



程式設計選手研習

講師:林易民

104 學年度程式設計職種競賽規則

LOGO

❖ 評分標準：

- 以學、術科比例合計為個人總成績，依參賽者總成績高低決定名次。若總成績相同則以術科成績較高者為先，**若術科成績仍相同，則依術科繳卷時間先者為先**，若繳卷時間仍相同則依序比較配分較高題得分者為領先。



104 學年度程式設計職種競賽規則

LOGO

❖ 競賽方式

1. 實地上機撰寫程式。
2. **每一試題提供 2組測試資料及其正確解答**以供學生測試用。
3. 試題之輸出入資料均有其格式及範圍之要求，選手應依指定之格式及範圍作答。
4. **各題需依試題說明之指定路徑儲存執行檔，執行時需直接讀取執行檔所在資料夾下的測試檔，並將其結果檔輸出至同一資料夾，未依規定者，該題不予計分。**
5. **程式執行過程中應先隱藏顯示介面或訊息，未依規定者，該題不予給分。**
6. **輸出內容與答案不符者，該題不予給分。**
7. **在限定時間內未執行結束者，該題不予給分。**

規則分析

❖ 分析重點

- 存檔很重要，不要存錯
- 程式不能出現任何訊息
- 只要你的程式能在規定時間內執行完畢而且執行正確，程式寫得快比跑得快重要



程式基本架構

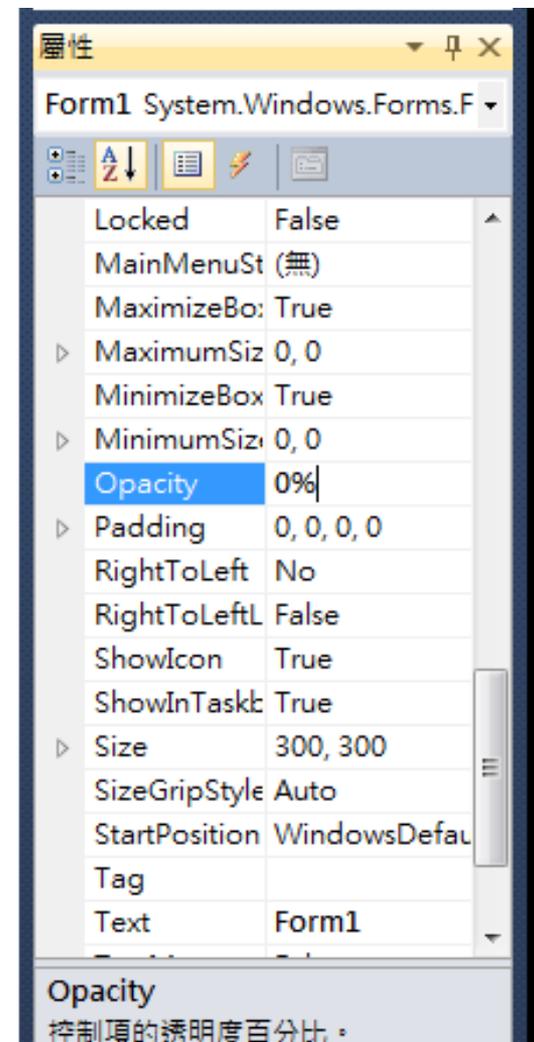
Enter ↵

T.C.R.

280-00

隱藏畫面

1. 設定表單屬性 **Opacity=0%**
2. 在**Form1_Load**副程式中最後一行加入
Me.Close() (關閉表單)



如何計算程式執行時間

Dim Watch As New Stopwatch (定義計時器)

Watch.Reset() (將計時器的時間歸零)

Watch.Start() (啟動計時器)

.....

(被監控的程式)

.....

.....

Watch.Stop() (停止計時器)

MsgBox("執行" & Watch.ElapsedMilliseconds & "毫秒")

如何讀取不同輸入格式資料

輸入檔案 1 :【檔名 : in1.txt】

2

{1, 3}, {2, 4}

{1, 2, 3}, {3, 4, 5}

❖ 程式範例

```
Dim Sep As String() = {"{", "}", ","}
```

```
Dim In() As String
```

```
in=St.Split(Sep,StringSplitOptions.RemoveEmptyEntries)
```

解題的七個步驟

1. 依據題目要求建立視窗專案 名稱
2. 設定表單程式透明度，並於Form1_Load副程式中加入Me.Close()
3. 設定必要的.NET 命名空間
4. 將測試資料檔放入Debug目錄中
5. 定義輸出入檔案變數及常用迴圈變數
6. 定義輸入分割符號
7. 不要忘記插入 分別兩個輸入檔的空白行

基本程式架構(練習用)

```
Private Sub Form1_Load(sender As System.Object, e As System.EventArgs) Handles MyBase.Load
```

```
    Dim MyWatch As New Stopwatch
```

```
    MyWatch.Start()
```

```
    Dim Reader(1) As StreamReader, SR As StreamReader
```

```
    Dim SW As New StreamWriter("out.txt")
```

```
    Reader(0) = New StreamReader("in1.txt")
```

```
    Reader(1) = New StreamReader("in2.txt")
```

```
    Dim Sep() As String = {" "} ,U As Integer, Fn As Integer
```

```
    Dim I As Integer, J As Integer, K As Integer, L As Integer
```

```
    For Fn = 0 To 1
```

```
        SR = Reader(Fn)
```

```
        .....
```

(主要解題的程式)

```
        .....
```

```
        SW.WriteLine()
```

```
    Next
```

```
    MyWatch.Stop()
```

```
    SW.WriteLine("總共花費{0}奈秒", ChaneTickToNanoSecond(MyWatch))
```

```
    Me.Close()
```

```
End Sub
```

```
Private Function ChaneTickToNanoSecond(MyStopWatch As Stopwatch) As Double
```

```
    Return (MyStopWatch.ElapsedTicks * 1000000000 / Stopwatch.Frequency)
```

```
End Function
```

基本程式架構(比賽用)

```
Private Sub Form1_Load(sender As System.Object, e As System.EventArgs) _  
    Handles
```

```
MyBase.Load
```

```
    Dim Reader(1) As StreamReader, SR As StreamReader
```

```
    Dim SW As New StreamWriter("out.txt")
```

```
    Reader(0) = New StreamReader("in1.txt")
```

```
    Reader(1) = New StreamReader("in2.txt")
```

```
    Dim Sep() As String = {" "}, U As Integer, Fn As Integer
```

```
    Dim I As Integer, J As Integer, K As Integer, L As Integer
```

```
    For Fn = 0 To 1
```

```
        SR = Reader(Fn)
```

```
        .....
```

(主要解題的程式)

```
        .....
```

```
        SW.WriteLine()
```

```
    Next
```

```
    Me.Close()
```

良好解題習慣

Enter ↵

T.C.R.

