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：

代码：
```
n,m=map(int,input().split())
d=[[0 for i in range(n)]for j in range(n)]
a=[[0 for i in range(n)]for j in range(n)]
op=[[0 for i in range(n)]for j in range(n)]
dic={}
for i in range(n):
    dic[i]=0
for p in range(m):
    l=list(map(int,input().split()))
    for k in l:
        dic[k]+=1
    a[l[0]][l[1]]=1
    a[l[1]][l[0]]=1
for i in range(n):
    d[i][i]=dic[i]
for i in range(n):
    s=''
    for j in range(n):
        s+=str(d[i][j]-a[i][j])+' '
    print(s[:-1])
```
代码运行截图 （至少包含有"Accepted"）

LC78.子集
backtracking, https://leetcode.cn/problems/subsets/

思路:

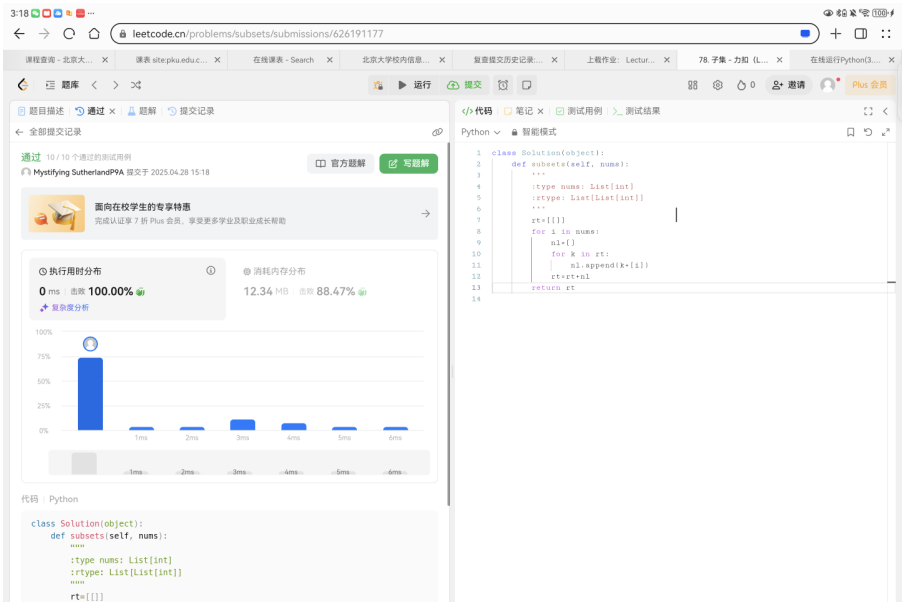
代码:

```
class Solution(object):

    def subsets(self, nums):

        """

        :type nums: List[int]

        :rtype: List[List[int]]

        """

        rt=[[]]

        for i in nums:

            nl=[]

            for k in rt:

                nl.append(k+[i])
```

rt=rt+nl

return rt

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/

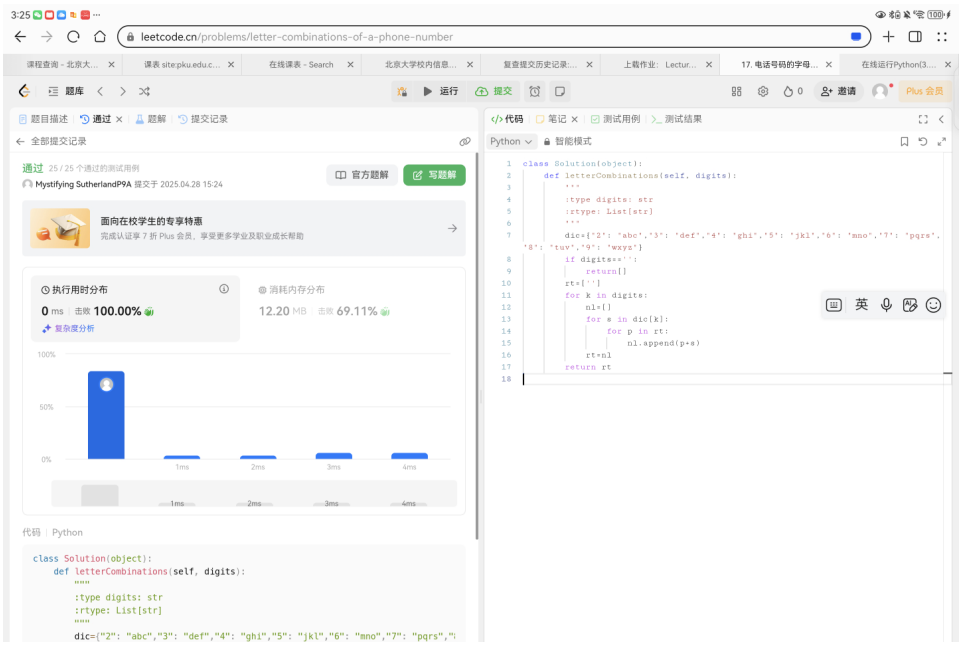
思路:

