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/

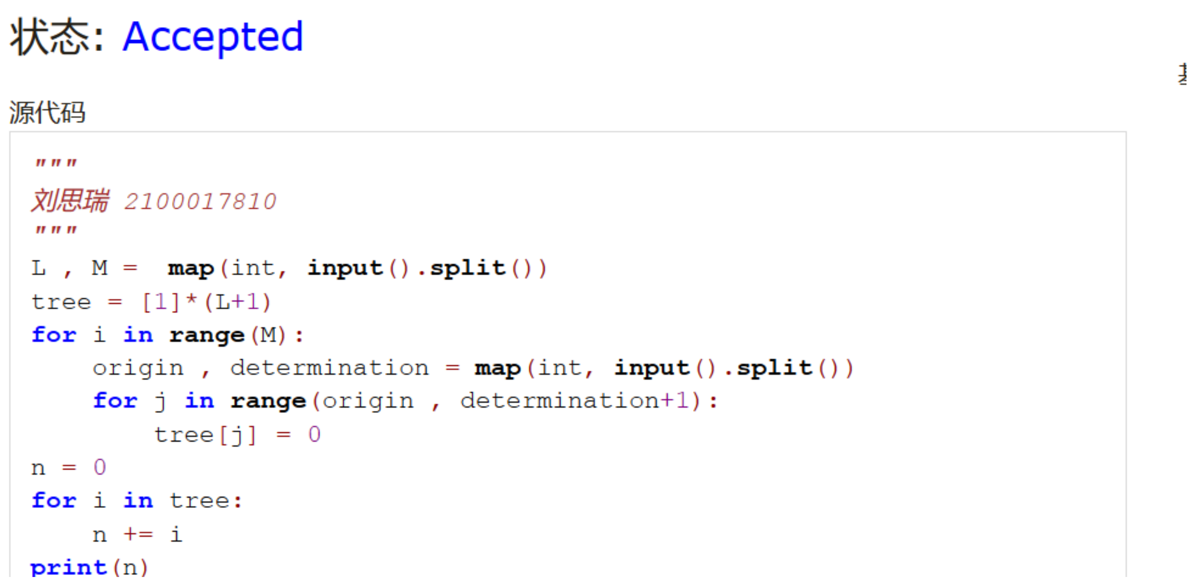
思路：

正常的操作问题

代码

```python
"""
刘思瑞 2100017810
"""
L , M =  map(int, input().split())
tree = [1]*(L+1)
for i in range(M):
    origin , determination = map(int, input().split())
    for j in range(origin , determination+1):
        tree[j] = 0
n = 0
for i in tree:
    n += i
print(n)
```

代码运行截图



状态: Accepted

源代码

```python
"""
刘思瑞 2100017810
"""
L , M =  map(int, input().split())
tree = [1]*(L+1)
for i in range(M):
    origin , determination = map(int, input().split())
    for j in range(origin , determination+1):
        tree[j] = 0
n = 0
for i in tree:
    n += i
print(n)
```

