

题目描述

五张牌，每张牌由牌大小和花色组成，牌大小2~10、J、Q、K、A，牌花色为红桃、黑桃、梅花、方块四种花色之一。

判断牌型:

- 牌型1，同花顺：同一花色的顺子，如红桃2红桃3红桃4红桃5红桃6。
- 牌型2，四条：四张相同数字 + 单张，如红桃A黑桃A梅花A方块A + 黑桃K。
- 牌型3，葫芦：三张相同数字 + 一对，如红桃5黑桃5梅花5 + 方块9梅花9。
- 牌型4，同花：同一花色，如方块3方块7方块10方块J方块Q。
- 牌型5，顺子：花色不一样的顺子，如红桃2黑桃3红桃4红桃5方块6。
- 牌型6，三条：三张相同+两张单。

说明:

- (1) 五张牌里不会出现牌大小和花色完全相同的牌。
- (2) 编号小的牌型较大，如同花顺比四条大，依次类推。
- (3) 包含A的合法的顺子只有10 J Q K A和A 2 3 4 5,类似K A 2 3 4的序列不认为是顺子。

输入描述

输入由5行组成，每行为一张牌大小和花色，牌大小为2~10、J、Q、K、A，花色分别用字符H、S、C、D表示红桃、黑桃、梅花、方块。

输出描述

输出牌型序号，5张牌符合多种牌型时，取最大的牌型序号输出。

用例

输入	4 H 5 S 6 C 7 D 8 D
输出	5
说明	4 5 6 7 8构成顺子，输出5

输入	9 S 5 S 6 S 7 S 8 S
输出	1
说明	既是顺子又是同花，输出1，同花顺

题目解析

这道题应该是一道 逻辑题，这道题目输入了五张牌，然后我们需要对这五张牌进行六种情况的分析，并且分析有优先级：

1. 是否为同花顺
2. 是否为四条
3. 是否为葫芦
4. 是否同花
5. 是否为顺子
6. 是否为三条

若满足了前面，则后面的就不需要再判断了。

另外，同花顺情况判断，其实就是 同花 + 顺子，因此在写代码时可以再拆下代码。

上面六种情况的判断，其实可以将五张牌的大小和花色分开来，分别检查。因此我将输入的五张牌的大小存入nums数组，花色存入colors数组。

其中顺子的判断，其实不关心花色，只关心牌大小，我们只需要为牌定义好大小后，进行大小升序排序，若后面一张牌的大小总是比其前面一张牌大1，则就是顺子。

另外还有一个特殊情况，即'A2345'也是顺子，这里我们给A定义的大小是14，因此nums升序排序后，只需要判断nums.join("")是否和'2345A'相等，即可判定为顺子。

而四条的判断，也不关心花色，只关系牌大小，四条即四个相同牌大小，一个不同牌大小。这个很好判断，我这里的方案是，new Set(nums)来去重，若去重后不是两张牌，那么就肯定不是四条，若有两张牌，则任取一张，看在nums中有几个，若为1个或4个，则可以判定为四条。

葫芦、三条的判定和四条类似。

同花的判定，即new Set(colors)，若去重后只有一个花色，则判定为同花。

另外，本题没有说如果输入的五张牌都不满足上面六种情况时，该输出啥，一个可能是，用例保证输入的五张牌肯定满足上面六种情况之一，因此不需要考虑这种异常场景。一个可能是，出题人遗漏了这种情况的输出说明，我这里把不满足上面六种情况的输出自定义为0。

