Assignment #F: All-Killed 满分

Updated 1844 GMT+8 May 20, 2024

2024 spring, Complied by 王申睿——物理学院

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 typorahttps://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

22485: 升空的焰火，从侧面看

http://cs101.openjudge.cn/practice/22485/

思路:

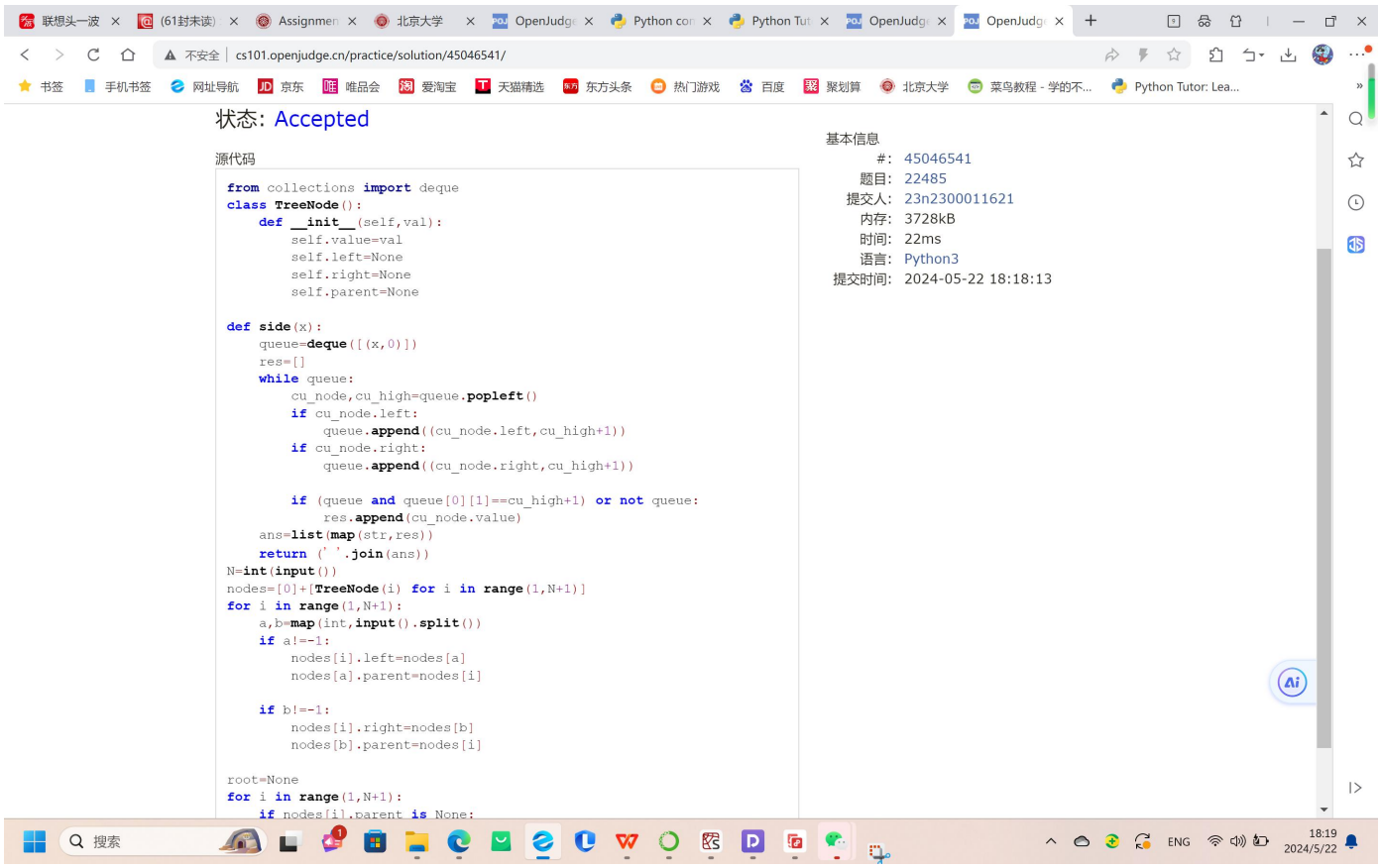
代码

```
from collections import deque

class TreeNode():
```

```
    def __init__(self,val):
        self.value=val
        self.left=None
        self.right=None
        self.parent=None


def side(x):
    queue=deque([(x,0)])
    res=[]
    while queue:
        cu_node,cu_high=queue.popleft()
        if cu_node.left:
            queue.append((cu_node.left,cu_high+1))
        if cu_node.right:
```

```
            queue.append((cu_node.right,cu_high+1))



        if (queue and queue[0][1]==cu_high+1) or not queue:

            res.append(cu_node.value)

    ans=list(map(str,res))

    return (' '.join(ans))

N=int(input())

nodes=[0]+[TreeNode(i) for i in range(1,N+1)]

for i in range(1,N+1):

    a,b=map(int,input().split())

    if a!=-1:

        nodes[i].left=nodes[a]

        nodes[a].parent=nodes[i]
```

```
    if b!=-1:
        nodes[i].right=nodes[b]
        nodes[b].parent=nodes[i]


root=None
for i in range(1,N+1):
    if nodes[i].parent is None:
        root=nodes[i]
print(side(root))
```


