

Assignment #A: Graph starts

Updated 1830 GMT+8 Apr 22, 2025

2025 spring, Compiled by 郑涵予 物理学院

说明:

1. 解题与记录:

对于每一个题目，请提供其解题思路（可选），并附上使用Python或C++编写的源代码（确保已在OpenJudge, Codeforces, LeetCode等平台上获得Accepted）。请将这些信息连同显示“Accepted”的截图一起填写到下方的作业模板中。（推荐使用Typora <https://typoraio.cn> 进行编辑，当然你也可以选择Word。）无论题目是否已通过，请标明每个题目大致花费的时间。

2. **提交安排**：提交时，请首先上传PDF格式的文件，并将.md或.doc格式的文件作为附件上传至右侧的“作业评论”区。确保你的Canvas账户有一个清晰可见的头像，提交的文件为PDF格式，并且“作业评论”区包含上传的.md或.doc附件。

3. **延迟提交**：如果你预计无法在截止日期前提交作业，请提前告知具体原因。这有助于我们了解情况并可能为你提供适当的延期或其他帮助。

请按照上述指导认真准备和提交作业，以保证顺利完成课程要求。

1. 题目

M19943:图的拉普拉斯矩阵

OOP, implementation, <http://cs101.openjudge.cn/practice/19943/>

要求创建Graph, Vertex两个类，建图实现。

思路：

直接实现就行，练习OOP写法（用时约10min）

代码：

```
class Vertex:
    def __init__(self,n):
        self.neighbors=[0]*n
        self.degree=0

class Graph:
    def __init__(self,n):
```

```

self.vertices=[Vertex(n) for _ in range(n)]

def add_edge(self,a,b):
    self.vertices[a].neighbors[b]=1
    self.vertices[b].neighbors[a] = 1
    self.vertices[a].degree+=1
    self.vertices[b].degree+=1

n,m=map(int,input().split())
g=Graph(n)
for _ in range(m):
    a,b=map(int,input().split())
    g.add_edge(a,b)
for i in range(n):
    for j in range(n):
        if i==j:
            print(g.vertices[i].degree-g.vertices[i].neighbors[j],end=" ")
        else:
            print(-g.vertices[i].neighbors[j],end=" ")
    print()

```

代码运行截图 (至少包含有"Accepted")

#48993398提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```

class Vertex:
    def __init__(self,n):
        self.neighbors=[0]*n
        self.degree=0

class Graph:
    def __init__(self,n):
        self.vertices=[Vertex(n) for _ in range(n)]

    def add_edge(self,a,b):
        self.vertices[a].neighbors[b]=1
        self.vertices[b].neighbors[a] = 1
        self.vertices[a].degree+=1
        self.vertices[b].degree+=1

n,m=map(int,input().split())
g=Graph(n)
for _ in range(m):
    a,b=map(int,input().split())
    g.add_edge(a,b)
for i in range(n):
    for j in range(n):
        if i==j:
            print(g.vertices[i].degree-g.vertices[i].neighbors[j],end=" ")
        else:

```

基本信息

#: 48993398
 题目: 19943
 提交人: 24n2400011325
 内存: 3648kB
 时间: 27ms
 语言: Python3
 提交时间: 2025-04-23 16:40:26

LC78.子集

backtracking, <https://leetcode.cn/problems/subsets/>

思路:

标准的回溯模板题 (用时约2min)

代码:

```

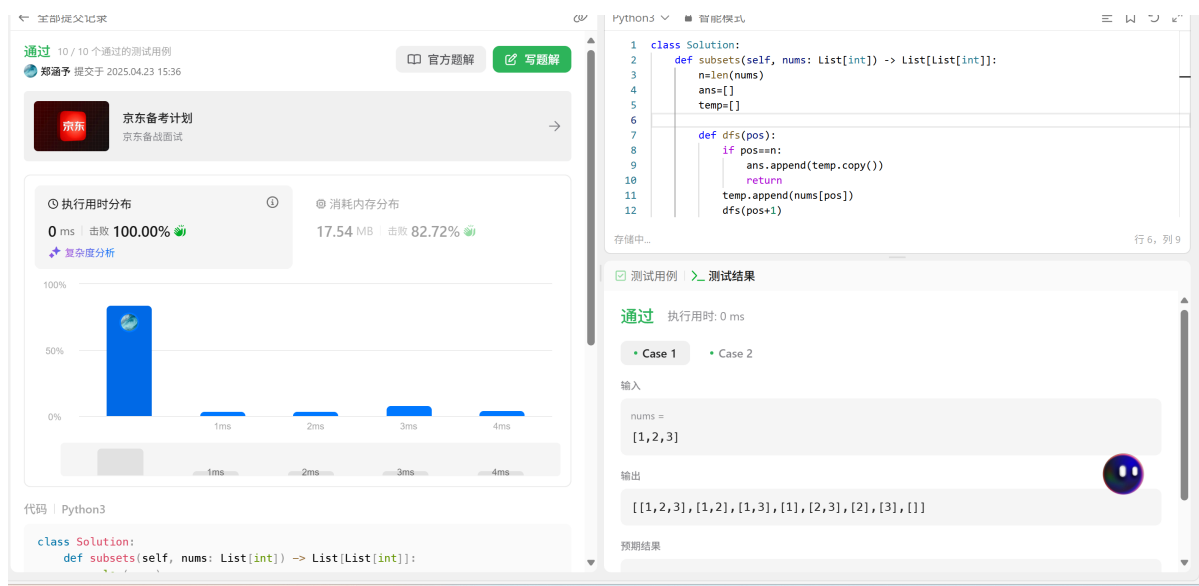
class Solution:
    def subsets(self, nums: List[int]) -> List[List[int]]:
        n=len(nums)
        ans=[]
        temp=[]

        def dfs(pos):
            if pos==n:
                ans.append(temp.copy())
                return
            temp.append(nums[pos])
            dfs(pos+1)
            temp.pop()
            dfs(pos+1)

        dfs(0)
        return ans

```

代码运行截图 (至少包含有"Accepted")



LC17.电话号码的字母组合

hash table, backtracking, <https://leetcode.cn/problems/letter-combinations-of-a-phone-number/>

思路:

同样是直接回溯即可, 可以用一个字典存一下每个数字对应的字母。(用时约5min)

代码:

```

class Solution:
    def letterCombinations(self, digits: str) -> List[str]:
        if not digits:
            return []
        phone = {
            '2': 'abc',
            '3': 'def',
            '4': 'ghi',
            '5': 'jkl',
            '6': 'mno',
            '7': 'pqrs',
            '8': 'tuv',
            '9': 'wxyz'
        }
        res = []
        def dfs(index, path):
            if index == len(digits):
                res.append(path)
                return
            for letter in phone[digits[index]]:
                dfs(index+1, path+letter)
        dfs(0, '')
        return res

```

```

class Solution:
    def letterCombinations(self, digits: str) -> List[str]:
        if not digits:
            return []