代码:

```
class Solution(object):

    def letterCombinations(self, digits):

        """

        :type digits: str

        :rtype: List[str]

        """

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

        if digits=='':

            return[]
```

```
        rt=['']
        for k in digits:
            nl=[]
            for s in dic[k]:
                for p in rt:
                    nl.append(p+s)
            rt=nl
        return rt
```

代码运行截图 （至少包含有"Accepted"）



M04089:电话号码
trie, http://cs101.openjudge.cn/practice/04089/

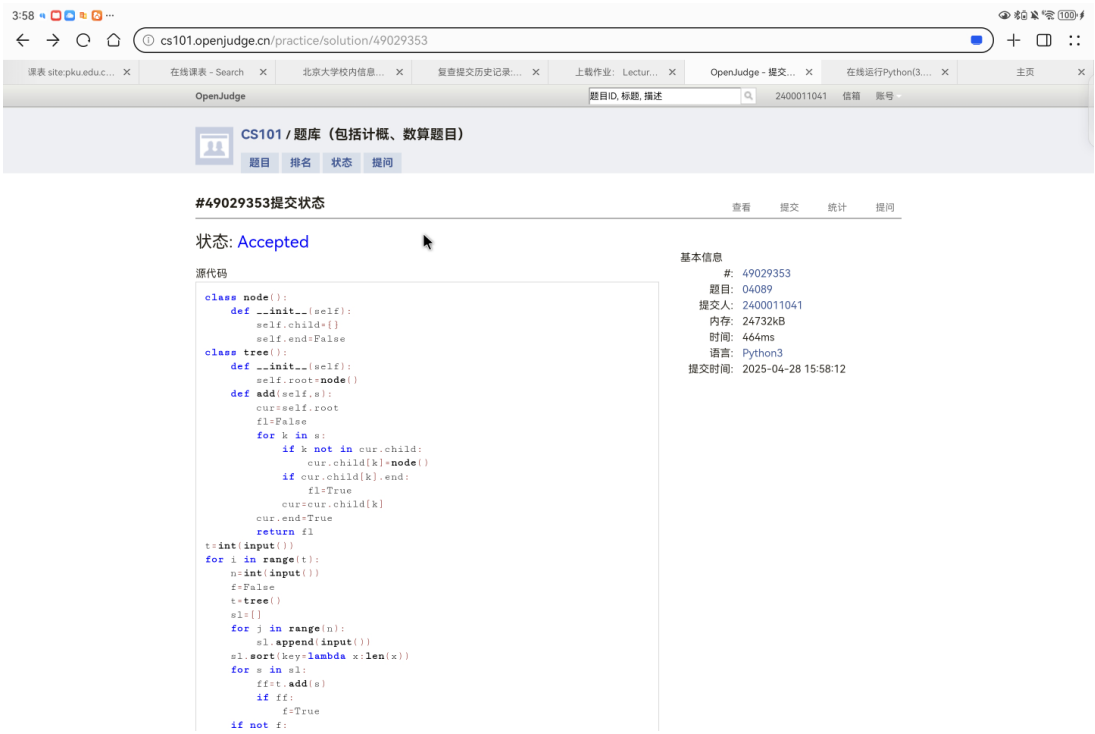
思路:

代码:

```
class node():
    def __init__(self):
        self.child={}
```

```
            self.end=False
class tree():
    def __init__(self):
        self.root=node()
    def add(self,s):
        cur=self.root
        fl=False
        for k in s:
            if k not in cur.child:
                cur.child[k]=node()
            if cur.child[k].end:
                fl=True
            cur=cur.child[k]
        cur.end=True
        return fl
t=int(input())
for i in range(t):
    n=int(input())
    f=False
    t=tree()
    sl=[]
    for j in range(n):
        sl.append(input())
    sl.sort(key=lambda x:len(x))
    for s in sl:
        ff=t.add(s)
        if ff:
            f=True
    if not f:
        print('YES')
    else:
        print('NO')
```

代码运行截图 （至少包含有"Accepted"）



T28046:词梯

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

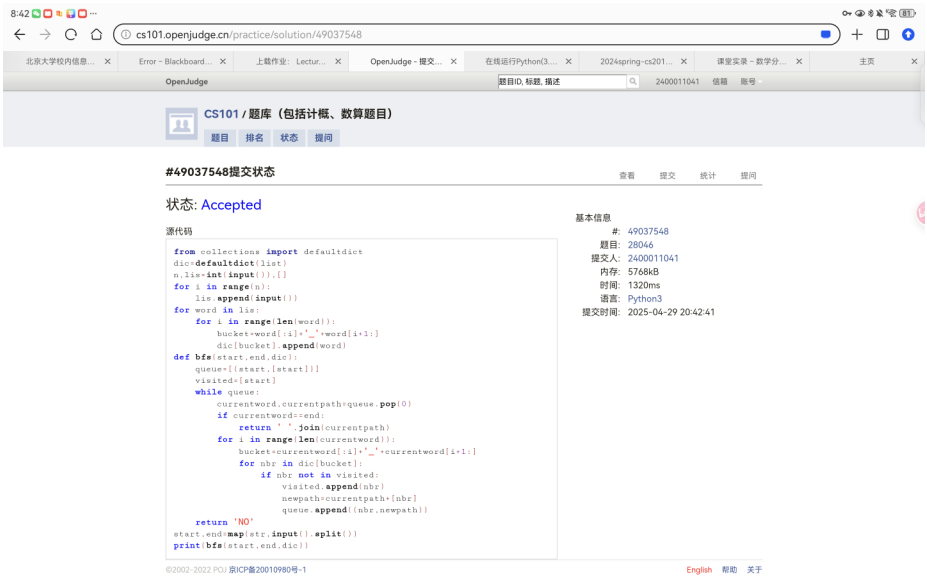
思路:

代码:

```
from collections import defaultdict
dic=defaultdict(list)
n,lis=int(input()),[]
for i in range(n):
    lis.append(input())
for word in lis:
    for i in range(len(word)):
        bucket=word[:i]+'_'+word[i+1:]
        dic[bucket].append(word)
def bfs(start,end,dic):
    queue=[(start,[start])]
    visited=[start]
    while queue:
        currentword,currentpath=queue.pop(0)
        if currentword==end:
            return ' '.join(currentpath)
        for i in range(len(currentword)):
            bucket=currentword[:i]+'_'+currentword[i+1:]
            for nbr in dic[bucket]:
                if nbr not in visited:
```

```
                    visited.append(nbr)
                    newpath=currentpath+[nbr]
                    queue.append((nbr,newpath))
    return 'NO'
start,end=map(str,input().split())
print(bfs(start,end,dic))
```

代码运行截图 （至少包含有"Accepted"）



T51.N 皇后

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

思路:

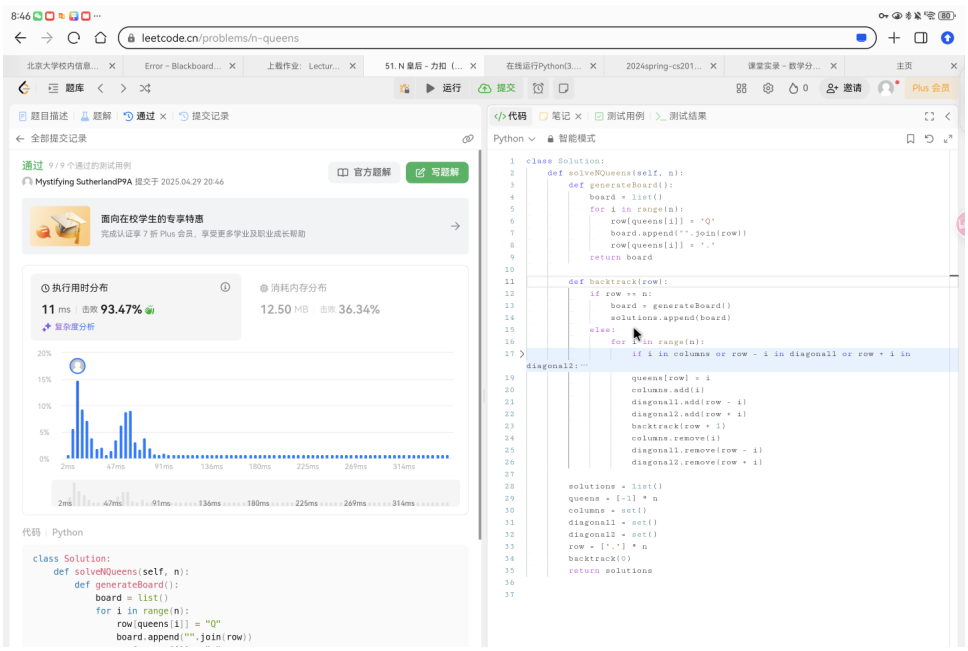
代码:

```
class Solution:

    def solveNQueens(self, n):

        def generateBoard():

            board = list()

            for i in range(n):

                row[queens[i]] = "Q"

                board.append("".join(row))

                row[queens[i]] = "."
```

```
        return board


def backtrack(row):
        if row == n:
                board = generateBoard()
                solutions.append(board)
        else:
                for i in range(n):
                        if i in columns or row - i in diagonal1 or row + i in diagonal2:
                                continue
                        queens[row] = i
                        columns.add(i)
                        diagonal1.add(row - i)
                        diagonal2.add(row + i)
                        backtrack(row + 1)
                        columns.remove(i)
                        diagonal1.remove(row - i)
                        diagonal2.remove(row + i)


solutions = list()
queens = [-1] * n
columns = set()
```

```
        diagonal1 = set()

        diagonal2 = set()

        row = ["."] * n

        backtrack(0)

        return solutions
```

代码运行截图 （至少包含有"Accepted"）



2. 学习总结和收获

词梯一题较难，参考了答案。本次作业让我对图的理解更加深刻，也熟悉了它的结构性打法。

如果发现作业题目相对简单，有否寻找额外的练习题目，如“数算 2025spring 每日选做”、LeetCode、Codeforces、洛谷等网站上的题目。

还在期中考，选做暂停