迴圈複習
&
結構式程式設計

資訊之芽語法班 2015 suhorng

重複動作

- **for** 通常被用來重複執行一些陳述句數次 (次數通常已知, 如 10, n, $(m \times 3 + 1)$)

  $$\begin{array}{l} \textbf{int } sum = 0; \\ \textbf{for } (\textbf{int } i = 0;\ \ i \le 3 \times n + 1;\ \ i = i + 1) \\ \quad sum = sum + i \times i; \end{array}$$

- 對比: $\displaystyle\sum_{i=0}^{3\times n+1} i^2 = 0^2 + 1^2 + 2^2 + \cdots + (3 \times n + 1)^2$

- 類比...

  $$\begin{array}{l} \textbf{int } sum = 0; \\ sum = sum + 0 \times 0; \\ sum = sum + 1 \times 1; \\ \ldots \\ sum = sum + (3 \times n + 1) \times (3 \times n + 1); \end{array}$$

程式片段: 迴圈與累積的值

$\textbf{int}\ acc = \text{初始值};$

$\textbf{for}\ (\textbf{int}\ i = begin;\ \ i \neq end;\ \ i = i + 1)\ \{$

$\quad //\ e.g.\ \ f\ \text{可以是某些算式如}\ (+), (\times), \ldots$

$\quad acc = f(acc, i);$

$\}$

$//\ acc\ \text{存著最後結果}$

- acc 等於 $f(\ldots f(f(\text{初始值}, begin), begin + 1) \ldots, end - 1)$
- 更多例子
  - 給定**整數** a, b, c 其中 $b, c > 0$, 求出 $a^b \textbf{ mod } c$
  - 利用 $(x \times y) \textbf{ mod } z = ((x \textbf{ mod } z) \times (y \textbf{ mod } z)) \textbf{ mod } z$ 邊乘邊取餘數避免溢位

程式片段: 找最大值

- 給定整數陣列 `int arr[100];`, 找出其中的最大值

```
int arr[100], m = arr[0];
for (int i = 1; i < 100; i = i+1) {
  if (arr[i] > m) {
    m = arr[i];
  }
}
```

- acc 是什麼(且代表什麼)? "算式" 是什麼?
- 為何 i 從 1 開始?

程式片段: 找最大值

- 給定整數陣列 `int arr[100];`, 找出最大值出現位置

```
int arr[100], max_idx = 0;
for (int i = 1; i < 100; i = i+1) {
  if (arr[i] > arr[max_idx]) {
    max_idx = i;
  }
}
```

- acc 是什麼(且代表什麼)? "算式" 是什麼?
- 為何 max_idx 從 0 開始?

歸納法與遞迴

```
int arr[100], m = arr[0];
for (int i = 1; i < 100; i = i+1)
  if (arr[i] > m)
    m = arr[i];
```

- Challenge: 說明這個程式會算出最大值
- 跑出來就這樣阿! 假設跑到某個 $1 \leq k < 100$, 則:
  - m 是到目前為止的最大值 (除了 $arr[k]$)
  - 若 $arr[k] > m$, 顯然 $arr[k]$ 是新的最大值. 否則 m 不變.
  - 所以當迴圈結束時 m 存著最大值

歸納法與遞迴

```
int arr[100], m = arr[0];
for (int i = 1; i < 100; i = i+1)
  if (arr[i] > m)
    m = arr[i];
```

- 至於你信不信, 我反正是信了

1. “當它跑到某個 $1 \leq k < 100$, m 是到目前為止的最大值”
   - 我們假設當 $i := k$ 時 m 就是 $arr[0], \ldots, arr[k-1]$ 中的最大值.
   - 我們要證當 $i := n$ 迴圈結束時 m 是 $arr[0], \ldots, arr[n-1]$ 中的最大值

1. “若 $arr[k] > m$ 則...”: 給定 k, 我們有 $m := \max\{\, m, arr[k] \,\}$

歸納法與遞迴

```
int arr[100], m = arr[0];
for (int i = 1; i < 100; i = i+1)
  if (arr[i] > m)
    m = arr[i];
```

1. 這個假設從何而來? 它是對的嗎?

1. 給定 $arr[0], \ldots, arr[k-1]$ 的最大值, 我們能算出 $arr[0], \ldots, arr[k]$ 的最大值.

