

Assignment #2: 编程练习

Updated 0953 GMT+8 Feb 24, 2024

2024 spring, Compiled by ==张宇帆 心理与认知科学学院==

操作系统: Windows 11 22H2 22621.3155

Python编程环境: Spyder IDE 5.2.2

C/C++编程环境: 无

1. 题目

27653: Fraction类

思路: 朴实无华的建类

代码

```
# Fraction类
def gcd(m, n):
    while m%n != 0:
        m, n = n, m%n
    return n

class Fraction:
    def __init__(self, num, den):
        self.num = num
        self.den = den

    def __add__(self, otherFract):
        newnum = self.num * otherFract.den + self.den * otherFract.num
        newden = self.den * otherFract.den
        common = gcd(newnum, newden)
        return Fraction(newnum//common, newden//common)

    def __str__(self):
        return str(self.num) + '/' + str(self.den)

first_num, first_den, second_num, second_den = map(int, input().split())
First = Fraction(first_num, first_den)
Second = Fraction(second_num, second_den)
Result = First + Second
print(Result)
```

代码运行截图 == (至少包含有"Accepted") ==

状态: Accepted

源代码

```
def gcd(m, n):
    while m % n != 0:
        m, n = n, m % n
    return n

class Fraction:
    def __init__(self, num, den):
        self.num = num
        self.den = den

    def __add__(self, otherFract):
        newnum = self.num * otherFract.den + self.den * otherFract.num
        newden = self.den * otherFract.den
        common = gcd(newnum, newden)
        return Fraction(newnum // common, newden // common)

    def __str__(self):
        return str(self.num) + '/' + str(self.den)

first_num, first_den, second_num, second_den = map(int, input().split())
First = Fraction(first_num, first_den)
Second = Fraction(second_num, second_den)
Result = First + Second
print(Result)
```

基本信息

#: 43943870
题目: 27653
提交人: 2200013720
内存: 3576kB
时间: 20ms
语言: Python3
提交时间: 2024-02-21 00:07:57

04110: 圣诞老人的礼物-Santa Clau's Gifts

思路: 计算均重价值, 然后排序, 选高的即可

代码

```
# 圣诞老人的礼物-Santa Clau's Gifts
n, w = map(int, input().split())
values = []
for _ in range(n):
    value, weight = map(int, input().split())
    values.append([value/weight, weight])
values.sort(reverse = True)
result = 0
remain = w
i = 0
while i < n and not remain == 0:
    result += values[i][0] * min(values[i][1], remain)
    remain -= min(values[i][1], remain)
    i += 1
print("%.1f\n" % result)
```

代码运行截图 == (至少包含有"Accepted") ==

状态: Accepted

源代码

```
n, w = map(int, input().split())
values = []
for _ in range(n):
    value, weight = map(int, input().split())
    values.append([value/weight, weight])
values.sort(reverse = True)
result = 0
remain = w
i = 0
while i < n and not remain == 0:
    result += values[i][0] * min(values[i][1], remain)
    remain -= min(values[i][1], remain)
    i += 1
print("%.1f\n" % result)
```

基本信息

#: 43982968
题目: 04110
提交人: 2200013720
内存: 3544kB
时间: 22ms
语言: Python3
提交时间: 2024-02-25 10:26:44

18182: 打怪兽

思路: 选择前m个时刻伤害最大的即可

代码

```
# 打怪兽
nCases = int(input())
for _ in range(nCases):
    n, m, HP = map(int, input().split())
    result = 0
    check = [result, HP, m]
    skill = []
    for k in range(n):
        time, reHP = map(int, input().split())
        skill.append([time, reHP])
    skill.sort()
    Die = False
    time = 0
    for i in range(n):
        if time != skill[i][0]:
            haveused = []
            check[2] = m
        if check[1] > 0 and check[2] > 0:
            check[0] = skill[i][0]
            check[1] -= skill[i][1]
            check[2] -= 1
            haveused.append(skill[i][1])
        elif check[1] > 0 and check[2] == 0:
            haveused.sort()
            if haveused[0] < skill[i][1]:
                check[0] = skill[i][0]
                check[1] -= skill[i][1] - haveused[0]
                haveused[0] = skill[i][1]
            time = skill[i][0]
        if check[1] <= 0:
            Die = True
            break
    if Die:
```

```
print(check[0])
else:
    print('alive')
```

