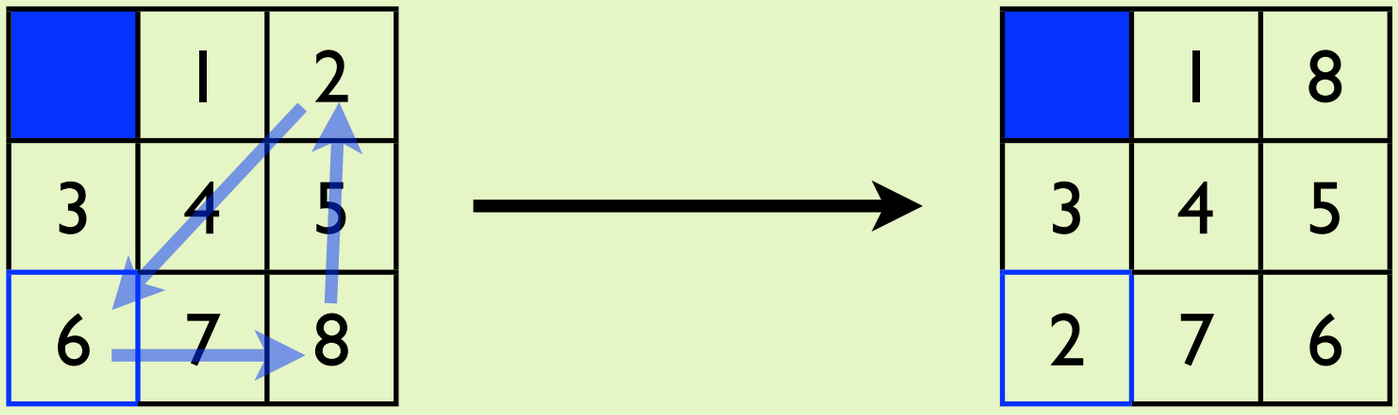
本日の課題

初期のコマの配置（ここでは[0, 1, 2, 3, 4, 5, 6, 7, 8]）から与えられた３つのコマを順に置換した配置にするための動きをもとめる関数を求めよ．たとえば，`kadai([2, 6, 8])`は左の状態から右の状態にするための動きを表現する文字列を返すものとする．ただし，そのような置換が不可能な場合にはnullを返すものとする．また，入力は常に要素が3つの配列であるとする.



`kadai([2, 6, 8]) = "llurdruluruddlluurddlurdr"`

解答例 (1)

一般的な配置から最終状態への最適手順を得るためのプログラムはあらかじめ与えられているので，ほとんどの部分はそれを流用する．具体的には以下のような関数が解となる．

```
function kadai(three_pos){
    function rotate_state(p1, p2, p3){
        var state = [0, 1, 2, 3, 4, 5, 6, 7, 8]
        var temp = state[p3]
        state[p3] = state[p2]
        state[p2] = state[p1]
        state[p1] = temp
        return state
    }
    var rdir = "dulr"
    var [p1, p2, p3] = three_pos
    var state1 = [0, 1, 2, 3, 4, 5, 6, 7, 8]
    var state = rotate_state(p1, p2, p3)
    var mm = []
    var queue = [make_node(null, state, null)]
    while (queue.length > 0){
        var node = queue.shift()
        if (eq_pat(node[1], state1) == 0) break
        var mlist = next_move_list(node[1], node[0])
        for (var i = 0; i < mlist.length; i++){
            var st = move(node[1], mlist[i])
            if (in_list(encode(st), mm)) continue
            var node1 = make_node(mlist[i], st, node)
            mm.push(encode(st))
            queue.push(node1)
        }
    }
    if (queue.length == 0) return null
    var action = ""
    while (true){
        if (node[0] == null) break
        action = action + rdir[node[0]]
        node = node[2]
    }
    return action
}
```

逆向きの動き

解答例 (2)

- これを実際に実行すれば，以下のようになる．

```
puts(kadai([2, 6, 8]))
```

↓

```
move = "luldruurdllurdrulddlurdr"
```