20449: 是否被5整除

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

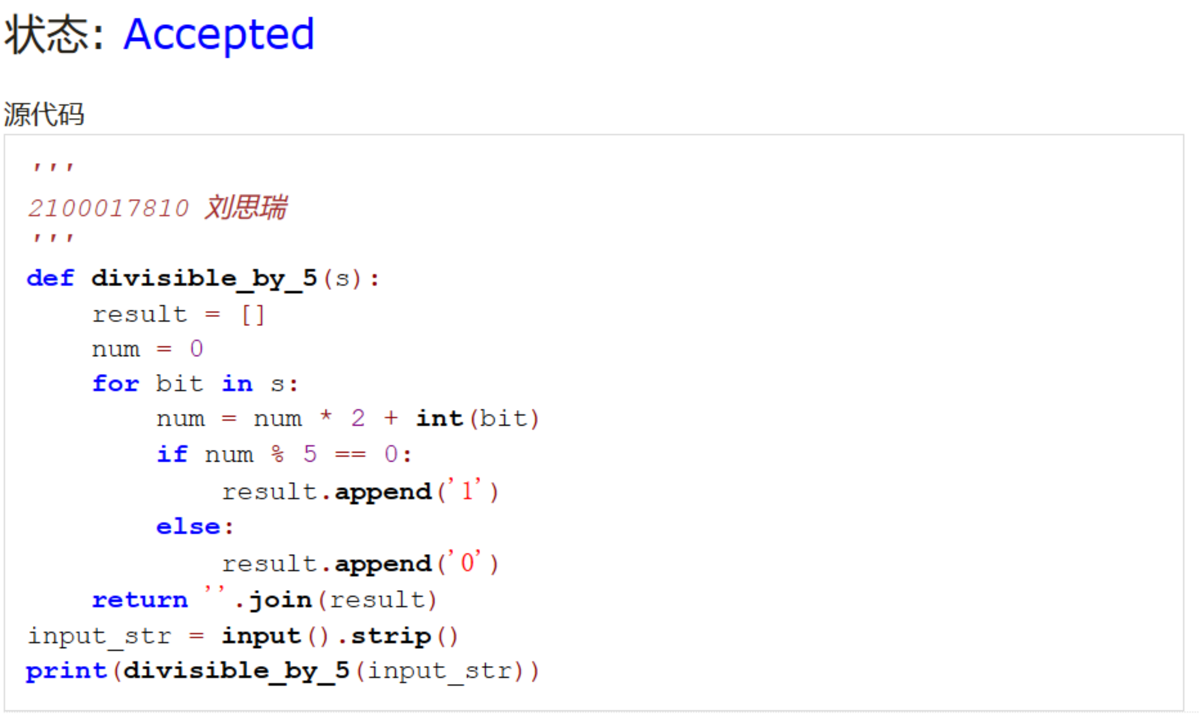
思路：

也是操作问题

代码

```python
'''
2100017810 刘思瑞
'''
def divisible_by_5(s):
    result = []
    num = 0
    for bit in s:
        num = num * 2 + int(bit)
        if num % 5 == 0:
            result.append('1')
        else:
            result.append('0')
    return ''.join(result)
input_str = input().strip()
print(divisible_by_5(input_str))
```

代码运行截图



状态: Accepted

源代码

```python
'''
2100017810 刘思瑞
'''
def divisible_by_5(s):
    result = []
    num = 0
    for bit in s:
        num = num * 2 + int(bit)
        if num % 5 == 0:
            result.append(’1’)
        else:
            result.append(’0’)
    return ’’.join(result)
input_str = input().strip()
print(divisible_by_5(input_str))
```

