韌體實驗第14題

Ting-Yu Lin, Adjunct Lecturer (林庭宇)

Office: AMOS Lab E517

Email: tonylin0413@gmail.com

AMOS Lab.
Advanced Mixed-Operation System Lab.
Dept. of Electrical and Computer Engineering,
Tamkang University, Taiwan

BIDAS Technology
GrounTruth Workshop
Computer Vision
Image Processing
Edge Computing

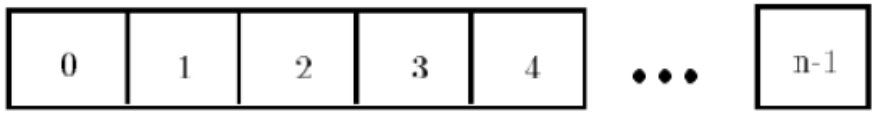
題目解析 程式詳解 輸出結果

第十四題

淡江大學電機工程學系106學年度韌體實驗題目

14. 【木塊問題/The Blocks Problem】

從左到右有 n 個木塊，編號為 0~n-1（如圖），$0 < n < 25$。要求模擬以下 4 種操作（下面的 a 和 b 都是木塊編號）。



(1) move a onto b：把 a 和 b 上方的木塊全部歸位，然後把 a 放在 b 上面。

(2) move a over b：把 a 上方的木塊全部歸位，然後把 a 放在 b 所在木塊堆的頂部。

(3) pile a onto b：把 b 上方的木塊全部歸位，然後把 a 及上面的木塊整體放在 b 上面。

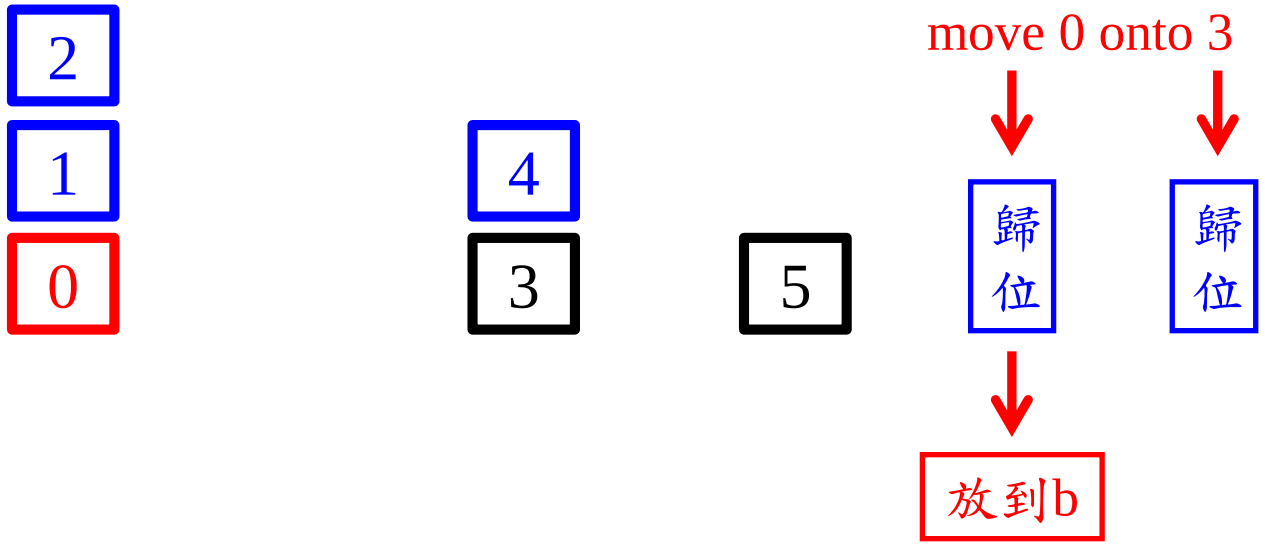
(4) pile a over b：把 a 及上面的木塊整體放在 b 所在木塊堆的頂部。

注意：a 和 b 在同一堆的指令是非法指令，應當忽略。

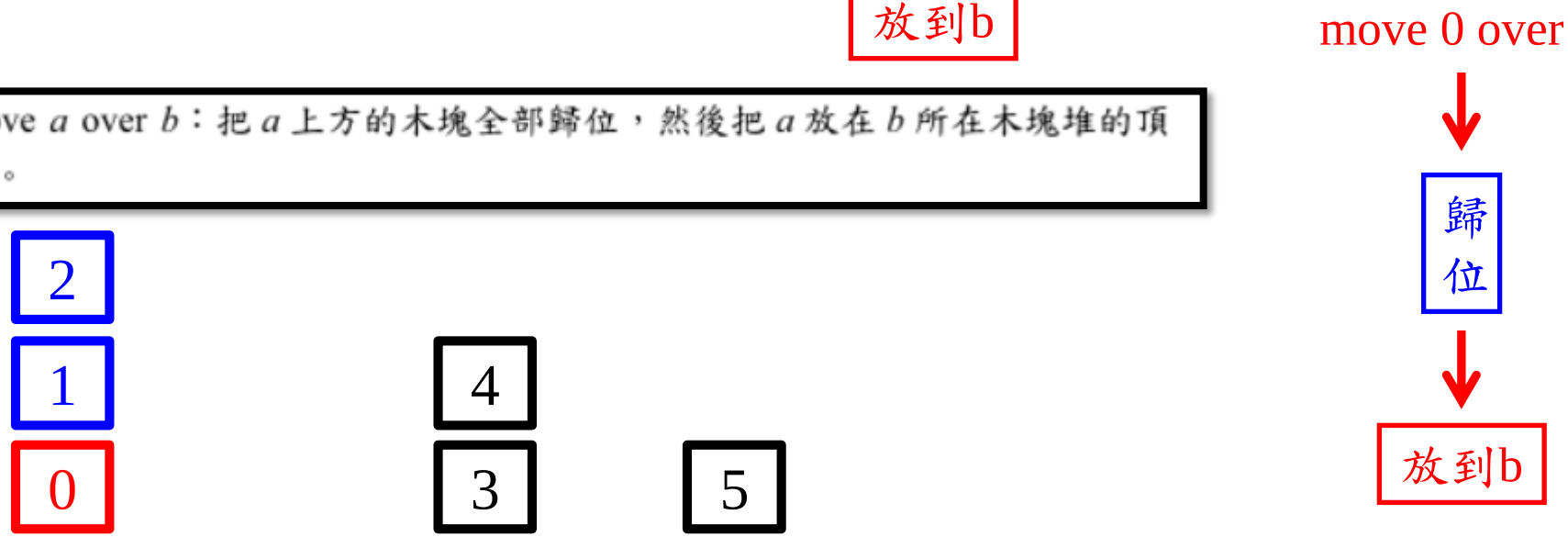
輸入（註1）：	輸出（註2）：
n個木塊 10	0: 0
move 9 onto 1	1: 1 9 2 4
move 8 over 1	2:
move 7 over 1	3: 3
move 6 over 1	4:
pile 8 over 6	5: 5 8 7 6
pile 8 over 5	6:
move 2 over 1	7:
move 4 over 9	8:
離開 quit	9:

第十四題

(1) move *a* onto *b*：把 *a* 和 *b* 上方的木塊全部歸位，然後把 *a* 放在 *b* 上面。



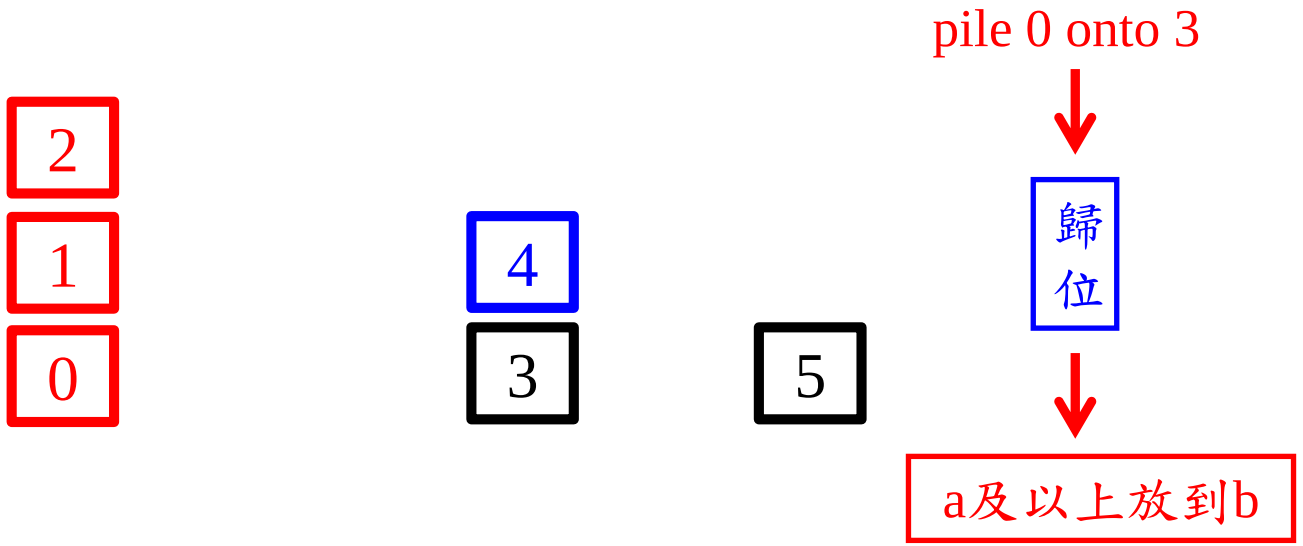
(2) move *a* over *b*：把 *a* 上方的木塊全部歸位，然後把 *a* 放在 *b* 所在木塊堆的頂部。