代码运行截图 == (AC代码截图, 至少包含有"Accepted") ==

状态: Accepted

源代码

```
nCases = int(input())
for _ in range(nCases):
    n, m, HP = map(int, input().split())
    result = 0
    check = [result, HP, m]
    skill = []
    for k in range(n):
        time, reHP = map(int, input().split())
        skill.append([time, reHP])
    skill.sort()
    Die = False
    time = 0
    for i in range(n):
        if time != skill[i][0]:
            haveused = []
            check[2] = m
        if check[1] > 0 and check[2] > 0:
            check[0] = skill[i][0]
            check[1] -= skill[i][1]
            check[2] -= 1
            haveused.append(skill[i][1])
        elif check[1] > 0 and check[2] == 0:
            haveused.sort()
            if haveused[0] < skill[i][1]:
                check[0] = skill[i][0]
                check[1] -= skill[i][1] - haveused[0]
                haveused[0] = skill[i][1]
            time = skill[i][0]
        if check[1] <= 0:
            Die = True
            break
    if Die:
        print(check[0])
    else:
        print('alive')
```

基本信息

#: 43989141
题目: 18182
提交人: 2200013720
内存: 3708kB
时间: 79ms
语言: Python3
提交时间: 2024-02-26 10:50:45

230B. T-primes

思路: 欧拉筛法打表即可

代码

```
# T-primes
n=int(input())
L=list(map(int,input().split()))
K=[1]*10**6
T_prime=set()
for i in range(2,10**6):
    if K[i]==1:
        T_prime.add(i*i)
        for j in range(i*i,10**6,i):
            K[j]=0
for x in L:
    print(['NO', 'YES'][x in T_prime])
```

代码运行截图 == (AC代码截图, 至少包含有"Accepted") ==

#	Author	Problem	Lang	Verdict
248375260	Practice: yunyu3	230B - 28	Python 3	Accepted

→ [Source](#)

```
n=int(input())
L=list(map(int,input().split()))
K=[1]*10**6
T_prime=set()
for i in range(2,10**6):
    if K[i]==1:
        T_prime.add(i*i)
        for j in range(i*i,10**6,i):
            K[j]=0
for X in L:
    print(['NO','YES'][X in T_prime])
```

1364A. XXXXX

思路: 如果x是1, 那么不存在对应子数组; 如果不是, 考虑A、B、C三个子数组, 三者合集为原数组且无交集, B为假设的最长的目标子数组, 即 $(A+B) \% x == 0$ 且 $(B+C) \% x == 0$, 则

①若A或C能够整除x, 则B必然也能够整除x, 与假设不符

②若A、C均不能被x整除, 则A+B+C必然不能够整除x

这说明A、C必有一个为空集, 换言之之最长目标子数组必然始于左端或终于右端

因此将原数组分为两段, 设置两个变量Sum和reSum代表左右两段的子数组和, 这样每次选择到元素就可以直接变更Sum和reSum值即可, 不必每次计算数组和, 最后选择符合“不整除”最大的长度即可 (Sum对应i, reSum对应n-i)

代码

```
# XXXXX
t = int(input())
for _ in range(t):
    n,x = map(int,input().split())
    values = list(map(int,input().split()))
    Sum = 0
    reSum = sum(values)
    result = -1
    if x == 1:
        print(result)
    else:
        for i in range(n):
            if reSum%x != 0:
                result = max(result, n-i)
            if Sum%x != 0 and i != 0:
                result = max(result, i)
            reSum -= values[i]
            Sum += values[i]
        print(result)
```