01258: Agri-Net

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

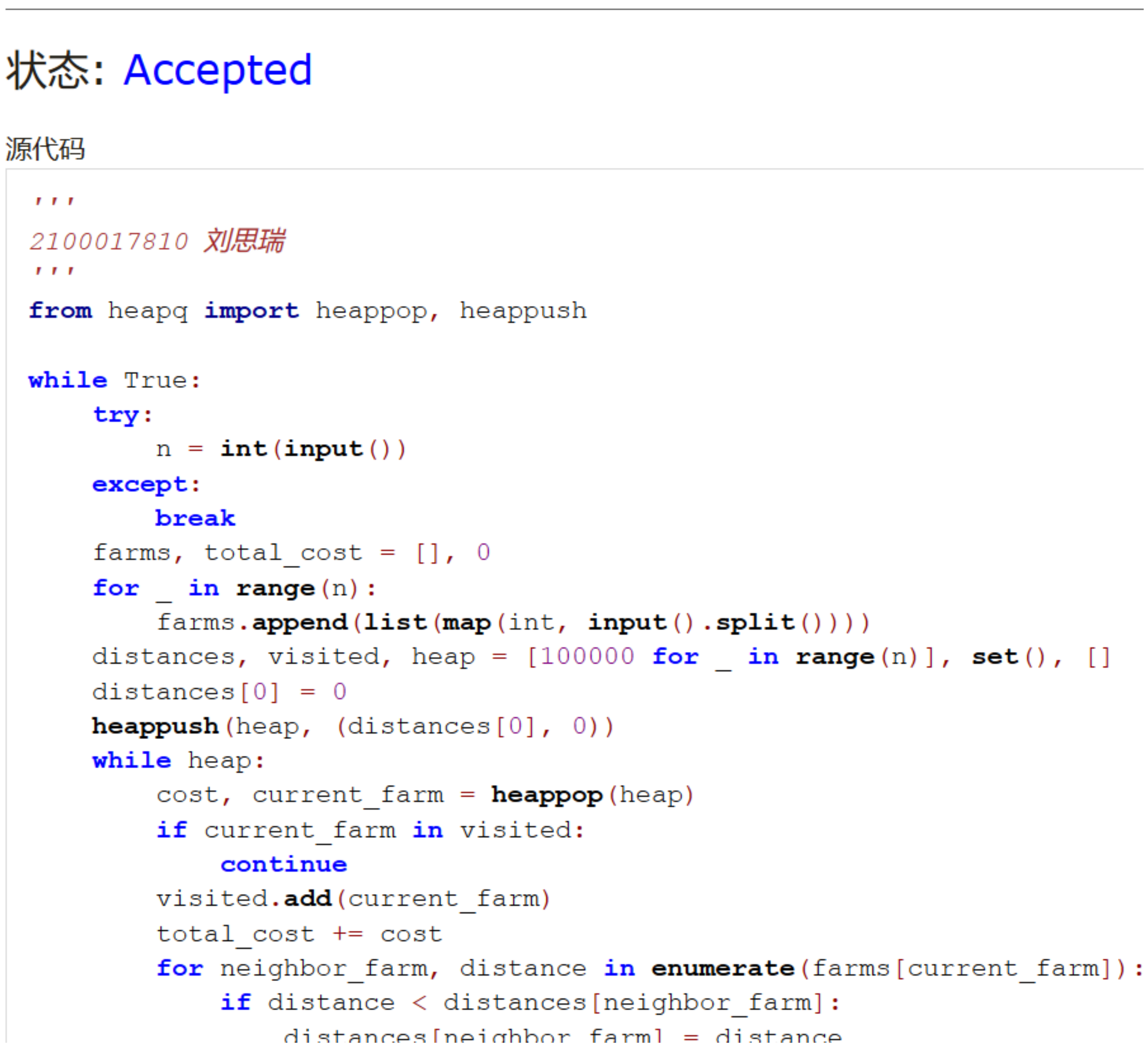
思路:

Dijstrack

代码

```python
'''
2100017810 刘思瑞
'''
from heapq import heappop, heappush

while True:
    try:
        n = int(input())
    except:
        break
    farms, total_cost = [], 0
    for _ in range(n):
        farms.append(list(map(int, input().split())))
    distances, visited, heap = [100000 for _ in range(n)], set(), []
    distances[0] = 0
    heappush(heap, (distances[0], 0))
    while heap:
        cost, current_farm = heappop(heap)
        if current_farm in visited:
            continue
        visited.add(current_farm)
        total_cost += cost
        for neighbor_farm, distance in enumerate(farms[current_farm]):
            if distance < distances[neighbor_farm]:
                distances[neighbor_farm] = distance
                heappush(heap, (distances[neighbor_farm], neighbor_farm))
    print(total_cost)
```