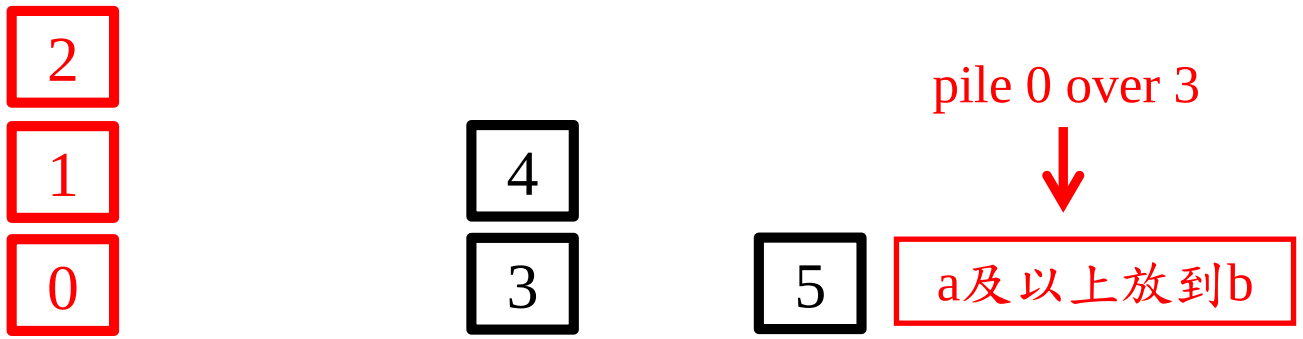
題目解析 程式詳解 輸出結果

第十四題

(3) pile *a* onto *b*：把 *b* 上方的木塊全部歸位，然後把 *a* 及上面的木塊整體放在 *b* 上面。



(4) pile *a* over b：把 *a* 及上面的木塊整體放在 *b* 所在木塊堆的頂部。
注意：*a* 和 *b* 在同一堆的指令是非法指令，應當忽略。



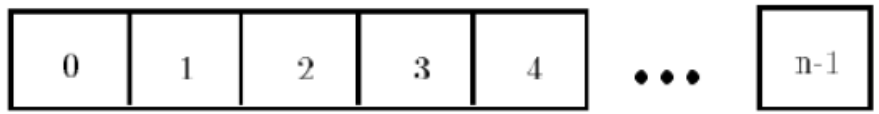
題目解析 程式詳解 輸出結果

第十四題

淡江大學電機工程學系 106 學年度韌體實驗題目

14. 【木塊問題/The Blocks Problem】

從左到右有 n 個木塊，編號為 0~n-1（如圖），$0 < n < 25$。要求模擬以下 4 種操作（下面的 a 和 b 都是木塊編號）。



(1) move a onto b：把 a 和 b 上方的木塊全部歸位，然後把 a 放在 b 上面。

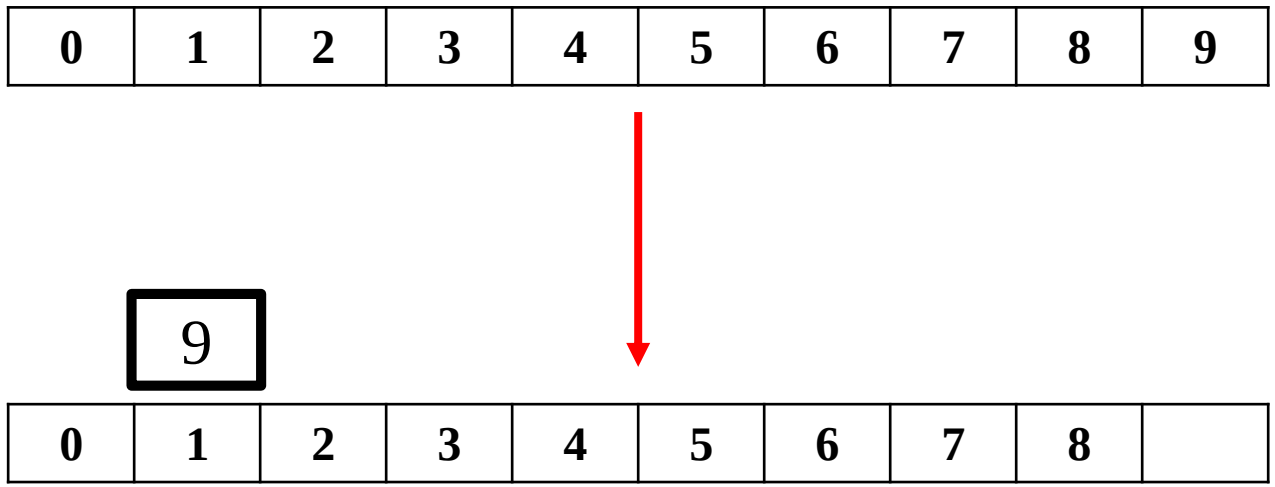
(2) move a over b：把 a 上方的木塊全部歸位，然後把 a 放在 b 所在木塊堆的頂部。

(3) pile a onto b：把 b 上方的木塊全部歸位，然後把 a 及上面的木塊整體放在 b 上面。

(4) pile a over b：把 a 及上面的木塊整體放在 b 所在木塊堆的頂部。

注意：a 和 b 在同一堆的指令是非法指令，應當忽略。

輸入（註 1）：	輸出（註 2）：
10	0: 0
move 9 onto 1	1: 1 9 2 4
move 8 over 1	2:
move 7 over 1	3: 3
move 6 over 1	4:
pile 8 over 6	5: 5 8 7 6
pile 8 over 5	6:
move 2 over 1	7:
move 4 over 9	8:
quit	9:



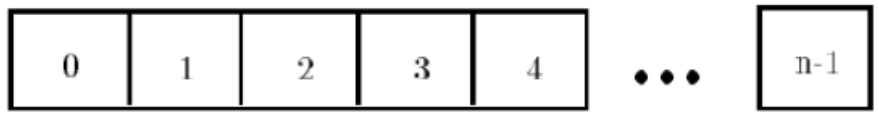
題目解析 程式詳解 輸出結果

第十四題

淡江大學電機工程學系 106 學年度韌體實驗題目

14. 【木塊問題/The Blocks Problem】

從左到右有 n 個木塊，編號為 0~n-1（如圖），$0 < n < 25$。要求模擬以下 4 種操作（下面的 a 和 b 都是木塊編號）。



(1) move a onto b：把 a 和 b 上方的木塊全部歸位，然後把 a 放在 b 上面。

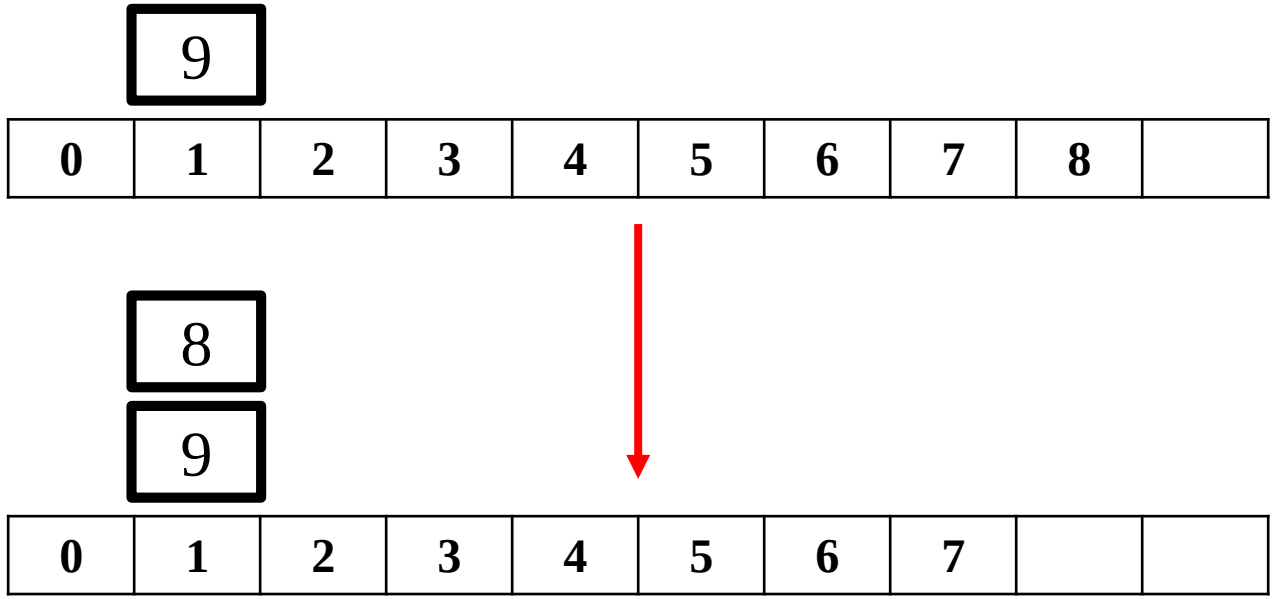
(2) move a over b：把 a 上方的木塊全部歸位，然後把 a 放在 b 所在木塊堆的頂部。

(3) pile a onto b：把 b 上方的木塊全部歸位，然後把 a 及上面的木塊整體放在 b 上面。

(4) pile a over b：把 a 及上面的木塊整體放在 b 所在木塊堆的頂部。

注意：a 和 b 在同一堆的指令是非法指令，應當忽略。

輸入（註 1）：	輸出（註 2）：
10	0: 0
move 9 onto 1	1: 1 9 2 4
move 8 over 1	2:
move 7 over 1	3: 3
move 6 over 1	4:
pile 8 over 6	5: 5 8 7 6
pile 8 over 5	6:
move 2 over 1	7:
move 4 over 9	8:
quit	9:



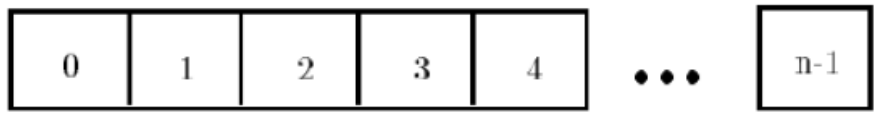
題目解析 程式詳解 輸出結果

第十四題

淡江大學電機工程學系106學年度韌體實驗題目

14. 【木塊問題/The Blocks Problem】

從左到右有 n 個木塊，編號為 0~n-1（如圖），$0 < n < 25$。要求模擬以下 4 種操作（下面的 a 和 b 都是木塊編號）。



(1) move a onto b：把 a 和 b 上方的木塊全部歸位，然後把 a 放在 b 上面。

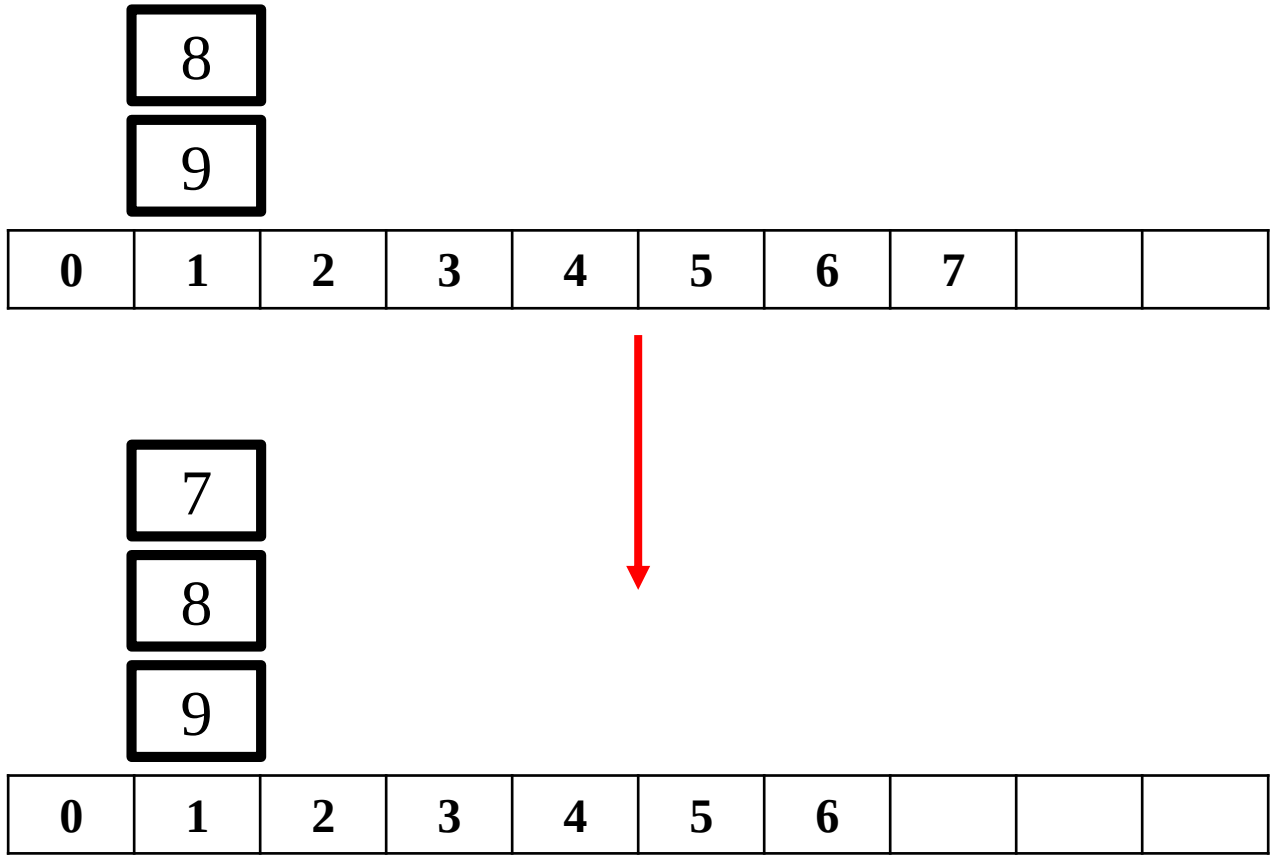
(2) move a over b：把 a 上方的木塊全部歸位，然後把 a 放在 b 所在木塊堆的頂部。

(3) pile a onto b：把 b 上方的木塊全部歸位，然後把 a 及上面的木塊整體放在 b 上面。

(4) pile a over b：把 a 及上面的木塊整體放在 b 所在木塊堆的頂部。

注意：a 和 b 在同一堆的指令是非法指令，應當忽略。

輸入（註1）：	輸出（註2）：
10	0: 0
move 9 onto 1	1: 1 9 2 4
move 8 over 1	2:
move 7 over 1	3: 3
move 6 over 1	4:
pile 8 over 6	5: 5 8 7 6
pile 8 over 5	6:
move 2 over 1	7:
move 4 over 9	8:
quit	9:



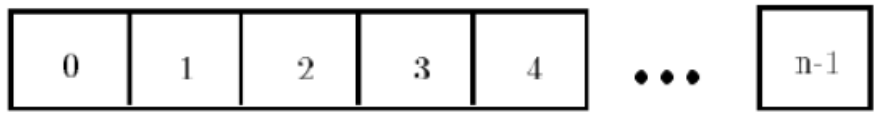
題目解析 程式詳解 輸出結果

第十四題

淡江大學電機工程學系 106 學年度韌體實驗題目

14. 【木塊問題/The Blocks Problem】

從左到右有 n 個木塊，編號為 0~n-1（如圖），$0 < n < 25$。要求模擬以下 4 種操作（下面的 a 和 b 都是木塊編號）。



(1) move a onto b：把 a 和 b 上方的木塊全部歸位，然後把 a 放在 b 上面。

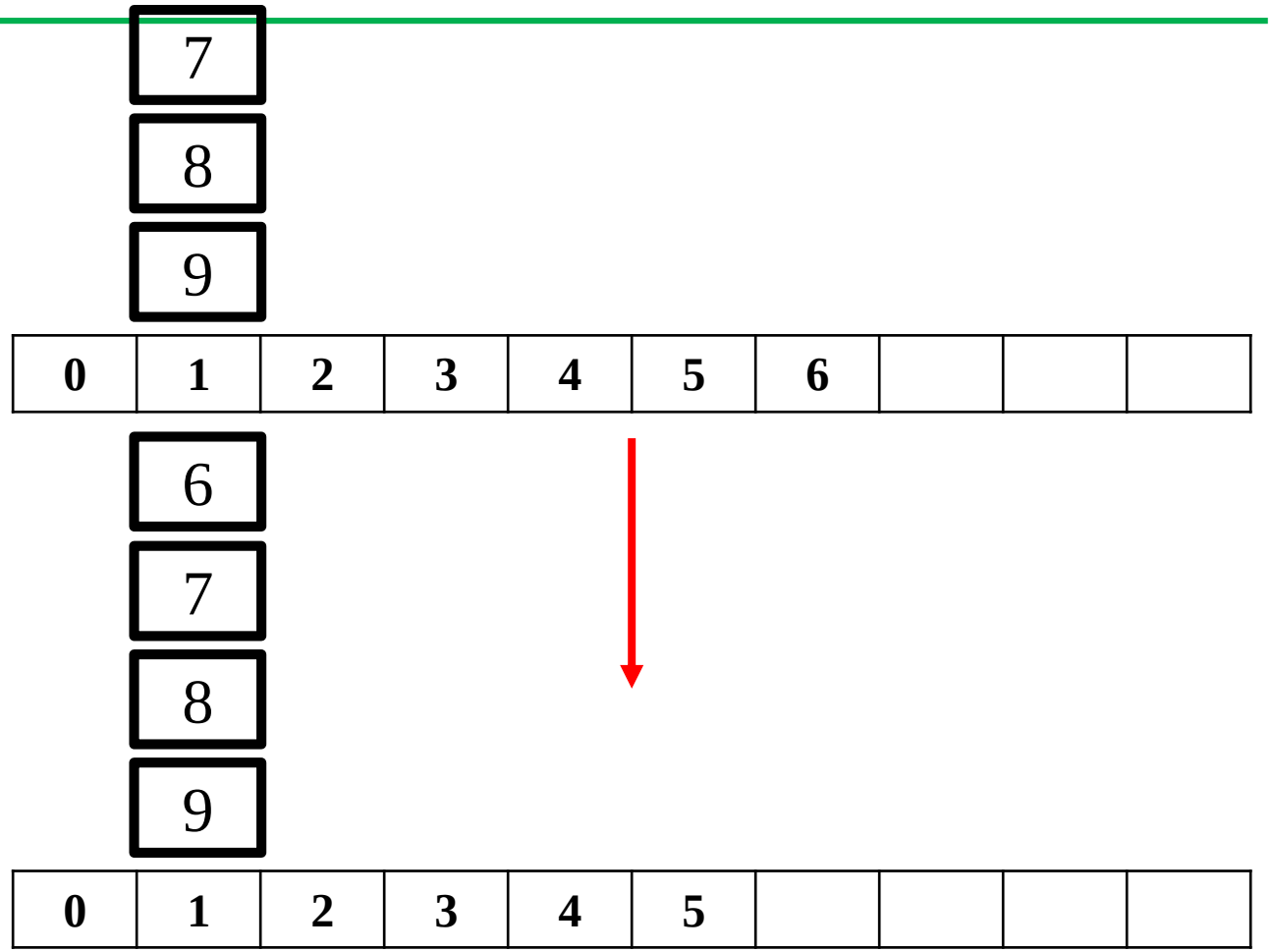
(2) move a over b：把 a 上方的木塊全部歸位，然後把 a 放在 b 所在木塊堆的頂部。

(3) pile a onto b：把 b 上方的木塊全部歸位，然後把 a 及上面的木塊整體放在 b 上面。

(4) pile a over b：把 a 及上面的木塊整體放在 b 所在木塊堆的頂部。

注意：a 和 b 在同一堆的指令是非法指令，應當忽略。

輸入（註 1）：	輸出（註 2）：
10	0: 0
move 9 onto 1	1: 1 9 2 4
move 8 over 1	2:
move 7 over 1	3: 3
move 6 over 1	4:
pile 8 over 6	5: 5 8 7 6
pile 8 over 5	6:
move 2 over 1	7:
move 4 over 9	8:
quit	9:



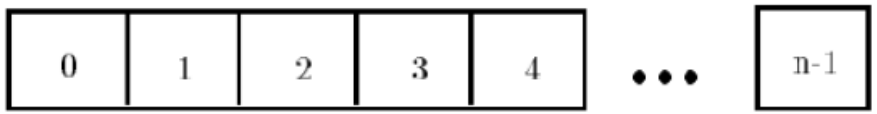
題目解析 程式詳解 輸出結果

第十四題

淡江大學電機工程學系106學年度韌體實驗題目

14. 【木塊問題/The Blocks Problem】

從左到右有 n 個木塊，編號為 0~n-1（如圖），$0 < n < 25$。要求模擬以下 4 種操作（下面的 a 和 b 都是木塊編號）。



(1) move a onto b：把 a 和 b 上方的木塊全部歸位，然後把 a 放在 b 上面。

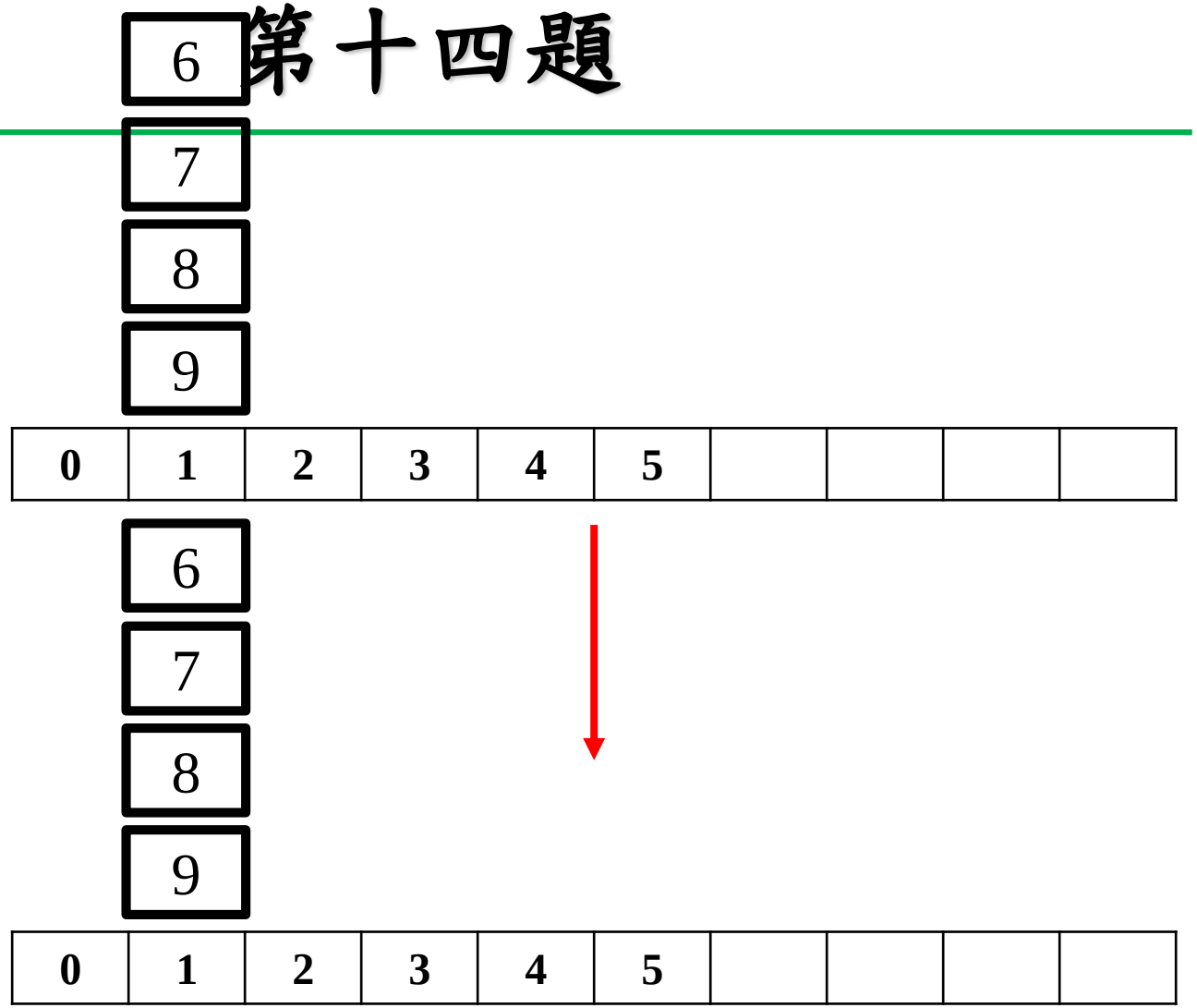
(2) move a over b：把 a 上方的木塊全部歸位，然後把 a 放在 b 所在木塊堆的頂部。

(3) pile a onto b：把 b 上方的木塊全部歸位，然後把 a 及上面的木塊整體放在 b 上面。

(4) pile a over b：把 a 及上面的木塊整體放在 b 所在木塊堆的頂部。

注意：a 和 b 在同一堆的指令是非法指令，應當忽略。

輸入（註1）：	輸出（註2）：
10	0: 0
move 9 onto 1	1: 1 9 2 4
move 8 over 1	2:
move 7 over 1	3: 3
move 6 over 1	4:
pile 8 over 6	5: 5 8 7 6
pile 8 over 5	6:
move 2 over 1	7:
move 4 over 9	8:
quit	9:



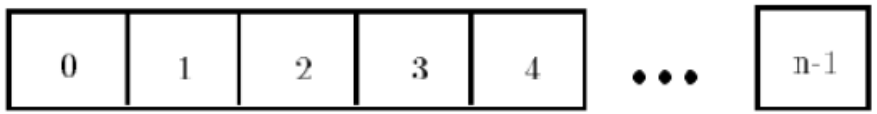
題目解析 程式詳解 輸出結果

第十四題

淡江大學電機工程學系106學年度韌體實驗題目

14. 【木塊問題/The Blocks Problem】

從左到右有 n 個木塊，編號為 0~n-1（如圖），$0 < n < 25$。要求模擬以下 4 種操作（下面的 a 和 b 都是木塊編號）。



(1) move a onto b：把 a 和 b 上方的木塊全部歸位，然後把 a 放在 b 上面。

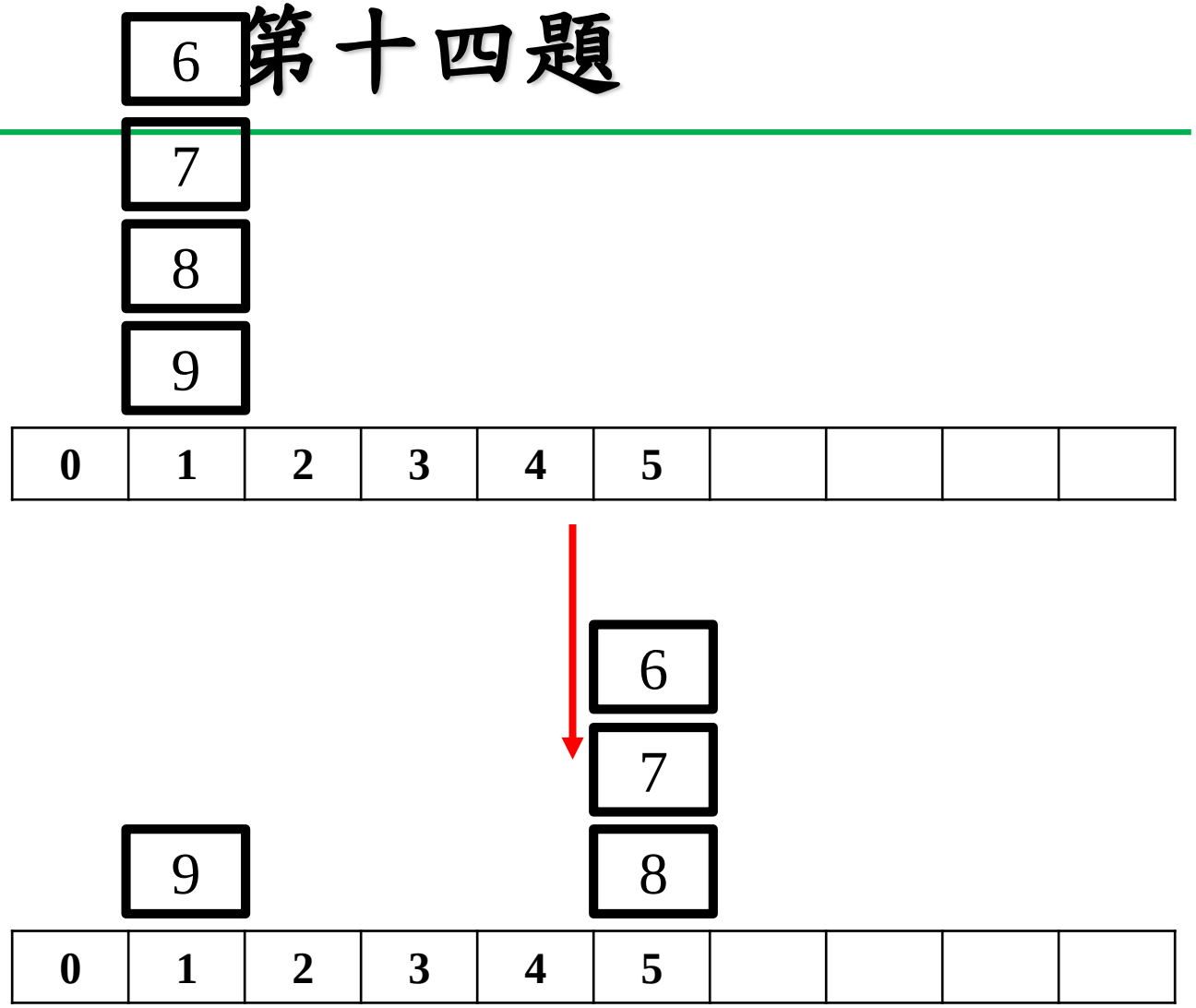
(2) move a over b：把 a 上方的木塊全部歸位，然後把 a 放在 b 所在木塊堆的頂部。