状态：Accepted
源代码
from collections import deque
class TreeNode():
def __init__(self,val):
self.value=val
self.left=None
self.right=None
self.parent=None
def side(x):
queue=deque([(x,0)])
res=[]
while queue:
cu_node,cu_high=queue.popleft()
if cu_node.left:
queue.append((cu_node.left,cu_high+1))
if cu_node.right:
queue.append((cu_node.right,cu_high+1))
if (queue and queue[0][1]==cu_high+1) or not queue:
res.append(cu_node.value)
ans=list(map(str,res))
return ' '.join(ans)
N=int(input())
nodes=[0]+[TreeNode(i) for i in range(1,N+1)]
for i in range(1,N+1):
a,b=map(int,input().split())
if a!=-1:
nodes[i].left=nodes[a]
nodes[a].parent=nodes[i]
if b!=-1:
nodes[i].right=nodes[b]
nodes[b].parent=nodes[i]
root=None
for i in range(1,N+1):
if nodes[i].parent is None:
基本信息
#: 45046541
题目: 22485
提交人: 23n2300011621
内存: 3728kB
时间: 22ms
语言: Python3
提交时间: 2024-05-22 18:18:13

28203: 【模板】单调栈

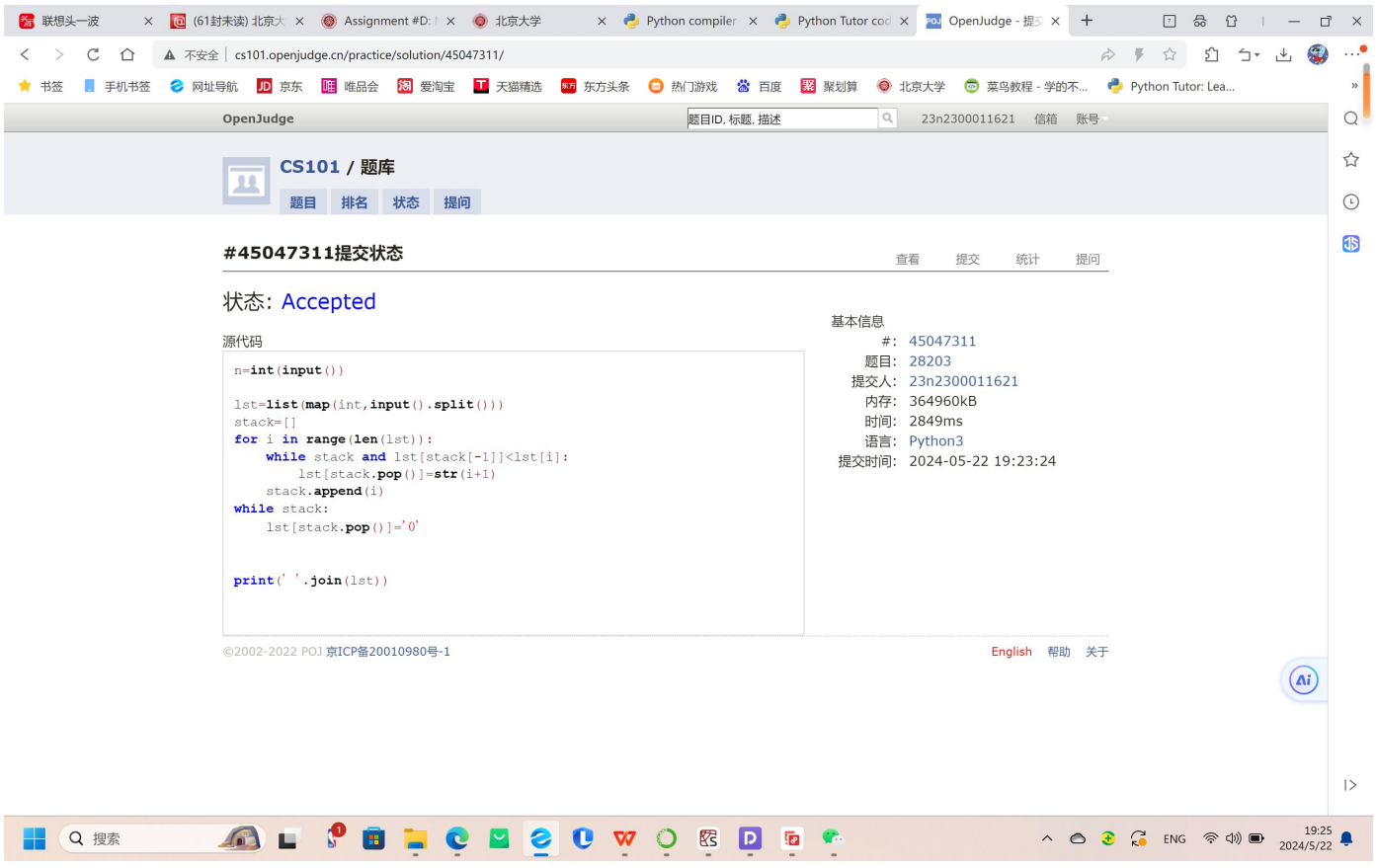
http://cs101.openjudge.cn/practice/28203/

思路:

代码

```
n=int(input())



lst=list(map(int,input().split()))

stack=[]

for i in range(len(lst)):

    while stack and lst[stack[-1]]<lst[i]:

        lst[stack.pop()]=str(i+1)

    stack.append(i)

while stack:

    lst[stack.pop()]='0'
```

print(' '.join(lst))



09202: 舰队、海域出击!

