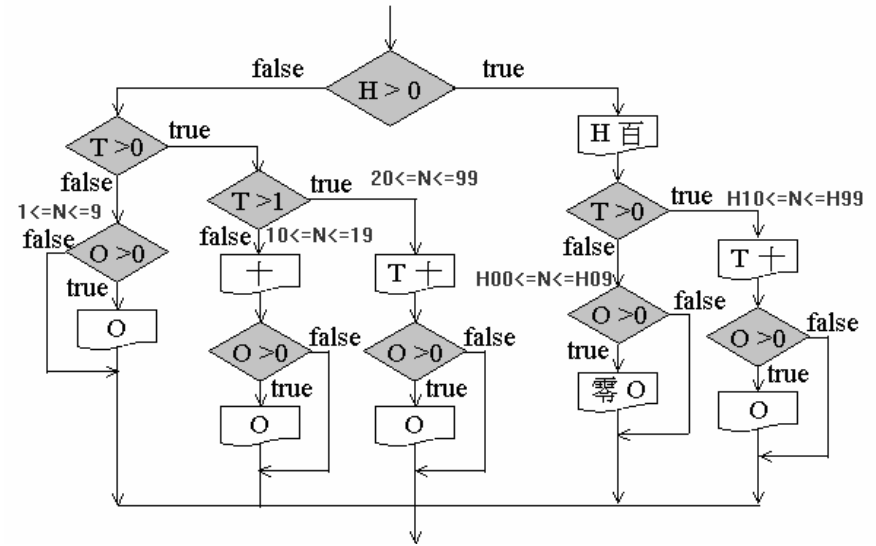
$$N = H*100 + T*10 + O$$

1. 這個流程圖其實把H, T, O 三個位數的數值分開來想了; 對於 "H" 和 "百" 的列印是對的, 但是列印 "T" 和 "十" 的時候, 不能不考量 "H" 和 "O" 的數值
2. 列印 "T" 和 "十" 在 $1 \leq N \leq 19$ 和 $101 \leq N \leq 119$ ($201 \leq N \leq 219, \dots$) 有特別的規則, 例如:
 $05 \implies$ 五, $105 \implies$ 一百零五
 $15 \implies$ 十五, $115 \implies$ 一百一十五
 所以在下一步驟裡我們檢查 $H > 0$ 然後個自判斷



5

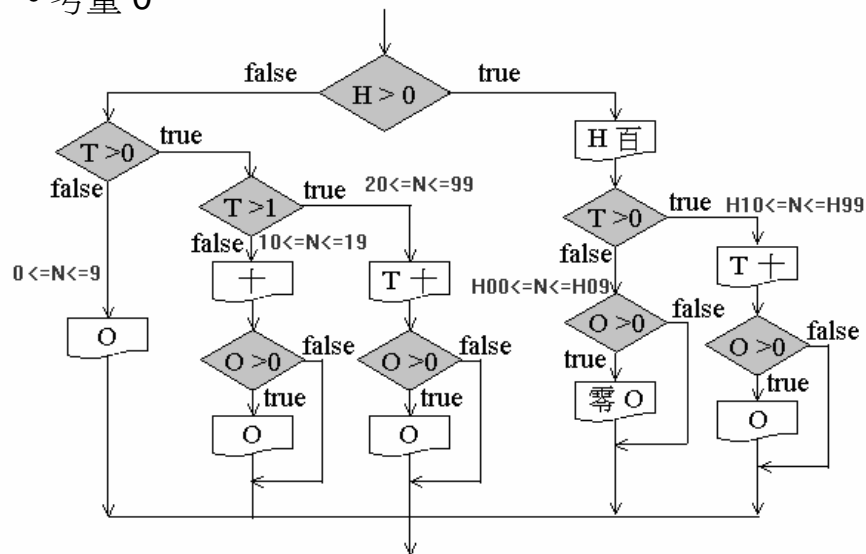
$$N = H*100 + T*10 + O$$



6

• 考量 0

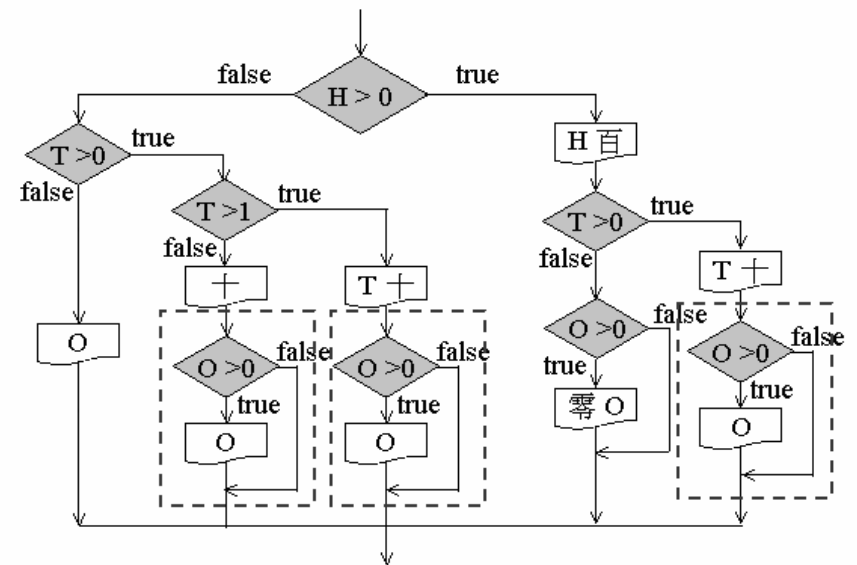
$$N = H*100 + T*10 + O$$



7

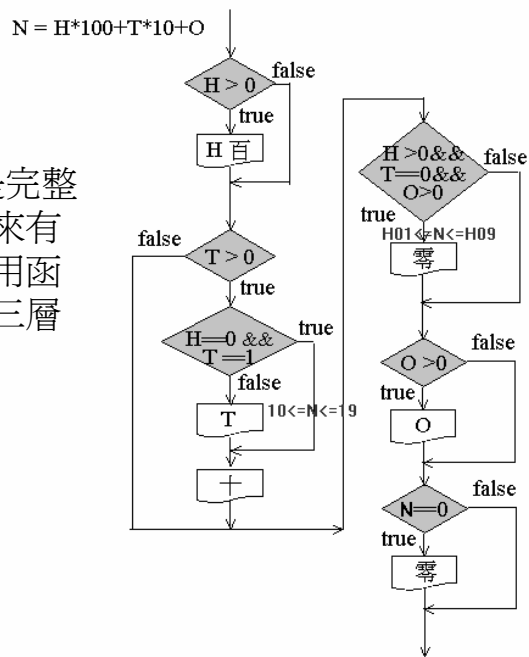
- 運用函式去除重複的程式碼, 簡化實作

$$N = H*100 + T*10 + O$$



8

- 前面的流程圖是完整的, 但是實作起來有點複雜, 如果不用函數的話, 會出現三層的 if 敘述



9

常數字串陣列運用

```
const char *mandarinDigits[] =
    {"零", "一", "二", "三", "四", "五", "六", "七", "八", "九"};
```

```
printf("%s百", mandarinDigits[h]);
```

例: h=5; 五百

可以取代前面的 switch 敘述以及函式 printMandarin()

10