

Assignment #D: May月考

Updated 1654 GMT+8 May 8, 2024

2024 spring, Compiled 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

```

1  """
2  刘思瑞 2100017810
3  """
4  L , M = map(int, input().split())
5  tree = [1]*(L+1)
6  for i in range(M):
7      origin , determination = map(int, input().split())
8      for j in range(origin , determination+1):
9          tree[j] = 0
10 n = 0
11 for i in tree:
12     n += i
13 print(n)

```

代码运行截图

状态: Accepted

1

源代码

```

"""
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

```

1  '''
2  2100017810 刘思瑞
3  '''
4  def divisible_by_5(s):
5      result = []
6      num = 0
7      for bit in s:
8          num = num * 2 + int(bit)

```

```

9         if num % 5 == 0:
10             result.append('1')
11         else:
12             result.append('0')
13     return ''.join(result)
14 input_str = input().strip()
15 print(divisible_by_5(input_str))

```

代码运行截图

状态: Accepted

源代码

```

'''
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

```

1  '''
2  2100017810 刘思瑞
3  '''
4  from heapq import heappop, heappush
5
6  while True:
7      try:
8          n = int(input())
9      except:
10         break

```

```

11     farms, total_cost = [], 0
12     for _ in range(n):
13         farms.append(list(map(int, input().split())))
14     distances, visited, heap = [100000 for _ in range(n)], set(), []
15     distances[0] = 0
16     heappush(heap, (distances[0], 0))
17     while heap:
18         cost, current_farm = heappop(heap)
19         if current_farm in visited:
20             continue
21         visited.add(current_farm)
22         total_cost += cost
23         for neighbor_farm, distance in enumerate(farms[current_farm]):
24             if distance < distances[neighbor_farm]:
25                 distances[neighbor_farm] = distance
26                 heappush(heap, (distances[neighbor_farm], neighbor_farm))
27     print(total_cost)

```

代码运行截图

状态: Accepted

源代码

```

'''
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

```
1 def dfs(graph, visited, current, parent):
2     visited[current] = True
3     for neighbor in graph[current]:
4         if not visited[neighbor]:
5             if dfs(graph, visited, neighbor, current):
6                 return True
7         elif neighbor != parent:
8             return True
9     return False
10
11 def is_connected(graph, n):
12     visited = [False] * n
13     stack = [0]
14     visited[0] = True
15     while stack:
16         current = stack.pop()
17         for neighbor in graph[current]:
18             if not visited[neighbor]:
19                 stack.append(neighbor)
20                 visited[neighbor] = True
21     return all(visited)
22
23 def has_loop(graph, n):
24     visited = [False] * n
25     for i in range(n):
26         if not visited[i]:
27             if dfs(graph, visited, i, -1):
28                 return True
29     return False
30
31 n, m = map(int, input().split())
32 graph = [[] for _ in range(n)]
33 for _ in range(m):
34     u, v = map(int, input().split())
35     graph[u].append(v)
36     graph[v].append(u)
37
38 if is_connected(graph, n):
39     print("connected:yes")
40 else:
41     print("connected:no")
42
43 if has_loop(graph, n):
44     print("loop:yes")
```

```
45 else:
46     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

```
1 '''
2 2100017810 刘思瑞
3 '''
4 import heapq
5
6 def dynamic_median(nums):
```

```
7     max_heap = []
8     min_heap = []
9     median = []
10    for i, num in enumerate(nums):
11        if i % 2 == 0:
12            if not max_heap or num <= -max_heap[0]:
13                heapq.heappush(max_heap, -num)
14            else:
15                heapq.heappush(min_heap, num)
16        else:
17            if num <= -max_heap[0]:
18                heapq.heappush(max_heap, -num)
19            else:
20                heapq.heappush(min_heap, num)
21
22            if len(max_heap) > len(min_heap) + 1:
23                heapq.heappush(min_heap, -heapq.heappop(max_heap))
24            elif len(min_heap) > len(max_heap):
25                heapq.heappush(max_heap, -heapq.heappop(min_heap))
26
27        if i % 2 == 0:
28            median.append(len(max_heap))
29        else:
30            median.append(-max_heap[0])
31
32    return median
33
34    T = int(input())
35    for _ in range(T):
36        nums = list(map(int, input().split()))
37        N = len(nums)
38        median = dynamic_median(nums)
39        print(N)
40        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

```
1 N = int(input())
2 heights = [int(input()) for _ in range(N)]
3
4 left_bound = [-1] * N
5 right_bound = [N] * N
6
7 stack = []
8
9 for i in range(N):
10     while stack and heights[stack[-1]] < heights[i]:
11         right_bound[stack.pop()] = i
12     left_bound[i] = stack[-1] if stack else -1
13     stack.append(i)
14
15 ans = max(i - left_bound[i] - 1 for i in range(N) if right_bound[i] > i)
16 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