- "假設 $P(i)$ 對於 $i := k-1$ 是真的. 由 ... 我們得知 $P(k)$ 是真的. 因此 $P(n)$ 是真的 ~~對所有~~ n."
  - 不就是歸納法嗎!
  - $n = 1$ 時是顯然的 -- 我們一開始就令 $m = arr[0]$

歸納法與遞迴

- 這就是歸納法跟遞迴的用處. 就算沒有去想, 畢竟是用了
- 命題 “m 是 XXX 中的最大值” 稱為 **迴圈不變量** (loop invariant): 它在迴圈初始化後, 每個迭代時, 迴圈結束後都成立.
- 如何計算 $a_1, \ldots, a_n$ 等 n 個數的最大值?
    - 若我們能計算 $a_2, \ldots, a_n$ 等 $n-1$ 個數的最大值, 算出來是 m, 那我們可以顯然看出 $\max\{ a_1, m \}$ 就是答案.
    - 那就直接假設我們能做到 -- 反正 $n=1$ 時我們會做, 這樣變小程式最後一定會終止.
    - 這就是遞迴...和歸納法!

for: "對所有" 和 "存在" (∀, ∃)

- 檢查是否 arr 陣列中 **所有** 元素都滿足 $P(x)$.
- 檢查是否 arr 陣列中 **存在** 元素滿足 $P(x)$.
- 就是計算 $P(arr[0]) \wedge P(arr[1]) \wedge \cdots \wedge P(arr[n-1])$ 跟 $P(arr[0]) \vee P(arr[1]) \vee \cdots \vee P(arr[n-1])$
  - 例如設 $P(x)$ 為 $x\%3 == 0$ (x 是否是 3 的倍數)
  - $a \wedge b \wedge c \wedge d = (((a \wedge b) \wedge c) \wedge d)$

```
int arr[100];
bool curr_cond = true;
for (int i = 0; i < 100; i = i+1)
  curr_cond = curr_cond && (arr[i]%3 == 0);
```

for: "對所有" 和 "存在" (∀, ∃)

- 檢查是否 arr 陣列中 **所有** 元素都滿足 $P(x)$.
- 檢查是否 arr 陣列中 **存在** 元素滿足 $P(x)$.
- 就是計算 $P(arr[0]) \wedge P(arr[1]) \wedge \cdots \wedge P(arr[n-1])$ 跟 $P(arr[0]) \vee P(arr[1]) \vee \cdots \vee P(arr[n-1])$
  - 例如設 $P(x)$ 為 $x\%3 == 0$ (x 是否是 3 的倍數)
  - $a \wedge b \wedge c \wedge d = (((a \wedge b) \wedge c) \wedge d)$

```
int arr[100];
bool curr_cond = true;
for (int i = 0; i < 100; i = i+1)
  if ( !(arr[i]%3 == 0) )
    curr_cond = false;
```

程式片段: 對每個元素獨立的操作

- “對每個 $k \in \{a_0, a_1, \ldots, a_{99}\}$, 做 ...”

```
int arr[100], ans[100];
for (int i = 0; i < 100; i = i+1) {
  int k = arr[i];
  std::cout << "arr[" << i << "] = " << k << "\n";
  ans[i] = k*k + 1;
}
```

程式判斷: 篩選

- “找到 $\{a_0, \ldots, a_{99}\}$ 所有讓 $P(x)$ 成立的 x ”

```
// `ans_total` 代表 `ans` 中元素的個數
int arr[100], ans[100], ans_total = 0;
for (int i = 0; i < 100; i = i+1) {
  if (arr[i]%2 == 0) {
    std::cout << "Find an even number!\n";
    ans[ans_total] = arr[i];
    ans_total = ans_total + 1;
  }
}
```

範例: 結合起來

- 找到陣列中所有的奇數然後把它們乘二

```
int arr[100], ans[100], ans_total = 0;
for (int i = 0; i < 100; i = i+1) {
  if (arr[i]%2 == 1) {
    ans[ans_total] = arr[i];
    ans_total = ans_total + 1;
  }
}
for (int i = 0; i < ans_total; i = i+1)
  ans[i] = ans[i] * 2;
```

