Assignment #A: 图论：遍历，树算及栈

Updated 2018 GMT+8 Apr 21, 2024

2024 spring, Complied by 余汶青 生命科学学院

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

操作系统：版本　Windows 11 家庭中文版
版本　22H2
安装日期 2023/7/18
操作系统版本 22621.2283
序列号　5CD323PJKL
体验　Windows Feature Experience Pack 1000.22662.1000.0

Python编程环境：* Spyder version: 5.4.3 (conda)

- Python version: 3.11.4 64-bit
- Qt version: 5.15.2
- PyQt5 version: 5.15.7
- Operating System: Windows 10

1. 题目

20743: 整人的提词本

http://cs101.openjudge.cn/practice/20743/

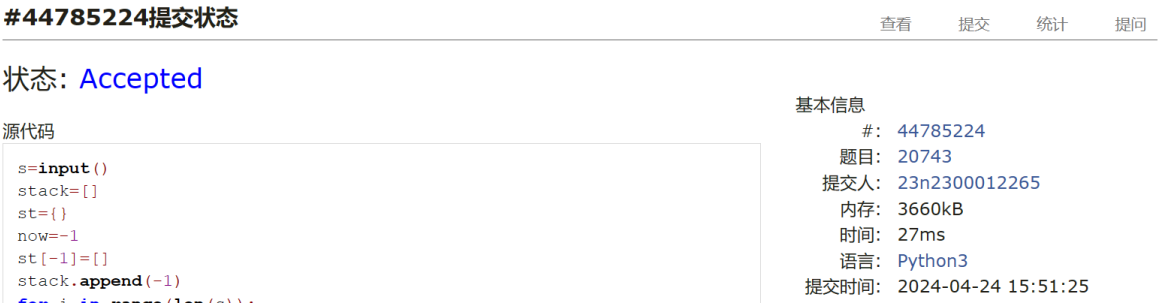
思路：

代码

```
s=input()
stack=[]
st={}
now=-1
st[-1]=[]
```

```
stack.append(-1)
for i in range(len(s)):
    if s[i]=='(':
        stack.append(i)
        now=i
        st[now]=[]
    elif s[i]==')':
        j=stack.pop()
        #print(j)
        a=st[j]
        a.reverse()
        k=stack[-1]
        st[k].extend(a)
        now=k
    else:

        st[now].append(s[i])
       # print(now,*st[now])
print(*st[-1],sep='')
```

代码运行截图



02255: 重建二叉树

http://cs101.openjudge.cn/practice/02255/

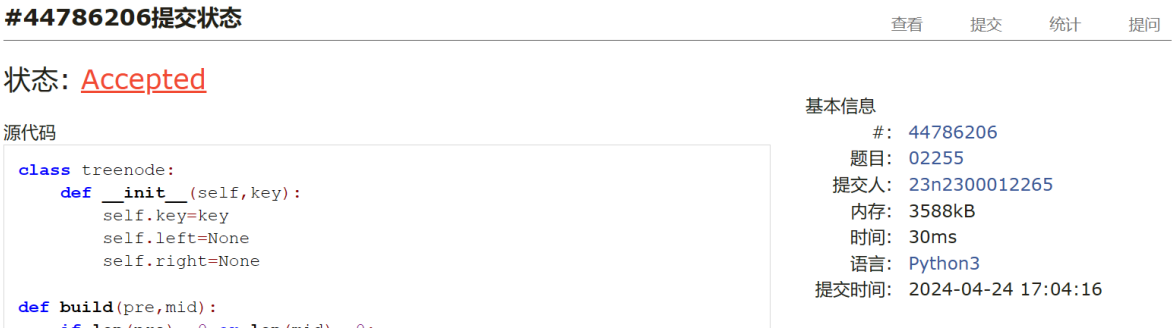
思路:

代码

```
class treenode:
    def __init__(self,key):
        self.key=key
        self.left=None
        self.right=None

def build(pre,mid):
    if len(pre)==0 or len(mid)==0:
        return None
    #print(pre,mid)
```

```
    root=treenode(pre[0])
    if len(mid)==1:
        return root
    ind=mid.index(pre[0])
    root.left=build(pre[1:],mid[:ind])
    root.right=build(pre[1+ind:],mid[ind+1:])
    return root

def post(root):
    if root.left:
        post(root.left)
    if root.right:
        post(root.right)
    print(root.key,end='')

while 1:
    try:
        pre,mid=input().split()
        root=build(pre,mid)
        post(root)
        print()
    except:
        break
```

代码运行截图



01426: Find The Multiple

http://cs101.openjudge.cn/practice/01426/

要求用bfs实现

思路:

其实这个题很简单，和上学期期末考试的那个跳房子很像

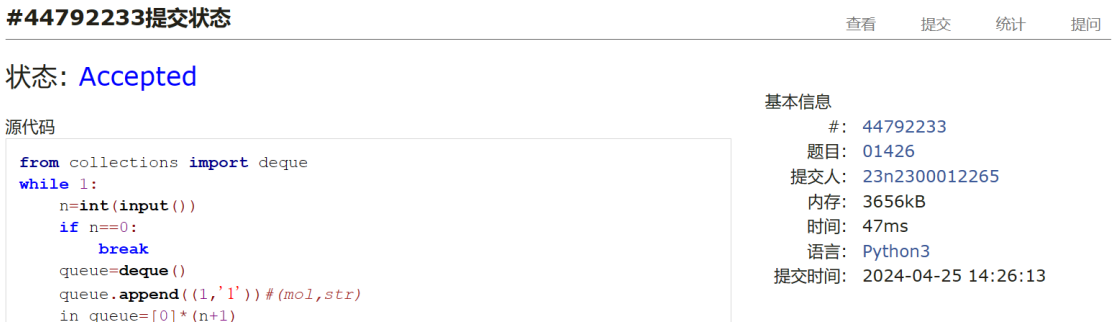
但是我一开始被样例输出的那一串1/0吓住了，不知道要怎么处理这么大的数据，后面看题解发现原来直接取模就好，感觉自己还是题目做少了，脑子转不过弯来唉

代码

```
from collections import deque
```

```
while 1:
    n=int(input())
    if n==0:
        break
    queue=deque()
    queue.append((1,'1'))#(mol,str)
    in_queue=[0]*(n+1)
    while queue:
        a,b=queue.popleft()
        if a==0:
            print(b)
            break
        a1=a*10%n
        a2=(a*10+1)%n
        if in_queue[a1]==0:
            in_queue[a1]=1
            queue.append((a1,b+'0'))
        if in_queue[a2]==0:
            in_queue[a2]=1
            queue.append((a2,b+'1'))
```

代码运行截图



04115: 鸣人和佐助

bfs, http://cs101.openjudge.cn/practice/04115/

思路:

这个题不能靠in_queue来判断是否应该把一个节点加入队伍，因为有可能有多条路径可以抵达这个节点，而每个路径的查克拉数目不同，选择需要最少的查克拉的数量才是正确的解。

代码

```
import sys
from collections import deque

d=[(1,0),(0,-1),(-1,0),(0,1)]
m,n,t=map(int,input().split())
```

```
mapp=[[] for i in range(m)]
startx,starty=0,0
endx,endy=0,0
for i in range(m):
    s=input()
    for j in range(len(s)):
        mapp[i].append(s[j])
        if s[j]=='@':
            startx=i
            starty=j
        if s[j]=='+':
            endx=i
            endy=j

queue=deque()
cc=[[11 for i in range(n)] for i in range(m)]
queue.append((startx,starty,0,0))
while queue:
    x,y,c,time=queue.popleft()
    for dx,dy in d:
        nc=c
        nx=x+dx
        ny=y+dy
        if 0<=nx<m and 0<=ny<n:
            if nx==endx and ny==endy:
                print(time+1)
                sys.exit()
            if mapp[nx][ny]=='#':
                nc+=1
            if nc<=t:
                if cc[nx][ny]>nc:
                    cc[nx][ny]=nc
                    queue.append((nx,ny,nc,time+1))
print(-1)
```

代码运行截图

#44791296提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
import sys
from collections import deque

d=[(1,0),(0,-1),(-1,0),(0,1)]
m,n,t=map(int,input().split())
mapp=[[] for i in range(m)]
startx,starty=0,0
endx,endy=0,0
for i in range(m):
```

基本信息

#: 44791296
题目: 04115
提交人: 23n2300012265
内存: 4512kB
时间: 99ms
语言: Python3
提交时间: 2024-04-25 12:08:07

20106: 走山路

Dijkstra, http://cs101.openjudge.cn/practice/20106/

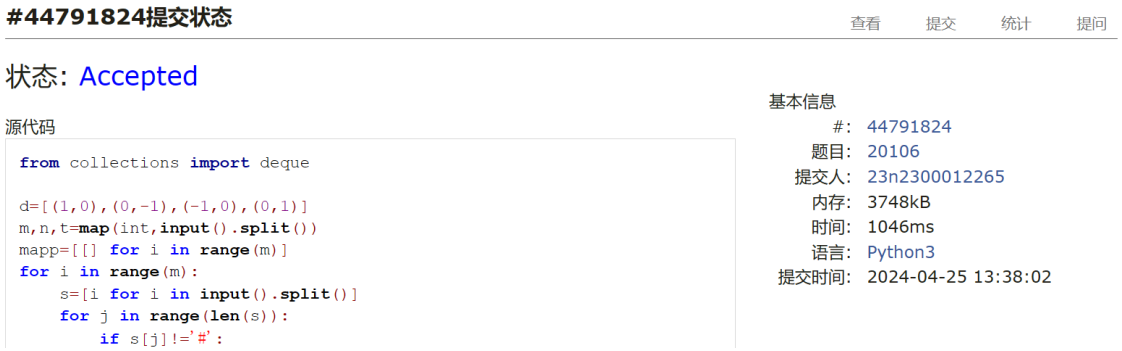
思路：

上学期就卡了好久，这次又卡了好久，一定一定要记得判断特殊情况！！！！！另外把体力值单独拿个矩阵记录比一起放进队列里面快，可能是因为这样可以一次性更新队列里面所有这个点的体力值。

代码