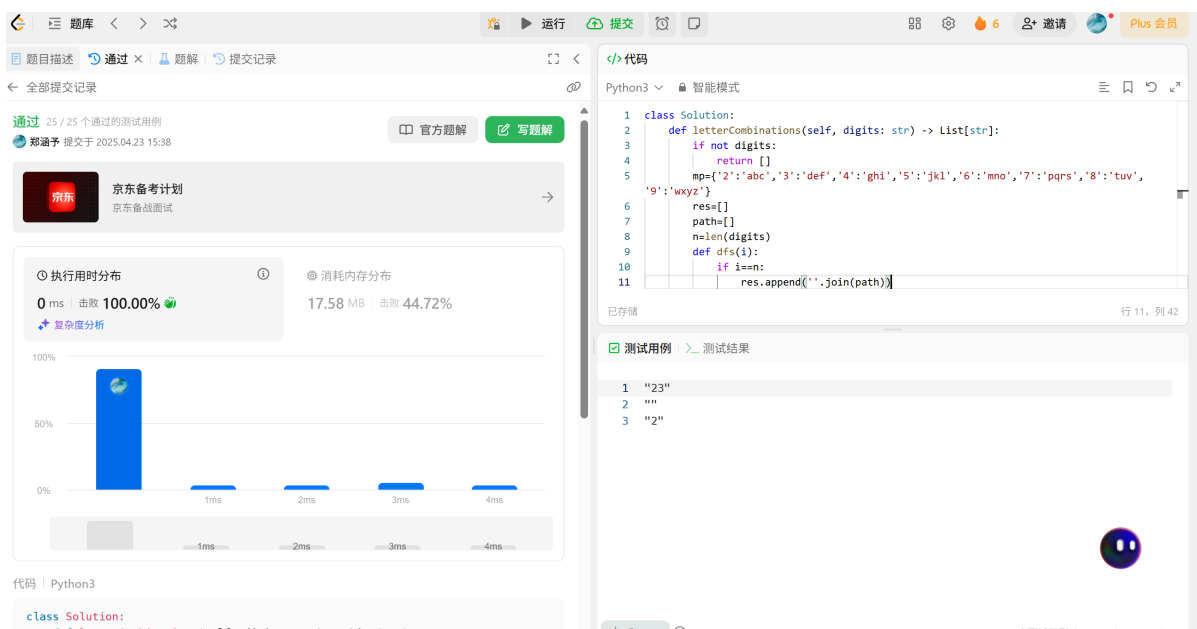
        mp=
        {'2': 'abc', '3': 'def', '4': 'ghi', '5': 'jkl', '6': 'mno', '7': 'pqrs', '8': 'tuv', '9': 'wxyz'}

        res=[]
        path=[]
        n=len(digits)
        def dfs(i):
            if i==n:
                res.append(''.join(path))
                return
            for x in mp[digits[i]]:
                path.append(x)
                dfs(i+1)
                path.pop()

        dfs(0)
        return res

```

代码运行截图 (至少包含有"Accepted")



M04089:电话号码

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

思路:

利用字典树可以很好的解决。需要注意的是不仅仅要考虑新增的电话号码是已存在电话号码的前缀，还要考虑已存在电话号码是新插入电话号码的前缀的情况。可以在插入时判断有没有新增节点来处理这个问题，也可以直接把数据读入后排序。（用时约15min）

代码：

```
delta=False
class Trie:
    def __init__(self):
        self.children=[None]*10
        self.isEnd = False

    def insert(self,word):
        global delta
        is_create=False
        p=self
        for x in word:
            x=int(x)
            if p.isEnd:
                delta=True
            if p.children[x] is None:
                is_create=True
                p.children[x]=Trie()
            p=p.children[x]
        p.isEnd=True
        if not is_create:
            delta=True

t=int(input())
for _ in range(t):
    n=int(input())
    root=Trie()
    delta=False
    for __ in range(n):
        s=input()
        root.insert(s)
    print('YES' if not delta else 'NO')
```

代码运行截图 (至少包含有"Accepted")

状态: Accepted

源代码

```
delta=False
class Trie:
    def __init__(self):
        self.children=[None]*10
        self.isEnd = False

    def insert(self,word):
        global delta
        is_create=False
        p=self
        for x in word:
            x=int(x)
            if p.isEnd:
                delta=True
            if p.children[x] is None:
                is_create=True
                p.children[x]=Trie()
            p=p.children[x]
        p.isEnd=True
        if not is_create:
            delta=True

t=int(input())
for _ in range(t):
    n=int(input())
    root=Trie()
    delta=False
    for _ in range(n):
        s=input()
```

基本信息

#: 48992802
题目: 04089
提交人: 24n2400011325
内存: 20252kB
时间: 473ms
语言: Python3
提交时间: 2025-04-23 16:04:01

T28046:词梯

bfs, <http://cs101.openjudge.cn/practice/28046/>

思路:

最大的难点在于建图。直接两两比较肯定会超时，看了课件后知道可以使用桶来排序,写起来就比较顺畅了。之后的bfs就是模板，比较简单。（用时约25min）

代码:

```
from collections import deque,defaultdict
n=int(input())
a=[[[] for _ in range(n)]
s=[]
pre=[-1]*n
buckets=defaultdict(list)
for i in range(n):
    s.append(input())
    for j in range(4):
        buckets[s[i][:j]+'.'+s[i][j+1:]].append(i)
for bucket in buckets.values():
    for i in range(len(bucket)):
        for j in range(i+1,len(bucket)):
            a[bucket[i]].append(bucket[j])
            a[bucket[j]].append(bucket[i])
x,y=input().split()
u,v=-1,-1
for i in range(n):
```

```

    if s[i]==x:
        u=i
    elif s[i]==y:
        v=i

def bfs(start,end):
    delta=False
    visited=[False]*n
    q=deque()
    q.append(start)
    visited[start]=True
    while q:
        p=q.popleft()
        if p==end:
            delta=True
            break
        for pos in a[p]:
            if visited[pos]:
                continue
            visited[pos] = True
            q.append(pos)
            pre[pos]=p
    if not delta:
        return 'NO'
    path=[]
    while end!=-1:
        path.append(s[end])
        end=pre[end]
    return ' '.join(path[::-1])

print(bfs(u,v))

```

代码运行截图 (至少包含有"Accepted")



#48994369提交状态

查看

提交

统计

提问

状态: Accepted

源代码

```
from collections import deque, defaultdict
n=int(input())
a=[] for _ in range(n):
    s=[]
    pre=[-1]*n
    buckets=defaultdict(list)
    for i in range(n):
        s.append(input())
        for j in range(4):
            buckets[s[i][j]+'.'+s[i][j+1:]].append(i)
    for bucket in buckets.values():
        for i in range(len(bucket)):
            for j in range(i+1, len(bucket)):
                a[bucket[i]].append(bucket[j])
                a[bucket[j]].append(bucket[i])
    x,y=input().split()
    u,v=-1,-1
    for i in range(n):
        if s[i]==x:
            u=i
        elif s[i]==y:
```

基本信息

#: 48994369

题目: 28046

提交人: 24n2400011325

内存: 6296kB

时间: 53ms

语言: Python3

提交时间: 2025-04-23 17:25:49

T51.N皇后

backtracking, <https://leetcode.cn/problems/n-queens/>

思路:

这应该也是经典试题了, 直接回溯搜索就行 (用时约10min)

代码:

```
class Solution:
    def solveNQueens(self, n: int) -> List[List[str]]:
        ans=[]
        a=[['.']*n for _ in range(n)]
        col=[False]*n

        def check(x,y):
            if col[y]:
                return False
            i,j=x-1,y-1
            while i>=0 and j>=0:
                if a[i][j]=='Q':
                    return False
                i-=1
                j-=1
            i,j=x-1,y+1
            while i>=0 and j<n:
                if a[i][j]=='Q':
                    return False
                i-=1
```


题目描述

通过

题解

提交记录

全部提交记录

通过

9 / 9 个通过的测试用例

郑潇予 提交于 2025.04.23 17:42

京东

京东备考计划

京东备战图试


执行用时分布

19 ms | 击败 45.15%

复杂度分析

消耗内存分布

17.97 MB | 击败 56.77%



代码 | Python3

```
class Solution:
```

</> 代码

Python3

智能模式

```
while i>=0 and j<=0:
    if a[i][j]=='Q':
        return False
    i-=1
    j-=1
i,j=x-1,y+1
while i>=0 and j<n:
    if a[i][j]=='Q':
        return False
    i-=1
    j+=1
return True
```

已存储

测试用例

测试结果

通过

执行用时: 0 ms

Case 1

Case 2

输入

n =

4

输出

["..Q..","...Q","Q...",".Q.."],["..Q..","Q...",".Q.."]

预期结果

这周又参加了力扣的周赛，发现力扣真的好喜欢考线段树，对着zkw线段树的模板还是能在比赛时间内调完代码的。现在开始慢慢整理期末考的cheat sheet了，希望五月的月考和期末机考都能发挥好。