代码运行截图 == (AC代码截图, 至少包含有"Accepted") ==

#	Author	Problem	Lang	Verdict	Time	Memory
248373722	Practice: yunyu3	1364A - 15	Python 3	Accepted	373 ms	17608 KB

→ Source

```
t = int(input())
for _ in range(t):
    n, x = map(int, input().split())
    values = list(map(int, input().split()))
    Sum = 0
    reSum = sum(values)
    result = -1
    if x == 1:
        print(result)
    else:
        for i in range(n):
            if reSum % x != 0:
                result = max(result, n-i)
            if Sum % x != 0 and i != 0:
                result = max(result, i)
            reSum -= values[i]
            Sum += values[i]
        print(result)
```

18176: 2050年成绩计算

思路：欧拉筛法打表即可

代码

```
# 2050年成绩计算
T_prime = [1]*(10**4)
for i in range(2,10**4):
    if T_prime[i] == 1:
        for j in range(i*i,10**4,i):
            T_prime[j] = 0
m,n = map(int,input().split())
for _ in range(m):
    grade = list(map(int,input().split()))
    result = 0
    for score in grade:
        if score >= 4 and score**((1/2))%1 == 0 and T_prime[int(score**((1/2)))]:
            result += score
    if result != 0:
        print("%.2f" % (result/len(grade)))
    else:
        print(0)
```

代码运行截图 == (AC代码截图, 至少包含有"Accepted") ==

状态: Accepted

源代码

```
T_prime = [1]*(10**4)
for i in range(2,10**4):
    if T_prime[i] == 1:
        for j in range(i*i,10**4,i):
            T_prime[j] = 0
m,n = map(int,input().split())
for _ in range(m):
    grade = list(map(int,input().split()))
    result = 0
    for score in grade:
        if score >= 4 and score**(1/2)%1 == 0 and T_prime[int(score**(1/2))]:
            result += score
    if result != 0:
        print("%.2f" % (result/len(grade)))
    else:
        print(0)
```

基本信息
#: 43977235
题目: 18176
提交人: 2200013720
内存: 7944kB
时间: 68ms
语言: Python3
提交时间: 2024-02-24 11:26:56

2. 学习总结和收获

==如果作业题目简单，有否额外练习题目，比如：OJ“2024spring每日选做”、CF、LeetCode、洛谷等网站题目。==

重新写代码做题又有不一样的感觉，此处作业确实简单，思路一看就有。有些题先前做过，将此次代码与先前的进行对比发现，先前更多地追求AC，而现今我更多地追求算法的改进（时间与空间上的简洁），当然题目思路也可能并非最佳。

额外练习

虽然看起来做得挺多，但其实很多都是常规思路与解法，对我而言有一定挑战性且有所收获的题目有：1、2、7、8、9、12，只放了一些周二周三上课时间段做的，正在努力地把假期放的题慢慢做，这学期选了matlab还有机器学习相关的一些课，之后应该会很忙，所以想着赶紧把数算先学学，已经在开始学习树和图了。

最后，我在一年半前选了闫老师的计概课，如今又选了数算，真的很感谢闫老师，我真的喜欢上了敲代码，在上学期有时候感到累了就会上OJ和CF挑几道题做做，AC了真的会治愈疲劳的！

1. 08758：2的幂次方表示

思路：先将输入数转化为二进制并存进表里，表的index即为指数。然后使用递归，基本结束条件为1，2，4，并注意到除首项外各项需前置+，因此用flag进行判断。

代码

```
# 2的幂次方表示
def DevideBy2(number):
    H = ['0','1']
    remain = []
    while number != 0:
        remain.append(H[number%2])
        number = number // 2
    result = ''
    flag = 0
    for i in range(len(remain)-1,-1,-1):
        if remain[i] == '1' and not flag:
            if i == 0:
                result += '2(0)'
            else:
                result += '+' + '2(' + str(i) + ')
```

```

        elif i == 1:
            result += '2'
        elif i == 2:
            result += '2(2)'
        else:
            result += '2(' + DevideBy2(i) + ')'
        flag = 1
    elif remain[i] == '1' and flag:
        if i == 0:
            result += '+2(0)'
        elif i == 1:
            result += '+2'
        elif i == 2:
            result += '+2(2)'
        else:
            result += '+2(' + DevideBy2(i) + ')'
    return result
print(DevideBy2(int(input())))

