Assignment #D: May月考

Updated 1654 GMT+8 May 8, 2024

2024 spring, Complied by ==同学的姓名、院系==

说明：

1）请把每个题目解题思路（可选），源码Python, 或者C++（已经在Codeforces/Openjudge上AC），截图（包含Accepted），填写到下面作业模版中（推荐使用 typora https://typoraio.cn ，或者用word）。AC 或者没有AC，都请标上每个题目大致花费时间。

2）提交时候先提交pdf文件，再把md或者doc文件上传到右侧“作业评论”。Canvas需要有同学清晰头像、提交文件有pdf、"作业评论"区有上传的md或者doc附件。

3）如果不能在截止前提交作业，请写明原因。

编程环境

==（请改为同学的操作系统、编程环境等）==

操作系统：macOS Ventura 13.4.1 (c)

Python编程环境：Spyder IDE 5.2.2, PyCharm 2023.1.4 (Professional Edition)

C/C++编程环境：Mac terminal vi (version 9.0.1424), g++/gcc (Apple clang version 14.0.3, clang-1403.0.22.14.1)

1. 题目

02808: 校门外的树

http://cs101.openjudge.cn/practice/02808/

思路：基本不需要优化。碰到移除区域直接在列表上进行处理就可以

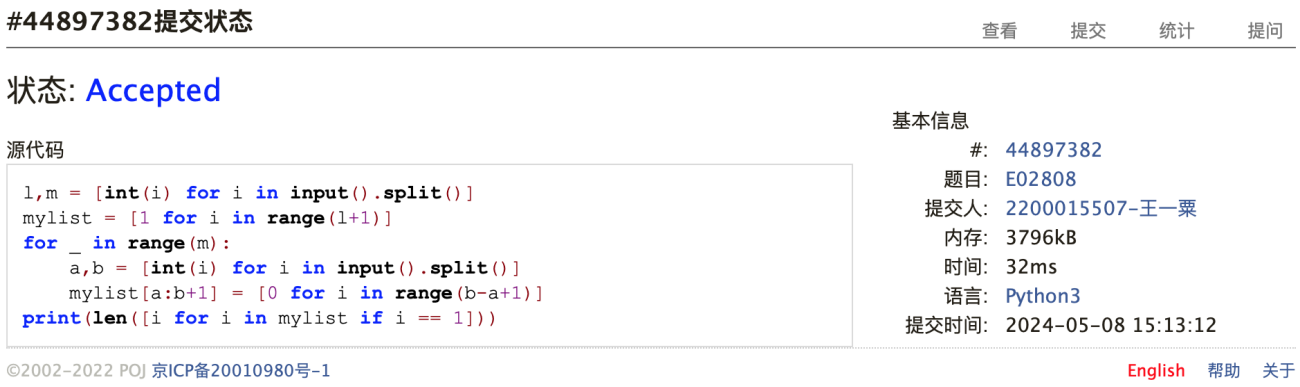
耗时：5min

代码

```
#2200015507 王一粟
l,m = [int(i) for i in input().split()]
mylist = [1 for i in range(l+1)]
for _ in range(m):
```

```
    a,b = [int(i) for i in input().split()]
    mylist[a:b+1] = [0 for i in range(b-a+1)]
print(len([i for i in mylist if i == 1]))
```

代码运行截图 ==（至少包含有"Accepted"）==



20449: 是否被5整除

http://cs101.openjudge.cn/practice/20449/

思路：从左到右做循环，每次移位*2即可

耗时：4min

代码

```
#2200015507 王一粟
result = ""
num_list = [int(i) for i in list(input())]
num = 0
for element in num_list:
    num = num*2 + element
    if num%5 == 0:
        result += str(1)
    else:
        result += str(0)
print(result)
```

代码运行截图 ==（至少包含有"Accepted"）==

#44897460提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
result = ""
num_list = [int(i) for i in list(input())]
num = 0
for element in num_list:
    num = num*2 + element
    if num%5 == 0:
        result += str(1)
    else:
        result += str(0)
print(result)
```

基本信息

#: 44897460
题目: E20449
提交人: 2200015507-王一粟
内存: 3600kB
时间: 19ms
语言: Python3
提交时间: 2024-05-08 15:16:59

English 帮助 关于

01258: Agri-Net

http://cs101.openjudge.cn/practice/01258/

思路：经典Prim算法。每次弹出距离最小的，在没有visited情况下，进行标记后访问未被标记的邻居节点。在全部访问之后，生成最小distance

耗时：30min

代码

