32、转骰子，考点 or 实现——逻辑分析

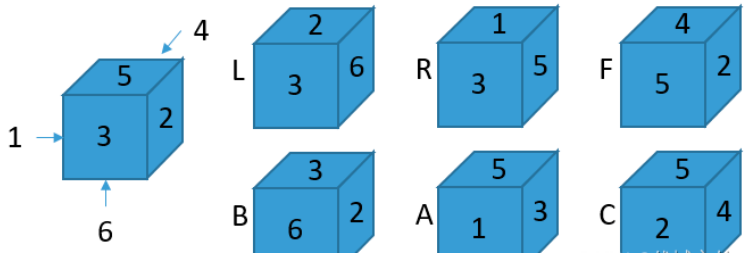
题目描述

骰子是一个立方体，每个面一个数字，初始为左1，右2，前3(观察者方向)，后4，上5，下6，用123456表示这个状态，放置在平面上，

- 可以向左翻转(用L表示向左翻转1次)，
- 可以向右翻转(用R表示向右翻转1次)，
- 可以向前翻转(用F表示向前翻转1次)，
- 可以向后翻转(用B表示向后翻转1次)，
- 可以逆时针旋转(用A表示逆时针旋转90度)，
- 可以顺时针旋转(用C表示顺时针旋转90度)，

现从123456这个初始状态开始，根据输入的动作序列，计算得到最终的状态。

骰子的初始状态和初始状态转动后的状态如图所示。



输入描述

输入一行，为只包含LRFBAC的字母序列，最大长度为50，字母可重复。

输出描述

输出最终状态

用例

输入	L R
输出	123456
说明	无

输入	F C R
输出	342156
说明	无

题目解析

本题感觉就是一道耗时题，考察细心程度的。具体的逻辑反而不是很难。

具体逻辑请看源码。

JavaScript算法源码

```javascript
/* JavaScript Node ACM模式 控制台输入获取 */
const readline = require("readline");

const rl = readline.createInterface({
  input: process.stdin,
  output: process.stdout,
});

rl.on("line", (line) => {
  const directives = line.split(" ");
  turnDice(directives);
});

function turnDice(directives) {
  const dice = new Dice();

  directives.forEach((directive) => {
    switch (directive) {
      case "L":
        dice.turnL();
        break;
      case "R":
        dice.turnR();
        break;
      case "F":
        dice.turnF();
        break;
      case "B":
        dice.turnB();
        break;
      case "A":
        dice.turnA();
        break;
      case "C":
        dice.turnC();
        break;
    }
  });

  dice.print();
```

```
}

class Dice {
  constructor() {
    this.left = 1;
    this.right = 2;
    this.front = 3;
    this.back = 4;
    this.top = 5;
    this.bottom = 6;
  }

  turnL() {
    // 前后不变，上变左，左变下，下变右，右变上
    let tmp = this.right;
    this.right = this.bottom;
    this.bottom = this.left;
    this.left = this.top;
    this.top = tmp;
  }

  turnR() {
    // 前后不变，上变右，右变下，下变左，左变上
    let tmp = this.left;
    this.left = this.bottom;
    this.bottom = this.right;
    this.right = this.top;
    this.top = tmp;
```

```
  }

  turnF() {
    // 左右不变，上变前，前变下，下变后，后变上
    let tmp = this.front;
    this.front = this.top;
    this.top = this.back;
    this.back = this.bottom;
    this.bottom = tmp;
  }

  turnB() {
    // 左右不变，前变上，上变后，后变下，下边前
    let tmp = this.top;
    this.top = this.front;
    this.front = this.bottom;
    this.bottom = this.back;
    this.back = tmp;
  }

  turnA() {
    // 上下不变，  前变右，右变后，后变左，左变前
    let tmp = this.right;
    this.right = this.front;
    this.front = this.left;
    this.left = this.back;
    this.back = tmp;
  }

  turnC() {
    // 上下不变，  右变前，前变左，左变后，后变右
    let tmp = this.front;
    this.front = this.right;
    this.right = this.back;
    this.back = this.left;
    this.left = tmp;
```

```
  }

  print() {
    let { left, right, front, back, top, bottom } = this;
    console.log(`${left}${right}${front}${back}${top}${bottom}`);
  }
}
```