代码运行截图



状态: Accepted

源代码

```python
'''
2100017810 刘思瑞
'''
from heapq import heappop, heappush

while True:
    try:
        n = int(input())
    except:
        break
    farms, total_cost = [], 0
    for _ in range(n):
        farms.append(list(map(int, input().split())))
    distances, visited, heap = [100000 for _ in range(n)], set(), []
    distances[0] = 0
    heappush(heap, (distances[0], 0))
    while heap:
        cost, current_farm = heappop(heap)
        if current_farm in visited:
            continue
        visited.add(current_farm)
        total_cost += cost
        for neighbor_farm, distance in enumerate(farms[current_farm]):
            if distance < distances[neighbor_farm]:
                distances[neighbor_farm] = distance
```

27635: 判断无向图是否连通有无回路(同23163)

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

思路：

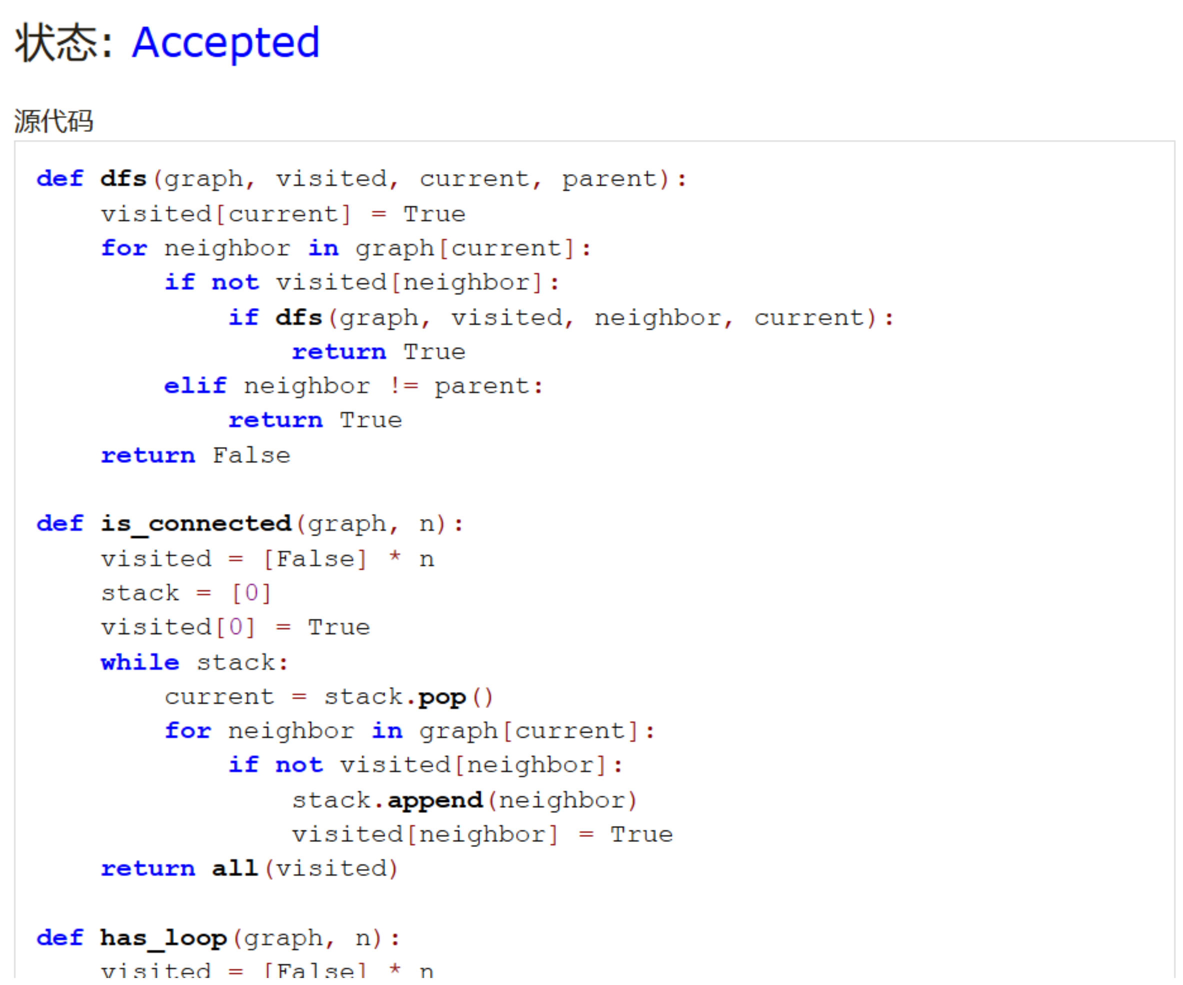
dfs，用图的邻接表来搜索

代码

```python
def dfs(graph, visited, current, parent):
    visited[current] = True
    for neighbor in graph[current]:
        if not visited[neighbor]:
            if dfs(graph, visited, neighbor, current):
                return True
        elif neighbor != parent:
            return True
    return False

def is_connected(graph, n):
    visited = [False] * n
    stack = [0]
    visited[0] = True
    while stack:
        current = stack.pop()
        for neighbor in graph[current]:
            if not visited[neighbor]:
                stack.append(neighbor)
                visited[neighbor] = True
    return all(visited)

def has_loop(graph, n):
    visited = [False] * n
    for i in range(n):
        if not visited[i]:
            if dfs(graph, visited, i, -1):
                return True
    return False

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

if is_connected(graph, n):
    print("connected:yes")
else:
    print("connected:no")

if has_loop(graph, n):
    print("loop:yes")
```

```
else:
    print("loop:no")
```

代码运行截图



状态: Accepted

源代码

```
def dfs(graph, visited, current, parent):
    visited[current] = True
    for neighbor in graph[current]:
        if not visited[neighbor]:
            if dfs(graph, visited, neighbor, current):
                return True
        elif neighbor != parent:
            return True
    return False

def is_connected(graph, n):
    visited = [False] * n
    stack = [0]
    visited[0] = True
    while stack:
        current = stack.pop()
        for neighbor in graph[current]:
            if not visited[neighbor]:
                stack.append(neighbor)
                visited[neighbor] = True
    return all(visited)

def has_loop(graph, n):
    visited = [False] * n
```

27947: 动态中位数

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

思路:

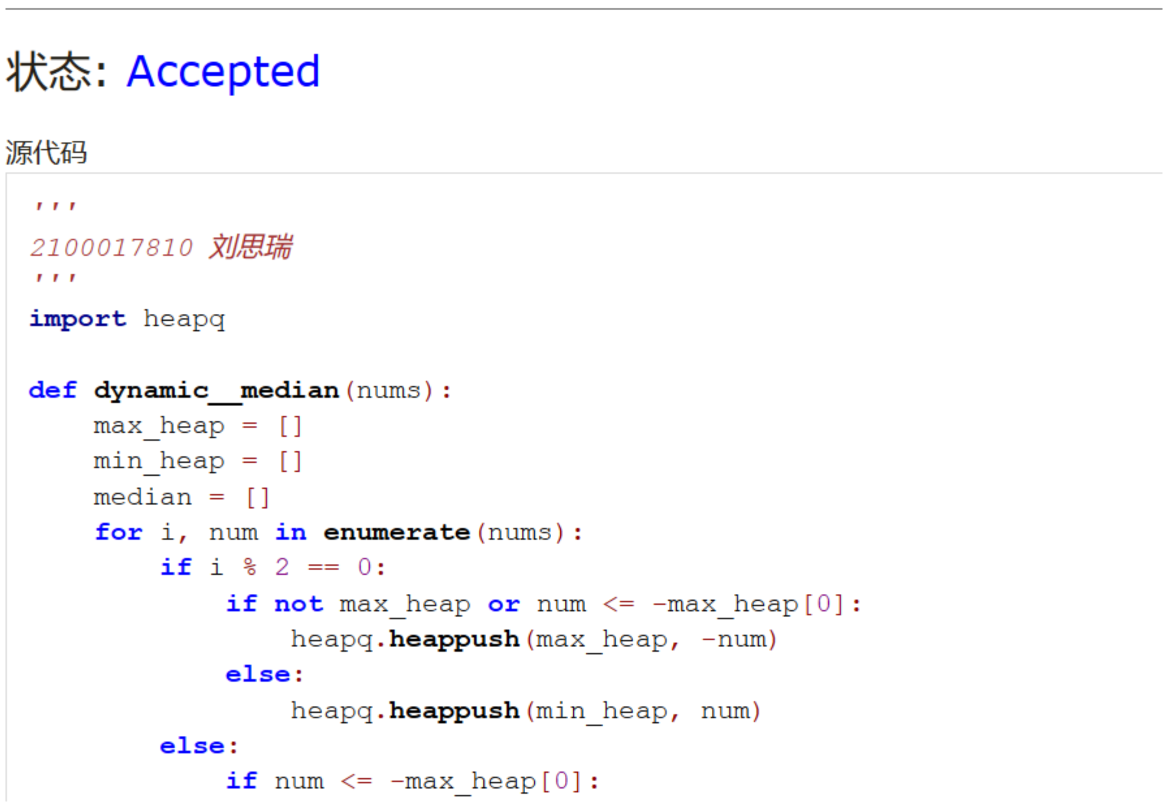
分为最大堆以及最小堆

代码

```
'''
2100017810 刘思瑞
'''
import heapq

def dynamic_median(nums):
```

```python
    max_heap = []
    min_heap = []
    median = []
    for i, num in enumerate(nums):
        if i % 2 == 0:
            if not max_heap or num <= -max_heap[0]:
                heapq.heappush(max_heap, -num)
            else:
                heapq.heappush(min_heap, num)
        else:
            if num <= -max_heap[0]:
                heapq.heappush(max_heap, -num)
            else:
                heapq.heappush(min_heap, num)

            if len(max_heap) > len(min_heap) + 1:
                heapq.heappush(min_heap, -heapq.heappop(max_heap))
            elif len(min_heap) > len(max_heap):
                heapq.heappush(max_heap, -heapq.heappop(min_heap))

        if i % 2 == 0:
            median.append(len(max_heap))
        else:
            median.append(-max_heap[0])

    return median

T = int(input())
for _ in range(T):
    nums = list(map(int, input().split()))
    N = len(nums)
    median = dynamic_median(nums)
    print(N)
    print(*median)
```

代码运行截图



状态: Accepted

源代码

```
'''
2100017810 刘思瑞
'''
import heapq

def dynamic__median(nums):
    max_heap = []
    min_heap = []
    median = []
    for i, num in enumerate(nums):
        if i % 2 == 0:
            if not max_heap or num <= -max_heap[0]:
                heapq.heappush(max_heap, -num)
            else:
                heapq.heappush(min_heap, num)
        else:
            if num <= -max_heap[0]:
```

28190: 奶牛排队

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

思路:

看了答案模仿单调栈的想法照着答案学习了一下

代码

```python
N = int(input())
heights = [int(input()) for _ in range(N)]

left_bound = [-1] * N
right_bound = [N] * N

stack = []

for i in range(N):
    while stack and heights[stack[-1]] < heights[i]:
        right_bound[stack.pop()] = i
    left_bound[i] = stack[-1] if stack else -1
    stack.append(i)

ans = max(i - left_bound[i] - 1 for i in range(N) if right_bound[i] > i)
print(ans)
```

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

这次月考还是前两个正常的操作题目，中间是bfsdfs以及树和图的操作问题，最后的题目比较灵活感觉看运气能不能想到，毕竟没有太多的题型的训练感觉这部分思路还是看感觉