(3) pile a onto b：把 b 上方的木塊全部歸位，然後把 a 及上面的木塊整體放在 b 上面。

(4) pile a over b：把 a 及上面的木塊整體放在 b 所在木塊堆的頂部。

注意：a 和 b 在同一堆的指令是非法指令，應當忽略。

輸入（註1）：	輸出（註2）：
10	0: 0
move 9 onto 1	1: 1 9 2 4
move 8 over 1	2:
move 7 over 1	3: 3
move 6 over 1	4:
pile 8 over 6	5: 5 8 7 6
pile 8 over 5	6:
move 2 over 1	7:
move 4 over 9	8:
quit	9:



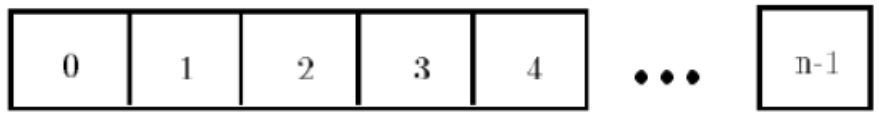
題目解析 程式詳解 輸出結果

第十四題

淡江大學電機工程學系 106 學年度韌體實驗題目

14. 【木塊問題/The Blocks Problem】

從左到右有 n 個木塊，編號為 0~n-1（如圖），$0 < n < 25$。要求模擬以下 4 種操作（下面的 a 和 b 都是木塊編號）。



(1) move a onto b：把 a 和 b 上方的木塊全部歸位，然後把 a 放在 b 上面。

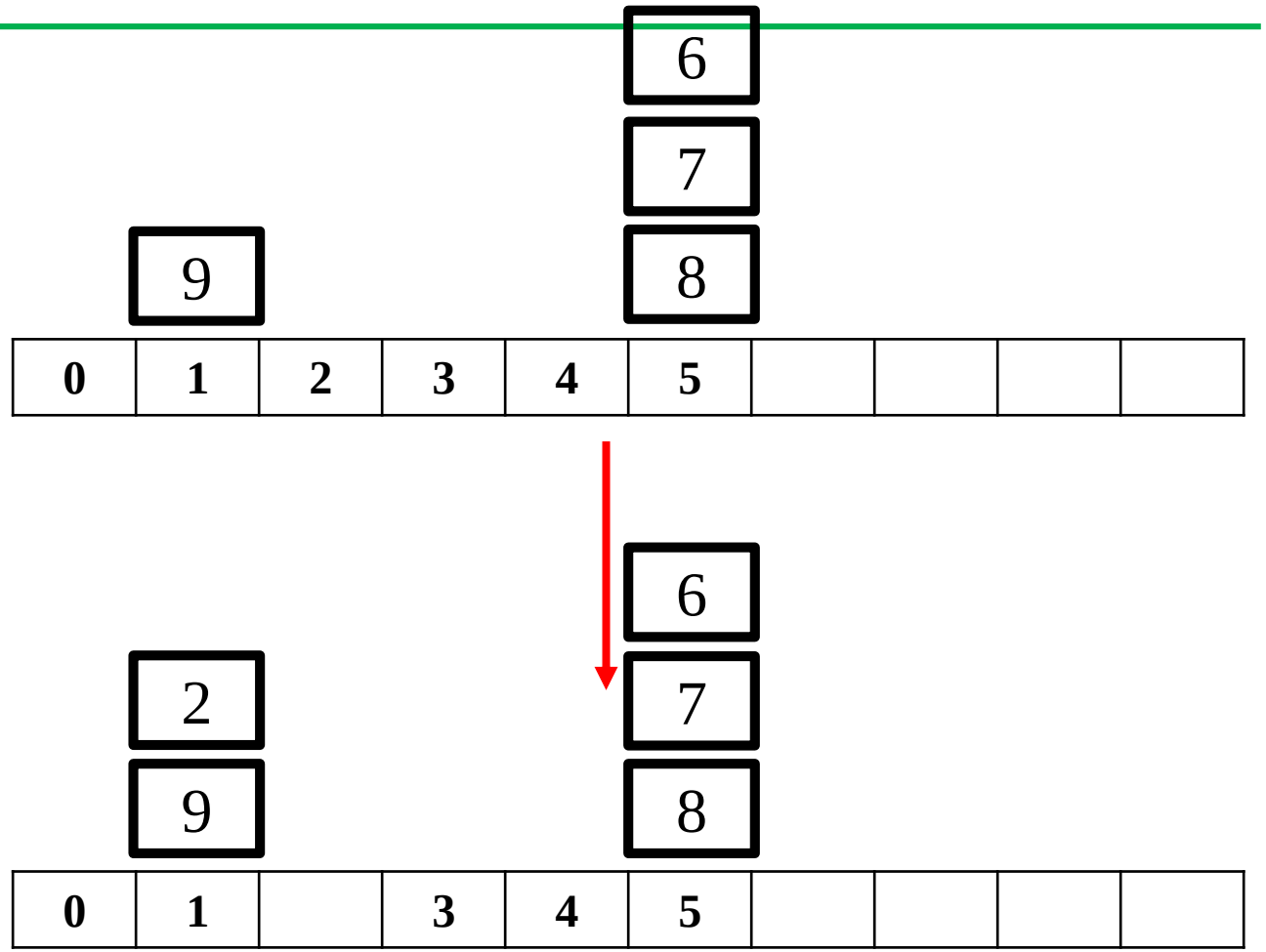
(2) move a over b：把 a 上方的木塊全部歸位，然後把 a 放在 b 所在木塊堆的頂部。

(3) pile a onto b：把 b 上方的木塊全部歸位，然後把 a 及上面的木塊整體放在 b 上面。

(4) pile a over b：把 a 及上面的木塊整體放在 b 所在木塊堆的頂部。

注意：a 和 b 在同一堆的指令是非法指令，應當忽略。

輸入（註 1）：	輸出（註 2）：
10	0: 0
move 9 onto 1	1: 1 9 2 4
move 8 over 1	2:
move 7 over 1	3: 3
move 6 over 1	4:
pile 8 over 6	5: 5 8 7 6
pile 8 over 5	6:
move 2 over 1	7:
move 4 over 9	8:
quit	9:



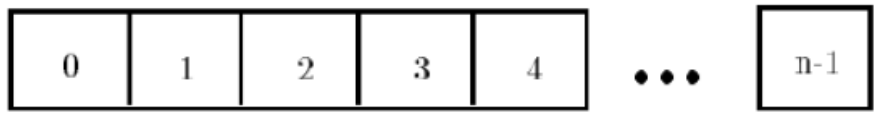
題目解析 程式詳解 輸出結果

第十四題

淡江大學電機工程學系 106 學年度韌體實驗題目

14. 【木塊問題/The Blocks Problem】

從左到右有 n 個木塊，編號為 0~n-1（如圖），$0 < n < 25$。要求模擬以下 4 種操作（下面的 a 和 b 都是木塊編號）。



(1) move a onto b：把 a 和 b 上方的木塊全部歸位，然後把 a 放在 b 上面。

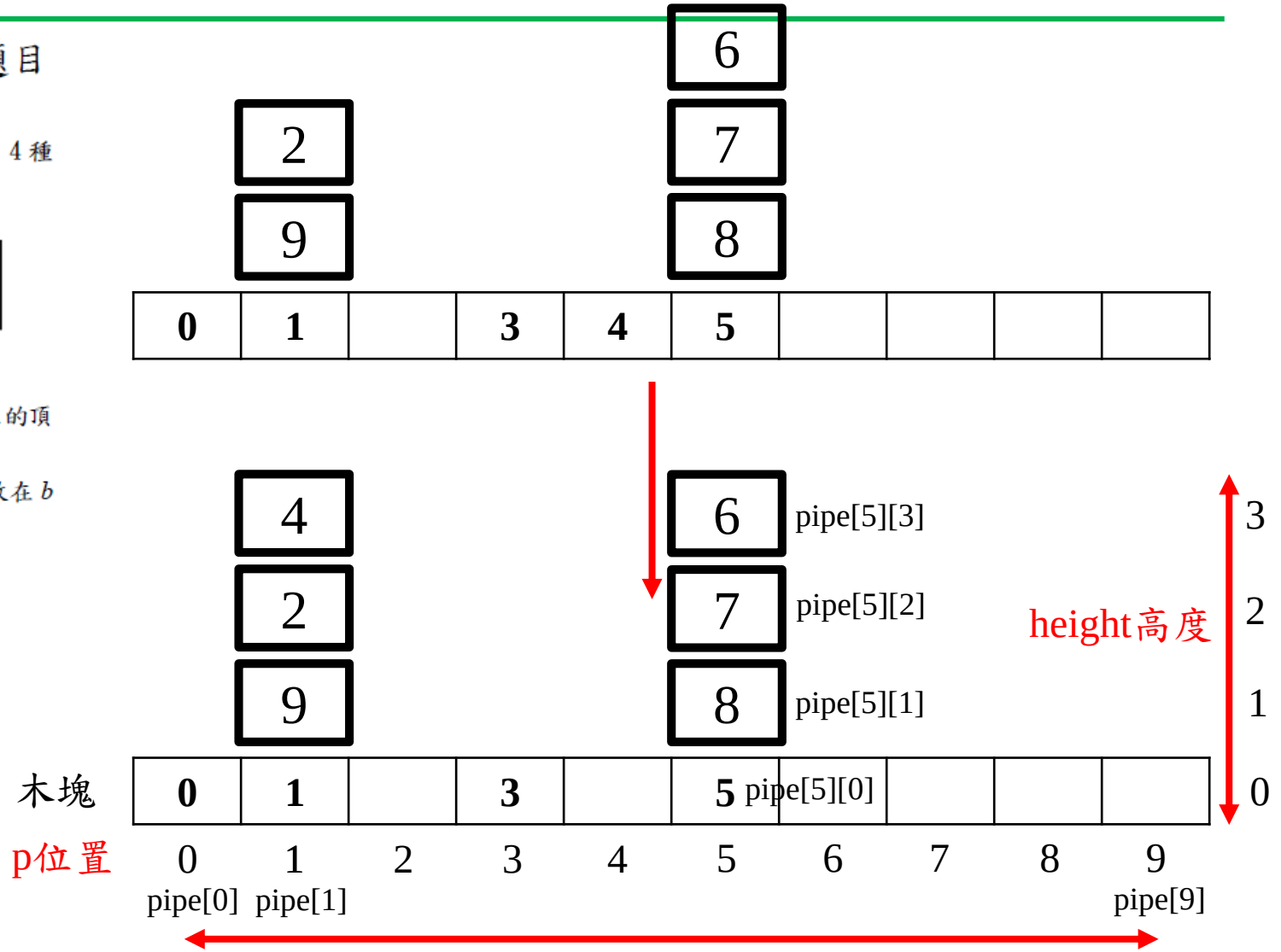
(2) move a over b：把 a 上方的木塊全部歸位，然後把 a 放在 b 所在木塊堆的頂部。

