

Assignment #A: 图论：算法，树算及栈

Updated 2018 GMT+8 Apr 21, 2024

2024 spring, Complied by 田济维 物理学院

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

(python pycharm)

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

20743: 整人的提词本

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

思路：

代码

```
1  #
2  s = input()
3  temp = []
4  string = []
5  for x in s:
6      if x!=")":
7          string.append(x)
8      else:
9          while string[-1]!="(":
```

```

10         temp.append(string.pop())
11     string.pop()
12     string.extend(temp)
13     temp.clear()
14 print("".join(string))

```

代码运行截图 (至少包含有"Accepted")

#44838579提交状态

查看 提交 统计

状态: Accepted

源代码

```

s = input()
temp = []
string = []
for x in s:
    if x!="(":
        string.append(x)
    else:
        while string[-1]!="(":
            temp.append(string.pop())
        string.pop()
        string.extend(temp)
        temp.clear()
print("".join(string))

```

基本信息

#: 44838579
 题目: 20743
 提交人: 23n2300011503
 内存: 3564kB
 时间: 21ms
 语言: Python3
 提交时间: 2024-04-30 23:07:01

02255: 重建二叉树

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

思路:

代码

```

1  #
2  def post(pro,mid):
3      if len(pro)>1:
4          s = pro[0]
5          l = mid.index(s)
6          return post(pro[1:l+1],mid[:l])+post(pro[l+1:],mid[l+1:])+s
7      else:
8          return pro
9  while True:
10     try:
11         pro,mid = input().split()
12     except EOFError:
13         break
14     else:
15         print(post(pro,mid))

```

代码运行截图 (至少包含有"Accepted")

状态: Accepted

源代码

```
def post(pro,mid):
    if len(pro)>1:
        s = pro[0]
        l = mid.index(s)
        return post(pro[1:l+1],mid[:l])+post(pro[l+1:],mid[l+1:])+s
    else:
        return pro
while True:
    try:
        pro,mid = input().split()
    except EOFError:
        break
    else:
        print(post(pro,mid))
```

基本信息

#: 44838606
题目: 02255
提交人: 23n2300011503
内存: 3520kB
时间: 22ms
语言: Python3
提交时间: 2024-04-30 23:12:58

01426: Find The Multiple

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

要求用bfs实现

思路:

代码

```
1 #
2 from collections import deque
3 def find(s):
4
5     que = deque(["1"])
6     while que:
7         t = que.popleft()
8         if int(t)%s==0 :
9             return t
10        else:
11            que.append(t+"0")
12            que.append(t+"1")
13    while True:
14        s = int(input())
15        if s == 0:
16            break
17        else:
18            print(find(s))
19
```

代码运行截图 (AC代码截图, 至少包含有"Accepted")

状态: Accepted

源代码

```
from collections import deque
def find(s):
    que = deque(["1"])
    while que:
        t = que.popleft()
        if int(t)%5==0 :
            return t
        else:
            que.append(t+"0")
            que.append(t+"1")
    while True:
        s = int(input())
        if s == 0:
            break
        else:
            print(find(s))
```

基本信息

#: 44838661
题目: 01426
提交人: 23n2300011503
内存: 49856kB
时间: 829ms
语言: Python3
提交时间: 2024-04-30 23:27:34

04115: 鸣人和佐助

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

思路:

代码

```
1 #
2 from collections import deque
3 m,n,T = map(int,input().split())
4 Map = [[0 for i in range(n)] for j in range(m)]
5 x1,y1=0,0
6 for i in range(m):
7     s = list(input())
8     for j in range(n):
9         if s[j]=="@":
10             x1,y1=i,j
11             Map[i][j]=s[j]
12 dx = [1,-1,0,0]
13 dy = [0,0,1,-1]
14 def bfs(x1,y1):
15     que = deque([(x1,y1,T)])
16     cnt =1
17     arrived = dict()
18     arrived[(x1,y1)]=T
19     while que:
20         for j in range(len(que)):
21             s = que.popleft()
22             x=s[0]
23             y=s[1]
24             t = s[2]
25             for i in range(4):
26                 tx = x+dx[i]
27                 ty = y+dy[i]
```

```

28         if 0<=tx<m and 0<=ty<n and ((tx,ty) not in arrived or
arrived[(tx,ty)]<t):
29             if Map[tx][ty]=="*":
30                 que.append((tx,ty,t))
31                 arrived[(tx,ty)]=t
32             elif Map[tx][ty]=="#":
33                 if t>0:
34                     que.append((tx,ty,t-1))
35                     arrived[(tx,ty)]=t-1
36             elif Map[tx][ty]=="+":
37                 return cnt
38         cnt+=1
39     return -1
40 print(bfs(x1,y1))

```

代码运行截图 (AC代码截图, 至少包含有"Accepted")

状态: Accepted

源代码

```

from collections import deque
m,n,T = map(int,input().split())
Map = [[0 for i in range(n)] for j in range(m)]
x1,y1=0,0
for i in range(m):
    s = list(input())
    for j in range(n):
        if s[j]=="@":
            x1,y1=i,j
        Map[i][j]=s[j]
dx = [1,-1,0,0]
dy = [0,0,1,-1]
def bfs(x1,y1):
    que = deque([(x1,y1,T)])
    cnt = 1
    arrived = dict()
    arrived[(x1,y1)]=T
    while que:
        for j in range(len(que)):
            s = que.popleft()
            x=s[0]
            y=s[1]
            t = s[2]
            for i in range(4):
                tx = x+dx[i]

