数据结构与算法B 作业2

20742: 泰波拿契数

20 min

http://cs101.openjudge.cn/practice/20742/

思路：创建一个数组，从n=3开始依次将3,4,5……n个斐波那契数append进数组中，直至第n个，输出第n个数即可。

代码：

```
class solution:
    def feibonaqie(self, n):
        feibo_list = [0, 1, 1]
        if n >= 3:
            for i in range(2, n):
                feibo_list.append(feibo_list[i-2]+feibo_list[i-
1]+feibo_list[i])
        return feibo_list[n]


if __name__ == '__main__':
    num = int(input())
    solut = solution()
    print(solut.feibonaqie(num))
```

58A. Chat room

40 min

greedy/strings, 1000, http://codeforces.com/problemset/problem/58/A

思路：没有想到怎么用贪心算法。最后用类似动态规划的算法解出题目。具体思路是，将正确的hello和输入的字符串都转化为列表，随后使用一个range(5)循环，将的第一个字母“h”在输入的字符串中遍历，如有匹配的，就终止此次循环，“正确匹配数（right_index）”+1，并将此次循环的末端设为下一个字母‘e’遍历比较的开端。如果“hello”全部都匹配上，即“正确匹配数”为5，就输出'YES‘，否则输出'NO'。

代码：

```
class solution:
    def hello_right_or_not(self, wrong_hello):
        right_hello = ['h','e','l','l','o']
        index_start = 0
        index_end = len(wrong_hello)
        right_index = 0
        for i in range(5):
            for j in range(index_start, index_end):
                if right_hello[i] == wrong_hello[j]:
                    index_start = j + 1
                    right_index += 1
                    break
        if right_index == 5:
            return 'YES'
        else:
            return 'NO'


if __name__ == '__main__':
    hello = list(input())
    solut = solution()
    print(solut.hello_right_or_not(hello))
```

By Sunswarmth, contest: Codeforces Beta Round 54 (Div. 2), problem: (A) Chat room, **Accepted**, #, Copy

```
class solution:
    def hello_right_or_not(self, wrong_hello):
        right_hello = ['h','e','l','l','o']
        index_start = 0
        index_end = len(wrong_hello)
        right_index = 0
        for i in range(5):
            for j in range(index_start, index_end):
                if right_hello[i] == wrong_hello[j]:
                    index_start = j + 1
                    right_index += 1
                    break
        if right_index == 5:
            return 'YES'
        else:
            return 'NO'


if __name__ == '__main__':
    hello = list(input())
    solut = solution()
    print(solut.hello_right_or_not(hello))
```

118A. String Task

20 min

implementation/strings, 1000, http://codeforces.com/problemset/problem/118/A

思路：倒着遍历给定的String，如果遍历的元素在元音之中，就将其pop。倒着遍历避免index发生变动。

代码：

```
class solution:
    def Task(self, string):
        vowels = ["a", "o", "y", "e", "u", "i"]
        for i in range(len(string)-1,-1,-1):
            if string[i] in vowels:
                string.pop(i)
        return '.'+'.'.join(string)



if __name__ == '__main__':
    String_for_deal = list(input().lower())
    Solution = solution()
    print(Solution.Task(String_for_deal))
```

By Sunswarmth, contest: Codeforces Beta Round 89 (Div. 2), problem: (A) String Task, **Accepted**, #, Copy

```
class solution:
    def Task(self, string):
        vowels = ["a", "o", "y", "e", "u", "i"]
        for i in range(len(string)-1,-1,-1):
            if string[i] in vowels:
                string.pop(i)
        return '.'+'.'.join(string)



if __name__ == '__main__':
    String_for_deal = list(input().lower())
    Solution = solution()
    print(Solution.Task(String_for_deal))
```

E23563: 多项式时间复杂度

20 min

http://cs101.openjudge.cn/pctbook/E23563/

思路：将输入的字符串先按照"+"split后，再对每个字符串按"n^"split，并将非空字符串的数取整型。接着，若第一个数不为0，则取最后一个数与原本储存的ans（初始值为0）比较，取大值。最后输出n^{ans}即可。

代码

```
class solution:
    def Task(self, string):
        split1 = string.split('+')
        ans = 0
        for i in range(len(split1)):
            split2 = [int(x) for x in split1[i].split('n^') if x != '']
            if split2[0] > 0:
                ans = max(ans, split2[-1])
        return f'n^{ans}'



if __name__ == '__main__':
    String_for_deal = input()
    Solution = solution()
    print(Solution.Task(String_for_deal))
```

#50041452提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
class solution:
    def Task(self, string):
        split1 = string.split('+')
        ans = 0
        for i in range(len(split1)):
            split2 = [int(x) for x in split1[i].split('n^') if x != '']
            if split2[0] > 0:
                ans = max(ans, split2[-1])
        return f'n^{ans}'


if __name__ == '__main__':
    String_for_deal = input()
    Solution = solution()
    print(Solution.Task(String_for_deal))
```

基本信息

#: 50041452
题目: E23563
提交人: 22n2200011816(略约横溪)
内存: 3616kB
时间: 22ms
语言: Python3
提交时间: 2025-09-18 19:33:58

24684: 直播计票

40 min（重新学习了字典的用法）

http://cs101.openjudge.cn/practice/24684/

思路：这题比较简单，就是把输入的计票的count存储到字典中的不同数字的key内，然后找到最大的count，输出对应的key即可。一开始想使用一个长度为最大编号的数组来储存count，但是由于输入最多有100个不同的编号，但是最大的编号可能达到100,000，于是果不其然地爆内存了（，最后还是选择了字典。

代码：

