Cube It

This problem will be worth 13 points You will write two methods in the `CubeIt` class. The two methods are `isPossible( int t, int f, int r)` and `getRightSide(int top, int front)`. The `CubeIt` class has a single constructor that takes a single `int[]` containing six values {1 – 6} representing a 6 sided die.

The `getRightSide(int top, int front)` returns the value on the right side of the die with the given `top` and `front` values. If it is not possible for the die to have both the given `top` and `front` value, return -1.

The `isPossible(int top, int front, int right)` returns a `boolean` value indicating if the configuration is possible. That is, it will return `true` if it is possible for the top of the dice to equal `top`, and the front of the die to equal `front` and the right side of the die to equal `right`. And return `false` otherwise.

For example, the image in Figure 1 is a standard 6-sided die and can be represented:

```
new int[] {6, 4, 5, 3, 1, 2};
```

The dice in Figure 2 are two possible images of the top, front and right side of the standard 6 sided dice.

The following sample code uses a standard 6-sided die.

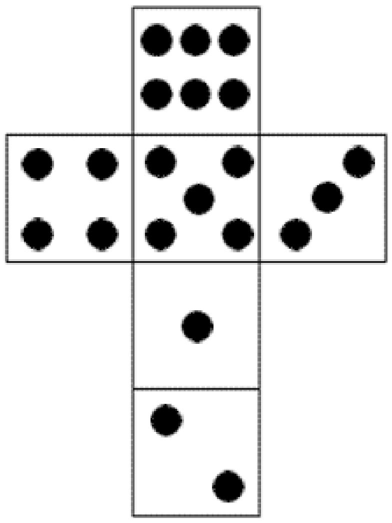
Figure 1

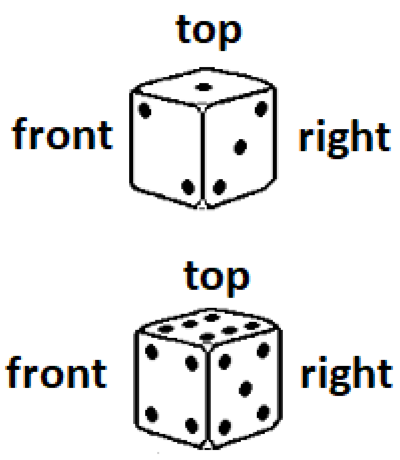

Figure 2

The following code shows the results of the `getRightSide(int top, int front)` method.

The following code	Returns
`CubeIt c = new CubeIt( new int[] {6, 4, 5, 3, 1, 2});`	
`c.getRightSide(1, 2);`	3
`c.getRightSide(6, 4);`	5

The following code shows the results of the `isPossible(int top, int front, int right)` method.

The following code	Returns
`CubeIt c = new CubeIt( new int[] {6, 4, 5, 3, 1, 2});`	
`c.isPossible(6, 4, 5);`	true
`c.isPossible(1, 2, 3);`	true

Cube It

The image in Figure 3 is represented by: `new int[] {1, 2, 3, 4, 5, 6}` and represents a 6-sided die.

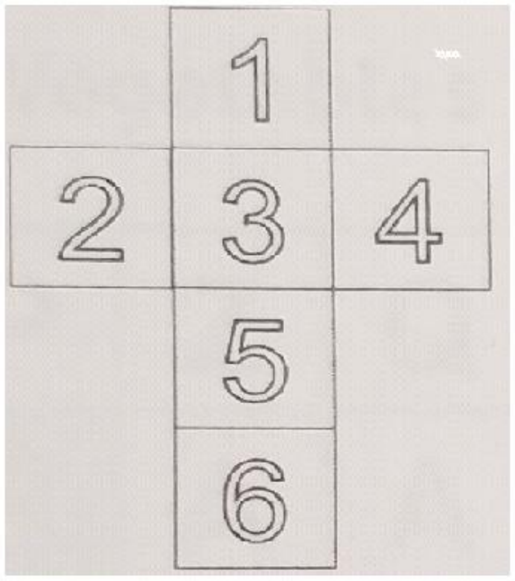

Figure 3

The die in Figure 4 is **<u>NOT</u>** a possible image of the top, front and right side of the 6 sided dice from figure 3.

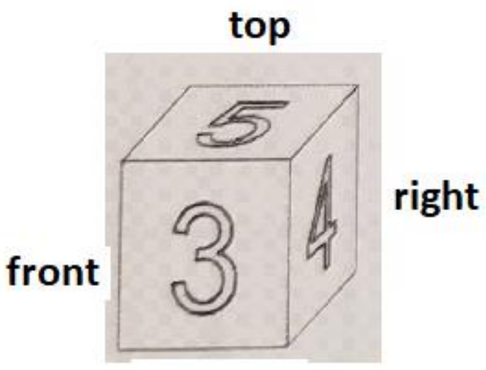

Figure 4

The following code shows the results of the `getRightSide(int top, int front)` method.

The following code	Returns
`CubeIt c = new CubeIt( new int[] {1, 2, 3, 4, 5, 6});`	
`c.getRightSide(5, 3);`	2

The following code shows the results of the `isPossible(int top, int front, int right)` method.

The following code	Returns
`CubeIt c = new CubeIt( new int[] {1, 2, 3, 4, 5, 6});`	
`c.isPossible(5, 3, 4);`	false
`c.isPossible(5, 3, 2);`	True

One more example on next page

Cube It

One last example, Figure 5 is example of a 6 sided die represented by: `new int[] {1, 5, 2, 6, 3, 4};`

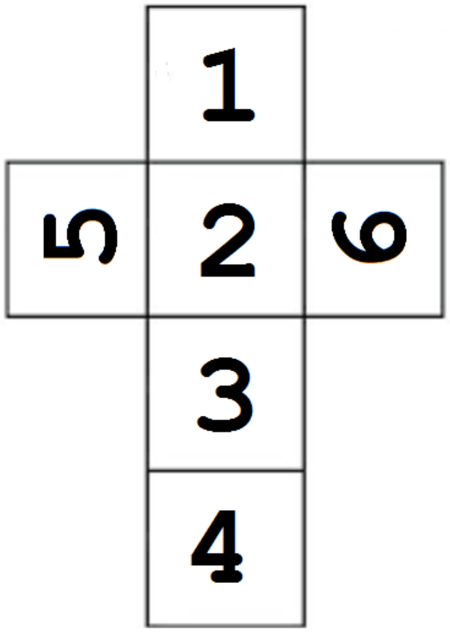

Figure 5

The following code shows the results of the `getRightSide(int top, int front)` method.

The following code	Returns
`CubeIt c = new CubeIt( new int[] {1, 5, 2, 6, 3, 4});`	
`c.getRightSide(1, 2);`	6
`c.getRightSide(5, 2);`	1
`c.getRightSide(5, 6);` // remember to return -1 if it is not possible for 5 to be on top and 6 in front.	-1

The following code shows the results of the `isPossible(int top, int front, int right)` method.

The following code	Returns
`CubeIt c = new CubeIt( new int[] {1, 5, 2, 6, 3, 4});`	
`c.isPossible(1, 2, 6);`	true
`c.isPossible(5, 2, 1);`	true
`c.isPossible(4, 1, 6);`	true