http://cs101.openjudge.cn/practice/09202/

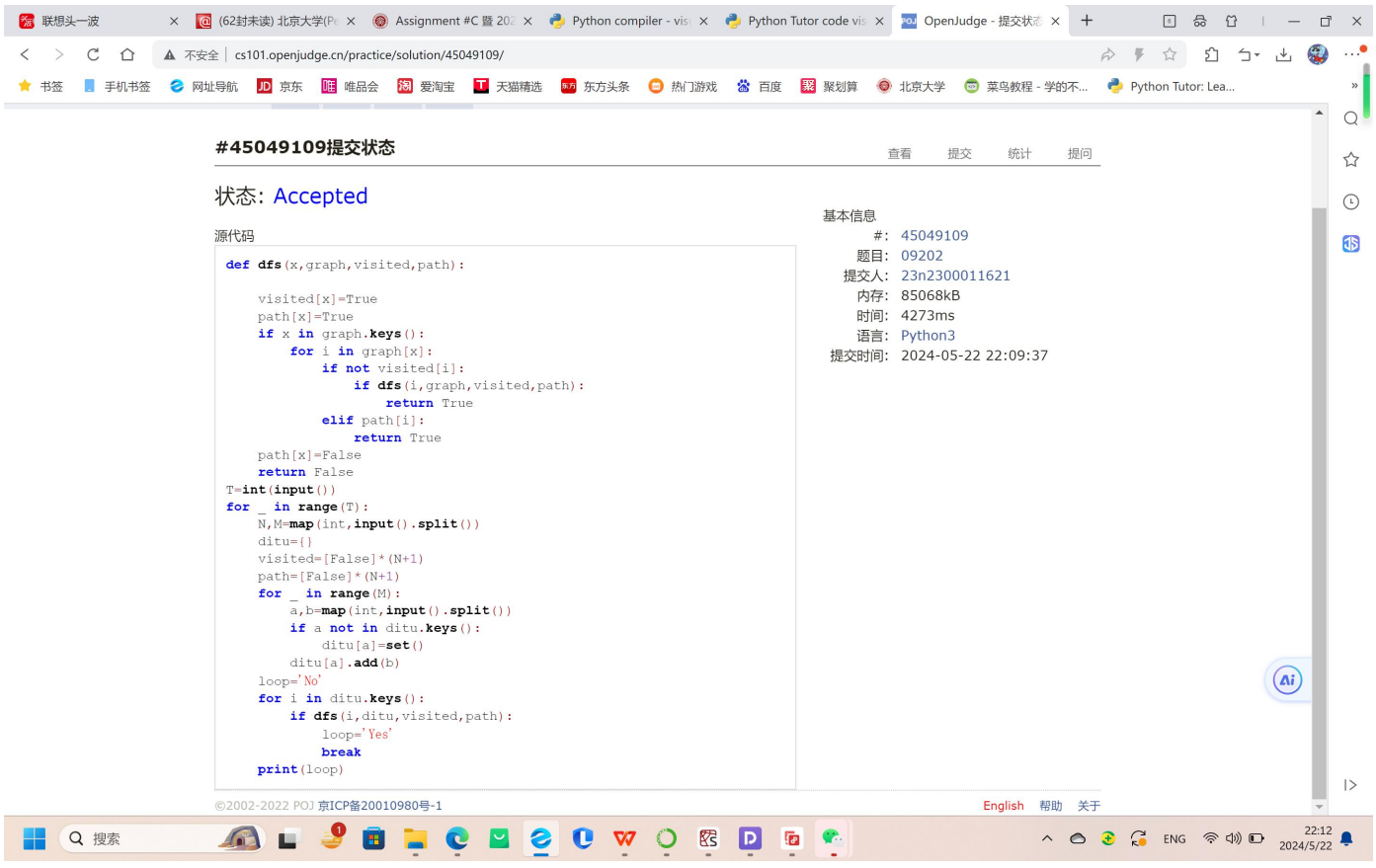
思路:

代码

```
def dfs(x,graph,visited,path):

    visited[x]=True
    path[x]=True
    if x in graph.keys():
        for i in graph[x]:
            if not visited[i]:
                if dfs(i,graph,visited,path):
                    return True
            elif path[i]:
                return True
```

```
    path[x]=False
    return False
T=int(input())
for _ in range(T):
    N,M=map(int,input().split())
    ditu={}
    visited=[False]*(N+1)
    path=[False]*(N+1)
    for _ in range(M):
        a,b=map(int,input().split())
        if a not in ditu.keys():
            ditu[a]=set()
        ditu[a].add(b)
    loop='No'
```

```
    for i in ditu.keys():
        if dfs(i,ditu,visited,path):
            loop='Yes'
            break
    print(loop)
```