```
class solution:
    def vote_counting(self, vote):
        vote_list = list(map(int, vote.split()))
        vote_count = dict()
        max_count = 0
        for i in range(len(vote_list)):
            if vote_list[i] not in vote_count.keys():
                vote_count[vote_list[i]] = 1
            else:
                vote_count[vote_list[i]] += 1
        for key in vote_count.keys():
            if vote_count[key] > max_count:
                max_count_index = [key]
                max_count = vote_count[key]
            elif vote_count[key] == max_count:
                max_count_index.append(key)
        return ' '.join([str(x) for x in sorted(max_count_index)])
```

```
if __name__ == '__main__':
    vote1 = input()
    Solution = solution()
    print(Solution.vote_counting(vote1))
```

#50042061提交状态

查看 提交 统计 提问

状态：Accepted

源代码

```
class solution:
    def vote_counting(self, vote):
        vote_list = list(map(int, vote.split()))
        vote_count = dict()
        max_count = 0
        for i in range(len(vote_list)):
            if vote_list[i] not in vote_count.keys():
                vote_count[vote_list[i]] = 1
            else:
                vote_count[vote_list[i]] += 1
        for key in vote_count.keys():
            if vote_count[key] > max_count:
                max_count_index = [key]
                max_count = vote_count[key]
            elif vote_count[key] == max_count:
                max_count_index.append(key)
        return ' '.join([str(x) for x in sorted(max_count_index)])


if __name__ == '__main__':
    vote1 = input()
    Solution = solution()
    print(Solution.vote_counting(vote1))
```

基本信息

#: 50042061
题目: 24684
提交人: 22n2200011816(略约横溪)
内存: 15312kB
时间: 63ms
语言: Python3
提交时间: 2025-09-18 20:08:21

2. 学习总结和个人收获

目前看起来，对这些较为简单的题目还是可以AC的（虽然对一些基础的知识还是需要不断地百度），目前也是成功热身了。希望后续的题目也可以顺利完成、掌握。除此之外，也写了部分的附加题目：

1.逆波兰表达式求值

150. 逆波兰表达式求值 - 力扣（LeetCode）

思路：非常好栈的题目。将数字添加入栈，如果遇到符号，则将最后两个数字出栈，并求出其结果再添加入栈即可。

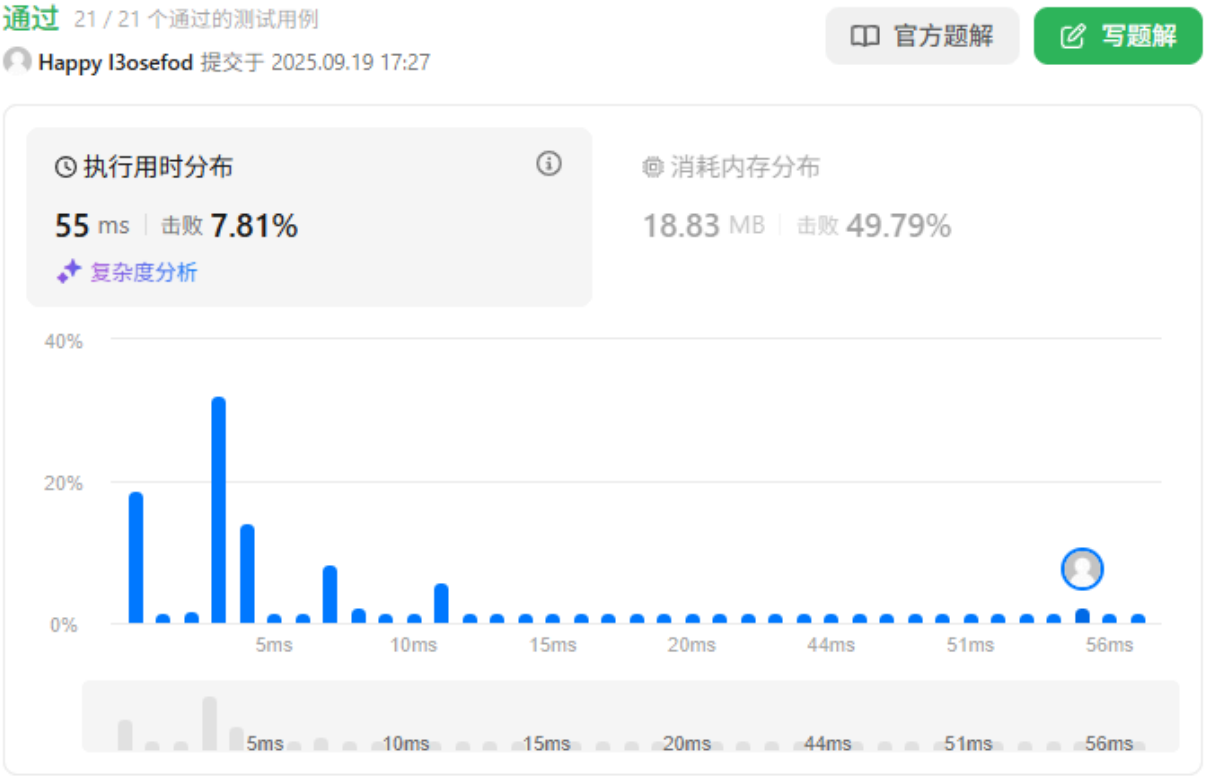
代码：

