Assignment #F: All-Killed 满分

Updated 1844 GMT+8 May 20, 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

22485: 升空的焰火，从侧面看

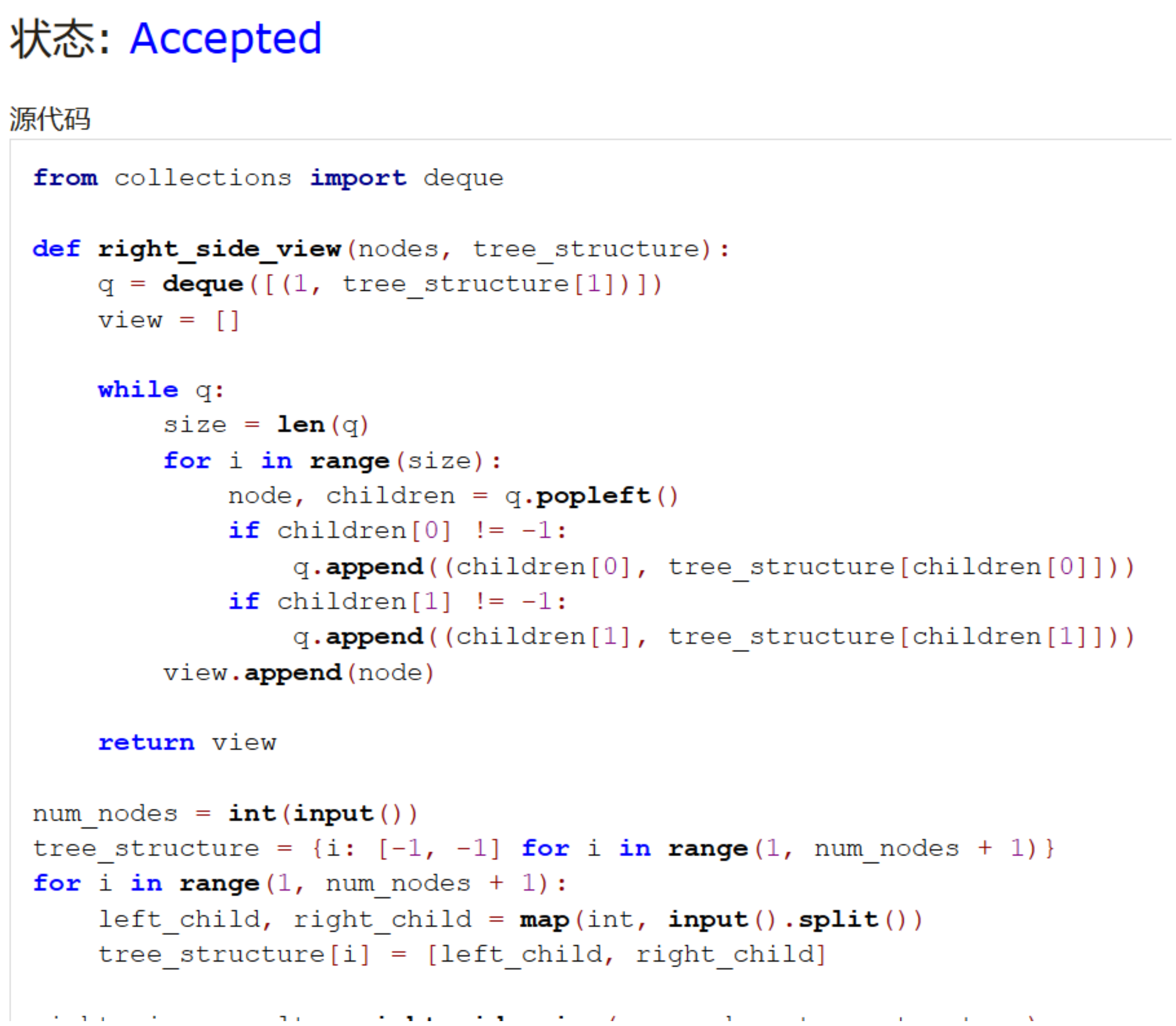
http://cs101.openjudge.cn/practice/22485/

