期末机考

第一题 Okabe and Boxes

http://cs101.openjudge.cn/cs2012024feclass12/E27933/

```
n = int(input())
count = 0
stack = []
now = 1
for _ in range(2*n):
    s = input()
    if s[0] == 'r':
        if stack[-1] != now:
            count += 1
            stack.sort()
            stack.reverse()
        stack.pop()
        now += 1
    else:
        item = int(s.split()[1])
        stack.append(item)
print(count)
```

第二题 分糖果

http://cs101.openjudge.cn/cs2012024feclass12/E28186/

```
n,m = map(int,input().split())
child = list(map(int,input().split()))
queue = list((x,i+1) for i,x in enumerate(child))
turn = []
while queue:
    c,index = queue.pop(0)
    if c <= m:
        turn.append(index)
    else:
        queue.append((c-m,index))
print(turn[-1])
```

第三题 首先条件下可到达结点的数目

http://cs101.openjudge.cn/cs2012024feclass12/M28012/

```
n = int(input())
graph = {i:[] for i in range(n)}
for _ in range(n-1):
    a,b = map(int,input().split())
    graph[a].append(b)
    graph[b].append(a)
cant = set(map(int,input().split()))
visited = set()
queue = [0]
while queue:
```

```
    node = queue.pop(0)
    if node in visited:
        continue
    else:
        visited.add(node)
        for i in graph[node]:
            if i not in cant:
                queue.append(i)
print(len(visited))
```

第四题 堆路径

http://cs101.openjudge.cn/cs2012024feclass12/M28013/

```
n = int(input())
heap = list(map(int,input().split()))
heap_list = []
def return_heap(heap,i,lst=[]):
    global n
    if 2*i+2 < n:
        return_heap(heap,2*i+2,lst+[i])
        return_heap(heap,2*i+1,lst+[i])
    elif 2*i+1 < n:
        return_heap(heap,2*i+1,lst+[i])
    else:
        lst.append(i)
        new_list = list(heap[j] for j in lst)
        heap_list.append(new_list)
        print(*new_list)
        return
return_heap(heap,0)
flag = None
for lst in heap_list:
    new = sorted(lst)
    new1 = list(reversed(new))
    if lst == new:
        if flag is None:
            flag = 'Min Heap'
        elif flag == 'Max Heap':
            flag = 'Not Heap'
            break
    elif lst == new1:
        if flag is None:
            flag = 'Max Heap'
        elif flag == 'Min Heap':
            flag = 'Not Heap'
            break
    else:
        flag = 'Not Heap'
        break
print(flag)
```

第五题 谣言

http://cs101.openjudge.cn/cs2012024feclass12/M28193/

```
n,m = map(int,input().split())
ci = list(map(int,input().split()))
cup1 = {i:i for i in range(1,n+1)}
cup2 = {i:{i} for i in range(1,n+1)}
for _ in range(m):
    a,b = map(int,input().split())
    a1,b1 = cup1[a],cup1[b]
    if a1 != b1:
        for j in cup2[b1]:
            cup1[j] = a1
            cup2[a1].add(j)
        del cup2[b1]
coins = 0
for k in cup2:
    coins += min(list(ci[j-1] for j in cup2[k]))
print(coins)
```

第六题 判断等价关系是否成立

刚出考场就会做了+1

http://cs101.openjudge.cn/cs2012024feclass12/M28276/

```
n = int(input())
dic = {}
not_euqal = []
def find(dic,x):
    if dic[x] != x:
        dic[x] = find(dic,dic[x])
    return dic[x]
def union(dic,x,y):
    rootx,rooty = find(dic,x),find(dic,y)
    if rootx != rooty:
        dic[rootx] = dic[rooty]

for _ in range(n):
    s = input()
    x,y = s[0],s[-1]
    if x not in dic: dic[x] = x
    if y not in dic: dic[y] = y
    if s[1:3] == '==':
        union(dic, x, y)
    else:
        not_euqal.append((x,y))
flag = True
for x,y in not_euqal:
    if find(dic,x) == find(dic,y):
        flag = False
print(flag)
```