```
from collections import deque

d=[(1,0),(0,-1),(-1,0),(0,1)]
m,n,t=map(int,input().split())
mapp=[[] for i in range(m)]
for i in range(m):
    s=[i for i in input().split()]
    for j in range(len(s)):
        if s[j]!='#':
            mapp[i].append(int(s[j]))
        else:
            mapp[i].append(s[j])
for _ in range(t):
    startx,starty,endx,endy=map(int,input().split())
    if mapp[startx][starty]=='#' or mapp[endx][endy]=='#':
        print("NO")
        continue
    if startx==endx and starty==endy:
        print(0)
        continue
    queue=deque()
    cc=[[99999999 for i in range(n)] for i in range(m)]
    cc[startx][starty]=0
    queue.append((startx,starty))
    while queue:
        x,y=queue.popleft()
        for dx,dy in d:
            nc=cc[x][y]
            nx=x+dx
            ny=y+dy
            if 0<=nx<m and 0<=ny<n and mapp[nx][ny]!='#':
                nc+=abs(mapp[nx][ny]-mapp[x][y])
                if cc[nx][ny]>nc:
                    cc[nx][ny]=nc
                    queue.append((nx,ny))

    if cc[endx][endy]!=99999999:
        print(cc[endx][endy])
    else:
        print("NO")
```

代码运行截图



#44791824提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
from collections import deque

d=[(1,0),(0,-1),(-1,0),(0,1)]
m,n,t=map(int,input().split())
mapp=[[] for i in range(m)]
for i in range(m):
    s=[i for i in input().split()]
    for j in range(len(s)):
        if s[j]!='#':
```

基本信息

#: 44791824
题目: 20106
提交人: 23n2300012265
内存: 3748kB
时间: 1046ms
语言: Python3
提交时间: 2024-04-25 13:38:02

05442: 兔子与星空

Prim, http://cs101.openjudge.cn/practice/05442/

思路:

代码

```
import sys
import heapq
class Vertex:
    def __init__(self,key):
        self.id=key
        self.connectedTo={}
        self.distance=sys.maxsize
        self.pre=None
    def addNeighbor(self,nbr,weight=0):
        self.connectedTo[nbr]=weight
    def __str__(self):
        return str(self.id)+'connectedTo:'+str([x.id for x in self.connectedTo])
    def getConnections(self):
        return self.connectedTo.keys()
    def getId(self):
        return self.id
    def getWeight(self,nbr):
        return self.connectedTo[nbr]
class Graph:
    def __init__(self):
        self.vertList={}
        self.numVertices=0
    def addVertex(self,key):
        self.numVertices+=1
        newVertex=Vertex(key)
        self.vertList[key]=newVertex
        return newVertex
    def getVertex(self,n):
        if n in self.vertList:
```

```
            return self.vertList[n]
        else:
            return None
    def __contains__(self,n):
        return n in self.vertList
    def addEdge(self,f,t,weight=0):
        if f not in self.vertList:
            nv=self.addVertex(f)
        if t not in self.vertList:
            nv=self.addVertex(t)
        self.vertList[f].addNeighbor(self.vertList[t],weight)
    def getVertices(self):
        return self.vertList.keys()
    def __iter__(self):
        return iter(self.vertList.values())

def prim(graph,start):
    pq=[]
    start.distance=0
    heapq.heappush(pq, (0,start))
    visited=set()
    while pq:
        currentDist,currentVert=heapq.heappop(pq)
        if currentVert in visited:
            continue
        visited.add(currentVert)
        for nextVert in currentVert.getConnections():
            weight=currentVert.getWeight(nextVert)
            if nextVert not in visited and weight<nextVert.distance:
                nextVert.distance=weight
                nextVert.pre=currentVert
                heapq.heappush(pq,(weight,nextVert))

n=int(input())
graph=Graph()
for _ in range(n-1):
    s=input().split()
    if int(s[1])>0:
        for i in range(2,len(s),2):
            graph.addEdge(s[0], s[i],int(s[i+1]))
            graph.addEdge(s[i], s[0],int(s[i+1]))
prim(graph,graph.getVertex('A'))
ans=0
for vertex in graph.vertList.values():
    ans+=vertex.distance
print(ans)
```

代码运行截图

#44798672提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
import sys
import heapq
class Vertex:
    def __init__(self,key):
        self.id=key
        self.connectedTo={}
        self.distance=sys.maxsize
```

基本信息

#: 44798672
题目: 05442
提交人: 23n2300012265
内存: 3792kB
时间: 30ms
语言: Python3
提交时间: 2024-04-26 00:30:12

2. 学习总结和收获

复习了递归和树的遍历

写了三道bfs的题目，04115和20106都需要单独开一个矩阵来记录当前位置的最小值，这两道题应该也可以视作最短路径，然后用 Dijkstra，后面有时间试试这样写

然后兔子与星空学到了用prim算法解决最小生成树