

数组的一些说明

过程

入桶与出桶

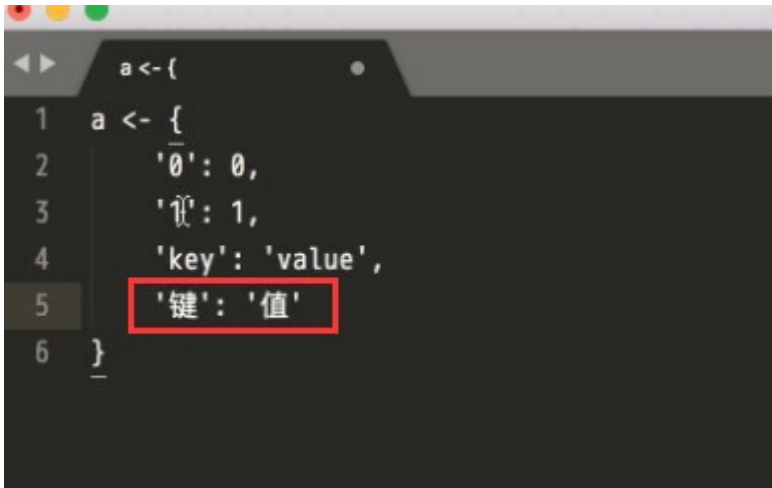
优化

整个代码

<https://github.com/MisterBooo/LeetCodeAnimation>

(学习数据结构的一个非常好的地方)

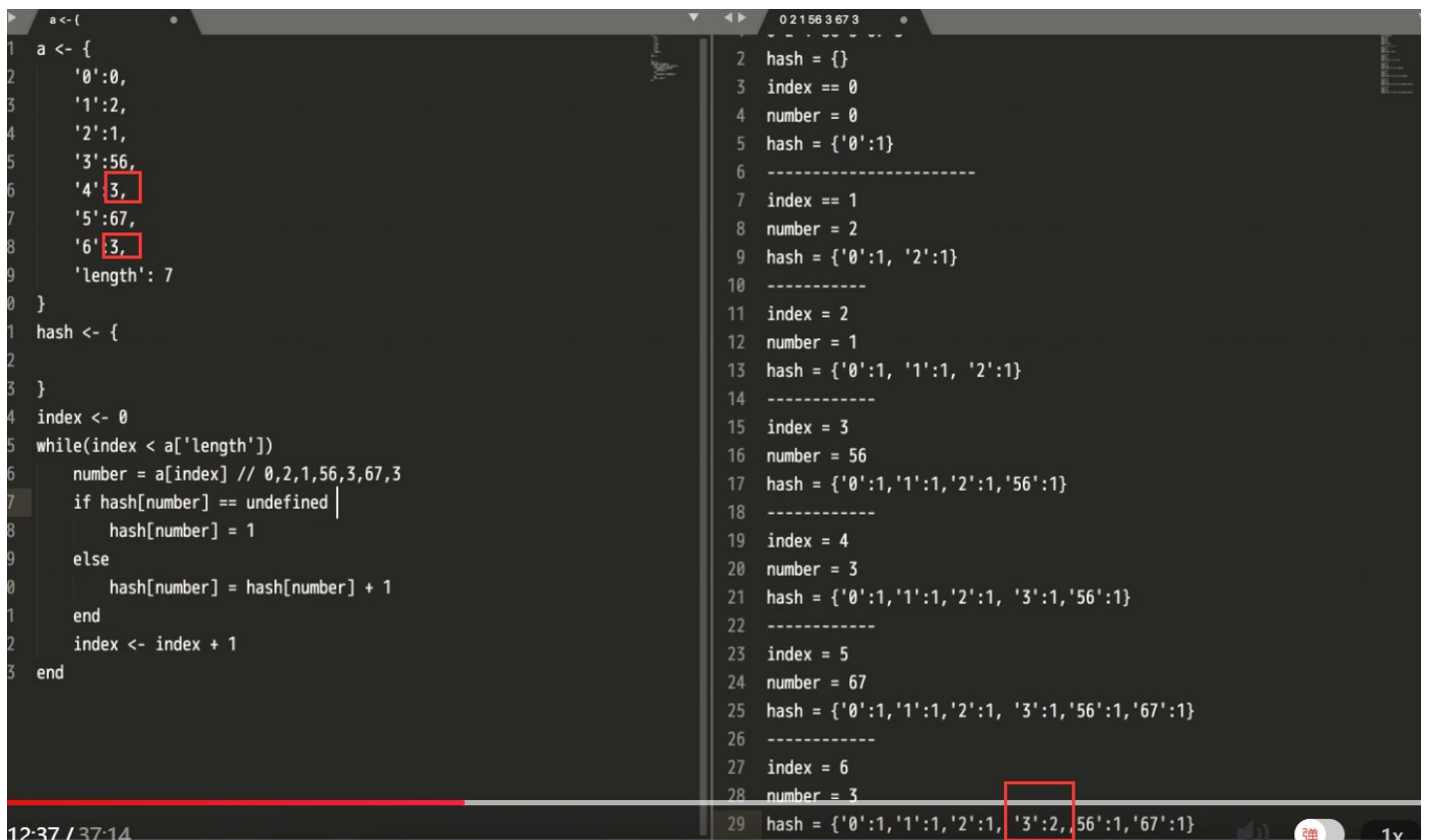
所有满足这种一个键一个值的都可以称为哈希;(数组也是哈希)



```
a <- {
1 a <- {
2   '0': 0,
3   '1': 1,
4   'key': 'value',
5   '键': '值',
6 }
```