```

AC截图

状态: Accepted

源代码

```

def DevideBy2(number):
    H = ['0', '1']
    remain = []
    while number != 0:
        remain.append(H[number%2])
        number = number // 2
    result = ''
    flag = 0
    for i in range(len(remain)-1, -1, -1):
        if remain[i] == '1' and not flag:
            if i == 0:
                result += '2(0)'
            elif i == 1:
                result += '2'
            elif i == 2:
                result += '2(2)'
            else:
                result += '2(' + DevideBy2(i) + ')'
            flag = 1
        elif remain[i] == '1' and flag:
            if i == 0:
                result += '+2(0)'
            elif i == 1:
                result += '+2'
            elif i == 2:
                result += '+2(2)'
            else:
                result += '+2(' + DevideBy2(i) + ')'
    return result
print(DevideBy2(int(input())))

```

基本信息

#: 43952331
 题目: 08758
 提交人: 2200013720
 内存: 3616kB
 时间: 21ms
 语言: Python3
 提交时间: 2024-02-21 18:57:19

2. 06250: 字符串最大跨距

思路: 使用found判断是否找到S1、S2, 并分为以下判断条件: ①找到S1而未找到S2, 记录S1位置; ②找到S1而已找到S2, 说明S2在S1前, 不满足条件, break; ③未找到S1而找到S2, 同②; ④找到S2而已找到S2, 记录S2位置并取最大值。总体思路较为常规。

代码

```

# 字符串最大跨距
string, start, end = map(str, input().split(', '))
lenstart = len(start)

```



```

flagstart = -1
foundstart = False
lenend = len(end)
flagend = -1
foundend = False
i = 0
while i < len(String):
    if String[i:i + lenstart] == start and not foundstart:
        flagstart = i + lenstart
        foundstart = True
        i += lenstart
    elif String[i:i + lenend] == end and foundstart:
        flagend = max(i, flagend)
        foundend = True
        i += lenend
    elif String[i:i + lenend] == end and not foundstart:
        break
    elif String[i:i + lenstart] == start and foundend:
        foundstart = False
        break
    else:
        i += 1
if foundstart and foundend:
    print(flagend - flagstart)
else:
    print(-1)

```

AC截图

状态: Accepted

源代码

```

String, start, end = map(str, input().split(' '))
lenstart = len(start)
flagstart = -1
foundstart = False
lenend = len(end)
flagend = -1
foundend = False
i = 0
while i < len(String):
    if String[i:i + lenstart] == start and not foundstart:
        flagstart = i + lenstart
        foundstart = True
        i += lenstart
    elif String[i:i + lenend] == end and foundstart:
        flagend = max(i, flagend)
        foundend = True
        i += lenend
    elif String[i:i + lenend] == end and not foundstart:
        break
    elif String[i:i + lenstart] == start and foundend:
        foundstart = False
        break
    else:
        i += 1
if foundstart and foundend:
    print(flagend - flagstart)
else:
    print(-1)

```

基本信息

#: 43951828
 题目: 06250
 提交人: 2200013720
 内存: 3636kB
 时间: 22ms
 语言: Python3
 提交时间: 2024-02-21 18:04:47

3. 24588: 后序表达式求值

思路：应用栈的思路去解决，使得每个运算符都能对应上正确的两个计算数。（这道题一开始直接把自己在假期写的Stack类直接用来，然后可能和OJ自带的冲突了，所以一直RE，麻烦了下助教和老师，以后还是要简单点，不能偷懒）