## Java算法源码

```
1 import java.util.Arrays;
2 import java.util.HashMap;
3 import java.util.HashSet;
4 import java.util.Scanner;
5
6 public class Main {
7     public static void main(String[] args) {
8         Scanner sc = new Scanner(System.in);
9
10        String[] nums = new String[5];
11        String[] colors = new String[5];
12
13        for (int i = 0; i < 5; i++) {
14            nums[i] = sc.next();
15            colors[i] = sc.next();
16        }
17
18        System.out.println(getResult(nums, colors));
19    }
20 }
```

```

21 public static int getResult(String[] nums, String[] colors) {
22     Arrays.sort(nums, (a, b) -> cards(a) - cards(b));
23
24     if (isShunzi(nums) && isTonghua(colors)) return 1;
25     else if (issitiao(nums)) return 2;
26     else if (isHulu(nums)) return 3;
27     else if (isTonghua(colors)) return 4;
28     else if (isShunzi(nums)) return 5;
29     else if (issantiao(nums)) return 6;
30     else return 0;
31 }
32
33 // 牌大小 映射为 数值
34 public static int cards(String num) {
35     switch (num) {
36         case "J":
37             return 11;
38         case "Q":
39             return 12;
40         case "K":
41             return 13;
42         case "A":
43             return 14;
44         default:
45             return Integer.parseInt(num);
46     }
47 }
48

```

```

49 // 顺子
50 public static boolean isShunzi(String[] nums) {
51     if ("2345A".equals(String.join("", nums))) return true;
52
53     for (int i = 1; i < nums.length; i++) {
54         int num1 = cards(nums[i - 1]);
55         int num2 = cards(nums[i]);
56         if (num1 + 1 != num2) return false;
57     }
58
59     return true;
60 }
61
62 // 同花
63 public static boolean isTonghua(String[] colors) {
64     // 同花牌的所有花色都一样
65     return new HashSet<String>(Arrays.asList(colors)).size() == 1;
66 }
67
68 // 四条
69 public static boolean issitiao(String[] nums) {
70     // 四条由两部分组成，一个部分四张相同牌，一个部分一张牌
71     return countNums(nums, 2, 4);
72 }
73
74 // 葫芦

```

```
74 // 葫芦
75 public static boolean isHulu(String[] nums) {
76     // 葫芦由两部分组成，一个部分三张牌相同，一个部分两张牌相同
77     return countNums(nums, 2, 3);
78 }
79
80 // 三条
81 public static boolean issantiao(String[] nums) {
82     // 三条由三部分组成，第一个部分由三张相同牌组成，第二个，第三个部分分别是两种不同的牌
83     return countNums(nums, 3, 3);
84 }
85
86 private static boolean countNums(String[] nums, int partCount, int maxSameNumCount) {
87     HashMap<String, Integer> count = new HashMap<>();
88
89     for (String num : nums) {
90         count.put(num, count.getOrDefault(num, 0) + 1);
91     }
92
93     if (count.keySet().size() != partCount) return false;
94
95     return count.containsValue(maxSameNumCount);
96 }
97 }
```

## JS算法源码

```
1  /* JavaScript Node ACM模式 控制台输入获取 */
2  const readline = require("readline");
3
4  const rl = readline.createInterface({
5    input: process.stdin,
6    output: process.stdout,
7  });
8
9  const lines = [];
10 rl.on("line", (line) => {
11   lines.push(line);
12
13   if (lines.length === 5) {
14     const arr = lines.map((line) => line.split(" "));
15     console.log(getResult(arr));
16     lines.length = 0;
17   }
18 });
19
20 function getResult(arr) {
21   const nums = [];
22   const colors = [];
23
24   for (let [num, color] of arr) {
25     nums.push(num);
26     colors.push(color);
27   }
28
29   nums.sort((a, b) => cards(a) - cards(b));
30 }
```

```

31     if (isShunzi(nums) && isTonghua(colors)) return 1;
32     else if (isSitiao(nums)) return 2;
33     else if (isHulu(nums)) return 3;
34     else if (isTonghua(colors)) return 4;
35     else if (isShunzi(nums)) return 5;
36     else if (isSantiao(nums)) return 6;
37     else return 0;
38 }
39
40 function cards(num) {
41     switch (num) {
42         case "J":
43             return 11;
44         case "Q":
45             return 12;
46         case "K":
47             return 13;
48         case "A":
49             return 14;
50         default:
51             return parseInt(num);
52     }
53 }
54

