Assignment #F: All-Killed 满分

Updated 1844 GMT+8 May 20, 2024

2024 spring, Complied by 周添 物理学院

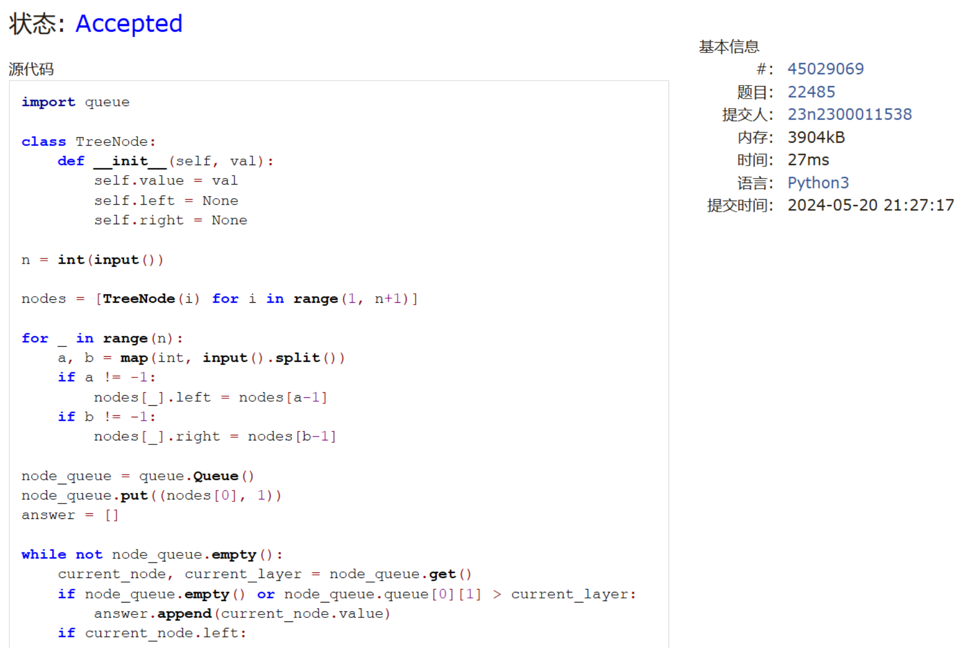
1. 题目

22485: 升空的焰火，从侧面看

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

代码

```python
import queue

class TreeNode:
    def __init__(self, val):
        self.value = val
        self.left = None
        self.right = None

n = int(input())

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

for _ in range(n):
    a, b = map(int, input().split())
    if a != -1:
        nodes[_].left = nodes[a-1]
    if b != -1:
        nodes[_].right = nodes[b-1]

node_queue = queue.Queue()
node_queue.put((nodes[0], 1))
answer = []

while not node_queue.empty():
    current_node, current_layer = node_queue.get()
    if node_queue.empty() or node_queue.queue[0][1] > current_layer:
        answer.append(current_node.value)
    if current_node.left:
        node_queue.put((current_node.left, current_layer+1))
    if current_node.right:
        node_queue.put((current_node.right, current_layer+1))

for item in answer:
    print(item, end=' ')
```



状态: Accepted

源代码

```
import queue

class TreeNode:
    def __init__(self, val):
        self.value = val
        self.left = None
        self.right = None

n = int(input())

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

for _ in range(n):
    a, b = map(int, input().split())
    if a != -1:
        nodes[_].left = nodes[a-1]
    if b != -1:
        nodes[_].right = nodes[b-1]

node_queue = queue.Queue()
node_queue.put((nodes[0], 1))
answer = []

while not node_queue.empty():
    current_node, current_layer = node_queue.get()
    if node_queue.empty() or node_queue.queue[0][1] > current_layer:
        answer.append(current_node.value)
    if current_node.left:
```

