2002 Sun Microsystems and TopCoder Collegiate Challenge – Problem Statement

Cubes PROBLEM STATEMENT

When constructing three-dimensional objects, it is often useful to think about folding a flat shape, or cutout, in a manner to produce the desired shape. For example, to produce a cube, you might fold a cutout composed of six squares. This is an example of such a cutout:

```
{"  C ",
 "ADEF",
 "B   "}
```

The six squares are labeled 'A' through 'F', with spaces filling the remainder of the area. For this problem all cutouts are one piece so all squares are connected horizontally or vertically to the other squares. There are many more cutouts than actual cubes, so some cutouts may not form cubes. Also, two cutouts may be different, yet produce identical cubes when folded at each edge. For example, this cutout produces the same cube as the cutout above:

```
{" D ",
 " C ",
 "EFA",
 " B "}
```

Determine whether each input String[] folds into a cube and if so, whether they both produce the same cube. Return:

0 - Neither cutout can form a cube
1 - Only cutoutA can form a cube
2 - Only cutoutB can form a cube
3 - Both cutouts can form cubes, but case #4 (next) does not apply
4 - Both cutouts can form cubes with faces in the same orientations, relative to each other

Note that folds can be made 90 degrees in either direction. Folding one cutout folded inward might match another cutout folded outward.

DEFINITION
Class name: Cubes
Method name: match
Parameters: String[], String[]
Returns: int
Method signature (be sure your method is public): int match(String[] cutoutA, String[] cutoutB)

TopCoder will ensure the validity of the inputs. Inputs are valid if all of the following criteria are met:
- Valid characters are: 'A', 'B', 'C', 'D', 'E', 'F', ' ' (space).
- Each String in each String[] argument:
  * Contains only valid characters.
  * Has a length between 1 and 6, inclusive.
  * Is the same length as other Strings in its String[].
- Each String[] contains each of the letters 'A', 'B', 'C', 'D', 'E', 'F' exactly once.
- Each String[] has between 1 and 6 elements, inclusive.

- Each cutout (represented by a String[]) is connected (there is a path of vertical and horizontal steps between each pair of letters, including only letters).
- A cutout will not contain a row composed entirely of spaces.
- A cutout will not contain a column composed entirely of spaces.

NOTES
- The length of the elements in cutoutA may be different than the length of the elements in cutoutB.
- The number of elements in cutoutA may be different than the number of elements in cutoutB.

EXAMPLE 1
```
{"  C ",   {" D ",
 "ADEF",     " C ",
 "B   "} ,  "EFA",  match should return 4
             " B "}
```

EXAMPLE 2
```
{"A  ",   {"AB ",
 "BC ",     " C ",
 " DE",     " D ",  match should return 3
 "  F"} ,  " EF "}
```

EXAMPLE 3
```
{"A  B",   {"  E ",
 "CDEF"} ,  "ABCD",  match should return 2
             " F  ")
```

EXAMPLE 4
```
{" B  ",   {"BEAD",
 "FACE",     "C   ",  match should return 1
 " D  "} ,  "F   "}
```

EXAMPLE 5
```
{"ABCDEF"},{"A",
            "B",
            "C",
            "D",     match should return 0
            "E",
            "F"}
```

EXAMPLE 6
```
{" BE",  {"EB ",
 " A ",    " A ",
 "CD ",    " DC",     match should return 4
 "F  "} , "  F"}
```