

# Assignment #B: 图 (1/4)

Updated 2031 GMT+8 Nov 17, 2025

2025 fall, Compiled by 杨浩、化院

## 1. 1. 题目

### 1.1 E07218: 献给阿尔吉侬的花束

bfs, <http://cs101.openjudge.cn/practice/07218/>

思路:

- 简单的bfs

代码:

```
1  from collections import deque
2  def find_start():
3      for j in range(r):
4          for k in range(c):
5              if matrix[j][k] == 'S':
6                  return j, k
7
8  def bfs(start, r, c):
9      queue = deque([(start[0], start[1], 0)])
10     finished = set()
11     finished.add(start)
12     delta = [(1, 0), (-1, 0), (0, 1), (0, -1)]
13     while queue:
14         x, y, steps = queue.popleft()
15         for dx, dy in delta:
16             if 0 <= x+dx < r and 0 <= y+dy < c:
17                 if (x+dx, y+dy) not in finished:
18                     finished.add((x+dx, y+dy))
19                     if matrix[x+dx][y+dy] == '.':
20                         queue.append((x+dx, y+dy, steps+1))
21                     if matrix[x+dx][y+dy] == 'E':
22                         return str(steps+1)
23     return 'oop!'
24 t = int(input())
25 for i in range(t):
26     r, c = map(int, input().split())
27     matrix = []
28     for j in range(r):
29         matrix.append(list(input()))
30     start = find_start()
31     print(bfs(start, r, c))
```

Fence 1

代码运行截图 (至少包含有"Accepted")

#50892823提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
from collections import deque
def find_start():
    for j in range(r):
        for k in range(c):
            if matrix[j][k] == 'S':
                return j, k
```

基本信息

#: 50892823  
 题目: 07218  
 提交人: 25n2400011769  
 内存: 5464kB  
 时间: 89ms  
 语言: Python3  
 提交时间: 2025-11-18 16:48:01

Figure 1

## 1.2 M27925: 小组队列

dict, queue, <http://cs101.openjudge.cn/practice/27925/>

思路:

- 构建组员到组的单向图
- 构建双端链表, 把组放进去, 在用一个字典实现组到已排队的组员列表的映射
- 散客放进特殊的组, 加入散客时总在组列表中加入散客组

代码:

```
1  from collections import deque
2  def enqueue(pr):
3      if pr in team:
4          team_name=team[pr]
5      else:
6          que.append(-1)
7          member__in__que[-1].append(pr)
8      return
9      if team_name in member__in__que:
10         member__in__que[team_name].append(pr)
11     else:
12         member__in__que[team_name] = deque([pr])
13         que.append(team_name)
14
15 def de():
16     t=que[0]
17     if t== -1:
18         que.popleft()
19         return member__in__que[-1].popleft()
20     else:
21         res=member__in__que[t].popleft()
22         if len(member__in__que[t])==0:
23             que.popleft()
24             member__in__que.pop(t)
25         return res
```

```

26
27     t=int(input())
28     team={}
29     for i in range(t):
30         data=list(input().split())
31         for j in data:
32             team[j]=i
33
34     que=deque()
35     member__in__que={-1:deque([])}
36     while True:
37         demand=input()
38         if demand=='STOP':
39             break
40         if demand[:7]=='ENQUEUE':
41             enqueue(demand[8:])
42         if demand[:7]=='DEQUEUE':
43             print(de())

```

Fence 2

### 代码运行截图 (至少包含有"Accepted")

#50894455提交状态

[查看](#) [提交](#) [统计](#) [提问](#)

状态: **Accepted**

源代码

```

from collections import deque
def enqueue(pr):
    if pr in team:
        team_name=team[pr]
    else:
        que.append(-1)
        member__in__que[-1].append(pr)

```

基本信息

#: 50894455  
 题目: 27925  
 提交人: 25n2400011769  
 内存: 5244kB  
 时间: 100ms  
 语言: Python3  
 提交时间: 2025-11-18 18:31:07

Figure 2

## 1.3 M04089: 电话号码

trie, <http://cs101.openjudge.cn/practice/04089/>

思路:

- 利用字典构造前缀树

代码:

