Assignment #D: May月考

Updated 1654 GMT+8 May 8, 2024

2024 spring, Complied by 陈奕好 工学院

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 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 Sonoma 14.4 (23E214)

Python编程环境：PyCharm 2023.3.1 (Professional Edition)

1. 题目

02808: 校门外的树

http://cs101.openjudge.cn/practice/02808/

思路：看图说话题

代码

```
L, M = map(int, input().split())
tree = [1] * (L + 1)
for i in range(M):
    start, end = map(int, input().split())
    tree[start: end + 1] = [0] * (end - start + 1)
print(sum(tree))

```

代码运行截图 （至少包含有"Accepted"）

#44897475提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
L, M = map(int, input().split())
tree = [1] * (L + 1)
for i in range(M):
    start, end = map(int, input().split())
    tree[start: end + 1] = [0] * (end - start + 1)
print(sum(tree))
```

基本信息

#: 44897475
题目: E02808
提交人: 23n2300011030(陈奕好)
内存: 3796kB
时间: 23ms
语言: Python3
提交时间: 2024-05-08 15:17:35

English 帮助 关于

20449: 是否被5整除

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

思路：int()用了都好说用

代码

```
A = input()
ans = ""
for i in range(len(A)):
    tmp = A[:i+1]
    if int(tmp, 2) % 5 == 0:
        ans += "1"
    else:
        ans += "0"
print(ans)

```

代码运行截图 （至少包含有"Accepted"）

#44897519提交状态 查看 提交 统计 提问

状态: Accepted

源代码

```
A = input()
ans = ""
for i in range(len(A)):
    tmp = A[:i+1]
    if int(tmp, 2) % 5 == 0:
        ans += "1"
    else:
        ans += "0"
print(ans)
```

基本信息

#: 44897519
题目: E20449
提交人: 23n2300011030(陈奕好)
内存: 3596kB
时间: 20ms
语言: Python3
提交时间: 2024-05-08 15:20:22

01258: Agri-Net

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

思路：prim最小生成树

代码

```
import heapq
def prim(graph, start):
    mst = []
    used = {start}
    edges = [
        (cost, start, to)
        for to, cost in graph[start].items()
    ]
    heapq.heapify(edges)

    while edges:
        cost, frm, to = heapq.heappop(edges)
        if to not in used:
            used.add(to)
            mst.append((frm, to, cost))
            for to_next, cost2 in graph[to].items():
                if to_next not in used:
                    heapq.heappush(edges, (cost2, to, to_next))
    return mst

while True:
    try:
        n = int(input())
        graph = {i:dict() for i in range(n)}
        for i in range(n):
            tmp = list(map(int, input().split()))
```

```
            node = i
            for j in range(n):
                if j != i:
                    graph[node][j] = tmp[j]

        # print(graph)
        mst = prim(graph, 0)
        ans = [cost for frm, to, cost in mst]
        print(sum(ans))
    except EOFError:
        break

```

代码运行截图 （AC代码截图，至少包含有"Accepted"）

#45030419提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
import heapq
def prim(graph, start):
    mst = []
    used = {start}
    edges = [
        (cost, start, to)
        for to, cost in graph[start].items()
    ]
    heapq.heapify(edges)

    while edges:
        cost, frm, to = heapq.heappop(edges)
        if to not in used:
            used.add(to)
            mst.append((frm, to, cost))
            for to_next, cost2 in graph[to].items():
                if to_next not in used:
                    heapq.heappush(edges, (cost2, to, to_next))
    return mst

while True:
    try:
        n = int(input())
        graph = {i:dict() for i in range(n)}
        for i in range(n):
            tmp = list(map(int, input().split()))
            node = i
            for j in range(n):
                if j != i:
```

基本信息

#: 45030419
题目: 01258
提交人: 23n2300011030(陈奕好)
内存: 4700kB
时间: 43ms
语言: Python3
提交时间: 2024-05-21 09:14:46

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找回路和连通性

代码

```
def is_connected(G):
    n = len(G)
    visited = [False for i in range(n)]
    total = 0

    def dfs(v):
        nonlocal total
        visited[v] = True
        total += 1
        for u in G[v]:
            if not visited[u]:
                dfs(u)

    dfs(0)
    return total == n

def hasloop(G):
    n = len(G)
    visited = [False for i in range(n)]

    def dfs(v, x):
        visited[v] = True
        for u in G[v]:
            if visited[u]:
                if u != x:
                    return True
            else:
                if dfs(u, v):
                    return True
        return False

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

n, m = map(int,input().split())
G = [[] for i in range(n)]
for _ in range(m):
    a, b = map(int ,input().split())
    G[a].append(b)
    G[b].append(a)

if is_connected(G):
    print("connected:yes")
else:
    print("connected:no")

if hasloop(G):
```