基本信息

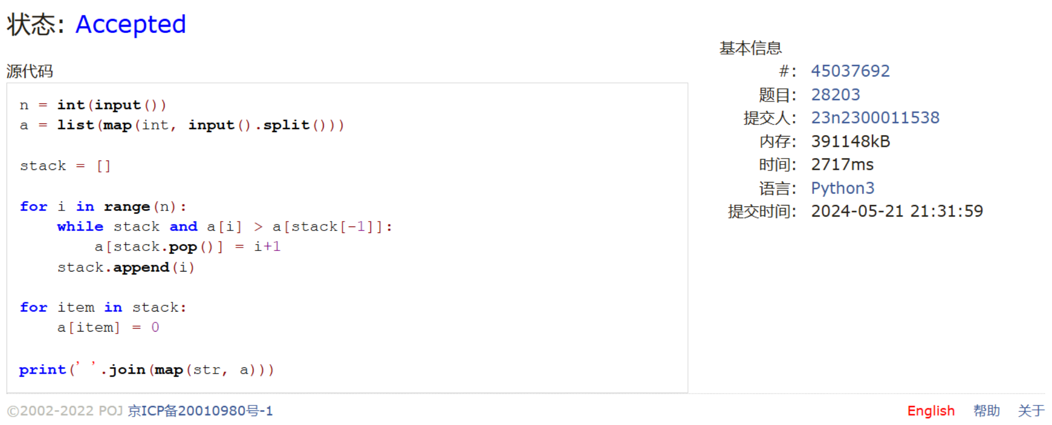
#:	45029069
题目:	22485
提交人:	23n2300011538
内存:	3904kB
时间:	27ms
语言:	Python3
提交时间:	2024-05-20 21:27:17

28203:【模板】单调栈

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

代码

```python
n = int(input())
a = list(map(int, input().split()))

stack = []

for i in range(n):
    while stack and a[i] > a[stack[-1]]:
        a[stack.pop()] = i+1
    stack.append(i)

for item in stack:
    a[item] = 0

print(' '.join(map(str, a)))
```



状态: Accepted

源代码

```
n = int(input())
a = list(map(int, input().split()))

stack = []

for i in range(n):
    while stack and a[i] > a[stack[-1]]:
        a[stack.pop()] = i+1
    stack.append(i)

for item in stack:
    a[item] = 0

print(' '.join(map(str, a)))
```

基本信息

#:	45037692
题目:	28203
提交人:	23n2300011538
内存:	391148kB
时间:	2717ms
语言:	Python3
提交时间:	2024-05-21 21:31:59

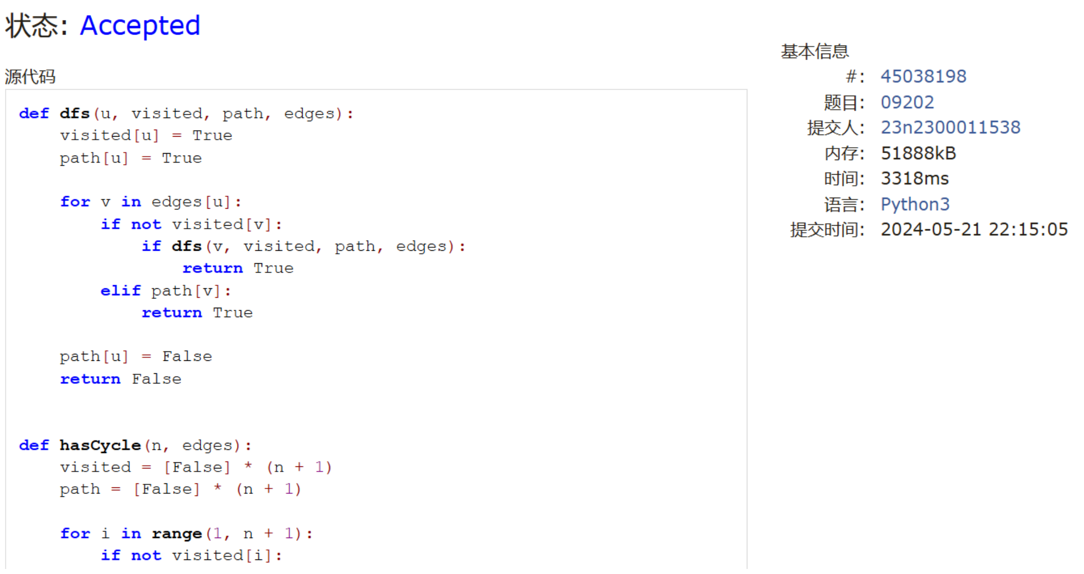
©2002-2022 POJ 京ICP备20010980号-1　English 帮助 关于

09202: 舰队、海域出击！

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

代码

```
def dfs(u, visited, path, edges):
    visited[u] = True
    path[u] = True

    for v in edges[u]:
        if not visited[v]:
            if dfs(v, visited, path, edges):
                return True
        elif path[v]:
            return True

    path[u] = False
    return False


def hasCycle(n, edges):
    visited = [False] * (n + 1)
    path = [False] * (n + 1)

    for i in range(1, n + 1):
        if not visited[i]:
            if dfs(i, visited, path, edges):
                return True

    return False


a = int(input())
for _ in range(a):
    n, m = map(int, input().split())
    edges = [[] for _ in range(n + 1)]
    for _ in range(m):
        u, v = map(int, input().split())
        edges[u].append(v)

    if hasCycle(n, edges):
        print('Yes')
    else:
        print('No')
```



