Assignment #B: 图为主

Updated 2223 GMT+8 Apr 29, 2025

2025 spring, Complied by **袁奕 数院 2400010766**

> **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

E07218:献给阿尔吉侬的花束

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

思路：

代码：

```
from collections import deque

graph, visited = [], set()
dir = [[0, 1], [1, 0], [0, -1], [-1, 0]]

def isvalid(nx, ny):
    return 0 <= nx < len(graph) and\
        0 <= ny < len(graph[0]) and\
        graph[nx][ny] != "#"

def bfs(start, end):
    que, visited = deque([(0,) + start]), {start}
    while que:
        step, lx, ly = que.popleft()
        if (lx, ly) == end:
            return step
```

```
        for dx, dy in dir:
            nx, ny = lx + dx, ly + dy
            if (nx, ny) not in visited and\
                isValid(nx, ny):
                que.append((step + 1, nx, ny))
                visited.add((nx, ny))
    return

def find(pat):
    for i in range(len(graph)):
        for j in range(len(graph[0])):
            if graph[i][j] == pat:
                return i, j

T = int(input())

for _ in range(T):
    graph, visited = [], set()
    m, n = map(int, input().split())
    for _ in range(m):
        graph.append(list(input()))
    start, end = find("S"), find("E")
    step = bfs(start, end)
    print(step) if step else print("oop!")
```

#49039326提交状态 查看 提交 统计 提问

状态：Accepted

源代码

```
from collections import deque

graph, visited = [], set()
dir = [[0, 1], [1, 0], [0, -1], [-1, 0]]

def isValid(nx, ny):
    return 0 <= nx < len(graph) and\
        0 <= ny < len(graph[0]) and\
        graph[nx][ny] != "#"

def bfs(start, end):
    que, visited = deque([(0,) + start]), {start}
    while que:
        step, lx, ly = que.popleft()
        if (lx, ly) == end:
            return step
        for dx, dy in dir:
            nx, ny = lx + dx, ly + dy
            if (nx, ny) not in visited and\
                isValid(nx, ny):
                que.append((step + 1, nx, ny))
                visited.add((nx, ny))
    return

def find(pat):
    for i in range(len(graph)):
        for j in range(len(graph[0])):
            if graph[i][j] == pat:
                return i, j

T = int(input())

for _ in range(T):
    graph, visited = [], set()
    m, n = map(int, input().split())
    for _ in range(m):
        graph.append(list(input()))
    start, end = find("S"), find("E")
    step = bfs(start, end)
    print(step) if step else print("oop!")
```

基本信息

#: 49039326
题目: 07218
提交人: 24n2400010766
内存: 5796kB
时间: 88ms
语言: Python3
提交时间: 2025-04-30 09:34:56