280-00

解題步驟

❖ 解題步驟

1. 先看完全部題目仔細閱讀並理解
2. 用自己熟悉的概念及術語來重新定義題目
3. 選擇簡單題目先做
4. 定義題目需要使用的演算法及資料結構
5. 想看看有沒有其他演算法。如果有，選擇最簡單的方法，如果解不出來，就用暴力法。
6. 確定好你所要使用演算法及資料結構，才開始寫程式
7. 測試

正確的儲存資料規定

- ❖ 請將資料同時存放於硬碟及隨身碟。
- ❖ 檔名請千萬不要存錯。
- ❖ 儲存格式如有疑惑，請於比賽開始前詢問評審教授。



好的程式習慣

- ❖ 盡量使用 .Net Framework 函數，少寫程式
- ❖ 維持相同的程式設計風格及命名變數的習慣
- ❖ 重複出現三次以上的程式就定義為函數
- ❖ 能用表格，就不要用IF
- ❖ 多用 AndAlso 及OrElse 來取代 And 及 Or
 - If i=-1 Or A(i)=-1
 - If i=-1 OrElse A(i)=-1
- ❖ 如果有需要使用常數，請先定義它在程式的最開頭
- ❖ 選擇合適的字體及大小寫來進行Coding，避免英文O與數字0、1與l、|與l，搞混，迴圈索引變數用大寫I、J、K 取代小寫ij

好的程式習慣

❖ 盡量使用 .Net Framework 函數，少寫程式

- 假設你要升冪排序陣列 B

方法一：自己寫

```
Dim i, j, count As Integer
Dim check As Boolean
For i = 0 To UBound(b) - 1
    check=false
    For j = 0 To UBound(b) - 1 - i
        If b(j) > b(j + 1) Then switch(b(j), b(j + 1))
        count += 1 : check = True
    Next
    If chk = False Then Exit For
Next
Private Sub switch(ByRef a as integer, ByRef b as integer)
Dim c As Integer
c = a : a = b : b = c
End Sub
```

方法二: 用 .Net Framework 函數

```
Array.Sort(B)
```

好的程式習慣

❖ 能用表格，就不要用IF

■ 少用

- If month=1 then return "Jan"
- If month=2 then return "February"
- ...
- If month=12 then return "December"

■ 改用

- MonthName() as String
= {"None", "January", "February", ..., "December"}
- Return(MonthName(month))

解題策略

❖ 養成良好編碼習慣，盡量減少Trival Error

■ 範例一

- 寫法一

```
return (x*y)+Func(t)/(1-s)
```

- 寫法二

```
temp = func(t)
```

```
RetVal = (x*y) + temp/(1-s)
```

```
return RetVal
```



❖ 養成良好編碼習慣，盡量減少 Trival Error

■ 範例一

- 寫法一

```
return (If(((x*8-111) mod 7)>5) , y , 8-x)
```

- 寫法二

```
Temp = ((x*8-111) mod 7)
```

```
If (5<Temp) Then
```

```
    return y
```

```
else
```

```
    return 8-x
```

```
End if
```

請注意一下

LOGO

❖ Counter = ?

```
Module Module1
```

```
    Const Max As Integer = 100000
```

```
    Sub Main()
```

```
        Dim I As Integer, Counter As Integer = 0
```

```
        For I = 1 To Max
```

```
            Dim Y As Double = 1.0 / I
```

```
            If I * Y = 1.0 Then Counter = Counter + 1
```

```
        Next
```

```
        Console.WriteLine(Counter)
```

```
        Console.Read()
```

```
    End Sub
```

```
End Module
```

盡量不要使用使用浮點數

❖ Counter = ?

```
Module Module1
```

```
    Const Max As Integer = 100000
```

```
    Sub Main()
```

```
        Dim I As Integer, Counter As Integer = 0
```

```
        For I = 1 To Max
```

```
            Dim Y As Double = 1.0 / I
```

```
            If I * Y = 1.0 Then Counter = Counter + 1
```

```
        Next
```

```
        Console.WriteLine(Counter)
```

```
        Console.Read()
```

```
    End Sub
```

```
End Module
```

❖ Counter = 86884

常見程式問題

LOGO

❖ Off By One Error

- 假設有一個籬笆長100公尺，每10公尺立一個柱子，請問總共要立多少柱子？

如何測試你的程式

❖ 測試資料

1. 使用比賽給的測試資料
2. 測試題目的臨界值，避免Off By One Error
3. 至少連續相同的兩筆測試資料
4. 測試已知正確答案的大數據
5. 分析題目可能有錯的部分
6. 亂數產生測試數據

至少連續相同的兩筆測試資料

❖ 求 $\text{Sum} = 1+2+3+ \dots + 10$

Dim K as Integer

Dim I as Integer, Sum as Integer

For K = 1 to 2

 For I = 1 to 10

 Sum = Sum+I

 Next I

Next

何時用遞迴或迴圈方式

- ❖ 如果解題概念比較簡單，請使用迴圈
 - Factorial、Fibonacci
- ❖ 如果解題概念比較複雜，請使用遞迴
 - Tower of Hanoi, Searching (DFS, BFS)

問題在那裡？

```
Sub Main()
```

```
    Dim T as Integer = LCM(50000,100000)
```

```
End Sub
```

```
Private Function LCM(ByVal N1 As Integer, ByVal N2  
As UInteger) As UInteger
```

```
    Return (N1* N2) / GCD(N1, N2)
```

```
End Function
```

小心Overflow

```
Sub Main()
```

```
    Dim T as Integer = LCM(50000,100000)
```

```
End Sub
```

```
Private Function LCM(ByVal N1 As Integer, ByVal N2  
As UInteger) As UInteger
```

```
    Return (N1 * / GCD(N1, N2)*N2)
```

```
End Function
```

解題考慮

1. 你遇到那一型的Bug，有沒有線索
2. 要不要修改資料結構或演算法
3. 你要Debug還是寫其他題目
4. 要不要解之前放棄的題目
5. 你還有多少時間可以用來解題

比賽常用函數

Enter ↵

T.C.R.