思路：

代码

```python
from collections import deque

def right_side_view(nodes, tree_structure):
    q = deque([(1, tree_structure[1])])
    view = []

    while q:
        size = len(q)
        for i in range(size):
            node, children = q.popleft()
            if children[0] != -1:
                q.append((children[0], tree_structure[children[0]]))
            if children[1] != -1:
                q.append((children[1], tree_structure[children[1]]))
        view.append(node)

    return view

num_nodes = int(input())
tree_structure = {i: [-1, -1] for i in range(1, num_nodes + 1)}
for i in range(1, num_nodes + 1):
    left_child, right_child = map(int, input().split())
    tree_structure[i] = [left_child, right_child]

right_view_result = right_side_view(num_nodes, tree_structure)
print(' '.join(map(str, right_view_result)))
```

代码运行截图



状态: Accepted

源代码

```python
from collections import deque

def right_side_view(nodes, tree_structure):
    q = deque([(1, tree_structure[1])])
    view = []

    while q:
        size = len(q)
        for i in range(size):
            node, children = q.popleft()
            if children[0] != -1:
                q.append((children[0], tree_structure[children[0]]))
            if children[1] != -1:
                q.append((children[1], tree_structure[children[1]]))
        view.append(node)

    return view

num_nodes = int(input())
tree_structure = {i: [-1, -1] for i in range(1, num_nodes + 1)}
for i in range(1, num_nodes + 1):
    left_child, right_child = map(int, input().split())
    tree_structure[i] = [left_child, right_child]
```

28203:【模板】单调栈

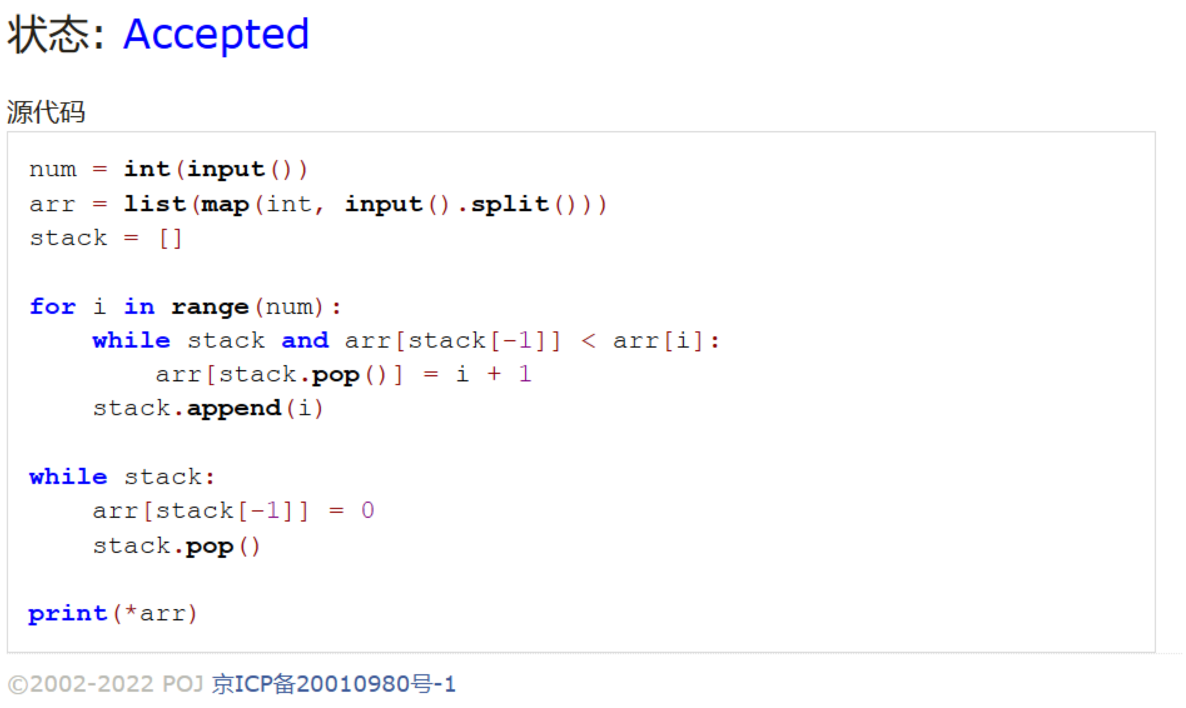
http://cs101.openjudge.cn/practice/28203/