```
from math import trunc


class Solution:
```

```python
    def evalRPN(self, tokens) -> int:
        stack = []
        sign = ['+','-','*','/']
        for i in tokens:
            if i in sign:
                b,a = stack.pop(),stack.pop()
                cal_result = trunc(eval(''.join([a,i,b])))
                stack.append(str(cal_result))

            else:
                stack.append(i)
        return int(stack[0])
```



代码 | Python3

```python
from math import trunc


class Solution:
    def evalRPN(self, tokens) -> int:
        stack = []
        sign = ['+','-','*','/']
        for i in tokens:
            if i in sign:
                b,a = stack.pop(),stack.pop()
                cal_result = trunc(eval(''.join([a,i,b])))
                stack.append(str(cal_result))

            else:
                stack.append(i)
        return int(stack[0])
```

2.字符串解码

30 min

思路：其实算是我这学期写的第一道栈的题目。一开始，还是用了倒着遍历字符串的方式避免了括号嵌套的情况（具体思路为：倒着遍历字符串，遇到数字，就单独设定一个index=k，继续倒着遍历数字，另设一个index=j，正着遍历字母，最后decode出密码），但是看了题解才知道要怎么用上栈。不过看懂之后就去练其他栈的题目去了（其实时间有限，只写了一道逆波兰表达式。。），没有再重写这道题的代码，但是看了题解+练习，后面第三周遇到栈也是没有再遇到困难。

```python
def is_number(s):
    try:
        int(s)
        return True
    except ValueError:
        pass
    return False

class Solution:
    def decodeString(self, s: str) -> str:
        string_coding = list(s)
        i = len(string_coding)-1
        while i > -1:
            if is_number(string_coding[i]) == 1:
                k = i
                while is_number(string_coding[k]) == 1:
                    k -= 1
                    if k == -1:
                        break
                repeat_number = int(''.join(string_coding[k+1:i+1]))
                j = i+1

                while string_coding[j] != ']':
                    j += 1
                str_decode = ''.join(string_coding[i+2:j]*repeat_number)
                i = k+1
                del string_coding[i:j+1]
                string_coding.insert(i,str_decode)
            i -= 1
        return ''.join(string_coding)
```

通过 34 / 34 个通过的测试用例

Happy l3osefod 提交于 2025.09.19 16:46

官方题解 写题解

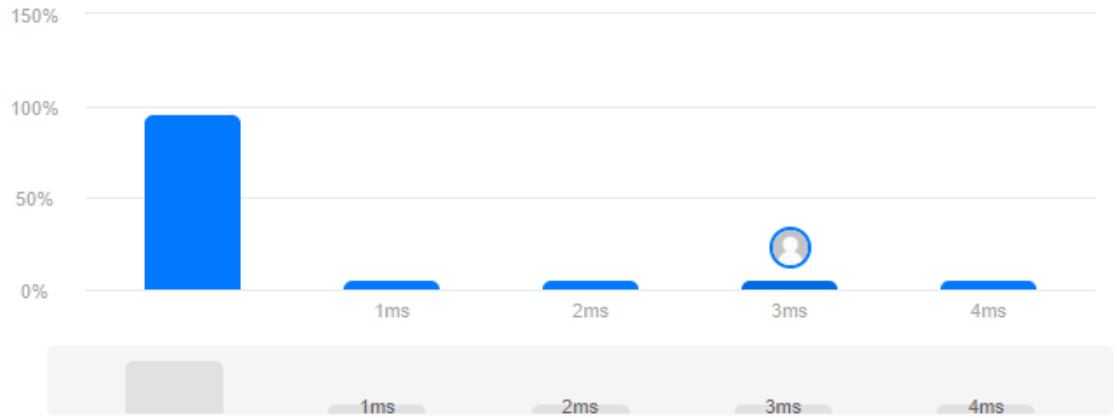
执行用时分布

3 ms | 击败 3.95%

复杂度分析

消耗内存分布

17.63 MB | 击败 33.48%



代码 | Python3

```python
def is_number(s):
    try:
        int(s)
        return True
    except ValueError:
        pass
    return False

class Solution:
    def decodeString(self, s: str) -> str:
        string_coding = list(s)
        i = len(string_coding)-1
        while i > -1:
            if is_number(string_coding[i]) == 1:
                k = i
                while is_number(string_coding[k]) == 1:
```