280-00

解題常用類別

❖ 常用類別

- Array
- String
- Convert
- Math

2、8、10、16進位轉換

❖ Convert.ToString(Value As Integer, Base As Integer)

- Base：傳回值的基底，必須是 2、8、10 或 16
- Value：要轉換的 32 位元帶正負號的整數
- 傳回值：Value所對應的Base為底數的字串
- Example
 - `Convert.ToString(100, 8) = "144"`

❖ Convert.ToUInt32(Value As String, Base As Integer)

- Base：傳回值的基底，必須是 2、8、10 或 16
- Value：要轉換的 32 位元帶正負號的整數
- 傳回值：Value所對應的Base為底數的數字
- Example
 - `Convert.ToUInt32("100101", 2) = 37`

解題常用類別

❖ System Collection

- ArrayList
- SortList
- Queue
- Stack

❖ System

- Math
- Array
- String

佇列(queue)

- ❖ **Def:** 具有**FIFO (first in-first out)** 性質的有序串列。其插入元素的動作稱為發生在**Rear (尾)端**, 刪除元素的動作發生在**Front (前)端**.



如何在VB2010中使用 Queue

❖ 佇列(Queue)常用的方法(函數) 如下

Dequeue	從佇列前端取出一個項目，同時將其移除
Enqueue	從佇列尾端加入一個項目
Peek	從佇列前端取出一個項目，但不移除

❖ 佇列(Queue)常用的屬性(變數) 如下

Count	取得佇列中目前的項目數量

如何在VB2010中使用 Queue

❖ 匯入 **System.Collections** 命名空間

- Imports System.Collections

❖ 引用 **Queue** 類別

- Dim myQueue As Queue(Of Integer) = New Queue (Of Integer)

❖ 用 **Enqueue** 加入佇列

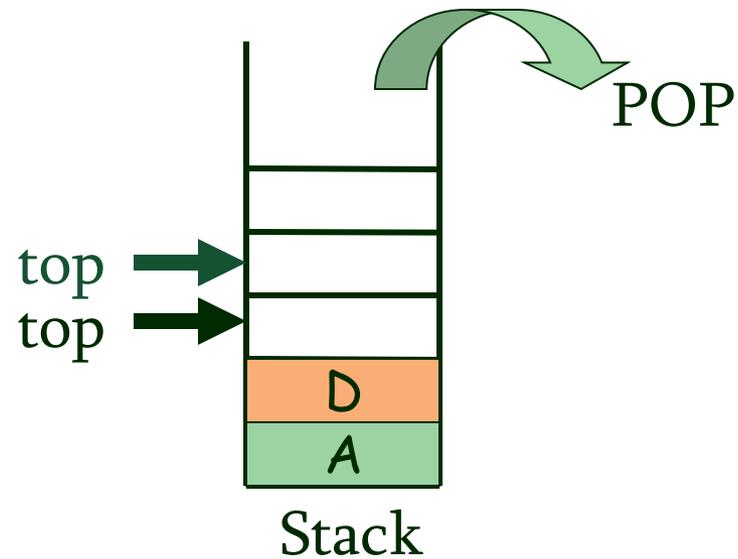
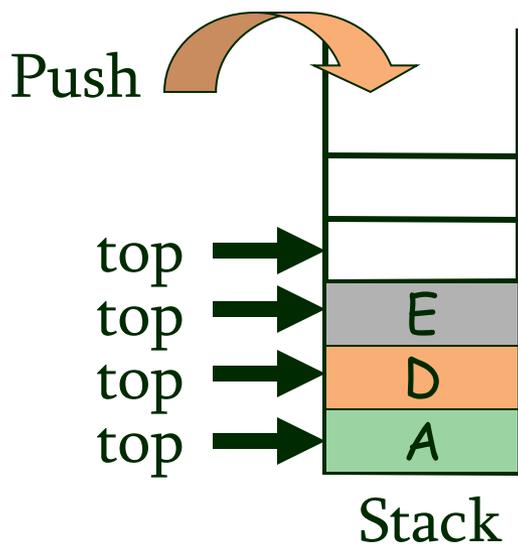
- myQueue.Enqueue(2012)

❖ 用 **Dequeue** 輸出資料

- item = myQueue.Dequeue()

堆疊(Stack)

- ❖ 將資料依序從堆疊下面儲存起來，並視需要從堆疊的上面將資料取出的方式之資料結構，稱為堆疊。



如何在VB2010中使用 Stack

❖ 堆疊(Stack)常用的方法(函數) 如下

Pop	從堆疊最頂端取出一個項目，同時將其移除
Push	從堆疊最頂端加入一個項目
Peek	從堆疊最頂端取出一個項目，但不移除

❖ 堆疊(Stack)常用的屬性 (變數) 如下

Count	取得堆疊中目前的項目數量

如何在VB2010中使用 Stack

❖ 匯入 **System.Collections** 命名空間

- Imports System.Collections

❖ 引用 **Stack** 類別

- Dim myStack As Stack(Of Integer) = New Stack (Of Integer)

❖ 用 **Push** 加入佇列

- myStack.push(2012)

❖ 用 **POP** 輸出資料

- item = myStack.pop()



104學年模擬試 題解析

Enter ↵

T.C.R.

280-00

Problem 1：數學問題

LOGO

❖ 子題1：質因數分解

■ 解題重點

- 使用Eratosthenes' Sieve，建立質數表

Problem 1 : 數學問題

❖ 子題2 : N! 尾數的 0 個數

■ 解題重點

1. $N! = 2^n \times 3^m \times 5^p \times \dots$ ($n > p$)

2. $10 = 2 \times 5$

3. N! 尾數的 0 個數 = p

Problem 2：其他

❖ 子題1：計程車費率計算

- 解題重點
 - 浮點數運算誤差

Problem 2 : 其他

❖ 迴文

- 解題重點
 - `String.toCharArray()`
 - `Array.Reverse()`

Problem 3：資料結構—樹

❖ 子題1：是否為樹

- 解題重點
- Tree的條件
 1. BFS Tree的點數 = Tree中所有點數
 2. Tree中所有邊數+1=Tree中所有點數

Problem 3：資料結構—樹

LOGO

❖ 二元搜尋樹的路徑長度最長的值

❖ 解題重點

❖ 使用 SortedList 來代表連接關係

Problem 4：其他

❖ 子題1：加減問題

■ 解題重點

• 鴿籠原理

- 若有 n 個籠子和 $n+1$ 隻鴿子，所有的鴿子都被關在鴿籠裡，那麼至少有一個籠子有至少2隻鴿子
- 將3個數字($C \geq B \geq A$)視為鴿子，正數及負數分別視為籠子，又3個正負數字和必須等於0，所以 $C=A+B$



Problem 4 : 其他

❖ 子題2：數字反轉後相加

■ 解題重點

- `String.toCharArray()`
- `Array.Reverse()`
- `Integer.Parse()` VAL

101-103學年度 試題類型

Enter



T.C.R.