```
#2200015507 王一粟
import heapq
while True:
    try:
        n = int(input())
        graph = []
        wait_list = []
        total = 0
        while True:
            a = [int(i) for i in input().split()]
            wait_list = wait_list + a
            total += len(a)
            if total == n ** 2:
                break
        cnt = 0
        for i in range(n):
            mylist = []
            for j in range(n):
                mylist.append(wait_list[cnt])
                cnt += 1
            graph.append(mylist)
        visited = [False for i in range(n)]
        mylist = []
        heapq.heappush(mylist, [0, 0])
        result = 0
        cnt = 0
        while cnt < n:
```

```
            distance, node = heapq.heappop(mylist)
            if visited[node] is True:
                continue
            visited[node] = True
            cnt += 1
            result += distance
            for idx, element in enumerate(graph[node]):
                if visited[idx] is False:
                    heapq.heappush(mylist, [element, idx])
        print(result)
    except:
        break
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

#44898128提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
import heapq

while True:
    try:
        n = int(input())
        graph = []
        wait_list = []
        total = 0
        while True:
            a = [int(i) for i in input().split()]
            wait_list = wait_list + a
            total += len(a)
            if total == n ** 2:
                break
        cnt = 0
        for i in range(n):
```

基本信息

#: 44898128
题目: M01258
提交人: 2200015507-王一粟
内存: 4548kB
时间: 44ms
语言: Python3
提交时间: 2024-05-08 15:46:28

27635: 判断无向图是否连通有无回路(同23163)

http://cs101.openjudge.cn/practice/27635/

思路：经典题目，经典思路，经典算法

耗时：13min

代码

```
def dfs(node):
    visited[node] = True
    for element in graph[node]:
        if visited[element] is False:
            dfs(element)
def loop(node,prev):
    visit[node] = True
    for element in graph[node]:
        if element == prev:
```

```
            continue
        if visit[element] is True:
            return True
        else:
            if loop(element,node):
                return True
    return False

n,m = [int(i) for i in input().split()]
graph = [[] for i in range(n)]
for _ in range(m):
    a,b = [int(i) for i in input().split()]
    graph[a].append(b)
    graph[b].append(a)
#connected
visited = [False for i in range(n)]
dfs(0)
result = [i for i in visited if i==False]
if result == []:
    print('connected:yes')
else:
    print("connected:no")
#loop
visit = [False for i in range(n)]
cnt = 0
for node in range(n):
    if visit[node] is False:
        if loop(node,-1):
            cnt = 1
            break
if cnt == 0:
    print("loop:no")
else:
    print("loop:yes")
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

#44898484提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
def dfs(node):
    visited[node] = True
    for element in graph[node]:
        if visited[element] is False:
            dfs(element)
def loop(node,prev):
    visit[node] = True
    for element in graph[node]:
        if element == prev:
            continue
        if visit[element] is True:
            return True
        else:
            if loop(element,node):
```

基本信息

#:	44898484
题目:	M27635
提交人:	2200015507-王一粟
内存:	3716kB
时间:	28ms
语言:	Python3
提交时间:	2024-05-08 15:59:30

27947: 动态中位数

http://cs101.openjudge.cn/practice/27947/

思路：我的核心想法是依据hint构建了左右两个二叉堆。每次弹入两个数据后，将其与中位数做比较，再决定如何出堆入堆

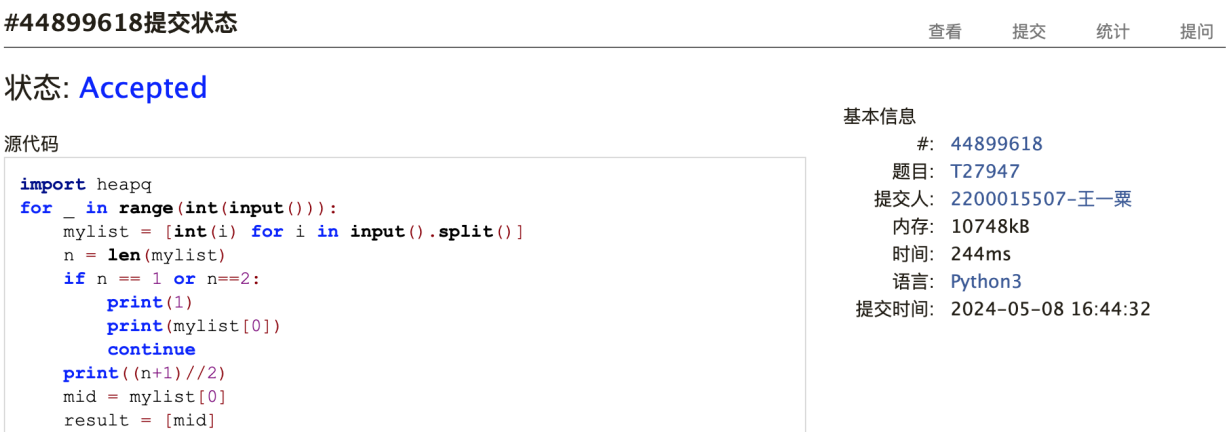
耗时：45min

代码