状态: Accepted

源代码

```
def dfs(u, visited, path, edges):
    visited[u] = True
    path[u] = True

    for v in edges[u]:
        if not visited[v]:
            if dfs(v, visited, path, edges):
                return True
        elif path[v]:
            return True

    path[u] = False
    return False


def hasCycle(n, edges):
    visited = [False] * (n + 1)
    path = [False] * (n + 1)

    for i in range(1, n + 1):
        if not visited[i]:
```

基本信息

#:	45038198
题目:	09202
提交人:	23n2300011538
内存:	51888kB
时间:	3318ms
语言:	Python3
提交时间:	2024-05-21 22:15:05

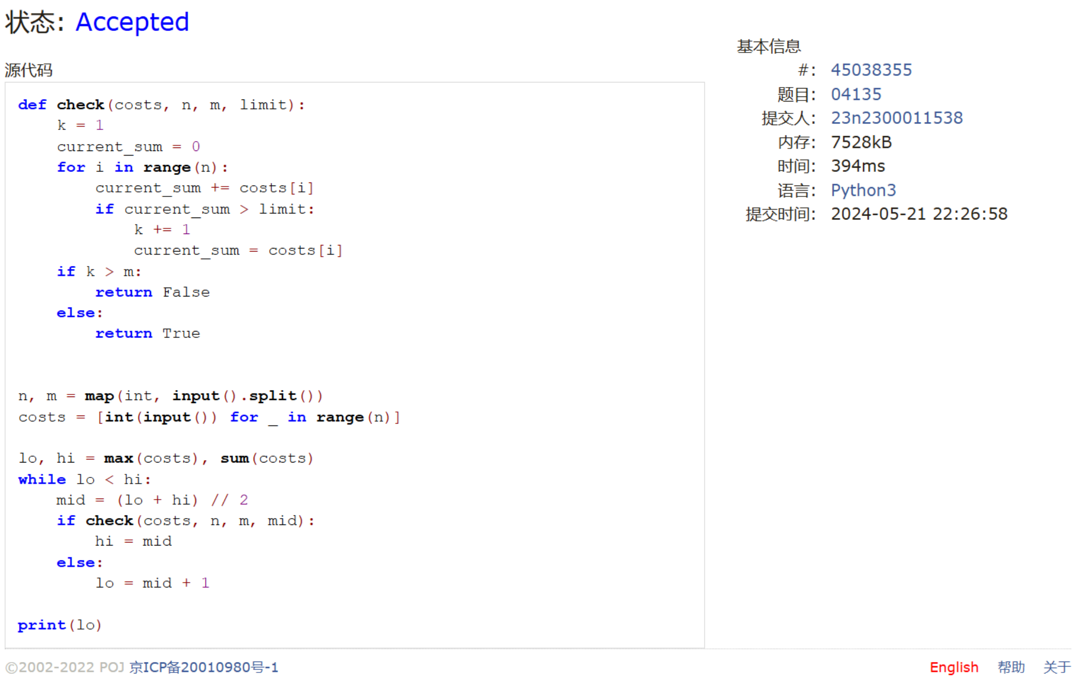
04135: 月度开销

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

代码

```
def check(costs, n, m, limit):
    k = 1
    current_sum = 0
    for i in range(n):
        current_sum += costs[i]
        if current_sum > limit:
            k += 1
            current_sum = costs[i]
    if k > m:
        return False
    else:
        return True


n, m = map(int, input().split())
costs = [int(input()) for _ in range(n)]
```

```
lo, hi = max(costs), sum(costs)
while lo < hi:
    mid = (lo + hi) // 2
    if check(costs, n, m, mid):
        hi = mid
    else:
        lo = mid + 1

print(lo)
```



状态: Accepted

源代码

```
def check(costs, n, m, limit):
    k = 1
    current_sum = 0
    for i in range(n):
        current_sum += costs[i]
        if current_sum > limit:
            k += 1
            current_sum = costs[i]
    if k > m:
        return False
    else:
        return True


n, m = map(int, input().split())
costs = [int(input()) for _ in range(n)]

lo, hi = max(costs), sum(costs)
while lo < hi:
    mid = (lo + hi) // 2
    if check(costs, n, m, mid):
        hi = mid
    else:
        lo = mid + 1

print(lo)
```

基本信息

#:	45038355
题目:	04135
提交人:	23n2300011538
内存:	7528kB
时间:	394ms
语言:	Python3
提交时间:	2024-05-21 22:26:58