280-00

101-103學年度試題類型

1. 質數(Eratosthenes' Sieve)
2. Cramer's Rule
3. 中序式轉後序式、後序式
4. 老鼠走迷宮
5. 找零錢 (動態規劃)
6. 撲克牌
7. 所有排列(暴力法)
8. 判斷是否為樹，路徑長度、內部節點、樹葉
9. 字串處理 (ArrayList, String, SortedList)

Eratosthenes' Sieve

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

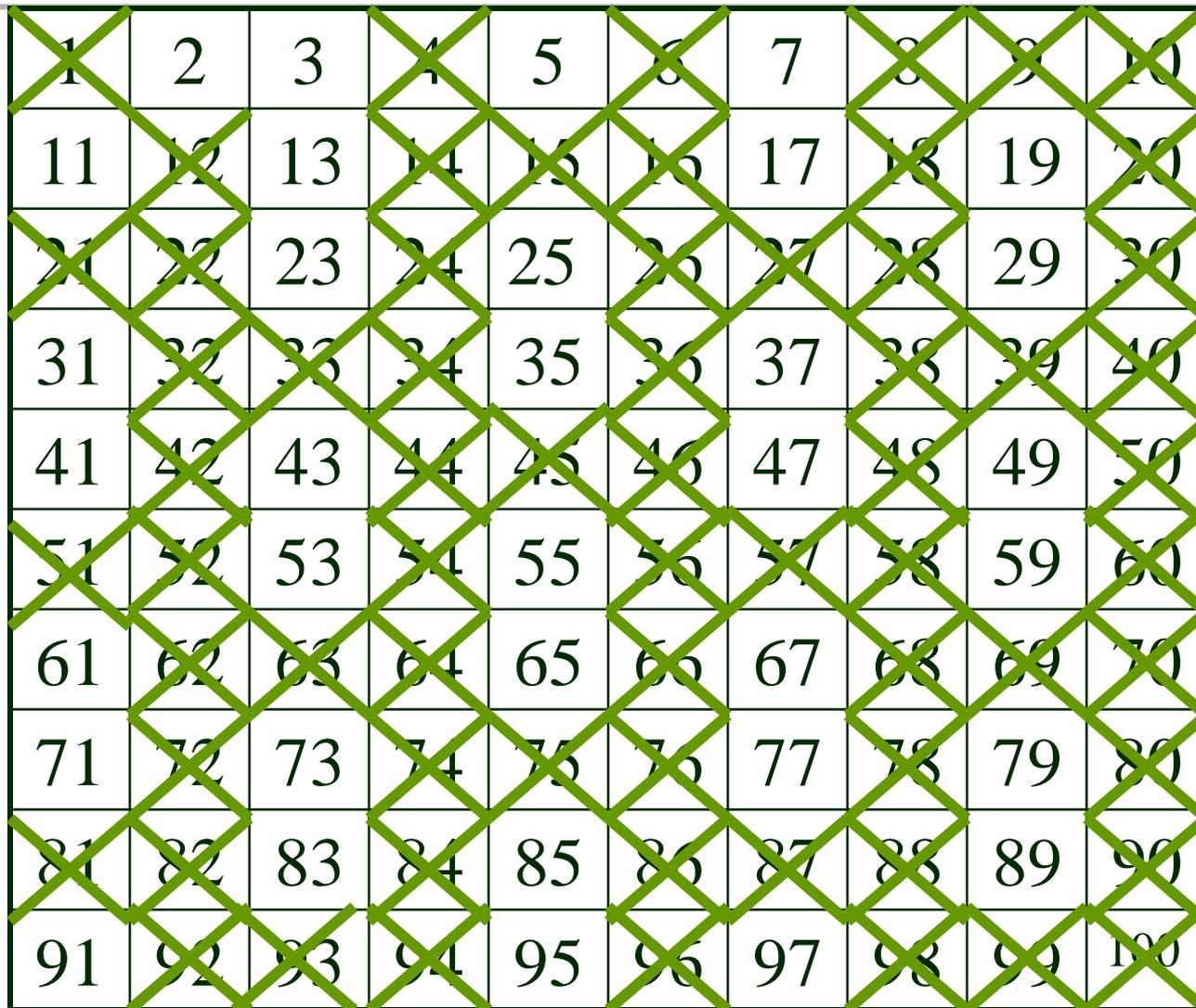
Eratosthenes' Sieve

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Eratosthenes' Sieve

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Eratosthenes' Sieve



1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Eratosthenes' Sieve

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Eratosthenes' Sieve

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Eratosthenes' Sieve

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Eratosthenes' Sieve

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Cramer's Rule

$$\text{若令 } \Delta = \begin{vmatrix} a_1 & b_1 \\ a_2 & b_2 \end{vmatrix}, \Delta_x = \begin{vmatrix} c_1 & b_1 \\ c_2 & b_2 \end{vmatrix}, \Delta_y = \begin{vmatrix} a_1 & c_1 \\ a_2 & c_2 \end{vmatrix}$$

則二元一次方程組 $\begin{cases} a_1x + b_1y = c_1 \\ a_2x + b_2y = c_2 \end{cases}$ 以二階行列式表示的

解為 (1) 當 $\Delta \neq 0$ 時 \Rightarrow 恰有一組解 $x = \frac{\Delta_x}{\Delta}$, $y = \frac{\Delta_y}{\Delta}$ 。

(2) 當 $\Delta = \Delta_x = \Delta_y = \mathbf{0}$ \Rightarrow 有無限多組解。

(3) 當 $\Delta = 0$, Δ_x 與 Δ_y 有任一個不為 0 \Rightarrow 無解。

中序式轉後序式

- ❖ 由左至右讀取中序式中元素 T
- ❖ 如果 T 是運算式結束符號，則輸出所有 Stack 中元素至後序式 Postfix 尾端
- ❖ 如果 T = "("，將 T 放入 Stack
- ❖ 如果 T 是運算元，將 T 輸出到後序式 Postfix 尾端
- ❖ 如果 T 是 ")"，則將 Stack 中的運算子輸出至後序式 Postfix 尾端，直到遇見 "("
- ❖ 如果 T 是運算子，和堆疊頂端運算子 P 比較優先順序。如果 T 的優先序 > P 的優先序，則將 T 放入 Stack。否則將 Stack 中所有優先序大於或等於 T 的運算子輸出到後序式 Postfix 尾端，或者是 Stack 中沒有任何其他元素，再將 T 輸入到 Stack 中。
- ❖ "(" 在堆疊中優先序最小。

後序式運算

- ❖ 由左至右讀取後序式中元素 T
- ❖ 如果 T 是運算元，將 T 放入 Stack
- ❖ 如果 T 是運算子，則由堆疊中取出兩個運算元進行對應的運算，然後將結果存回堆疊
- ❖ 當運算式讀取完畢，堆疊中的值就是運算結果