思路：

代码

```python
num = int(input())
arr = list(map(int, input().split()))
stack = []

for i in range(num):
    while stack and arr[stack[-1]] < arr[i]:
        arr[stack.pop()] = i + 1
    stack.append(i)

while stack:
    arr[stack[-1]] = 0
    stack.pop()

print(*arr)
```

代码运行截图



状态: Accepted

源代码

```python
num = int(input())
arr = list(map(int, input().split()))
stack = []

for i in range(num):
    while stack and arr[stack[-1]] < arr[i]:
        arr[stack.pop()] = i + 1
    stack.append(i)

while stack:
    arr[stack[-1]] = 0
    stack.pop()

print(*arr)
```

©2002-2022 POJ 京ICP备20010980号-1

09202: 舰队、海域出击！

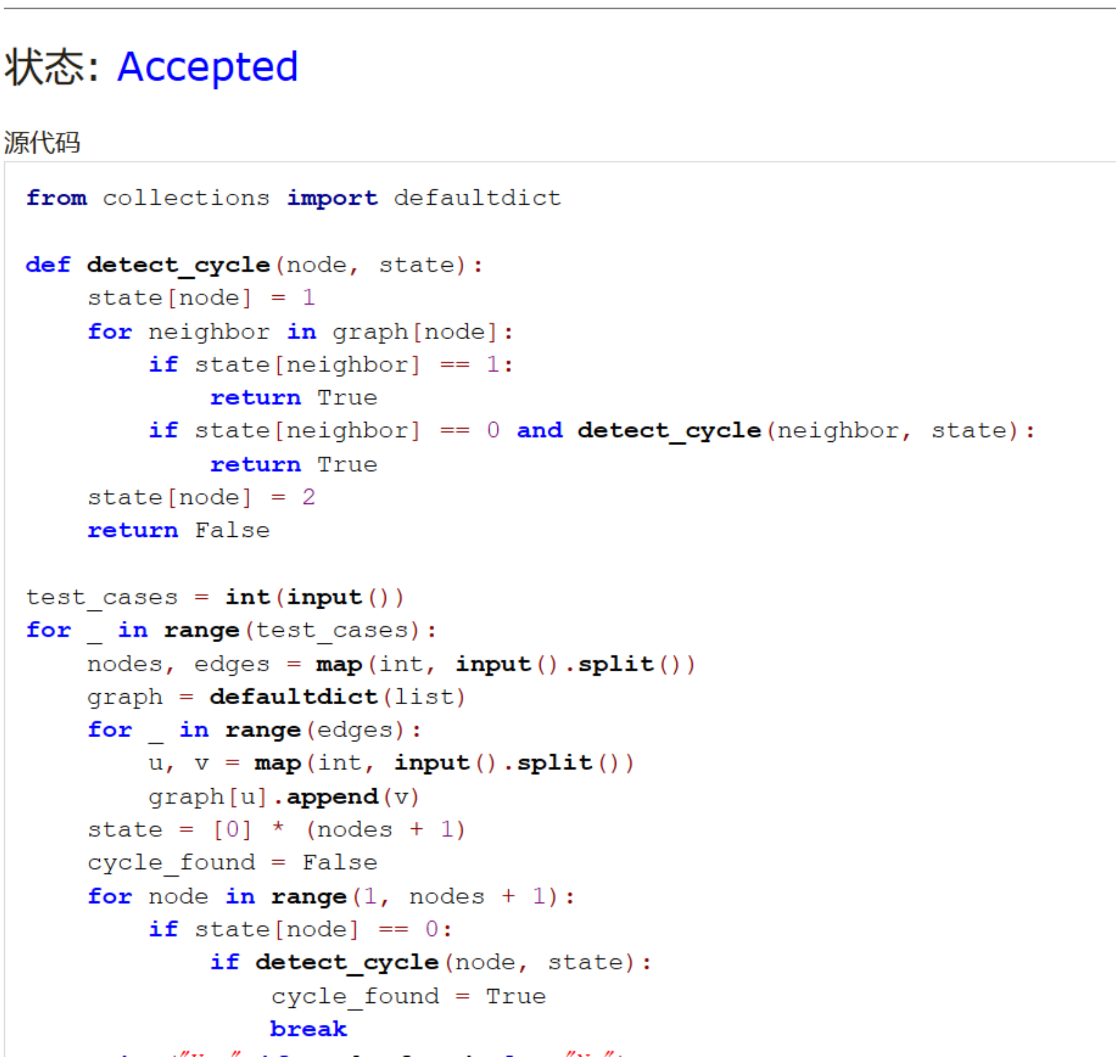
http://cs101.openjudge.cn/practice/09202/