©2002-2022 POJ 京ICP备20010980号-1 English 帮助 关于

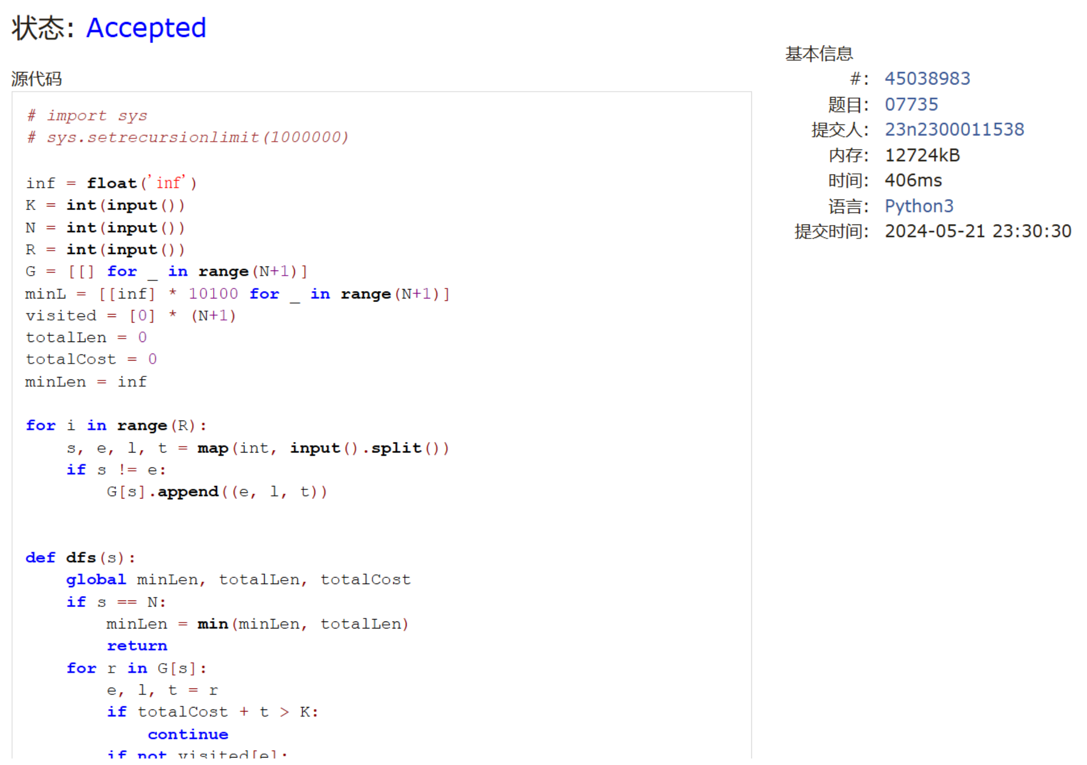
07735: 道路

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

代码

```
# import sys
# sys.setrecursionlimit(1000000)

inf = float('inf')
K = int(input())
N = int(input())
R = int(input())
G = [[] for _ in range(N+1)]
minL = [[inf] * 10100 for _ in range(N+1)]
visited = [0] * (N+1)
totalLen = 0
totalCost = 0
minLen = inf

for i in range(R):
```

```python
    s, e, l, t = map(int, input().split())
    if s != e:
        G[s].append((e, l, t))


def dfs(s):
    global minLen, totalLen, totalCost
    if s == N:
        minLen = min(minLen, totalLen)
        return
    for r in G[s]:
        e, l, t = r
        if totalCost + t > K:
            continue
        if not visited[e]:
            if totalLen + l >= minLen:
                continue
            if totalLen + l >= minL[e][totalCost+t]:
                continue
            minL[e][totalCost+t] = totalLen + l
            totalLen += l
            totalCost += t
            visited[e] = 1
            dfs(e)
            visited[e] = 0
            totalLen -= l
            totalCost -= t


visited[1] = 1
dfs(1)

if minLen < inf:
    print(minLen)
else:
    print(-1)
```


状态: Accepted

源代码

```
# import sys
# sys.setrecursionlimit(1000000)

inf = float('inf')
K = int(input())
N = int(input())
R = int(input())
G = [[] for _ in range(N+1)]
minL = [[inf] * 10100 for _ in range(N+1)]
visited = [0] * (N+1)
totalLen = 0
totalCost = 0
minLen = inf

for i in range(R):
    s, e, l, t = map(int, input().split())
    if s != e:
        G[s].append((e, l, t))


def dfs(s):
    global minLen, totalLen, totalCost
    if s == N:
        minLen = min(minLen, totalLen)
        return
    for r in G[s]:
        e, l, t = r
        if totalCost + t > K:
            continue
        if not visited[e]:
```