(3) pile a onto b：把 b 上方的木塊全部歸位，然後把 a 及上面的木塊整體放在 b 上面。

(4) pile a over b：把 a 及上面的木塊整體放在 b 所在木塊堆的頂部。

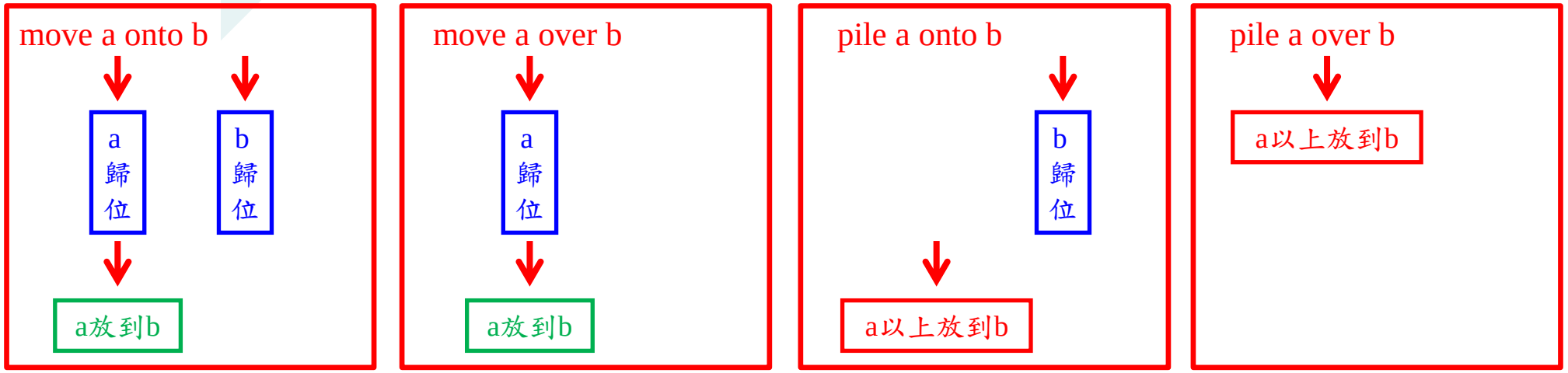
注意：a 和 b 在同一堆的指令是非法指令，應當忽略。

輸入（註 1）：	輸出（註 2）：
10	0: 0
move 9 onto 1	1: 1 9 2 4
move 8 over 1	2:
move 7 over 1	3: 3
move 6 over 1	4:
pile 8 over 6	5: 5 8 7 6
pile 8 over 5	6:
move 2 over 1	7:
move 4 over 9	8:
quit	9:



題目解析 程式詳解 輸出結果

第十四題



- 程式撰寫邏輯：

✓ 有move就將a以上歸位 (使用clear_above()函式)。

✓ 有onto就將b以上歸位 (使用clear_above()函式)。

✓ 四步驟最後皆需要pile over (使用pipe_over()函式)，將a及上面的木塊整體放在b 所在木塊堆的頂部，只差在有沒有先將自己以上的木塊歸位。

✓ 三個副函式表達4種操作。

題目解析 | 程式詳解 | 輸出結果

第十四題

```
1  #define _CRT_SECURE_NO_WARNINGS
2  //[木塊問題/The Blocks Problem](1/3)
3  #define IN "P14IN.txt"
4  #define OUT "P14OUT.txt"
5  //****************************************
6  #include <iostream>
7  #include <ctime>
8  using namespace std;
9  void redir(void);
10
11 //****************************************
12 /* Work Space*/
13 #include <string>
14 #include <vector>
15 void find_block(int a, int &pa, int &ha);
16 void clear_above(int p, int h);
17 void pipe_over(int p, int h, int p2);
18 void show(void);
19 int n;
20 vector<int> pipe[25]; //二維陣列(第一維大小根據題意，第二維大小不固定)
21 //****************************************
22
23 int main(void) {
24     redir(); //redirection
25
```

需要修改或自行撰寫的部分

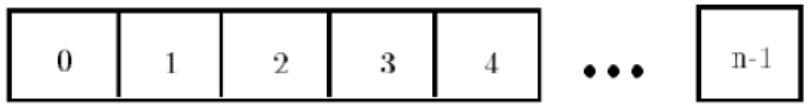
- 初行加入crt secure no warnings。
- 記得修改輸入和輸出txt檔案名稱。
- #define為C++直接定義，變數型態會依照定義自動判定。

- #include <string>
- #include <vector>
- 宣告四個副函式 (3個操作+1個顯示)。
- 宣告全域整數變數n：木塊數。
- 宣告名為pipe、型態為int陣列的vector(向量)，此vector可視為二維陣列，先宣告第一個維度依題目大小25，第二個維度為堆放其餘木塊的高度。

淡江大學電機工程學系106學年度韌體實驗題目

14.【木塊問題/The Blocks Problem】

從左到右有 n 個木塊，編號為 0~n-1（如圖），$0 < n < 25$。要求模擬以下 4 種操作（下面的 a 和 b 都是木塊編號）。



題目解析 | 程式詳解 | 輸出結果

第十四題

```
26  //***************************************
27  /* Work Space*/
28      int i, a, b;
29      string s1, s2;
30      int pa, pb; //pa, pb: position of a and b
31      int ha, hb; //ha, hb: height of a and b
32
33      cin >> n;
34      for (i = 0; i < n; i++) {
35          pipe[i].push_back(i);
36      }
37      while (1) {
38          cin >> s1;
39          if (s1 == "quit") { // '==' is overloaded for type string
40              break;
41          }
42          else {
43              cin >> a >> s2 >> b;
44              find_block(a, pa, ha);
45              find_block(b, pb, hb);
46              if (pa == pb) {
47                  continue; //非法指令
48              }
49              if (s1 == "move") {
50                  clear_above(pa, ha);
51              }
52              if (s2 == "onto") {
53                  clear_above(pb, hb);
54              }
55              pipe_over(pa, ha, pb);
56          }
57      }
58      show();
59  //***************************************
```

需要修改或自行撰寫的部分

- 解題思考：

(1) 變數宣告、初始化vector

(2) while迴圈讀取輸入資料

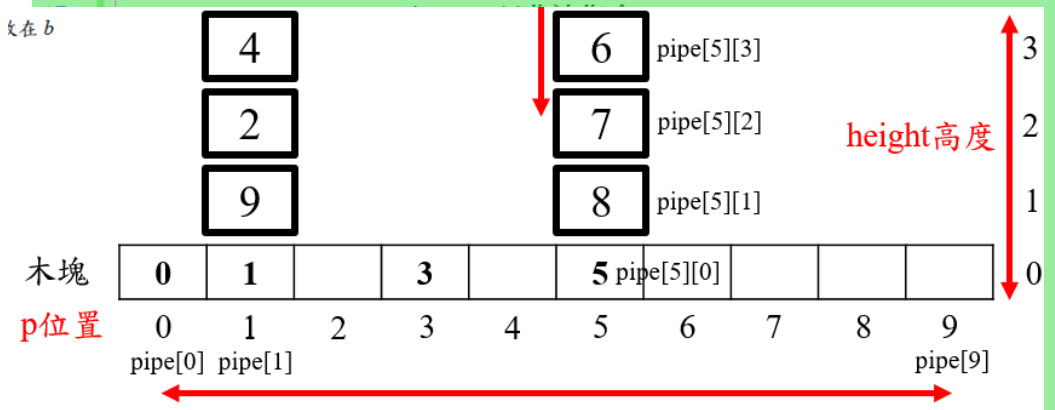
(3) 三副函式find_ab、clear_ab、pipe_ab

(4) 顯示結果

題目解析 | 程式詳解 | 輸出結果

第十四題

```
26  //****************************************
27  /* Work Space*/
28      int i, a, b;
29      string s1, s2;
30      int pa, pb; //pa, pb: position of a and b
31      int ha, hb; //ha, hb: height of a and b
32
33      cin >> n;
34      for (i = 0; i < n; i++) {
35          pipe[i].push_back(i);
36      }
37      while (1) {
38          cin >> s1;
39          if (s1 == "quit") { // '==' is overloaded for type string
40              break;
41          }
42          else {
43              cin >> a >> s2 >> b;
44              find_block(a, pa, ha);
45              find_block(b, pb, hb);
46              if (pa == pb) {
```



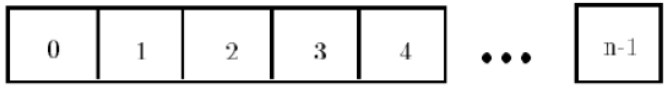

```
59  //****************************************
```