- 或把迴圈合起來

```
int arr[100], ans[100], ans_total = 0;
for (int i = 0; i < 100; i = i+1) {
  if (arr[i]%2 == 1) {
    ans[ans_total] = arr[i] * 2;
    ans_total = ans_total+ + 1;
  }
```

Dijkstra 的結構式程式設計

- 用簡單的邏輯把程式寫出來: `if` 跟 `for`
- 用有意義的小片段組出程式...
  - 已知:
    - 質數判定

```
bool found_factor = false;
for (int i = 2; i<p && !found_factor; i = i+1)
  if (p%i == 0) found_factor = true;
```

    - ⇒ 找到所有質數

```
for (int p = 2; p < n; p = p+1) {
  ...
  if (!found_factor) { ... }
}
```

結構式程式設計 - 交換數字

- 每一個小片段都**做某些事**, 有一些**前條件**跟**後條件**
- (Tricky) 交換數字:

$\textbf{int}\ a = \ldots,\ b = \ldots;$

$//$ 前條件 : $a = x \wedge b = y$

$a = a + b;$

$b = a - b;$

$a = a - b;$

$//$ 後條件 : $a = y \wedge b = x$

結構式程式設計 - 交換數字

- 每一個小片段都**做某些事**, 有一些**前條件**跟**後條件**
- (Tricky) 交換數字:

$\textbf{int}\ a = \ldots,\ b = \ldots;$

$//$ 前條件 : $a = x \wedge b = y$

$a = a + b;$

$b = a - b;$

$//\ a - b = y \wedge b = x$

$a = a - b;$

$//$ 後條件 : $a = y \wedge b = x$

結構式程式設計 - 交換數字

- 每一個小片段都**做某些事**, 有一些**前條件**跟**後條件**
- (Tricky) 交換數字:

$\textbf{int } a = \ldots,\ b = \ldots;$

$//$ 前條件 : $a = x \wedge b = y$

$a = a + b;$

$//\ a - (a - b) = y \wedge a - b = x$

$b = a - b;$

$//\ a - b = y \wedge b = x$

$a = a - b;$

$//$ 後條件 : $a = y \wedge b = x$

結構式程式設計 - 交換數字

- 每一個小片段都**做某些事**, 有一些**前條件**跟**後條件**
- (Tricky) 交換數字:

$$\textbf{int } a = \ldots,\ b = \ldots;$$
$$// \text{ 前條件}: a = x \wedge b = y$$

$$a = a + b;$$
$$// \ b = y \wedge a - b = x$$
$$b = a - b;$$
$$// \ a - b = y \wedge b = x$$
$$a = a - b;$$
$$// \text{ 後條件}: a = y \wedge b = x$$

結構式程式設計 - 交換數字

- 每一個小片段都**做某些事**, 有一些**前條件**跟**後條件**
- (Tricky) 交換數字:

$\textbf{int}\ a = \ldots,\ b = \ldots;$

$//\ \text{前條件}: a = x \land b = y$

$//\ b = y \land (a + b)\ - b = x$

$a = a + b;$

$//\ b = y \land a - b = x$

$b = a - b;$

$//\ a - b = y \land b = x$

$a = a - b;$

$//\ \text{後條件}: a = y \land b = x$

結構式程式設計 - 交換數字

- 每一個小片段都**做某些事**, 有一些**前條件**跟**後條件**
- (Tricky) 交換數字:

$\textbf{int } a = \ldots,\ b = \ldots;$

$//\ 前條件：a = x \wedge b = y$

$//\ b = y \wedge a = x$

$a = a + b;$

$//\ b = y \wedge a - b = x$

$b = a - b;$

$//\ a - b = y \wedge b = x$

$a = a - b;$

$//\ 後條件：a = y \wedge b = x$

- 所以這段程式確實正確!