随机快排的复杂度: $N\log N$

但是计数快排可以更快;



```
a <- {
1 a <- {
2   '0': 0,
3   '1': 2,
4   '2': 1,
5   '3': 56,
6   '4': 3,
7   '5': 67,
8   '6': 3,
9   'length': 7
10 }
11 hash <- {
12 }
13 index <- 0
14 while(index < a['length'])
15   number = a[index] // 0,2,1,56,3,67,3
16   if hash[number] == undefined
17     hash[number] = 1
18   else
19     hash[number] = hash[number] + 1
20   end
21   index <- index + 1
22 end

23 hash = {}
24 index == 0
25 number = 0
26 hash = {'0':1}
27 -----
28 index == 1
29 number = 2
30 hash = {'0':1, '2':1}
31 -----
32 index = 2
33 number = 1
34 hash = {'0':1, '1':1, '2':1}
35 -----
36 index = 3
37 number = 56
38 hash = {'0':1, '1':1, '2':1, '56':1}
39 -----
40 index = 4
41 number = 3
42 hash = {'0':1, '1':1, '2':1, '3':1, '56':1}
43 -----
44 index = 5
45 number = 67
46 hash = {'0':1, '1':1, '2':1, '3':1, '56':1, '67':1}
47 -----
48 index = 6
49 number = 3
50 hash = {'0':1, '1':1, '2':1, '3':2, '56':1, '67':1}
```

数组的一些说明

```
1 a <- {  
2   '0':0,  
3   '1':2,  
4   '2':1,  
5   '3':56,  
6   '4':4,  
7   '5':67,  
8   '6':3,  
9   '66': 8,  
10  'length': 67  
11 }  
12 hash <- {  
13  
14 }  
15 index <- 0  
16 while(index < a['length'])  
17   number = a[index] // 0,2,1,56,3,67,3  
18   if hash[number] == undefined // hash[number] 不存在  
19     hash[number] = 1  
20   else // hash[number] 存在  
21     hash[number] = hash[number] + 1  
22   end  
23   index <- index + 1  
24 end  
25 index2 <- 0  
26 max <- findMax(a)  
27 while(index2 < max+1) // index2 == max(67) 是可以出现的, index2<68
```