老鼠走迷宮

Algorithm Maze (*maze*, *start*)

Find the goal in a maze using a queue (breadth first search)

create an empty queue named *Q*;

enqueue *start* in *Q*;

while *Q* is not empty **do**

current ← dequeue from *Q*;

if *current* is the goal **then**

 output “Success!”;

Q ← empty queue; {to end the loop}

else if *current* is not a wall and *current* is not marked as visited **then**

 mark *current* as visited;

 enqueue in *Q* the point to the right of *current*;

 enqueue in *Q* the point to the left of *current*;

 enqueue in *Q* the point above *current*;

 enqueue in *Q* the point below *current*;

end if

end while

老鼠走迷宮

Algorithm *Maze (maze, start)*

Find the goal in a maze using a stack (depth first search)

```
create an empty stack named S;  
push start onto S;  
while S is not empty do  
  current ← pop from S;  
  if current is the goal then  
    output "Success!";  
    S ← empty stack; {to end the loop}  
  else if current is not a wall and current is not marked as visited then  
    mark current as visited;  
    push onto S the point to the right of current;  
    push onto S the point to the left of current;  
    push onto S the point above current;  
    push onto S the point below current;  
  end if  
end while
```

撲克牌 (同花順)

		1	2	3	4	5	6	7	8	9	10	11	12	13	14
黑桃	0	1									1	1	1	1	1
紅桃	1														
方塊	2														
梅花	3														
合計	4	1									1	1	1	1	1

撲克牌 (順子)

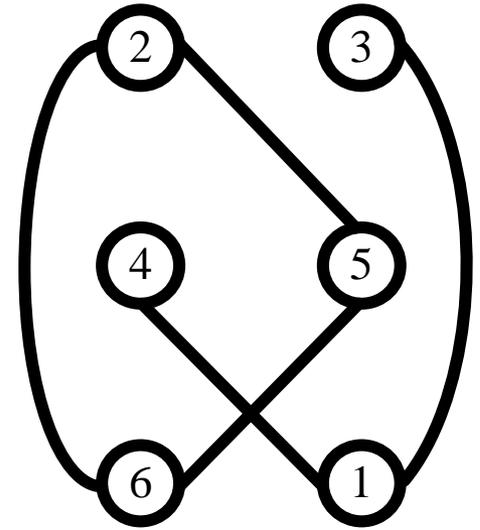
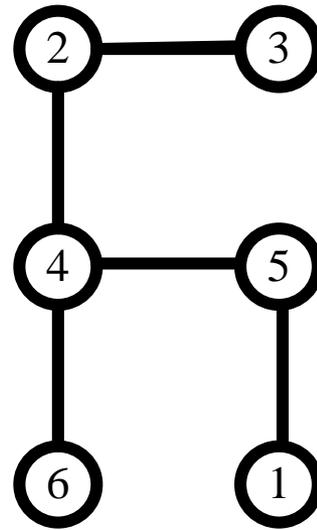
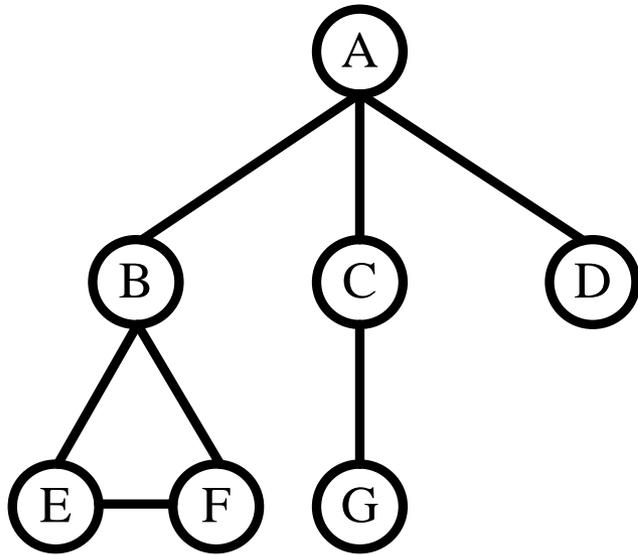
		1	2	3	4	5	6	7	8	9	10	11	12	13	14
黑桃	0						1	1	1						
紅桃	1				1										
方塊	2					1									
梅花	3														
合計	4				1	1	1	1	1						

撲克牌 (葫蘆)

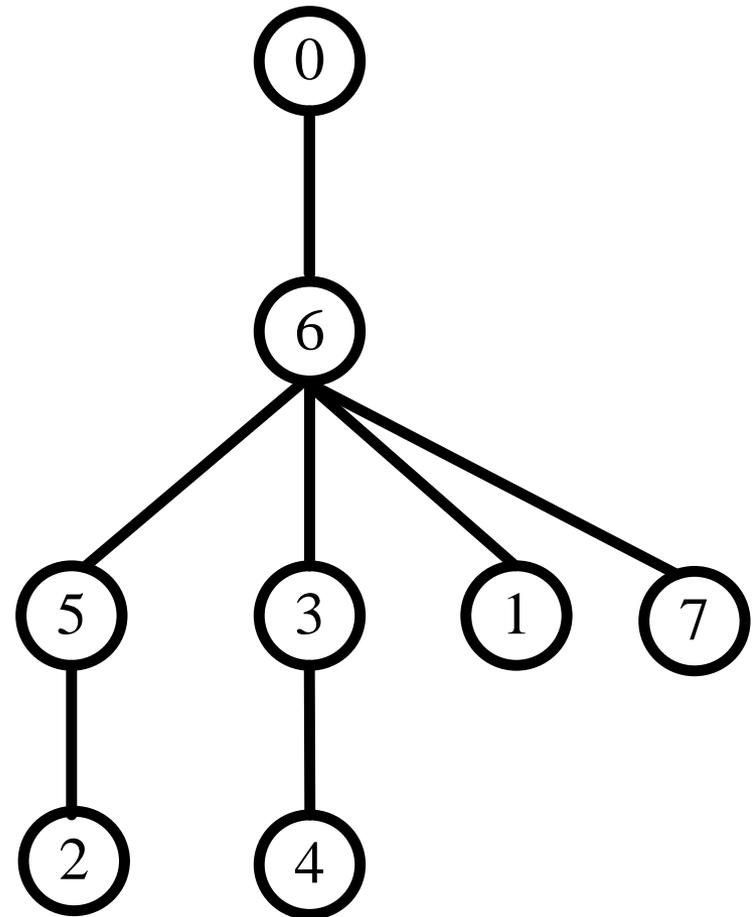
LOGO

		1	2	3	4	5	6	7	8	9	10	11	12	13	14
黑桃	0														
紅桃	1												1	1	
方塊	2												1	1	
梅花	3												1		
合計	4												3	2	

樹



樹 (樹葉、內部節點)



參考資料及網站

Enter



T.C.R.