結構式程式設計 - 冪次

$\textbf{int } a = \ldots,\ n = \ldots, r = 1;$

$//$ 前條件：$r = a^0 \wedge n = N$

$\textbf{for } (\textbf{int } i = 0;\ i < n;\ i = i + 1)$

$\{$

$\quad r = r \times a;$

$\}$

$//$ 後條件：$i = N \wedge r = a^i \wedge n = N$

結構式程式設計 - 冪次

$\textbf{int}\ a = \ldots,\ n = \ldots, r = 1;$

// 前條件：$r = a^0 \wedge n = N$

$\textbf{for}\ (\textbf{int}\ i = 0;\ i < n;\ i = i + 1\ /\!*\ r = a^i\ *\!/)$

$\{$

$\quad r = r \times a;$

$\}$

// 後條件：$i = N \wedge r = a^i \wedge n = N$

結構式程式設計 - 冪次

$\textbf{int } a = \ldots,\ n = \ldots, r = 1;$

$//$ 前條件：$r = a^0 \wedge n = N$

$\textbf{for } (\textbf{int } i = 0;\ i < n;\ /{*}\ r = a^{i+1}\ {*}/\ i = i + 1\ /{*}\ r = a^i\ {*}/)$

$\{$

$\quad r = r \times a;$

$\}$

$//$ 後條件：$i = N \wedge r = a^i \wedge n = N$

結構式程式設計 - 冪次

$\textbf{int}\ a = \ldots,\ n = \ldots, r = 1;$

// 前條件：$r = a^0 \wedge n = N$

$\textbf{for}\ (\textbf{int}\ i = 0;\ i < n;\ /\!*\, r = a^{i+1}\, *\!/\ i = i + 1\ /\!*\, r = a^{i}\, *\!/)$

$\{$

$\quad r = r \times a;$

$\quad // \ r = a^{i+1}$

$\}$

// 後條件：$i = N \wedge r = a^i \wedge n = N$

結構式程式設計 - 冪次

$\textbf{int}\ a = \ldots,\ n = \ldots, r = 1;$

// 前條件：$r = a^0 \wedge n = N$

$\textbf{for}\ (\textbf{int}\ i = 0;\ i < n;\ /\!*\ r = a^{i+1}\ */\ i = i + 1\ /\!*\ r = a^i\ */)$

$\{$

  // $r \times a = a^{i+1}$

  $r = r \times a;$

  // $r = a^{i+1}$

$\}$

// 後條件：$i = N \wedge r = a^i \wedge n = N$

結構式程式設計 - 冪次

$\mathbf{int}\ a = \ldots,\ n = \ldots, r = 1;$

// 前條件：$r = a^0 \wedge n = N$

$\mathbf{for}\ (\mathbf{int}\ i = 0;\ i < n;\ /\!*\ r = a^{i+1}\ *\!/\ i = i + 1\ /\!*\ r = a^i\ *\!/)$