```
import heapq
for _ in range(int(input())):
    mylist = [int(i) for i in input().split()]
    n = len(mylist)
    if n == 1 or n==2:
        print(1)
        print(mylist[0])
        continue
    print((n+1)//2)
    mid = mylist[0]
    result = [mid]
    left = []
    right = []
    num1, num2 = mylist[1], mylist[2]
    if num2 < num1:
        num1, num2 = num2, num1
    if num1 <= mid <= num2:
        result.append(mid)
        left.append(-num1)
        right.append(num2)
    elif num2 < mid:
        result.append(num2)
        left.append(-num1)
        right.append(mid)
        mid = num2
    else:
        result.append(num1)
        left.append(-mid)
        right.append(num2)
        mid = num1
    for i in range(3,n-1,2):
        num1,num2 = mylist[i],mylist[i+1]
        if num2<num1:
            num1,num2 = num2,num1
        if num1<=mid<=num2:
            heapq.heappush(left,-num1)
            heapq.heappush(right,num2)
            result.append(mid)
        elif num2 < mid:
            heapq.heappush(right,mid)
            heapq.heappush(left,-num1)
            wait_for_mid = -heapq.heappop(left)
            if num2<wait_for_mid:
```

```
                mid = wait_for_mid
            else:
                mid = num2
                num2 = wait_for_mid
            heapq.heappush(left,-num2)
            result.append(mid)
        else:
            heapq.heappush(left,-mid)
            heapq.heappush(right,num2)
            wait_for_mid = heapq.heappop(right)
            if wait_for_mid < num1:
                mid = wait_for_mid
            else:
                mid = num1
                num1 = wait_for_mid
            heapq.heappush(right,num1)
            result.append(mid)
    print(" ".join(str(i) for i in result))
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==



#44899618提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
import heapq
for _ in range(int(input())):
    mylist = [int(i) for i in input().split()]
    n = len(mylist)
    if n == 1 or n==2:
        print(1)
        print(mylist[0])
        continue
    print((n+1)//2)
    mid = mylist[0]
    result = [mid]
```

基本信息

#:	44899618
题目:	T27947
提交人:	2200015507-王一粟
内存:	10748kB
时间:	244ms
语言:	Python3
提交时间:	2024-05-08 16:44:32

28190: 奶牛排队

http://cs101.openjudge.cn/practice/28190/

思路：维护一个栈。当碰到不大于第一个栈中数的情况，就全部弹出。特别要注意的是对于可能出现的后续序列中的最大值的处理

耗时：40min

代码

```
#2200015507 王一粟
n = int(input())
mylist = []
max_num = 0
for i in range(n):
    t = int(input())
```

```
    mylist.append(t)
max_list = sorted(mylist)
total_max = max_list[-1]
stack = [mylist[0]]
if mylist[0] == total_max:
    max_list.pop()
    total_max = max_list[-1]
result = 0
max_num = stack[0]
cnt = 1
for _ in range(1,n):
    num = mylist[_]
    if num == total_max:
        result = max(result,cnt+1)
        stack = [num]
        max_list.pop()
        total_max = max_list[-1]
        max_num = num
        continue
    if num <= stack[0]:
        stack = [num]
        max_num = num
        cnt = 1
        continue
    stack.append(num)
    cnt += 1
    if num > max_num:
        max_num = num
        result = max(result,cnt)
if result != 1:
    print(result)
else:
    print(0)
```

代码运行截图 ==（AC代码截图，至少包含有"Accepted"）==

#44900670提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
n = int(input())
mylist = []
max_num = 0
for i in range(n):
    t = int(input())
    mylist.append(t)
max_list = sorted(mylist)
total_max = max_list[-1]
stack = [mylist[0]]
if mylist[0] == total_max:
    max_list.pop()
    total_max = max_list[-1]
result = 0
max_num = stack[0]
cnt = 1
for _ in range(1,n):
    num = mylist[_]
```

基本信息

#:	44900670
题目:	28190
提交人:	2200015507-王一粟
内存:	4080kB
时间:	37ms
语言:	Python3
提交时间:	2024-05-08 17:23:16

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。==

三次月考都卡在AC5...很难绷

第五第六题做的时间比较久，都大概40分钟。第六题是因为自己以前还没碰到过类似于这种单调栈的题目；第五题其实比较早想到了方法（老师的hint），但是一直处于debug的状态...

第三题也做了30min，其实比较经典的算法，应该是需要节省出来一些时间的

第五题这种老师后面有给关于heap的提示，不知道正式机考的时候会不会有这种hint，没有的话感觉其实很难想到这一层...