```

基本信息

#: 44838795
 题目: 04115
 提交人: 23n2300011503
 内存: 4752kB
 时间: 101ms
 语言: Python3
 提交时间: 2024-05-01 00:22:37

20106: 走山路

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

思路:

代码

```

1 #
2 from heapq import *
3 m,n,p = map(int,input().split())
4 #创建地图

```

```

5 Map = [["#"]*(n+2) for i in range(m+2)]
6 for i in range(1,m+1):
7     Map[i][1:-1]=input().split()
8
9 def bfs(x1,y1,x2,y2):
10     #一开始能到达的最近点是起始点，耗体力0
11     q = [(0,x1,y1)]
12     #对q堆排序
13     heapify(q)
14     #需要一个数据容器记录处理过的结点processed
15     sured = set()
16     tx = [1,-1,0,0]
17     ty = [0,0,1,-1]
18     while q:
19         # 找到目前为止能到达的耗总体力最少的结点，拜访它，把它加入已处理的集合中，后面到达
        此点不可能有更短的步骤了
20         t,x,y = heappop(q)
21         sured.add((x,y))
22         #如果是终点
23         if x == x2 and y == y2:
24             return t
25         # 然后看看加入了此结点后，有哪些子结点可以到达了
26         # heap的好处体现出来了，如果子节点之前就可以到达，我们不用管更新后子节点所需的体
        力是否下降了，因为heap会自动选出最小的
27         # 比如(12,3,4) 在q中，现在又加入了(15,3,4)，heap在取时还是会取(12,3,4)
        不用我们担心
28         for i in range(4):
29             nx = x+tx[i]
30             ny = y+ty[i]
31             #不用管之前到没到过，就是要加入更新
32             if Map[nx][ny]!="#" and (nx,ny) not in sured:
33                 heappush(q,(t+abs(int(Map[nx][ny])-int(Map[x][y])),nx,ny))
34             # 现在所有的新的产生的可到达的位置又加入q中了，重复上述操作
35             #如果q都空了还没return 说明根本到不了终点，因为只要能到终点，即使t很大，也最终被取出
        来
36     return "NO"
37 for i in range(p):
38     x1,y1,x2,y2 = map(int,input().split())
39     if Map[x1+1][y1+1]=="#" or Map[x2+1][y2+1]=="#":
40         print("NO")
41     else:
42         print(bfs(x1+1,y1+1,x2+1,y2+1))

```

代码运行截图 (AC代码截图，至少包含有"Accepted")

状态: Accepted

基本信息

#: 4323325

题目: 20106

提交人: 23n230c

内存: 4168kB

时间: 941ms

语言: Python3

提交时间: 2023-12

源代码

```
from heapq import *
m,n,p = map(int,input().split())
#创建地图
Map = [['#']*(n+2) for i in range(m+2)]
for i in range(1,m+1):
    Map[i][1:-1]=input().split()

def bfs(x1,y1,x2,y2):
    #一开始能到达的最近点是起始点, 耗体力0
    q = [(0,x1,y1)]
    #对q堆排序
    heapify(q)
    #需要一个数据容器记录处理过的结点processed
    sured = set()
    tx = [1,-1,0,0]
    ty = [0,0,1,-1]
    while q:
        #找到目前为止能到达的耗总体力最少的结点, 拜访它, 把它加入已处理的集合中, 后
        t,x,y = heappop(q)
```

05442: 兔子与星空

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

思路:

不会做, 图做的太少了, 多刷题

代码

```
1 #
2 import heapq
3
4 def prim(graph, start):
5     mst = []
6     used = set([start])
7     edges = [
8         (cost, start, to)
9         for to, cost in graph[start].items()
10    ]
11    heapq.heapify(edges)
12
13    while edges:
14        cost, frm, to = heapq.heappop(edges)
15        if to not in used:
16            used.add(to)
17            mst.append((frm, to, cost))
18            for to_next, cost2 in graph[to].items():
19                if to_next not in used:
20                    heapq.heappush(edges, (cost2, to, to_next))
21
22    return mst
23
24 def solve():
```

```

25     n = int(input())
26     graph = {chr(i+65): {} for i in range(n)}
27     for i in range(n-1):
28         data = input().split()
29         star = data[0]
30         m = int(data[1])
31         for j in range(m):
32             to_star = data[2+j*2]
33             cost = int(data[3+j*2])
34             graph[star][to_star] = cost
35             graph[to_star][star] = cost
36     mst = prim(graph, 'A')
37     print(sum(x[2] for x in mst))
38
39 solve()

```

代码运行截图 (AC代码截图, 至少包含有"Accepted")

#44838800提交状态

[查看](#) [提交](#) [统计](#)

状态: **Accepted**

源代码

```

import heapq

def prim(graph, start):
    mst = []
    used = set([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))

```

基本信息

#: 44838800
 题目: 05442
 提交人: 23n2300011503
 内存: 3676kB
 时间: 19ms
 语言: Python3
 提交时间: 2024-05-01 00:26

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。

关于带权的最值还是熟悉得多练, 最后一题反复学习以下