思路：

代码

```python
from collections import defaultdict

def detect_cycle(node, state):
    state[node] = 1
    for neighbor in graph[node]:
        if state[neighbor] == 1:
            return True
        if state[neighbor] == 0 and detect_cycle(neighbor, state):
            return True
    state[node] = 2
    return False

test_cases = int(input())
for _ in range(test_cases):
    nodes, edges = map(int, input().split())
    graph = defaultdict(list)
    for _ in range(edges):
        u, v = map(int, input().split())
        graph[u].append(v)
    state = [0] * (nodes + 1)
    cycle_found = False
    for node in range(1, nodes + 1):
        if state[node] == 0:
            if detect_cycle(node, state):
                cycle_found = True
                break
    print("Yes" if cycle_found else "No")
```

代码运行截图


状态: Accepted

源代码

```python
from collections import defaultdict

def detect_cycle(node, state):
    state[node] = 1
    for neighbor in graph[node]:
        if state[neighbor] == 1:
            return True
        if state[neighbor] == 0 and detect_cycle(neighbor, state):
            return True
    state[node] = 2
    return False

test_cases = int(input())
for _ in range(test_cases):
    nodes, edges = map(int, input().split())
    graph = defaultdict(list)
    for _ in range(edges):
        u, v = map(int, input().split())
        graph[u].append(v)
    state = [0] * (nodes + 1)
    cycle_found = False
    for node in range(1, nodes + 1):
        if state[node] == 0:
            if detect_cycle(node, state):
                cycle_found = True
                break
```

04135: 月度开销

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

思路:

代码

```python
def can_divide(days, max_expenditure, m):
    current_sum = 0
    count = 1
    for expense in days:
        if current_sum + expense > max_expenditure:
            count += 1
            current_sum = expense
            if count > m:
                return False
        else:
            current_sum += expense
    return True
```

```python

def find_min_max_expenditure(n, m, expenditures):
    low = max(expenditures)
    high = sum(expenditures)

    while low < high:
        mid = (low + high) // 2
        if can_divide(expenditures, mid, m):
            high = mid
        else:
            low = mid + 1

    return low
n, m = map(int, input().split())
expenditures = [int(input()) for _ in range(n)]
print(find_min_max_expenditure(n, m, expenditures))
```