```

1     def add(nums,pr):
2         for k in range(len(nums)):
3             if nums[k] not in pr:
4                 pr[nums[k]] = {}
5                 pr=pr[nums[k]]
6             else:
7                 pr=pr[nums[k]]
8                 if pr=={}:

```

```

9         return False
10    if pr!={}:
11        return False
12    else:
13        return True
14    t=int(input())
15    for i in range(t):
16        n=int(input())
17        root={}
18        judge=True
19        for j in range(n):
20            nums=input()
21            if judge:
22                if not add(nums,root):
23                    print('NO')
24                    judge=False
25        if judge:
26            print('YES')

```

Fence 3

代码运行截图 (至少包含有"Accepted")

#50901680提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```

def add(nums,pr):
    for k in range(len(nums)):
        if nums[k] not in pr:
            pr[nums[k]] = {}
            pr=pr[nums[k]]
        else:
            pr=pr[nums[k]]

```

基本信息

#: 50901680  
 题目: 04089  
 提交人: 25n2400011769  
 内存: 15120kB  
 时间: 175ms  
 语言: Python3  
 提交时间: 2025-11-19 13:11:16

Figure 3

## 1.4 M3532.针对图的路径存在性查询I

disjoint set, <https://leetcode.cn/problems/path-existence-queries-in-a-graph-i/>

思路:

- 并查集

代码

```

1 class Solution:
2     def pathExistenceQueries(self, n: int, nums: List[int],
3                               maxDiff: int, queries: List[List[int]]) -> List[bool]:
4         class DisjointSet:
5             def __init__(self, n):
6                 self.root = [x for x in range(n)]
7                 self.rank = [0 for x in range(n)]

```

```
8         def find(self, x):
9             if self.root[x] != x:
10                 self.root[x] = self.find(self.root[x])
11             return self.root[x]
12
13         def union(self, x, y):
14             xroot = self.find(x)
15             yroot = self.find(y)
16             if xroot == yroot:
17                 return
18             if self.rank[xroot] < self.rank[yroot]:
19                 self.root[xroot] = yroot
20             elif self.rank[yroot] < self.rank[xroot]:
21                 self.root[yroot] = xroot
22             else:
23                 self.root[yroot] = xroot
24                 self.rank[xroot] += 1
25
26         disjointset = DisjointSet(n)
27         for i in range(1, n):
28             if nums[i] - nums[i - 1] <= maxDiff:
29                 disjointset.union(i, i - 1)
30         res = []
31         for u, v in queries:
32             if disjointset.find(u) == disjointset.find(v):
33                 res.append(True)
34             else:
35                 res.append(False)
36         return res
```

Fence 4

代码运行截图 (至少包含有"Accepted")



Figure 4

## 1.5 M19943: 图的拉普拉斯矩阵

OOP, graph, implementation, <http://cs101.openjudge.cn/pctbook/E19943/>

要求创建Graph, Vertex两个类，建图实现。

思路：

- 略

代码

```
1 class Vertex():
2     def __init__(self, name):
3         self.name = name
4         self.neighbors = {}
5
6     def connect(self, other, weight):
7         self.neighbors[other] = weight
8
9
10 class Graph():
11     def __init__(self, n):
12         self.vertices = {}
13         for i in range(n):
14             self.addVertex(i)
15     def addVertex(self, name):
16         vertex = Vertex(name)
17         self.vertices[name] = vertex
```

```

18
19     def connect(self, name1, name2, weight):
20         vertex1 = self.vertices[name1]
21         vertex2 = self.vertices[name2]
22         vertex1.connect(vertex2, weight)
23         vertex2.connect(vertex1, weight)
24     def build_laplace_matrix(self):
25         laplace_matrix = [[0 for _ in
26                             range(len(self.vertices))] for _ in
27                             range(len(self.vertices))]
28         for i in range(len(self.vertices)):
29             neighbors = self.vertices[i].neighbors
30             for neighbor in neighbors:
31                 laplace_matrix[i][i] += 1
32                 laplace_matrix[i][neighbor.name] -= 1
33         return '\n'.join(map(lambda x: '
34             '.join(map(str,x)), laplace_matrix))
35
36 n,m = map(int, input().split())
37 graph = Graph(n)
38 for i in range(m):
39     a,b= map(int, input().split())
40     graph.connect(a,b,0)
41 print(graph.build_laplace_matrix())