代码

```
# 后序表达式求值
def compute(stack, operator):
    op1 = float(stack.pop())
    op2 = float(stack.pop())
    if operator == '+':
        return op2 + op1
    elif operator == '-':
        return op2 - op1
    elif operator == '*':
        return op2 * op1
    elif operator == '/':
        return op2 / op1
    elif operator == '^':
        return op2 ** op1

def post_eva(formula):
    comp = '+-*/^'
    wordlist = formula.split()
    opStack = []
    for word in wordlist:
        if word not in comp:
            opStack.append(float(word))
        else:
            op = compute(opStack, word)
            opStack.append(op)
    return opStack[0]

n = int(input())
for _ in range(n):
    result = post_eva(input())
    print(f"{result:.2f}")
```

AC截图

状态: Accepted

源代码

```
def compute(stack, operator):
    op1 = float(stack.pop())
    op2 = float(stack.pop())
    if operator == '+':
        return op2 + op1
    elif operator == '-':
        return op2 - op1
    elif operator == '*':
        return op2 * op1
    elif operator == '/':
        return op2 / op1
    elif operator == '^':
        return op2 ** op1

def post_eva(formula):
    comp = '+-*/^'
    wordlist = formula.split()
    opStack = []
    for word in wordlist:
        if word not in comp:
            opStack.append(float(word))
        else:
            op = compute(opStack, word)
            opStack.append(op)
    return opStack[0]

n = int(input())
for _ in range(n):
    result = post_eva(input())
    print(f"{result:.2f}")
```

基本信息

#: 43953198
题目: 24588
提交人: 2200013720
内存: 3656kB
时间: 25ms
语言: Python3
提交时间: 2024-02-21 20:22:06

4. 05345: 位查询

思路: 定义操作函数operate, 然后依据题目要求操作即可, 需要注意i的数位是从word的末尾以0开始, 因此对应位置的index应为-value-1。

代码

```
# 位查询
def operate(operate, wordlist, value):
    if operate == 'C':
        for idx in range(len(wordlist)):
            wordlist[idx] = (wordlist[idx] + value) % 65536
    elif operate == 'Q':
        result = 0
        for word in wordlist:
            if value < len(bin(word)[2:]):
                result += (bin(word)[2:][-value-1] == '1')
        print(result)

N,M = map(int,input().split())
wordlist = list(map(int,input().split()))
for _ in range(M):
    op, value = map(str,input().split())
    operate(op, wordlist, int(value))
```

AC截图

状态: Accepted

源代码

```
def operate(operate, wordlist, value):
    if operate == 'C':
        for idx in range(len(wordlist)):
            wordlist[idx] = (wordlist[idx] + value) % 65536
    elif operate == 'Q':
        result = 0
        for word in wordlist:
            if value < len(bin(word)[2:]):
                result += (bin(word)[2:][-value-1] == '1')
        print(result)

N,M = map(int,input().split())
wordlist = list(map(int,input().split()))
for _ in range(M):
    op, value = map(str,input().split())
    operate(op, wordlist, int(value))
```

基本信息

#: 43956349
题目: 05345
提交人: 2200013720
内存: 3600kB
时间: 21ms
语言: Python3
提交时间: 2024-02-22 10:35:31

5. 07745: 整数奇偶排序

思路：判断奇偶性并分别储存与排序，随后链接即可，本来打算用join链接但不知道怎么把两个列表连接起来并变成字符串，于是干脆分别进行。

代码

```
# 整数奇偶排序
L = list(map(int,input().split()))
odd = []
even = []
for i in L:
    if i % 2 == 0:
        even.append(i)
    else:
        odd.append(i)
even.sort()
odd.sort(reverse = True)
result = ''
for i in odd:
    if len(result) == 0:
        result += str(i)
    else:
        result += ' ' + str(i)
for j in even:
    if len(result) == 0:
        result += str(j)
    else:
        result += ' ' + str(j)
print(result)
```