280-00

參考網站

❖ 演算法筆記

- <http://www.csie.ntnu.edu.tw/~u91029/book.html>

❖ 非關語言：常見程式演算

- <http://openhome.cc/Gossip/AlgorithmGossip/>

❖ Lucky貓的 UVA (ACM) 園地

- <http://luckycat.kshs.kh.edu.tw/>

❖ Ruby兔的ACM園地

- <http://rubyacm.blogspot.com/>

❖ UVA Tool Kit

- <http://uvatoolkit.com/>

參考網站

❖ NPSC補完計劃

- <http://www3.tcgs.tc.edu.tw/npsc/>

❖ USACO Training

- <http://ace.delos.com/usacogate>

❖ IOI

- <http://www.ioinformatics.org/index.shtml>

參考網站

LOGO

❖ USACO Training (中文簡體)

- http://www.nocow.cn/index.php/USACO_Training

❖ USACO Training (中文簡體)

- <http://www.wzoi.org/usaco/>

❖ ACM ICPC <http://www.acm-icpc.tw/>

❖ Nordic Collegiate Programming Contest

- <https://ncpc.idi.ntnu.no/>

❖ The Central European Olympiad in Informatics

- <http://ceoi.inf.elte.hu/>

❖ Internet Problem Solving Contest

- (<http://ipsc.ksp.sk/>)

❖ CROATIAN OPEN COMPETITION IN INFORMATICS

- <http://hsin.hr/coci/>

參考書籍

Enter ↵

T.C.R.