```

Fence 5

代码运行截图 (至少包含有"Accepted")

#50903398提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```

class Vertex():
    def __init__(self, name):
        self.name = name
        self.neighbors = {}

    def connect(self, other, weight):
        self.neighbors[other] = weight

```

基本信息

#: 50903398  
 题目: E19943  
 提交人: 25n2400011769  
 内存: 3712kB  
 时间: 30ms  
 语言: Python3  
 提交时间: 2025-11-19 14:19:24

Figure 5

## 1.6 T25353: 排队

<http://cs101.openjudge.cn/pctbook/T25353/>

思路:

- 先分组，再排序。指针存储当前的最大值（包括在组里的和未在组里的）和最小值（仅需要未在组里的），依次判断剩余元素是否可以入组。
- 类似quicksort的方法也可以解决，但是用时会长很多，显然这里分成的两部分会非常不均匀。

代码:

```

1  n,d=map(int,input().split())
2  height_list=[]
3  for i in range(n):
4      height_list.append(int(input()))
5  team_list=[]
6  hax=[False]*n
7  cnt=0
8  while cnt < n:
9      team_list.append([])
10
11     for i in range(n):
12         if hax[i]:
13             continue
14         if not team_list[-1]:
15             team_list[-1].append(height_list[i])
16             maxi=height_list[i]
17             mini=height_list[i]
18             cnt+=1
19             hax[i] = True
20             continue
21         if height_list[i] < maxi - d:
22             if height_list[i] < mini:
23                 mini = height_list[i]
24             continue
25         elif height_list[i] > mini + d:
26             if height_list[i] > maxi:
27                 maxi = height_list[i]
28             continue
29         else:
30             team_list[-1].append(height_list[i])
31             hax[i] = True
32             cnt += 1
33             if height_list[i] > maxi:
34                 maxi = height_list[i]
35 res=[]
36 for i in team_list:
37     i.sort()
38     res.extend(i)
39 print('\n'.join(map(str,res)))

```

Fence 6

quicksort

```

1  def quick_sort(d,num_list):
2      if not num_list:
3          return []
4      left=[]
5      right=[]
6      pr=num_list[0]

```



```

7     maxi=pr
8     mini=pr
9     for num in num_list[1:]:
10        if num < pr and maxi-num <= d and num-mini <= d:
11            left.append(num)
12        else:
13            right.append(num)
14            if num > maxi:
15                maxi = num
16            if num < mini:
17                mini = num
18        return quick_sort(d,left)+[pr]+quick_sort(d,right)
19
20    n,d=map(int,input().split())
21    num_list=[]
22    for i in range(n):
23        num_list.append(int(input()))
24    print('\n'.join(map(str,quick_sort(d,num_list))))

```

Fence 7

### 代码运行截图 (至少包含有"Accepted")

#50906593提交状态

[查看](#)
[提交](#)
[统计](#)
[提问](#)

状态: Accepted

源代码

```

n,d=map(int,input().split())
height_list=[]
for i in range(n):
    height_list.append(int(input()))
team_list=[]
hax=[False]*n
cnt=0

```

基本信息

#: 50906593  
 题目: T25353  
 提交人: 25n2400011769  
 内存: 18100kB  
 时间: 276ms  
 语言: Python3  
 提交时间: 2025-11-19 17:00:57

Figure 6

#50906840提交状态

[查看](#)
[提交](#)
[统计](#)
[提问](#)

状态: Accepted

源代码

```

def quick_sort(d,num_list):
    if not num_list:
        return []
    if len(num_list) == 1:
        return num_list
    left=[]
    right=[]

```

基本信息

#: 50906840  
 题目: T25353  
 提交人: 25n2400011769  
 内存: 54420kB  
 时间: 1605ms  
 语言: Python3  
 提交时间: 2025-11-19 17:08:13

Figure 7

## 2. 2. 学习总结和个人收获

本周练习了一些图的题目，部分题目与树的部分有很一定关联，做起来相对容易。还有一些题目涉及一些图的算法，诸如拓扑排序，最短路径，这些题目的算法还没有掌握，做起来有一定困难。