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

```

代码运行截图 （AC代码截图，至少包含有"Accepted"）

#45030438提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
def is_connected(G):
    n = len(G)
    visited = [False for i in range(n)]
    total = 0

    def dfs(v):
        nonlocal total
        visited[v] = True
        total += 1
        for u in G[v]:
            if not visited[u]:
                dfs(u)

    dfs(0)
    return total == n

def hasloop(G):
    n = len(G)
    visited = [False for i in range(n)]

    def dfs(v, x):
        visited[v] = True
        for u in G[v]:
            if visited[u]:
                if u != x:
                    return True
            else:
                if dfs(u, v):
                    return True
```

基本信息

#: 45030438
题目: 27635
提交人: 23n2300011030(陈奕好)
内存: 3720kB
时间: 25ms
语言: Python3
提交时间: 2024-05-21 09:20:13

27947: 动态中位数

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

思路：其实是“洗牌排序”，在min里洗出最小的牌，放入max再洗出max最大的牌，min中的每一项一定大于max中的，维持两个堆长度差一即可洗出中位数。

代码

```
import heapq


```

```python
def find_median(numbers):
    min_heap = []
    max_heap = []
    for i, number in enumerate(numbers):
        heapq.heappush(max_heap, -heapq.heappushpop(min_heap, number))
        if len(max_heap) > len(min_heap):
            heapq.heappush(min_heap, -heapq.heappop(max_heap))

        if i % 2 == 0:
            ans.append(min_heap[0])


T = int(input())
for i in range(T):
    ans = []
    arr = list(map(int, input().split()))
    find_median(arr)
    print(len(ans))
    print(*ans)

```

代码运行截图 （AC代码截图，至少包含有"Accepted"）

#45030446提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```python
import heapq


def find_median(numbers):
    min_heap = []
    max_heap = []
    for i, number in enumerate(numbers):
        heapq.heappush(max_heap, -heapq.heappushpop(min_heap, number))
        if len(max_heap) > len(min_heap):
            heapq.heappush(min_heap, -heapq.heappop(max_heap))

        if i % 2 == 0:
            ans.append(min_heap[0])


T = int(input())
for i in range(T):
    ans = []
    arr = list(map(int, input().split()))
    find_median(arr)
    print(len(ans))
    print(*ans)
```

基本信息

#: 45030446
题目: 27947
提交人: 23n2300011030(陈奕好)
内存: 9856kB
时间: 300ms
语言: Python3
提交时间: 2024-05-21 09:21:23

English 帮助 关于

28190: 奶牛排队

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

思路：跟股票系列有点像，可惜没想出来。

代码

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

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

stack = []  # 单调栈，存储索引

# 求左侧第一个≥h[i]的奶牛位置
for i in range(N):
    while stack and heights[stack[-1]] < heights[i]:
        stack.pop()

    if stack:
        left_bound[i] = stack[-1]

    stack.append(i)

stack = []  # 清空栈以供寻找右边界使用

# 求右侧第一个≤h[i]的奶牛位
for i in range(N-1, -1, -1):
    while stack and heights[stack[-1]] > heights[i]:
        stack.pop()

    if stack:
        right_bound[i] = stack[-1]

    stack.append(i)

ans = 0

# for i in range(N-1, -1, -1):  # 从大到小枚举是个技巧
#     for j in range(left_bound[i] + 1, i):
#         if right_bound[j] > i:
#             ans = max(ans, i - j + 1)
#             break
#
#     if i <= ans:
#         break

for i in range(N):  # 枚举右端点 B寻找 A，更新 ans
    for j in range(left_bound[i] + 1, i):
        if right_bound[j] > i:
            ans = max(ans, i - j + 1)
```

```
            break
print(ans)
```

代码运行截图 （AC代码截图，至少包含有"Accepted"）

#45030553提交状态

查看 提交 统计 提问

状态: Accepted

源代码

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

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

stack = []  # 单调栈，存储索引

# 求左侧第一个≥h[i]的奶牛位置
for i in range(N):
    while stack and heights[stack[-1]] < heights[i]:
        stack.pop()

    if stack:
        left_bound[i] = stack[-1]

    stack.append(i)

stack = []  # 清空栈以供寻找右边界使用

# 求右侧第一个≤h[i]的奶牛位
for i in range(N-1, -1, -1):
    while stack and heights[stack[-1]] > heights[i]:
        stack.pop()

    if stack:
        right_bound[i] = stack[-1]

    stack.append(i)
```

基本信息

#: 45030553
题目: 28190
提交人: 23n2300011030(陈奕好)
内存: 82472kB
时间: 2711ms
语言: Python3
提交时间: 2024-05-21 09:43:27

2. 学习总结和收获

如果作业题目简单，有否额外练习题目，比如：OJ“2024spring每日选做”、CF、LeetCode、洛谷等网站题目。

最近在复习树和图