280-00

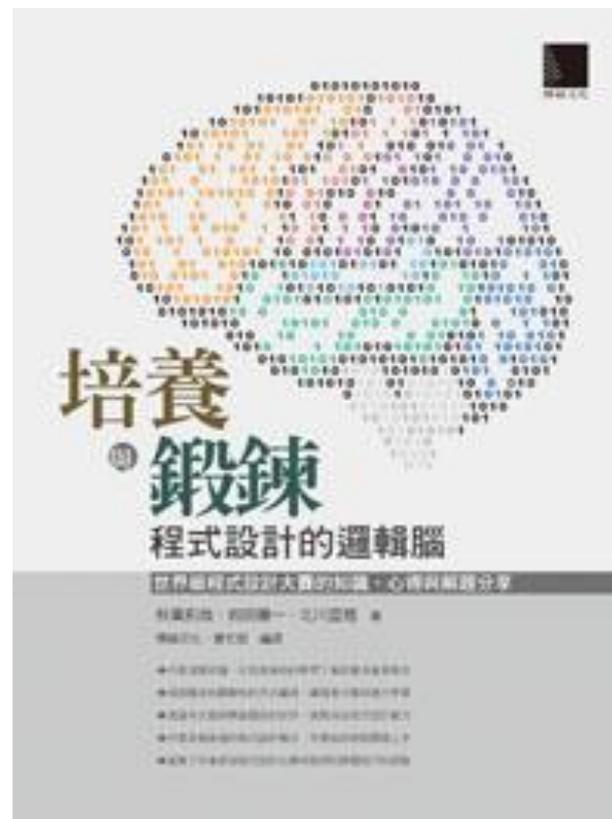
參考書籍

- ❖ 書名：演算法之道--讓你學不會演算法都難
- ❖ 作者：鄒恆明
- ❖ 出版社：碁峯出版社



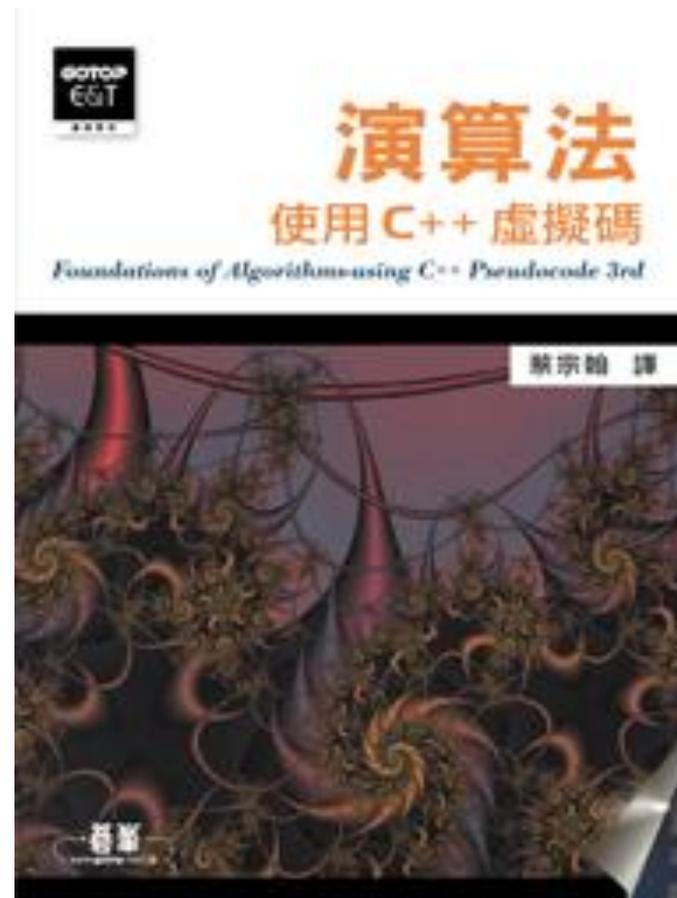
參考書籍

- ❖ 書名：培養與鍛鍊程式設計的邏輯腦：世界級程式設計大賽的知識、心得與解題分享
- ❖ 作者：秋葉拓哉、岩田陽一、北川宜稔
- ❖ 出版社：博碩文化



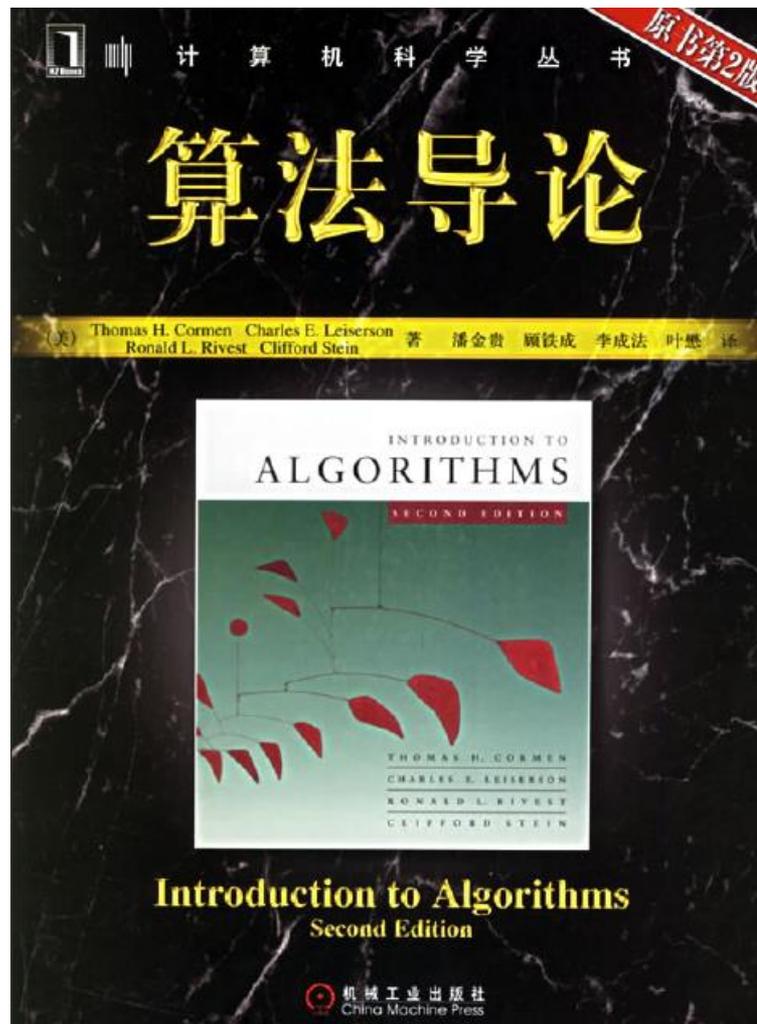
參考書籍

- ❖ 書名：演算法--使用C++虛擬碼
- ❖ 作者：Richard Neapolitan
- ❖ 出版社：基峯出版社



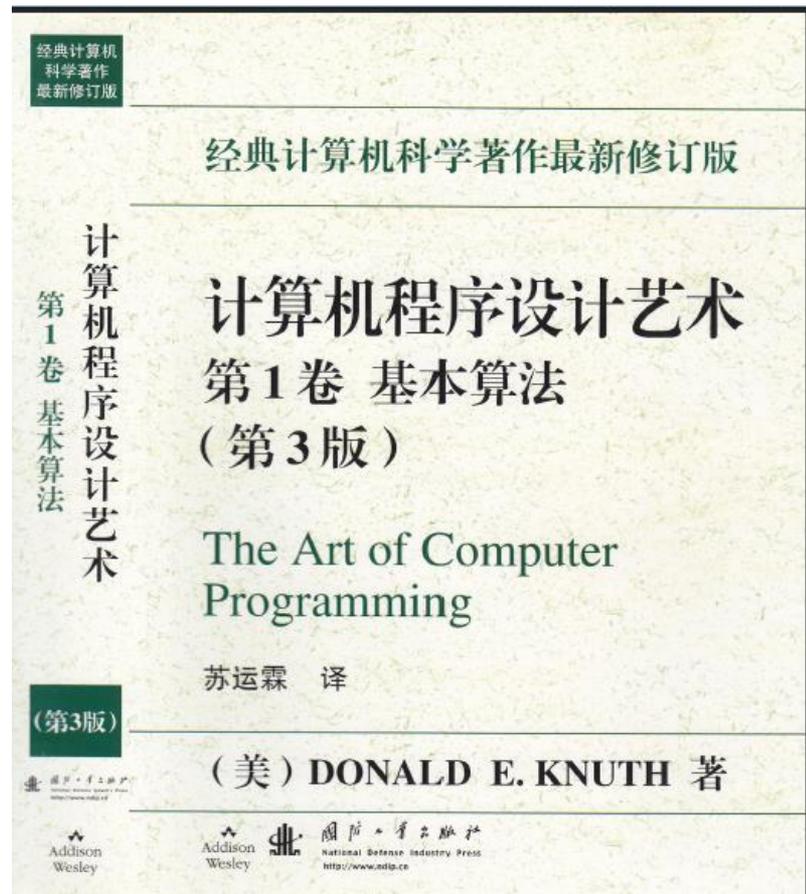
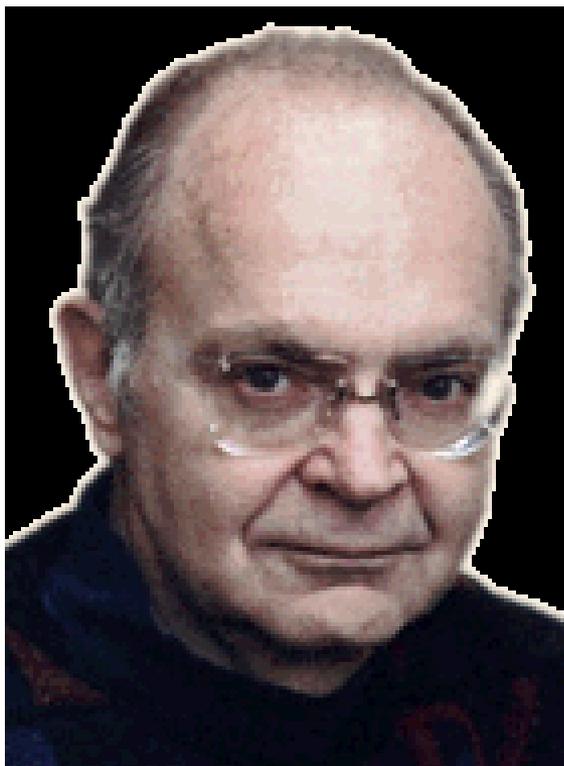
參考書籍

