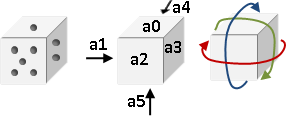
有兩顆骰子，上面的點數順序，可能是亂的、重複的。請辨別兩顆骰子一不一樣。骰子經過旋轉後，如果六個對應的面，上面的點數皆相同，則骰子視為相同。

要辨別兩顆骰子一不一樣，一種方式是旋轉其中一顆骰子，再跟另一顆比對。必須將骰子所有可能的情形都轉出來才行。



骰子一共有六個面：上下前後左右。建立六個變數、或者六格的陣列，分別儲存六個面的點數。陣列的第 0 格存入上面的點數、第 1 格存左面的點數、 …… 。當然也可以採用不同的儲存方法。

骰子一共有三種旋轉方向：東西方向、南北方向、時鐘方向。我個人偏好的轉法是：東西方向轉一圈，順時針方向轉一圈，南北方向轉一下，以上動作重複四次，就能轉出所有情形。

亦得採用其他轉法，最好的轉法只需轉 24 次，讀者可以想想看怎麼做。

亦得預先計算所有旋轉結果，儲存於 lookup table ，以節省旋轉時間。

1. // 往西轉一下
2. void turn\_west(int dice[6])
3. {
4. int temp = dice[1];
5. dice[1] = dice[2];
6. dice[2] = dice[3];
7. dice[3] = dice[4];
8. dice[4] = temp;
9. }
11. // 六個面是否一一對應
12. bool equal(int a[6], int b[6])
13. {
14. for (int i=0; i<6; i++)
15. if (a[i] != b[i])
16. return false;
17. return true;
18. }
20. // 兩顆骰子是否相同
21. bool check(int a[6], int b[6])
22. {
23. for (int i=0; i<4; i++)
24. {
25. for (int j=0; j<4; j++)
26. {
27. turn\_west();
28. if (equal(a,b)) return true;
29. }
31. for (int j=0; j<4; j++)
32. {
33. turn\_clockwise();
34. if (equal(a,b)) return true;
35. }
37. turn\_north();
38. if (equal(a,b)) return true;
39. }
40. return false;
41. }