Java算法源码

```
import java.util.Scanner;

public class Main {
  public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    String[] directives = sc.nextLine().split(" ");
    turnDice(directives);
  }

  public static void turnDice(String[] directives) {
    Dice dice = new Dice();

    for (String directive : directives) {
      switch (directive) {
        case "L":
          dice.turnL();
          break;
        case "R":
          dice.turnR();
          break;
        case "F":
          dice.turnF();
          break;
        case "B":
          dice.turnB();
          break;
        case "A":
          dice.turnA();
          break;
        case "C":
          dice.turnC();
          break;
      }
    }

```

```
36      dice.print();
37    }
38  }
39
40  class Dice {
41    int left = 1;
42    int right = 2;
43    int front = 3;
44    int back = 4;
45    int top = 5;
46    int bottom = 6;
47
48    public void turnL() {
49      // 前后不变，上变左，左变下，下变右，右变上
50      int tmp = this.right;
51      this.right = this.bottom;
52      this.bottom = this.left;
53      this.left = this.top;
54      this.top = tmp;
55    }
56
57    public void turnR() {
58      // 前后不变，上变右，右变下，下变左，左变上
59      int tmp = this.left;
60      this.left = this.bottom;
61      this.bottom = this.right;
62      this.right = this.top;
63      this.top = tmp;
64    }
65
```

```
66    public void turnF() {
67      // 左右不变，上变前，前变下，下变后，后变上
68      int tmp = this.front;
69      this.front = this.top;
70      this.top = this.back;
71      this.back = this.bottom;
72      this.bottom = tmp;
73    }
74
75    public void turnB() {
76      // 左右不变，前变上，上变后，后变下，下边前
77      int tmp = this.top;
78      this.top = this.front;
79      this.front = this.bottom;
80      this.bottom = this.back;
81      this.back = tmp;
82    }
83
84    public void turnA() {
85      // 上下不变，  前变右，右变后，后变左，左变前
86      int tmp = this.right;
87      this.right = this.front;
88      this.front = this.left;
89      this.left = this.back;
90      this.back = tmp;
91    }
92
93    public void turnC() {
```

```
94      // 上下不变，  右变前，前变左，左变后，后变右
95      int tmp = this.front;
96      this.front = this.right;
97      this.right = this.back;
98      this.back = this.left;
99      this.left = tmp;
100   }
101
102   public void print() {
103     String sb =
104         String.valueOf(this.left) + this.right + this.front + this.back + this.top + this.bottom;
105     System.out.println(sb);
106   }
107 }
```

Python算法源码

```python
# 算法入口
def turnDice(directives):
    dice = Dice()

    for directive in directives:
        if directive == "L":
            dice.turnL()
        elif directive == "R":
            dice.turnR()
        elif directive == "F":
            dice.turnF()
        elif directive == "B":
            dice.turnB()
        elif directive == "A":
            dice.turnA()
        elif directive == "C":
            dice.turnC()

    return str(dice)

# 骰子实现类
class Dice:
    def __init__(self):
        self.left = 1
        self.right = 2
        self.front = 3
        self.back = 4
        self.top = 5
        self.bottom = 6

    def turnL(self):
        # 前后不变，上变左，左变下，下变右，右变上
        self.right, self.bottom, self.left, self.top = self.bottom, self.left, self.top, self.right

    def turnR(self):
        # 前后不变，上变右，右变下，下变左，左变上
        self.left, self.bottom, self.right, self.top = self.bottom, self.right, self.top, self.left

```

```python
    def turnF(self):
        # 左右不变，上变前，前变下，下变后，后变上
        self.front, self.top, self.back, self.bottom = self.top, self.back, self.bottom, self.front

    def turnB(self):
        # 左右不变，前变上，上变后，后变下，下边前
        self.top, self.front, self.bottom, self.back = self.front, self.bottom, self.back, self.top

    def turnA(self):
        # 上下不变， 前变右，右变后，后变左，左变前
        self.right, self.front, self.left, self.back = self.front, self.left, self.back, self.right

    def turnC(self):
        # 上下不变， 右变前，前变左，左变后，后变右
        self.front, self.right, self.back, self.left = self.right, self.back, self.left, self.front

    def __str__(self):
        return f"{self.left}{self.right}{self.front}{self.back}{self.top}{self.bottom}"


# 输入获取
directives = input().split()

# 算法调用
print(turnDice(directives))
```