$\{$

  // $r = a^i$

  $r = r \times a;$

  // $r = a^{i+1}$

$\}$

// 後條件：$i = N \wedge r = a^i \wedge n = N$

結構式程式設計 - Maximum

- 我們令 $M(k)$ 為 $\max\{a[0], \ldots, a[k]\}$

$\textbf{int } a[100],\ i = 1,\ m = a[0];$

$//$ 前條件：$m = M(0)$

$\textbf{for } (\textbf{int } i = 1;\ i < n;\ i = i + 1)$

$\{$

$\quad \textbf{if } (a[i] > m)\ \{$

$\quad\quad m = a[i];$

$\quad \}$

$\}$

$//$ 後條件：$i = n \wedge m = M(i - 1)$

結構式程式設計 - Maximum

- 我們令 $m(k)$ 為 $\max\{a[0], \ldots, a[k]\}$

$\textbf{int}\ a[100],\ i = 1,\ m = a[0];$

// 前條件：$m = M(0)$

$\textbf{for}\ (\textbf{int}\ i = 1;\ i < n;\ i = i + 1)$

$\{$

$\quad \textbf{if}\ (a[i] > m)\ \{$

$\qquad m = a[i];$

$\quad \}$

$\quad // m = M(i)$

$\}$

// 後條件：$i = n \land m = M(i-1)$

結構式程式設計 - Maximum

- 我們令 $m(k)$ 為 $\max\{a[0], \ldots, a[k]\}$

$\textbf{int } a[100],\ i = 1,\ m = a[0];$

$//$ 前條件：$m = M(0)$

$\textbf{for } (\textbf{int } i = 1;\ i < n;\ i = i + 1)$

$\{$

$\quad \textbf{if } (a[i] > m)\ \{$

$\qquad m = a[i];$

$\quad \}\ //\ \textbf{else},\ \ a[i] \leq m \wedge m = M(i) = M(i-1)$

$\quad //\ m = M(i)$

$\}$

$//$ 後條件：$i = n \wedge m = M(i-1)$

結構式程式設計 - Maximum

- 我們令 $m(k)$ 為 $\max\{a[0], \ldots, a[k]\}$

$\textbf{int}\ a[100],\ i = 1,\ m = a[0];$

$//$ 前條件：$m = M(0)$

$\textbf{for}\ (\textbf{int}\ i = 1;\ i < n;\ i = i + 1)$

$\{$

$\quad \textbf{if}\ (a[i] > m)\ \{$

$\qquad m = a[i];$

$\qquad //\ \textbf{then}\ a[i] > m \wedge m = M(i)$

$\quad \}\ //\ \textbf{else},\ a[i] \leq m \wedge m = M(i) = M(i-1)$

$\quad //\ m = M(i)$

$\}$

$//$ 後條件：$i = n \wedge m = M(i-1)$

結構式程式設計 - Maximum

- 我們令 $m(k)$ 為 $\max\{a[0], \ldots, a[k]\}$

$\textbf{int } a[100],\ i = 1,\ m = a[0];$

$//$ 前條件：$m = M(0)$

$\textbf{for } (\textbf{int } i = 1;\ i < n;\ i = i + 1)$

$\{$

$\quad \textbf{if } (a[i] > m)\ \{$

$\quad\quad //\ \textbf{then } a[i] > m \wedge a[i] = M(i)\ (\Leftarrow a[i] > m \wedge m = M(i-1))$

$\quad\quad m = a[i];$

$\quad\quad //\ \textbf{then } a[i] > m \wedge m = M(i)$

$\quad \}\ //\ \textbf{else},\ a[i] \leq m \wedge m = M(i) = M(i-1)$

$\quad //\ m = M(i)$

$\}$

$//$ 後條件：$i = n \wedge m = M(i-1)$

結構式程式設計 - Maximum

- 我們令 $m(k)$ 為 $\max\{a[0], \ldots, a[k]\}$

$$
\begin{array}{l}
\textbf{int } a[100],\ i = 1,\ m = a[0]; \\
//\ \text{前條件：} m = M(0) \\
\textbf{for } (\textbf{int } i = 1;\ i < n;\ i = i + 1) \\
\{ \\
\quad //\ m = M(i-1) \\
\quad \textbf{if } (a[i] > m)\ \{ \\
\qquad //\ \textbf{then } a[i] > m \wedge a[i] = M(i) \\
\qquad m = a[i]; \\
\qquad //\ \textbf{then } a[i] > m \wedge m = M(i) \\
\quad \}\ //\ \textbf{else},\ a[i] \leq m \wedge m = M(i) = M(i-1) \\
\quad //\ m = M(i) \\
\} \\
//\ \text{後條件：} i = n \wedge m = M(i-1)
\end{array}
$$

結構式程式設計 - 質數測試

$\textbf{bool } flag = \textbf{true};$

$//$ 前條件：$flag = 2 \not| p \wedge \cdots \wedge 1 \not| p$

$\textbf{for } (\textbf{int } i = 2;\ i < p \,\&\&\, flag;\ i = i + 1)\ \{$

$\quad \textbf{if } (p \% i == 0)\ \{$

$\qquad flag = \textbf{false};$

$\quad \}$

$\}$

$//$ 後條件：$flag = 2 \not| p \wedge \cdots \wedge (i-1) \not| p$

結構式程式設計 - 質數測試

$\textbf{bool}\ flag = \textbf{true};$

$//$ 前條件：$flag = 2 \not| p \wedge \cdots \wedge 1 \not| p$

$\textbf{for}\ (\textbf{int}\ i = 2;\ i < p\ \&\&\ flag;\ i = i + 1)\ \{$

$\quad \textbf{if}\ (p \% i == 0)\ \{$

$\quad\quad flag = \textbf{false};$

$\quad \}$

$\quad //\ flag = 2 \not| p \wedge \cdots \wedge i \not| p$

$\}$

$//$ 後條件：$flag = 2 \not| p \wedge \cdots \wedge (i - 1) \not| p$

結構式程式設計 - 質數測試

$\textbf{bool}\ flag = \textbf{true};$

$//$ 前條件：$flag = 2 \nmid p \wedge \cdots \wedge 1 \nmid p$

$\textbf{for}\ (\textbf{int}\ i = 2;\ i < p\ \&\&\ flag;\ i = i + 1)\ \{$

$\quad \textbf{if}\ (p \% i == 0)\ \{$

$\qquad flag = \textbf{false};$

$\quad \}\ //\ \textbf{else} : i \nmid p \Rightarrow flag = 2 \nmid p \wedge \cdots \wedge i \nmid p = 2 \nmid p \wedge \cdots \wedge (i-1) \nmid p$