基本信息

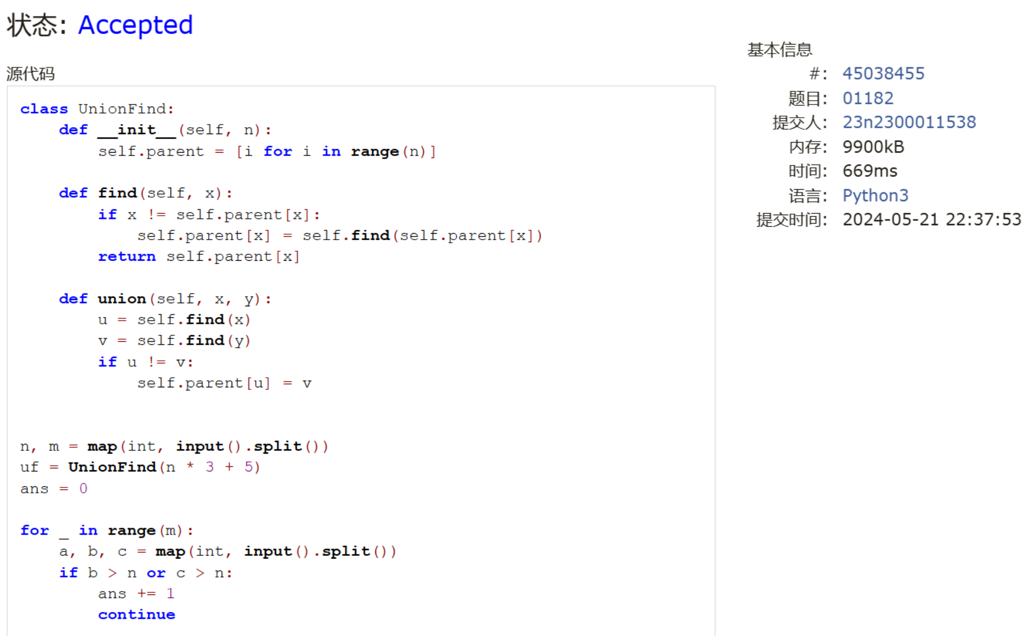
#:	45038983
题目:	07735
提交人:	23n2300011538
内存:	12724kB
时间:	406ms
语言:	Python3
提交时间:	2024-05-21 23:30:30

01182: 食物链

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

代码

```python
class UnionFind:
    def __init__(self, n):
        self.parent = [i for i in range(n)]

    def find(self, x):
        if x != self.parent[x]:
            self.parent[x] = self.find(self.parent[x])
        return self.parent[x]

    def union(self, x, y):
        u = self.find(x)
        v = self.find(y)
        if u != v:
            self.parent[u] = v


n, m = map(int, input().split())
uf = UnionFind(n * 3 + 5)
ans = 0

for _ in range(m):
    a, b, c = map(int, input().split())
    if b > n or c > n:
        ans += 1
        continue
    if a == 1:
        if uf.find(b + n) == uf.find(c) or uf.find(b) == uf.find(c + n):
            ans += 1
            continue
        uf.union(b, c)
        uf.union(b + n, c + n)
        uf.union(b + 2 * n, c + 2 * n)
    else:
        if b == c or uf.find(b) == uf.find(c) or uf.find(b + 2 * n) == 
uf.find(c):
            ans += 1
            continue
        uf.union(b + n, c)
        uf.union(c + 2 * n, b)
        uf.union(b + 2 * n, c + n)

print(ans)
```



状态: Accepted

源代码

```
class UnionFind:
    def __init__(self, n):
        self.parent = [i for i in range(n)]

    def find(self, x):
        if x != self.parent[x]:
            self.parent[x] = self.find(self.parent[x])
        return self.parent[x]

    def union(self, x, y):
        u = self.find(x)
        v = self.find(y)
        if u != v:
            self.parent[u] = v


n, m = map(int, input().split())
uf = UnionFind(n * 3 + 5)
ans = 0

for _ in range(m):
    a, b, c = map(int, input().split())
    if b > n or c > n:
        ans += 1
        continue
```

基本信息

#:	45038455
题目:	01182
提交人:	23n2300011538
内存:	9900kB
时间:	669ms
语言:	Python3
提交时间:	2024-05-21 22:37:53

2. 学习总结和收获

单调栈卡得真死，用cpp写倒是过了，python就不行，得牺牲一点点时间来优化空间