需要修改或自行撰寫的部分

淡江大學電機工程學系106學年度韌體實驗題目

14.【木塊問題/The Blocks Problem】

從左到右有 n 個木塊，編號為 0~n-1（如圖），$0 < n < 25$。要求模擬以下 4 種操作（下面的 a 和 b 都是木塊編號）。



➢ 解題思考：

(1) 變數宣告、初始化vector

(2) while迴圈讀取輸入資料

(3) 三副函式find_ab、clear_ab、pipe_ab

(4) 顯示結果

- Line 28-31:
  - ✓ 宣告整數變數i，編號a木塊、編號b木塊。
  - ✓ 宣告字串變數s1, s2。
  - ✓ 宣告整數變數pa：木塊a所在的位置、pb：木塊b所在的位置、ha：a木塊的高度、hb：b木塊的高度。
- Line 33-36:
  - ✓ cin輸入木塊數量n。
  - ✓ for迴圈初始化木塊vector，將n個木塊push進vector中。

題目解析 | 程式詳解 | 輸出結果

第十四題

```
輸入（註1）:
10
move 9 onto 1
move 8 over 1
move 7 over 1
move 6 over 1
pile 8 over 6
pile 8 over 5
move 2 over 1
move 4 over 9
quit
```

```
輸出（註2）:
0: 0
1: 1 9 2 4
2:
3: 3
4:
5: 5 8 7 6
6:
7:
8:
9:
```

```
26  //****************************************
27  /* Work Space*/
28      int i, a, b;
29      string s1, s2;
30      int pa, pb; //pa, pb: position of a and b
31      int ha, hb; //ha, hb: height of a and b
32
33      cin >> n;
34      for (i = 0; i < n; i++) {
35          pipe[i].push_back(i);
36      }
37      while (1) {
38          cin >> s1;
39          if (s1 == "quit") { // '==' is overloaded for type string
40              break;
41          }
42          else {
43              cin >> a >> s2 >> b;
44              find_block(a, pa, ha);
45              find_block(b, pb, hb);
46              if (pa == pb) {
47                  continue; //非法指令
48              }
49              if (s1 == "move") {
50                  clear_above(pa, ha);
51              }
52              if (s2 == "onto") {
53                  clear_above(pb, hb);
54              }
55              pipe_over(pa, ha, pb);
56          }
57      }
58      show();
59  //****************************************
```

需要修改或自行撰寫的部分

➢ 解題思考：

(1) 變數宣告、初始化vector

(2) while迴圈讀取輸入資料

(3) 三副函式find_ab、clear_ab、pipe_ab

(4) 顯示結果

- Line 37:

✓ 無窮迴圈。

- Line 38-41:

✓ cin輸入第一個指令字串至變數s1，如果為”quit”則break跳出無窮迴圈。

題目解析 | 程式詳解 | 輸出結果

第十四題

```
26  //***************************************
27  /* Work Space*/
28      int i, a, b;
29      string s1, s2;
30      int pa, pb; //pa, pb: position of a and b
31      int ha, hb; //ha, hb: height of a and b
32
33      cin >> n;
34      for (i = 0; i < n; i++) {
35          pipe[i].push_back(i);
36      }
37      while (1) {
38          cin >> s1;
39          if (s1 == "quit") { // '==' is overloaded for type string
40              break;
41          }
42          else {
43              cin >> a >> s2 >> b;
44              find_block(a, pa, ha);
45              find_block(b, pb, hb);
46              if (pa == pb) {
47                  continue; //非法指令
48              }
49              if (s1 == "move") {
50                  clear_above(pa, ha);
51              }
52              if (s2 == "onto") {
53                  clear_above(pb, hb);
54              }
55              pipe_over(pa, ha, pb);
56          }
57      }
58      show();
59  //***************************************
```

需要修改或自行撰寫的部分

➢ 解題思考：

(1) 變數宣告、初始化vector

(2) while迴圈讀取輸入資料

(3) 三副函式find_ab、clear_ab、pipe_ab

(4) 顯示結果

- Line 43-45:

✓ cin輸入編號a木塊、s2第二個指令字串、編號b木塊。

✓ find_block()搜尋a木塊與b木塊的所在的位置和高度。

題目解析 | 程式詳解 | 輸出結果

第十四題

```
73  //*****************************************
74  /* Work Space*/
75  //發現木塊a的位置p和高度h(皆從0算起)
76  void find_block(int a, int &p, int &h) {
77      for (p = 0; p < n; p++) {
78          for (h = 0; h < pipe[p].size(); h++) {
79              if (pipe[p][h] == a) {
80                  return;
81              }
82          }
83      }
84  }
```

副函式

搜尋a木塊與b木塊的所在的位置和高度。

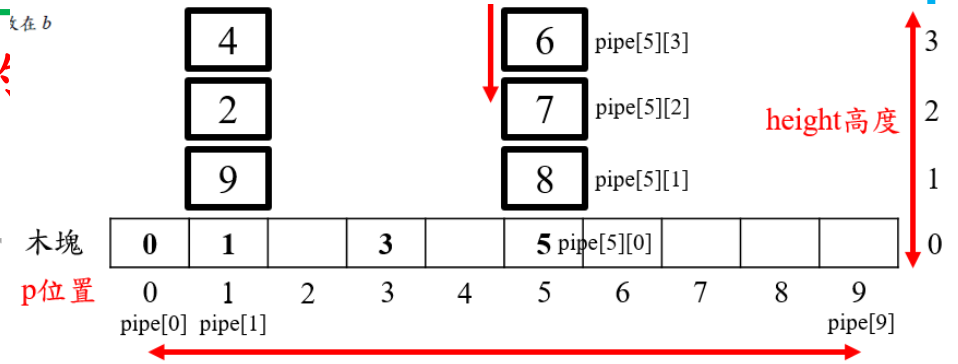
標頭檔

```
15  void find_block(int a, int &pa, int &ha);
16  void clear_above(int p, int h);
17  void pipe_over(int p, int h, int p2);
18  void show(void);
19  int n;
20  vector<int> pipe[25]; //二維陣列(第一維大
```

Main主函式

```
44              find_block(a, pa, ha);
45              find_block(b, pb, hb);
```

需要修改或自行撰寫的



➢ 解題思考：

(1) 變數宣告、初始化vector

(2) while迴圈讀取輸入資料

(3) 三副函式find_ab、clear_ab、pipe_ab

(4) 顯示結果

- 副函式參數(編號a木塊, 位置p, 高度h)
- Line 77-78:
  - ✓ 外層for迴圈找位置p，範圍為0~n。
  - ✓ 內層for迴圈找高度h，範圍為0~pipe[p].size()-1。
- Line 79-80:
  - ✓ 如果在vector中找到，則回傳空值，結束搜索。

題目解析 | 程式詳解 | 輸出結果

第十四題

```
26  //***************************************
27  /* Work Space*/
28      int i, a, b;
29      string s1, s2;
30      int pa, pb; //pa, pb: position of a and b
31      int ha, hb; //ha, hb: height of a and b
32
33      cin >> n;
34      for (i = 0; i < n; i++) {
35          pipe[i].push_back(i);
36      }
37      while (1) {
38          cin >> s1;
39          if (s1 == "quit") { // '==' is overloaded for type string
40              break;
41          }
42          else {
43              cin >> a >> s2 >> b;
44              find_block(a, pa, ha);
45              find_block(b, pb, hb);
46              if (pa == pb) {
47                  continue; //非法指令
48              }
49              if (s1 == "move") {
50                  clear_above(pa, ha);
51              }
52              if (s2 == "onto") {
53                  clear_above(pb, hb);
54              }
55              pipe_over(pa, ha, pb);
56          }
57      }
58      show();
59  //***************************************
```

需要修改或自行撰寫的部分

➢ 解題思考：

(1) 變數宣告、初始化vector

(2) while迴圈讀取輸入資料

(3) 三副函式find_ab、clear_ab、pipe_ab

(4) 顯示結果

- Line 46-48:

✓ 如果 a 和 b 在同一個位置是非法指令。

- Line 49-54:

✓ 如果 s1 指令為move則將 a 以上歸位。

✓ 如果 s2 指令為onto則將 b 以上歸位。

題目解析 | 程式詳解 | 輸出結果

第十四題

```
86  //把第p堆高度為h的木塊"上方的"所有木塊歸位
87  void clear_above(int p, int h) {
88      int i, b;
89      for (i = pipe[p].size() - 1; i > h; i--) {
90          b = pipe[p][i];
91          pipe[b].push_back(b);
92      }
93      pipe[p].resize(h + 1);
94  }
```

將第p個位置高度為h的木塊(a或b)上方所有木塊歸位。

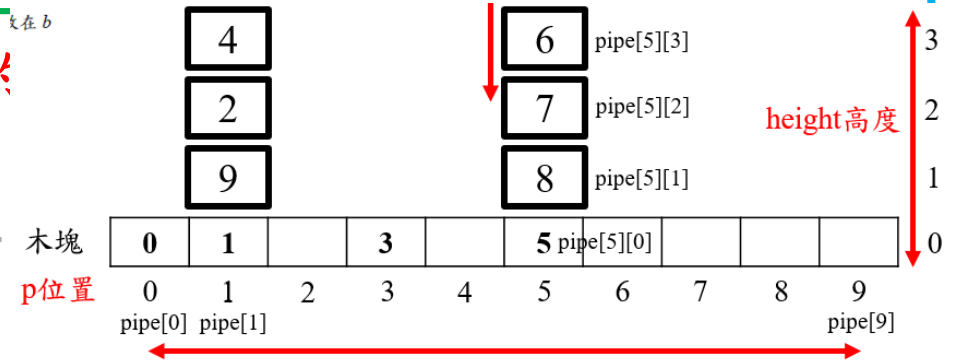
標頭檔

```
15  void find_block(int a, int &pa, int &ha);
16  void clear_above(int p, int h);
17  void pipe_over(int p, int h, int p2);
18  void show(void);
19  int n;
20  vector<int> pipe[25]; //二維陣列(第一維大
```

Main主函式

```
49              if (s1 == "move") {
50                  clear_above(pa, ha);
51              }
52              if (s2 == "onto") {
53                  clear_above(pb, hb);
54              }
```