AC截图

状态: Accepted

源代码

```
L = list(map(int, input().split()))
odd = []
even = []
for i in L:
    if i % 2 == 0:
        even.append(i)
    else:
        odd.append(i)
even.sort()
odd.sort(reverse = True)
result = ''
for i in odd:
    if len(result) == 0:
        result += str(i)
    else:
        result += ' ' + str(i)
for j in even:
    if len(result) == 0:
        result += str(j)
    else:
        result += ' ' + str(j)
print(result)
```

基本信息

#: 43956010
题目: 07745
提交人: 2200013720
内存: 3608kB
时间: 21ms
语言: Python3
提交时间: 2024-02-22 10:09:51

6. 20449: 是否被5整除

思路: 很简单的一道题, 将字符串转为十进制即可, 一开始忘了int里面加2以至于WA了一次, 有点粗心了)

代码

```
# 是否被5整除
A = input()
answer = ''
for i in range(len(A)):
    answer += str(int(int(A[:i+1], 2)%5 == 0))
print(answer)
```

AC截图

状态: Accepted

源代码

```
A = input()
answer = ''
for i in range(len(A)):
    answer += str(int(int(A[:i+1], 2)%5 == 0))
print(answer)
```

基本信息

#: 43964303
题目: 20449
提交人: 2200013720
内存: 3600kB
时间: 21ms
语言: Python3
提交时间: 2024-02-22 23:31:52

7. 06364: 牛的选举

思路: 两次排序即可。关键在于储存牛的编号 (idx+1) 以及第二次判断的写法, 学到了lambda的使用方法:

lambda本质上和def相似, 后接<函数名>: <参数>, 并直接返回结果, 此题中用于sort的key中即为: 选取x时默认选取x[1]进行操作。如果不用key和lambda那么直接遍历也可。

代码

```
# 牛的选举
N,k = map(int,input().split())
votes = []
for idx in range(N):
    first, second = map(int,input().split())
    votes.append([first,second,idx+1])
votes.sort(reverse = True)
votes = votes[:k]
votes.sort(key = lambda x:x[1])
print(votes[-1][2])
```

AC截图

状态: Accepted

源代码

```
N,k = map(int,input().split())
Votes = []
for idx in range(N):
    first, second = map(int,input().split())
    Votes.append([first,second,idx+1])
Votes.sort(reverse = True)
Votes = Votes[:k]
Votes.sort(key = lambda x:x[1])
print(Votes[-1][2])
```

基本信息

#: 43964350
 题目: 06364
 提交人: 2200013720
 内存: 13712kB
 时间: 170ms
 语言: Python3
 提交时间: 2024-02-22 23:48:27

8. 20472: 死循环的机器人

思路：一开始想着判断机器人跑一次之后的位置即可，但是WA了，后来想到还得考虑朝向，如果机器人跑一次后朝向发生变化，那么再经过1/3次他将回到原点，因此机器人的朝向turn在结束后应为0（即仍朝向北）且不在原点，才能保证机器人走出。

代码

```
# 死循环的机器人
move = [[0,1],[-1,0],[0,-1],[1,0]]
turn = 0
start = [0,0]
operate = input()
for op in operate:
    if op == 'G':
        start[0] += move[turn%4][0]
        start[1] += move[turn%4][1]
    elif op == 'L':
        turn += 1
    elif op == 'R':
        turn += 3
print(int(start == [0,0] or turn%4 in [1,2,3]))
```

AC截图

状态: Accepted

源代码

```
move = [[0,1],[-1,0],[0,-1],[1,0]]
turn = 0
start = [0,0]
operate = input()
for op in operate:
    if op == 'G':
        start[0] += move[turn%4][0]
        start[1] += move[turn%4][1]
    elif op == 'L':
        turn += 1
    elif op == 'R':
        turn += 3
print(int(start == [0,0] or turn%4 in [1,2,3]))
```