$\quad //\ flag = 2 \nmid p \wedge \cdots \wedge i \nmid p$

$\}$

$//$ 後條件：$flag = 2 \nmid p \wedge \cdots \wedge (i-1) \nmid p$

結構式程式設計 - 質數測試

$\textbf{bool}\ flag = \textbf{true};$

$//$ 前條件：$flag = 2 \not| p \wedge \cdots \wedge 1 \not| p$

$\textbf{for}\ (\textbf{int}\ i = 2;\ i < p\, \&\&\ flag;\ i = i + 1)\ \{$

$\quad \textbf{if}\ (p \% i == 0)\ \{$

$\qquad flag = \textbf{false};$

$\qquad //\ (i \mid p) \wedge flag = 2 \not| p \wedge \cdots \wedge i \not| p$

$\quad \}\ //\ \textbf{else} : i \not| p \Rightarrow flag = 2 \not| p \wedge \cdots \wedge i \not| p = 2 \not| p \wedge \cdots \wedge (i-1) \not| p$

$\quad //\ flag = 2 \not| p \wedge \cdots \wedge i \not| p$

$\}$

$//$ 後條件：$flag = 2 \not| p \wedge \cdots \wedge (i-1) \not| p$

結構式程式設計 - 質數測試

$\textbf{bool}\ flag = \textbf{true};$

$//$ 前條件：$flag = 2 \nmid p \wedge \cdots \wedge 1 \nmid p$

$\textbf{for}\ (\textbf{int}\ i = 2;\ i < p\, \&\&\ flag;\ i = i + 1)\ \{$

$\quad \textbf{if}\ (p\%i == 0)\ \{$

$\qquad // \ (i \mid p) \wedge \textbf{false} = 2 \nmid p \wedge \cdots \wedge i \nmid p$

$\qquad\qquad (\Leftarrow (i \mid p) \wedge flag = 2 \nmid p \wedge \cdots \wedge (i-1) \nmid p)$

$\qquad flag = \textbf{false};$

$\qquad // \ (i \mid p) \wedge flag = 2 \nmid p \wedge \cdots \wedge i \nmid p$

$\quad \} \ // \ \textbf{else} : i \nmid p \Rightarrow flag = 2 \nmid p \wedge \cdots \wedge i \nmid p = 2 \nmid p \wedge \cdots \wedge (i-1) \nmid p$

$\quad // \ flag = 2 \nmid p \wedge \cdots \wedge i \nmid p$

$\}$

$//$ 後條件：$flag = 2 \nmid p \wedge \cdots \wedge (i-1) \nmid p$

結構式程式設計 - 質數測試

$\textbf{bool}\ flag = \textbf{true};$

$//$ 前條件：$flag = 2\nmid p \wedge \cdots \wedge 1\nmid p$

$\textbf{for}\ (\textbf{int}\ i = 2;\ i < p\ \&\&\ flag;\ i = i + 1)\ \{$

$\quad // \ flag = 2\nmid p \wedge \cdots \wedge (i-1)\nmid p$

$\quad \textbf{if}\ (p\%i == 0)\ \{$

$\qquad // \ (i \mid p) \wedge \textbf{false} = 2\nmid p \wedge \cdots \wedge i\nmid p$

$\qquad flag = \textbf{false};$

$\qquad // \ (i \mid p) \wedge flag = 2\nmid p \wedge \cdots \wedge i\nmid p$

$\quad \}\ // \ \textbf{else} : i\nmid p \Rightarrow flag = 2\nmid p \wedge \cdots \wedge i\nmid p = 2\nmid p \wedge \cdots \wedge (i-1)\nmid p$

$\quad // \ flag = 2\nmid p \wedge \cdots \wedge i\nmid p$

$\}$

$//$ 後條件：$flag = 2\nmid p \wedge \cdots \wedge (i-1)\nmid p$