这里的值其实就是下标的最大值+1, 而非这里数组的key的个数

这边是遍历

过程

```

14 index <- 0
15 while(index < a['length'])
16   number = a[index] // 0,2,1,56,3,67,3
17   if hash[number] == undefined // hash[number] 不存在
18     hash[number] = 1
19   else // hash[number] 存在
20     hash[number] = hash[number] + 1
21   end
22   index <- index + 1
23 end
24 index2 <- 0
25 max <- findMax(a)
26 newArr <- {}
27 while(index2 < max+1)
28   count = hash[index2]
29   if count == 1
30     newArr.push(index2)
31   elseif count == 2
32     newArr.push(index2)
33     newArr.push(index2)
34   else count == 3
35     newArr.push(index2)
36     newArr.push(index2)
37     newArr.push(index2)
38   end
39   index2 <- index2 + 1
40 end
41 print newArr

```

```

33 '3':2,
34 '56':1,
35 '67':1
36 }
37
38 入桶
39 -----
40 出桶
41 index2 = 0
42 newArr = [0]
43 index2 = 1
44 newArr = [0,1]
45 index =2
46 newArr = [0,1,2]
47 index2 =3
48 newArr = [0,1,2,3,3]
49 index2 = 4
50 什么也不做
51 index2 = 5
52 什么也不做
53 ...
54 ...
55 index2 = 56
56 newArr = [0,1,2,3,3,56]
57 ...
58 index2 = 67
59 newArr = [0,1,2,3,3,56,67]
60

```

入桶与出桶

```
27 index = 6
28 number = 3
29 hash = {
30     '0':1,
31     '1':1,
32     '2':1,
33     '3':2,
34     '56':1,
35     '67':1
36 }
37
38 入桶
```

```
39 -----
40 出桶
41 index2 = 0
42 newArr = [0]
43 index2 = 1
44 newArr = [0,1]
45 index =2
46 newArr = [0,1,2]
47 index2 =3
48 newArr = [0,1,2,3,3]
49 index2 = 4
50 什么也不做
51 index2 = 5
52 什么也不做
53 ...
54 ...
55 index2 = 56
```

优化

```

1 hash <- {
2
3 }
4 index <- 0
5 while(index < a['length'])
6   number = a[index] // 0,2,1,56,3,67,3
7   if hash[number] == undefined // hash[number] 不存在
8     hash[number] = 1
9   else // hash[number] 存在
10    hash[number] = hash[number] + 1
11  end
12  index <- index + 1
13 end
14 index2 <- 0
15 max <- findMax(a)
16 newArr <- {}
17 while(index2 < max+1)
18   count = hash[index2]
19   if count != undefined
20     countIndex = 0
21     while(countIndex < count)
22       newArr.push(index2)
23       countIndex <- countIndex + 1
24     end
25   end
26   index2 <- index2 + 1
27 end
28
29
30
31
32
33
34
35
36
37 end
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55 index2 = 56
56 newArr = [0,1,2,3,3,56]
57 ...
58 index2 = 67
59 newArr = [0,1,2,3,3,56,67]
60
61 ///
62
63
64 if count == 1
65   newArr.push(index2)
66 elseif count == 2
67   newArr.push(index2)
68   newArr.push(index2)
69 elseif count == 3
70   newArr.push(index2)
71   newArr.push(index2)
72   newArr.push(index2)
73 end
74
75
76
77
78
79
80
81
82

```

整个代码

```
1  a <- {'0':0, '1':2, '2':1, '3':56, '4':4 '5':67, '6':3, 'length':
    67 }
2  hash <- {}
3  index <- 0
4  while(index < a['length'])
5      number = a[index] // 0,2,1,56,3,67,3
6      if hash[number] == undefined // hash[number] 不存在
7          hash[number] = 1
8      else // hash[number] 存在
9          hash[number] = hash[number] + 1
10     end
11     index <- index + 1
12 end
13 index2 <- 0
14 max <- findMax(a)
15 newArr <- {}
16 while(index2 < max+1)
17     count = hash[index2]
18     if count != undefined
19         countIndex = 0
20         while(countIndex < count)
21             newArr.push(index2)
22             countIndex <- countIndex + 1
23         end
24     end
25     index2 <- index2 + 1
26 end
27 print newArr
```