基本信息

#: 43964455
题目: 20472
提交人: 2200013720
内存: 3596kB
时间: 20ms
语言: Python3
提交时间: 2024-02-23 00:42:29

9. 02039: 反反复复

思路: 第一反应是转为矩阵然后检索, 但转念一想是否可以直接把信息从密码里面一个个摘出来, 在纸上寻找下规律很容易就能写出来了, 关键在于找到步进increase的规律。

代码

```
# 反反复复
cols = int(input())
Password = input()
Message = ''
for col in range(cols):
    row = col
    count = 1
    increase = [2*col+1, 2*cols-2*col-1]
    while row < len(Password):
        Message += Password[row]
        row += increase[count%2]
        count += 1
print(Message)
```

AC截图

状态: Accepted

源代码

```
cols = int(input())
Password = input()
Message = ''
for col in range(cols):
    row = col
    count = 1
    increase = [2*col+1, 2*cols-2*col-1]
    while row < len(Password):
        Message += Password[row]
        row += increase[count%2]
        count += 1
print(Message)
```

基本信息

#: 43966290
题目: 02039
提交人: 2200013720
内存: 3612kB
时间: 22ms
语言: Python3
提交时间: 2024-02-23 12:16:09

10. 02810: 完美立方

思路: 朴实无华的遍历, 本来担心会TLE一直在想怎么减小时间复杂度, 想不出来于是直接提交, 结果没有TLE。

代码

```
# 反反复复
n=int(input())
a=4
while a<n:
    a=a+1
    for b in range(2,a):
        for c in range(b,a):
            for d in range(c,a):
                if d**3+c**3+b**3==a**3:
                    print('Cube = '+str(a)+' Triple = ('+str(b)+' '+str(c)+' '+str(d)+')')
```

AC截图

状态: Accepted

源代码

```
n=int(input())
a=4
while a<n:
    a=a+1
    for b in range(2,a):
        for c in range(b,a):
            for d in range(c,a):
                if d**3+c**3+b**3==a**3:
                    print('Cube = '+str(a)+' Triple = ('+str(b)+' '+str(c)+')
```

基本信息

#: 43964947
 题目: 02810
 提交人: 2200013720
 内存: 3616kB
 时间: 3750ms
 语言: Python3
 提交时间: 2024-02-23 09:51:47

11. 02808: 校门外的树

思路：第一反应就是直接构建列表然后进行区域性更改，最后用Sum即可，后来考虑是否可以在每次输入时更新并储存要移除的范围（以列表的形式），但这样对于每个新输入的移除范围，均要与先前的移除范围进行比较（包含or扩张or不相干），较为麻烦，于是最后直接构建列表了。

代码

```
# 校门外的树
L,M = map(int,input().split())
Area = [1]*(L+1)
for _ in range(M):
    start,end = map(int,input().split())
    Area[start:end+1] = [0]*(end + 1 -start)
print(sum(Area))
```

AC截图

状态: Accepted

源代码

```
cols = int(input())
Password = input()
Message = ''
for col in range(cols):
    row = col
    count = 1
    increase = [2*col+1,2*cols-2*col-1]
    while row < len(Password):
        Message += Password[row]
        row += increase[count%2]
        count += 1
print(Message)
```

基本信息

#: 43966290
 题目: 02039
 提交人: 2200013720
 内存: 3612kB
 时间: 22ms
 语言: Python3
 提交时间: 2024-02-23 12:16:09

12. 04143: 和为给定数

思路:

①第一反应就是对每个数进行遍历, 看看后面有没有某个数使得两者之和为Sum, 但是TLE了, 时间复杂度为 $O(n^2)$;

②于是改为列表, 简单的排序然后选取前半部分, 对Sum-i之后的数是否在后半部分进行判断, 但是TLE了, 上网查了下才发现in在列表中的复杂度为 $O(n)$, 这样复杂度又回归到 $O(n^2)$;