- ❖ 書名：算法導論
- ❖ 出版社：機械工業出版社



參考書籍

- ❖ 書名：計算機程序設計藝術
- ❖ 出版社：國防工業出版社



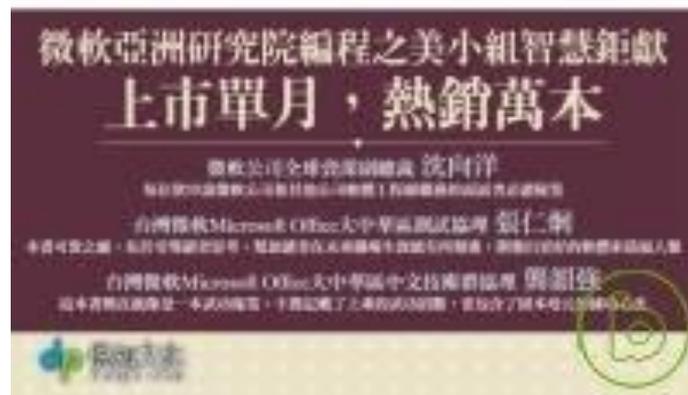
參考書籍

- ❖ 書名：名題精選百則:技巧篇
- ❖ 作者：冼鏡光
- ❖ 出版社：儒林出版社



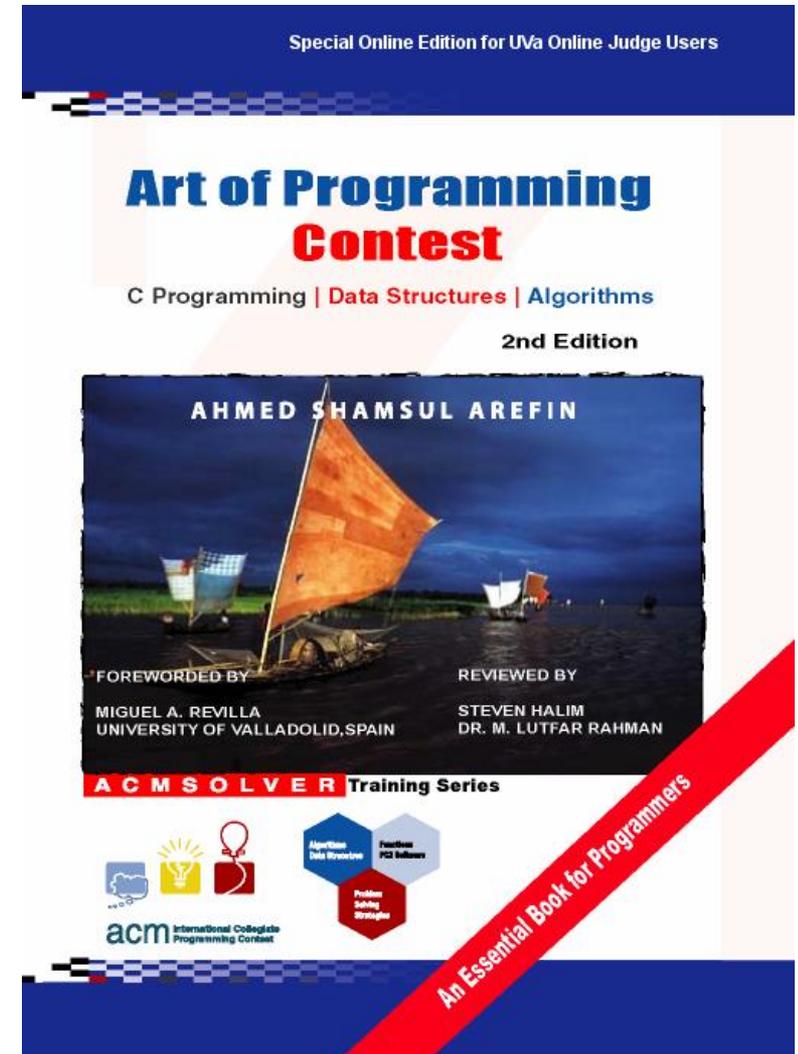
參考書籍

- ❖ 書名：程式之美
- ❖ 作者：《編程之美》小組
- ❖ 出版社：悅知文化



參考書籍

- ❖ 書名：Art of Programming Contest
- ❖ 作者：Ahmed Shamsul Arefin
- ❖ 下載網址：http://www.comp.nus.edu.sg/~stevenha/database/Art_of_Programming_Contest_SE_for_uva.pdf



參考書籍

- ❖ 書名: 啊哈！圖解演算法必學基礎
- ❖ 作者: 啊哈磊
- ❖ 出版社: 碁峯出版社



參考書籍

- ❖ 書名:打下好基礎—程式設計與演算法競賽入門經典
- ❖ 作者:劉汝佳
- ❖ 出版社:碁峯出版社



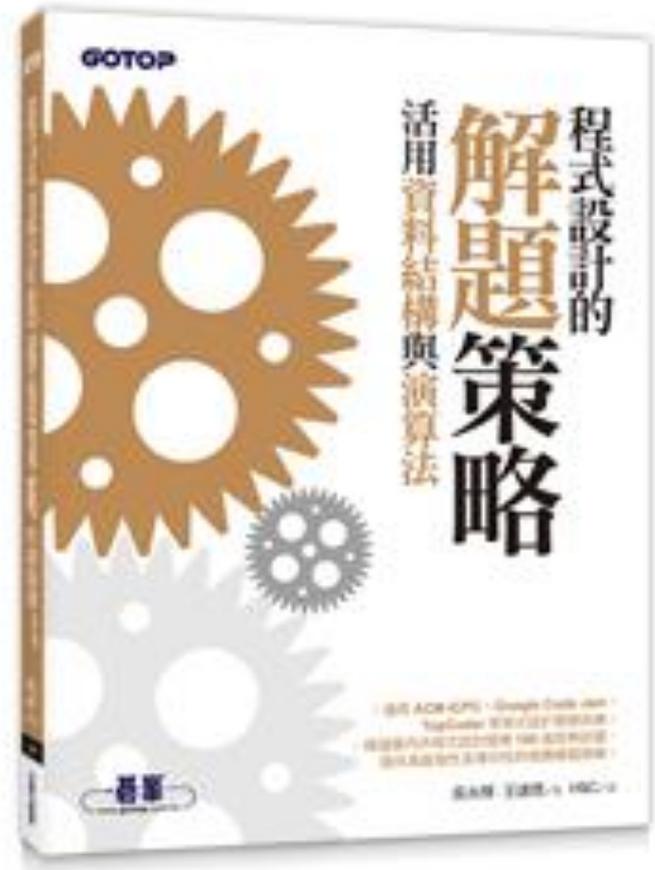
參考書籍

- ❖ 書名:提升程式設計的解題思考力--國際演算法程式設計競賽訓練指南
- ❖ 作者:劉汝佳/陳鋒
- ❖ 出版社:碁峯出版社



參考書籍

- ❖ 書名:程式設計的解題策略—活用資料結構與演算法
- ❖ 作者:吳永輝/王建德
- ❖ 出版社:碁峯出版社



參考書籍

- ❖ 書名:提升程式設計的資料結構力 — 國際程式設計競賽之資料結構原理、題型、解題技巧與重點解析
- ❖ 作者:吳永輝/王建德
- ❖ 出版社:碁峯出版社



歡迎指教

LOGO