需要修改或自行撰寫的



➢ 解題思考：

(1) 變數宣告、初始化vector

(2) while迴圈讀取輸入資料

(3) 三副函式find_ab、clear_ab、pipe_ab

(4) 顯示結果

- 副函式參數(編號a或b木塊的位置p, 編號a或b木塊的高度h)
- Line 89-92:
  - ✓ 設定for迴圈i的範圍從pipe[p].size() - 1遞減至h+1，代表a木塊或b木塊上方木塊的高度範圍。
  - ✓ 將vector pipe第p位置第i個高度木塊值給變數b。
  - ✓ 將b值push進vector pipe[b]歸位。
- Line 93:
  - ✓ 將vector pipe第p位置大小resize為h+1，移動後要將h位置以上的木塊刪除並保留h位置的木塊，直接使用resize即可，resize為h+1實際保留為0~h。

題目解析 | 程式詳解 | 輸出結果

第十四題

```
26  //***************************************
27  /* Work Space*/
28      int i, a, b;
29      string s1, s2;
30      int pa, pb; //pa, pb: position of a and b
31      int ha, hb; //ha, hb: height of a and b
32
33      cin >> n;
34      for (i = 0; i < n; i++) {
35          pipe[i].push_back(i);
36      }
37      while (1) {
38          cin >> s1;
39          if (s1 == "quit") { // '==' is overloaded for type string
40              break;
41          }
42          else {
43              cin >> a >> s2 >> b;
44              find_block(a, pa, ha);
45              find_block(b, pb, hb);
46              if (pa == pb) {
47                  continue; //非法指令
48              }
49              if (s1 == "move") {
50                  clear_above(pa, ha);
51              }
52              if (s2 == "onto") {
53                  clear_above(pb, hb);
54              }
55              pipe_over(pa, ha, pb);
56          }
57      }
58      show();
59  //***************************************
```

需要修改或自行撰寫的部分

➢ 解題思考：

(1) 變數宣告、初始化vector

(2) while迴圈讀取輸入資料

(3) 三副函式find_ab、clear_ab、pipe_ab

(4) 顯示結果

- Line 55:

✓ pipe a over b：把a及上面的木塊整體放在b 所在木塊堆的頂部。

題目解析 | 程式詳解 | 輸出結果

第十四題

```
96  //把第p堆高度h"及其上方"的木塊整體移動到第p2堆的頂部
97  void pipe_over(int p, int h, int p2) {
98      int i;
99      for (i = h; i < pipe[p].size(); i++) {
100         pipe[p2].push_back(pipe[p][i]);
101     }
102     pipe[p].resize(h);
103 }
```

把a及上面的木塊整體放在b所在木塊堆的頂部。

標頭檔

```
15  void find_block(int a, int &pa, int &ha);
16  void clear_above(int p, int h);
17  void pipe_over(int p, int h, int p2);
18  void show(void);
19  int n;
20  vector<int> pipe[25]; //二維陣列(第一維大
```

Main主函式

```
55              pipe_over(pa, ha, pb);
```

需要修改或自行撰寫的

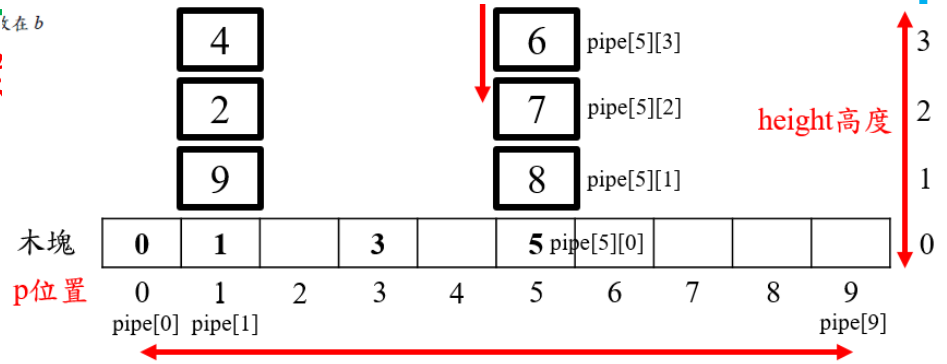
- 解題思考：

(1) 變數宣告、初始化vector

(2) while迴圈讀取輸入資料

(3) 三副函式find_ab、clear_ab、pipe_ab

(4) 顯示結果



- 副函式參數(編號a木塊的位置p, 編號a木塊的高度h, 位置p2)
- Line 99-101:
  - ✓ 設定for迴圈i範圍從h遞增至pipe[p].size() - 1，代表包含a木塊及上方木塊高度範圍。
  - ✓ 將vector pipe第p個位置第i個高度木塊值push進vector pipe[p2]。
- Line 102:
  - ✓ 將vector pipe第p位置大小resize為h，移動後要將包含h及以上的木塊刪除並保留h-1位置的木塊，直接使用resize即可，resize為h實際保留為0~h-1。

題目解析 | 程式詳解 | 輸出結果

第十四題

```
26  //****************************************
27  /* Work Space*/
28      int i, a, b;
29      string s1, s2;
30      int pa, pb; //pa, pb: position of a and b
31      int ha, hb; //ha, hb: height of a and b
32
33      cin >> n;
34      for (i = 0; i < n; i++) {
35          pipe[i].push_back(i);
36      }
37      while (1) {
38          cin >> s1;
39          if (s1 == "quit") { // '==' is overloaded for type string
40              break;
41          }
42          else {
43              cin >> a >> s2 >> b;
44              find_block(a, pa, ha);
45              find_block(b, pb, hb);
46              if (pa == pb) {
47                  continue; //非法指令
48              }
49              if (s1 == "move") {
50                  clear_above(pa, ha);
51              }
52              if (s2 == "onto") {
53                  clear_above(pb, hb);
54              }
55              pipe_over(pa, ha, pb);
56          }
57      }
58      show();
59  //****************************************
```

需要修改或自行撰寫的部分

➢ 解題思考：

(1) 變數宣告、初始化vector

(2) while迴圈讀取輸入資料

(3) 三副函式find_ab、clear_ab、pipe_ab

(4) 顯示結果

- Line 58:

✓ 顯示結果

題目解析 | 程式詳解 | 輸出結果

第十四題

```
105  //輸出結果
106  void show(void) {
107      int i, j;
108      for (i = 0; i < n; i++) {
109          printf("%d: ", i);
110          for (j = 0; j < pipe[i].size(); j++) {
111              printf("%d ", pipe[i][j]);
112          }
113          printf("\n");
114      }
115  }
```

需要修改或自行撰寫的部分

- 解題思考：

(1) 變數宣告、初始化vector

(2) while迴圈讀取輸入資料

(3) 三副函式find_ab、clear_ab、pipe_ab

(4) 顯示結果

- Line 108-109:

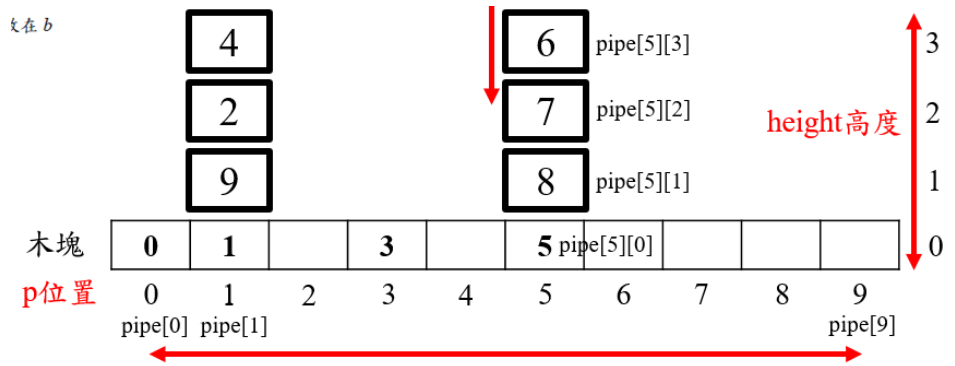
✓ 設定外層for迴圈範圍i從0遞增至n-1，位置數量。

✓ 設定內層for迴圈範圍j從0遞增pipe[p].size()-1，每個位置的木塊數量。

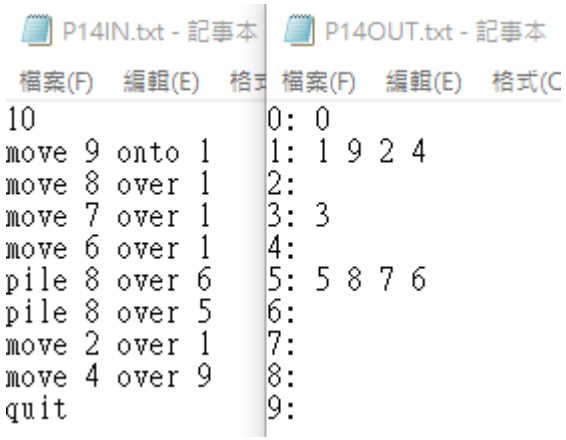
✓ 依序顯示每一個位置所含木塊的結果。

```
輸入（註1）:
10
move 9 onto 1
move 8 over 1
move 7 over 1
move 6 over 1
pile 8 over 6
pile 8 over 5
move 2 over 1
move 4 over 9
quit
```

```
輸出（註2）:
0: 0
1: 1 9 2 4
2:
3: 3
4:
5: 5 8 7 6
6:
7:
8:
9:
```



第十四題



P14IN.txt - 記事本

檔案(F) 編輯(E) 格式

```
10
move 9 onto 1
move 8 over 1
move 7 over 1
move 6 over 1
pile 8 over 6
pile 8 over 5
move 2 over 1
move 4 over 9
quit
```

P14OUT.txt - 記事本

檔案(F) 編輯(E) 格式(C

```
0: 0
1: 1 9 2 4
2:
3: 3
4:
5: 5 8 7 6
6:
7:
8:
9:
```

Q&A

2022/5/6