04135: 月度开销

http://cs101.openjudge.cn/practice/04135/

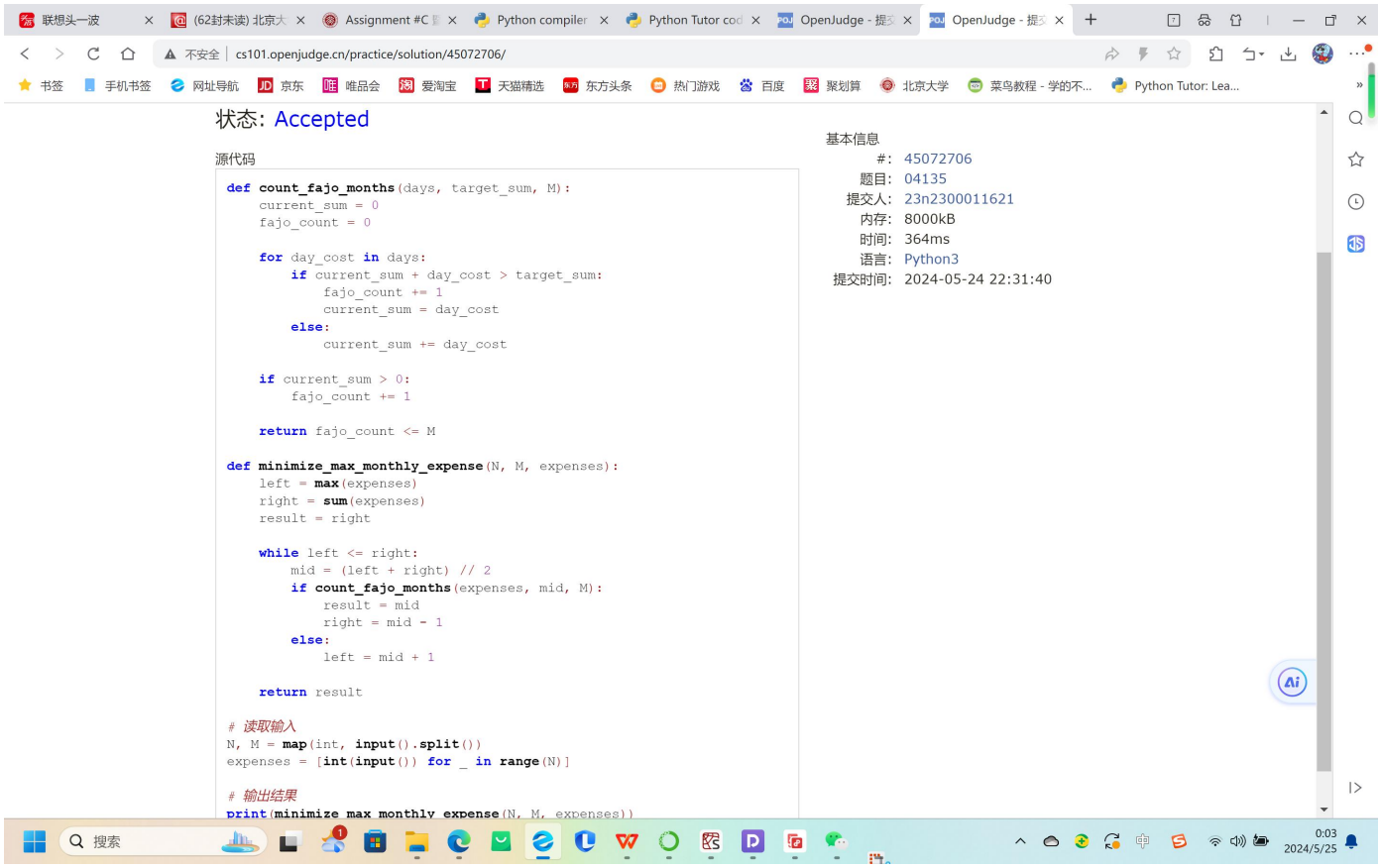
思路：我一开始以为是对每一天二分，实际上是在最大值和开销之和之间二分搜索。因为每一天不能跳着过，所以可以确定的是每一次都可以从头开始迭代。如果中间值不能满足划分月份的具体要求，就向上搜索，反之向下。上界为N-M-1，下界为max（每日开销）。