代码运行截图 （AC代码截图，至少包含有"Accepted"）

07735: 道路

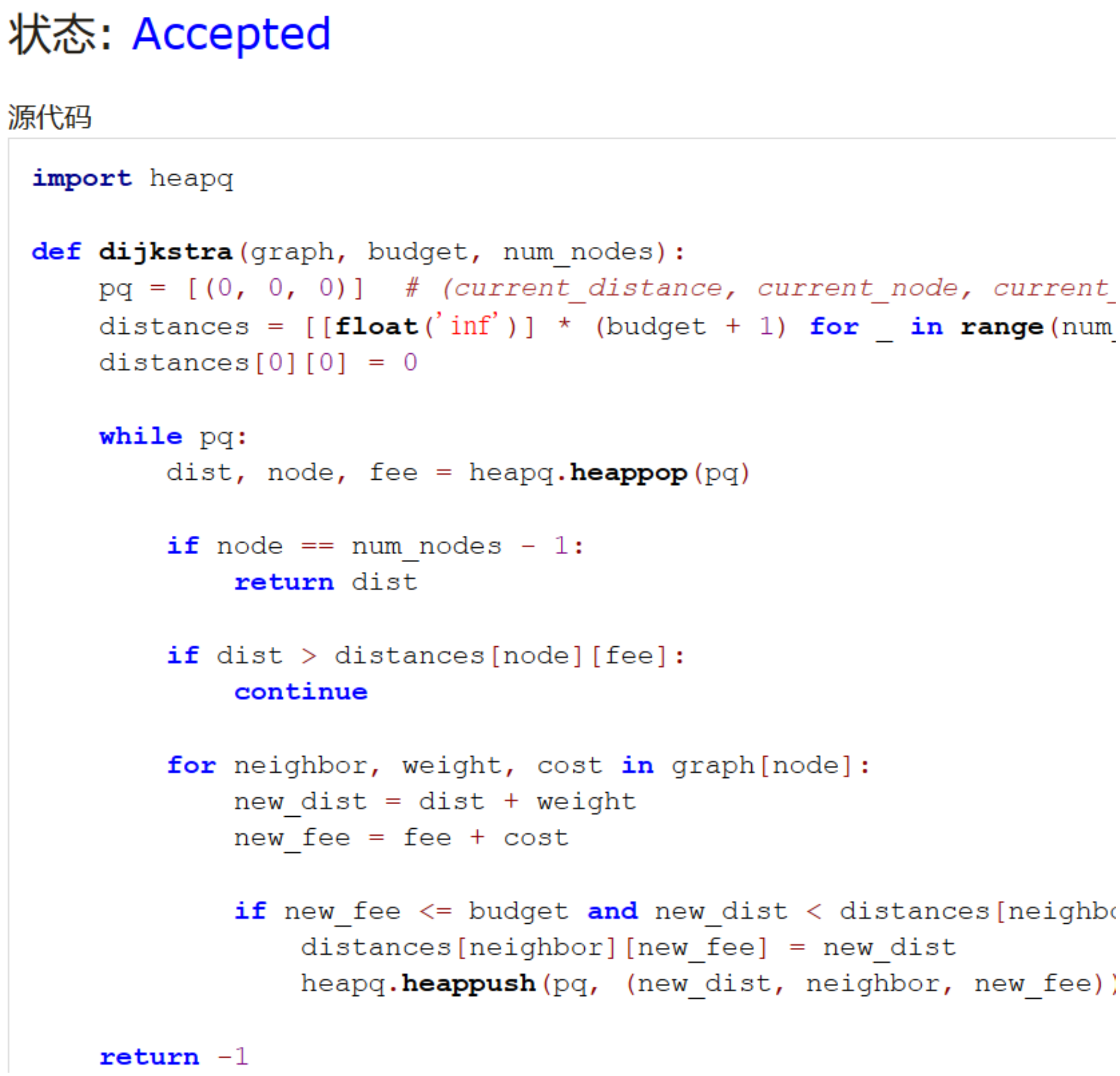
http://cs101.openjudge.cn/practice/07735/

思路：

代码

```python
import heapq

def dijkstra(graph, budget, num_nodes):
    pq = [(0, 0, 0)]  # (current_distance, current_node, current_fee)
    distances = [[float('inf')] * (budget + 1) for _ in range(num_nodes)]
    distances[0][0] = 0

    while pq:
        dist, node, fee = heapq.heappop(pq)

        if node == num_nodes - 1:
            return dist

        if dist > distances[node][fee]:
            continue

        for neighbor, weight, cost in graph[node]:
            new_dist = dist + weight
            new_fee = fee + cost

            if new_fee <= budget and new_dist < distances[neighbor]
[new_fee]:
                distances[neighbor][new_fee] = new_dist
                heapq.heappush(pq, (new_dist, neighbor, new_fee))

    return -1

budget = int(input())
num_nodes = int(input())
num_edges = int(input())
graph = [[] for _ in range(num_nodes)]
for _ in range(num_edges):
    src, dst, length, fee = map(int, input().split())
    graph[src - 1].append((dst - 1, length, fee))


result = dijkstra(graph, budget, num_nodes)
print(result)
```

代码运行截图



状态: Accepted

源代码

```python
import heapq

def dijkstra(graph, budget, num_nodes):
    pq = [(0, 0, 0)]  # (current_distance, current_node, current_
    distances = [[float('inf')] * (budget + 1) for _ in range(num_
    distances[0][0] = 0

    while pq:
        dist, node, fee = heapq.heappop(pq)

        if node == num_nodes - 1:
            return dist

        if dist > distances[node][fee]:
            continue

        for neighbor, weight, cost in graph[node]:
            new_dist = dist + weight
            new_fee = fee + cost

            if new_fee <= budget and new_dist < distances[neighbo
                distances[neighbor][new_fee] = new_dist
                heapq.heappush(pq, (new_dist, neighbor, new_fee)

    return -1
```

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

期末了先漏一道题（（（