③同时发现in在集合、字典中的复杂度为 $O(1)$, 又由于不知道怎么读取集合中的元素, 所以考虑使用字典, key对应的值即为自身, 由于没有进行排序, 故需要对所有数字进行遍历, 构建result储存较小数最小的那组数对即可。

④最后上网看了下如果用二分查找也可以过, 这样排序和二分时间复杂度均为 $O(n\log n)$, 但是一开始确实没想到用二分查找简化。

代码

```
# 和为给定数
n = int(input())
inter = list(map(int, input().split()))
Sum = int(input())
Dict = {}
result = [Sum, 0]
found = False
for i in range(len(inter)):
    if Sum - inter[i] in Dict:
        found = True
        if min(Sum - inter[i], inter[i]) < result[0]:
            result[0] = min(Sum - inter[i], inter[i])
            result[1] = max(Sum - inter[i], inter[i])
        else:
            Dict[inter[i]] = inter[i]
if found:
    print("%s %s"%(result[0], result[1]))
else:
    print("No")
```

AC截图

状态: Accepted

源代码

```
n = int(input())
inter = list(map(int, input().split()))
Sum = int(input())
Dict = {}
result = [Sum, 0]
found = False
for i in range(len(inter)):
    if Sum - inter[i] in Dict:
        found = True
        if min(Sum - inter[i], inter[i]) < result[0]:
            result[0] = min(Sum - inter[i], inter[i])
            result[1] = max(Sum - inter[i], inter[i])
        else:
            Dict[inter[i]] = inter[i]
if found:
    print("%s %s"%(result[0], result[1]))
else:
    print("No")
```

基本信息

#: 43965818
题目: 04143
提交人: 2200013720
内存: 18092kB
时间: 73ms
语言: Python3
提交时间: 2024-02-23 11:16:59

13. 24591: 中序表达式转后序表达式

思路：利用栈进行结果拼接，通过字典对符号优先级进行判断。应当注意的是所给数字可能为小数与多位数，因此在将数字压入栈时还应赋值变量flag以判断下一个数字该不该接上所压入的数字，如果为符号则无需接上，且应将flag变为False。

代码

```
# 中序表达式转后序表达式
def post(formula):
    prec = {}
    prec["("] = 1
    prec["+"] = 2
    prec["-"] = 2
    prec["*"] = 3
    prec["/"] = 3
    opStack = []
    result = []
    flag = False
    for word in formula:
        if word == '(':
            opStack.append(word)
            flag = False
        elif word == ')':
            topword = opStack.pop()
            while not topword == '(':
                result.append(topword)
                topword = opStack.pop()
            flag = False
        elif word == '.':
            flag = True
            result[-1] += word
        elif word in '+-*/':
            while (not opStack == []) and (prec[opStack[-1]] >= prec[word]):
                result.append(opStack.pop())
            opStack.append(word)
            flag = False
        elif flag:
            result[-1] += word
        elif word not in '+-*/()':
            result.append(word)
            flag = True
    while not opStack == []:
        result.append(opStack.pop())
    return ' '.join(result)

n = int(input())
for _ in range(n):
    print(post(input()))
```

AC截图

状态: Accepted

源代码

```
def post(formula):
    prec = {}
    prec["("] = 1
    prec["+"] = 2
    prec["-"] = 2
    prec["*"] = 3
    prec["/"] = 3
    opStack = []
    result = []
    flag = False
    for word in formula:
        if word == '(':
            opStack.append(word)
            flag = False
        elif word == ')':
            topword = opStack.pop()
            while not topword == '(':
                result.append(topword)
                topword = opStack.pop()
            flag = False
```

基本信息

#: 43969817
题目: 24591
提交人: 2200013720
内存: 3680kB
时间: 25ms
语言: Python3
提交时间: 2024-02-23 17:49:20