```

```

55 // 顺子
56 function isShunzi(nums) {
57     if (nums.join("") === "2345A") return true;
58
59     for (let i = 1; i < nums.length; i++) {
60         const num1 = cards(nums[i - 1]);
61         const num2 = cards(nums[i]);
62
63         if (num1 + 1 !== num2) return false;
64     }
65     return true;
66 }
67
68 // 同花
69 function isTonghua(colors) {
70     // 同花牌的所有花色都一样
71     return new Set(colors).size === 1;
72 }
73
74 // 四条
75 function isSitiao(nums) {
76     // 四条由两部分组成，一个部分四张相同牌，一个部分一张牌
77     return countNums(nums, 2, 4);
78 }
79

```

```

80 // 葫芦
81 function isHulu(nums) {
82     // 葫芦由两部分组成。一个部分三张牌相同，一个部分两张牌相同
83     return countNums(nums, 2, 3);
84 }
85
86 // 三条
87 function isSantiao(nums) {
88     // 三条由三部分组成。第一个部分由三张相同牌组成，第二个，第三个部分分别是两种不同的牌
89     return countNums(nums, 3, 3);
90 }
91
92 function countNums(nums, partCount, maxSameNumCount) {
93     const count = {};
94
95     for (let num of nums) {
96         count[num] = (count[num] ?? 0) + 1;
97     }
98
99     if (Object.keys(count).size !== partCount) return false;
100
101     return Object.values(count).includes(maxSameNumCount);
102 }

```

## Python算法源码

```

1 # 输入获取
2 arr = [input().split() for _ in range(5)]
3
4
5 # 牌大小 映射为 数值
6 def cards(num):
7     if num == "J":
8         return 11
9     elif num == "Q":
10        return 12
11    elif num == "K":
12        return 13
13    elif num == "A":
14        return 14
15    else:
16        return int(num)
17
18

```



```

19 def countNums(nums, partCount, maxSameNumCount):
20     count = {}
21
22     for num in nums:
23         if count.get(num) is None:
24             count[num] = 0
25             count[num] += 1
26
27     if len(count.keys()) != partCount:
28         return False
29
30     return maxSameNumCount in count.values()
31
32
33 # 三条
34 def isSantiao(nums):
35     # 三条由三部分组成，第一个部分由三张相同牌组成，第二个，第三个部分分别是两种不同的牌
36     return countNums(nums, 3, 3)
37
38
39 # 葫芦
40 def isHulu(nums):
41     # 葫芦由两部分组成，一个部分三张牌相同，一个部分两张牌相同
42     return countNums(nums, 2, 3)
43
44

```

```

45 # 四条
46 def isSitiaio(nums):
47     # 四条由两部分组成，一个部分四张相同牌，一个部分一张牌
48     return countNums(nums, 2, 4)
49
50
51 # 同花
52 def isTonghua(colors):
53     # 同花牌的所有花色都一样
54     return len(set(colors)) == 1
55
56
57 # 顺子
58 def isShunzi(nums):
59     if "".join(nums) == "2345A":
60         return True
61
62     for i in range(1, len(nums)):
63         if cards(nums[i - 1]) + 1 != cards(nums[i]):
64             return False
65     return True
66
67

```



```
68 # 算法入口
69 def getResult():
70     nums = []
71     colors = []
72
73     for num, color in arr:
74         nums.append(num)
75         colors.append(color)
76
77     nums.sort(key=lambda x: cards(x))
78
79     if isShunzi(nums) and isTonghua(colors):
80         return 1
81     elif isSitiao(nums):
82         return 2
83     elif isHulu(nums):
84         return 3
85     elif isTonghua(colors):
86         return 4
87     elif isShunzi(nums):
88         return 5
89     elif isSantiao(nums):
90         return 6
91     else:
92         return 0
93
94
95 # 算法调用
96 print(getResult())
```