代码

```
def count_fajo_months(days, target_sum, M):

    current_sum = 0

    fajo_count = 0



    for day_cost in days:

        if current_sum + day_cost > target_sum:

            fajo_count += 1

            current_sum = day_cost

        else:

            current_sum += day_cost



    if current_sum > 0:
```

```
        fajo_count += 1

    return fajo_count <= M

def minimize_max_monthly_expense(N, M, expenses):
    left = max(expenses)
    right = sum(expenses)
    result = right

    while left <= right:
        mid = (left + right) // 2
        if count_fajo_months(expenses, mid, M):
            result = mid
            right = mid - 1
```

```
        else:
            left = mid + 1

    return result

# 读取输入
N, M = map(int, input().split())
expenses = [int(input()) for _ in range(N)]

# 输出结果
print(minimize_max_monthly_expense(N, M, expenses))
```



07735: 道路

http://cs101.openjudge.cn/practice/07735/

思路：已参考题解，正常Dijkstra用已访问集合防止走回头路，这里用总花费剪枝和终止（也对，不停地走回头路总会花光路费的）。

代码

```
#

```

代码运行截图 （AC代码截图，至少包含有"Accepted"）

01182: 食物链

http://cs101.openjudge.cn/practice/01182/

思路:

代码

```
#

```

。

代码运行截图 （AC代码截图，至少包含有"Accepted"）

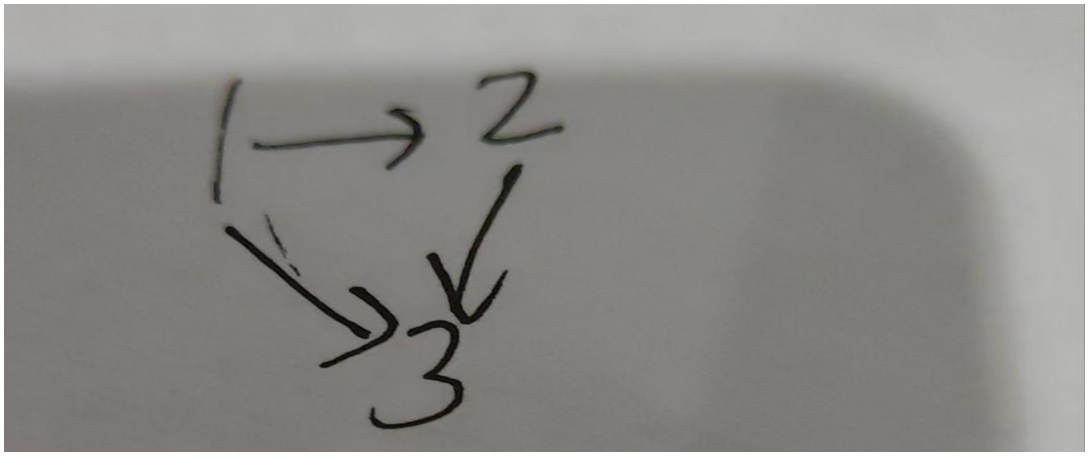
2. 学习总结和收获

关于拓扑排序的必要性，可以举出一个简单的例子：
输入：

```
1
3 3
1 2
2 3
1 3
```

输出：

```
Yes
```

但这显然不对， 

。

原因是未能保存路径信息，沿不同路径到达相同节点时条件为真。因此，我们需要在搜索过程中保存路径，这样才能保证沿相同路径到达，即成环。实际上，这就是忽略了DFS最基本的原则：回溯！

图的算法还有困难，接下来的一周会重点复习。