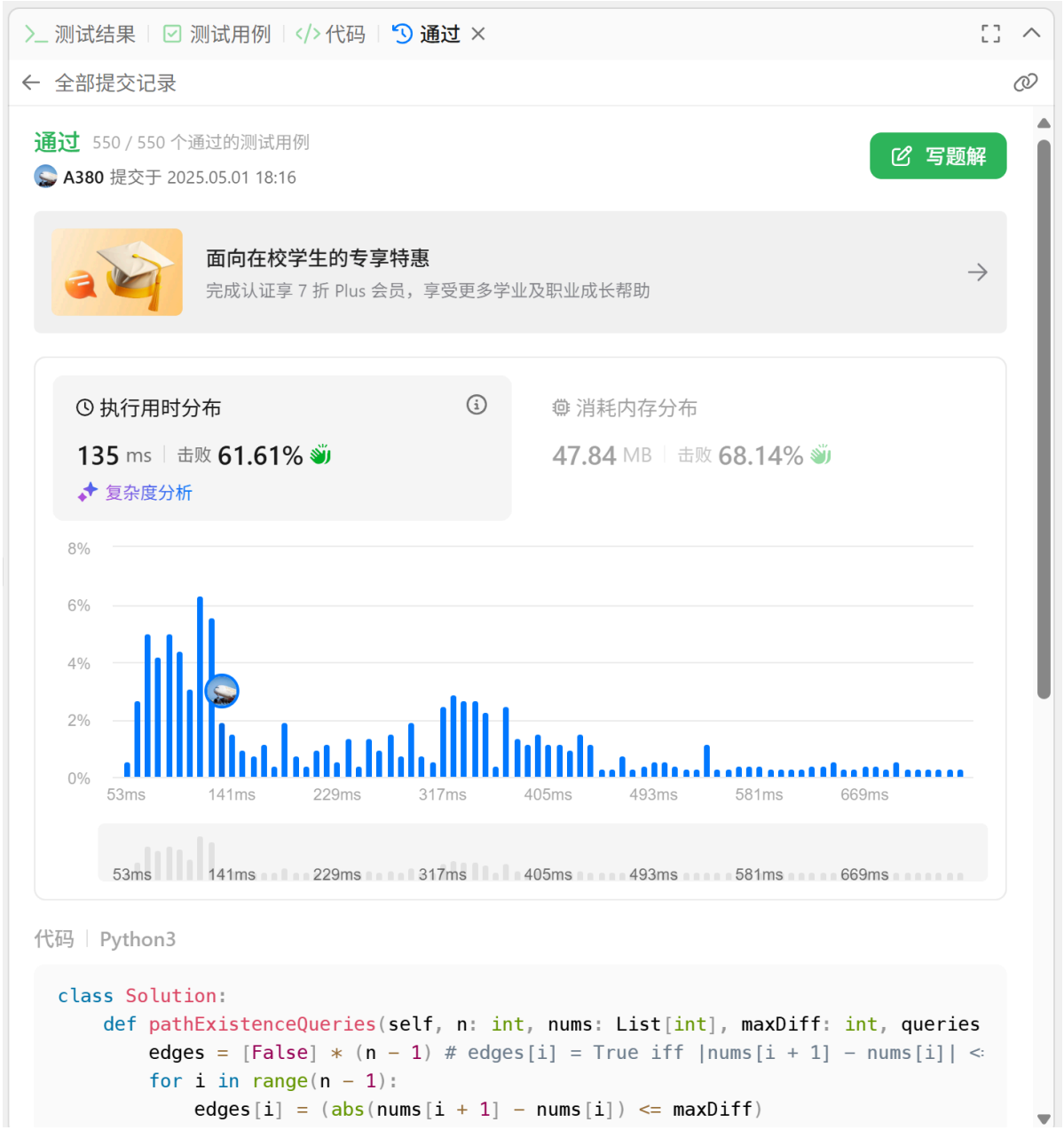
M3532.针对图的路径存在性查询I

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

思路：

代码：

```python
class Solution:
    def pathExistenceQueries(self, n: int, nums: List[int], maxDiff: int, queries: List[List[int]]) -> List[bool]:
        edges = [False] * (n - 1) # edges[i] = True iff |nums[i + 1] - nums[i]| <= maxDiff
        for i in range(n - 1):
            edges[i] = (abs(nums[i + 1] - nums[i]) <= maxDiff)
        prefix = [0]
        for i in range(n - 1):
            new = prefix[-1] + 1 if edges[i] else prefix[-1]
            prefix.append(new)
        res = []
        for u, v in queries:
            res.append(abs(u - v) == abs(prefix[u] - prefix[v]))
        return res
```



M22528:厚道的调分方法

binary search, http://cs101.openjudge.cn/practice/22528/

思路：

代码：

```python
nums = sorted(list(map(float, input().split())))
x = nums[(2 * len(nums)) // 5]

def isValid(b):
    a = float(b / 1000000000.0)
    return a * x + 1.1 ** (a * x) >= 85

low, high = 1, 1000000000

while low + 1 < high:
    mid = (low + high) // 2
    if isValid(mid):
        high = mid
    else:
        low = mid

if isValid(high):
    low = high

print(low)
```

#49040600提交状态

查看 提交 统计 提问

状态：Accepted

源代码

```python
nums = sorted(list(map(float, input().split())))
x = nums[(2 * len(nums)) // 5]

def isValid(b):
    a = float(b / 1000000000.0)
    return a * x + 1.1 ** (a * x) >= 85

low, high = 1, 1000000000

while low + 1 < high:
    mid = (low + high) // 2
    if isValid(mid):
        high = mid
    else:
        low = mid

if isValid(high):
    low = high

print(low)
```

基本信息

#:	49040600
题目:	22528
提交人:	24n2400010766
内存:	17976kB
时间:	92ms
语言:	Python3
提交时间:	2025-04-30 15:41:25

English 帮助 关于

Msy382: 有向图判环

dfs, https://sunnywhy.com/sfbj/10/3/382

思路:

代码:

```python
from collections import deque

n, m = map(int, input().split())
neighbour = [set() for _ in range(n)]
for _ in range(m):
    u, v = map(int, input().split())
    neighbour[u].add(v)

def bfs(start):
    dis = [-1] * n
    dis[start] = 0
    que = deque([(start, None)])
    while que:
        v, u = que.popleft() # edge : u -> v
        for w in neighbour[v]:
            if dis[w] == -1:
                que.append((w, v))
                dis[w] = dis[v] + 1
            elif w == start:
                return True
    return False

def isCycle():
    for i in range(n):
        if bfs(i):
            return True
    return False

print("Yes") if isCycle() else print("No")
```

代码书写 Python

```python
from collections import deque

n, m = map(int, input().split())
neighbour = [set() for _ in range(n)]
for _ in range(m):
    u, v = map(int, input().split())
    neighbour[u].add(v)

def bfs(start):
    dis = [-1] * n
    dis[start] = 0
    que = deque([(start, None)])
    while que:
        v, u = que.popleft() # edge : u -> v
        for w in neighbour[v]:
            if dis[w] == -1:
                que.append((w, v))
                dis[w] = dis[v] + 1
            elif w == start:
                return True
    return False

def isCycle():
    for i in range(n):
        if bfs(i):
            return True
```

测试输入 历史提交

提交时间	结果	时长(ms)	语言	
2025-05-01 18:44:08	完美通过	0	Python	查看

收起面板 运行 提交

M05443:兔子与樱花

Dijkstra, http://cs101.openjudge.cn/practice/05443/

思路:

代码:

```python
from heapq import heappush, heappop

class Vertex:
```

```python
    def __init__(self, name):
        self.name = name
        self.neighbour = {}

class Graph:
    def __init__(self):
        self.vertices = {}
    def edge(self, start : str, end : str, dis):
        start = self.vertices[start]
        end = self.vertices[end]
        start.neighbour[end] = dis
        end.neighbour[start] = dis

graph = Graph()

def dijkstra(start, end):
    que = [(0, [start], start)]
    while que:
        length, path, s = heappop(que)
        if s == end:
            return path
        for node, dis in graph.vertices[s].neighbour.items():
            heappush(que, (length + dis, path + [node.name], node.name))

V = int(input())
for _ in range(V):
    s = input()
    graph.vertices[s] = Vertex(s)

E = int(input())
for _ in range(E):
    start, end, dis = input().split()
    graph.edge(start, end, int(dis))

T = int(input())
for _ in range(T):
    start, end = input().split()
    path = dijkstra(start, end)
    for i in range(len(path) - 1):
        s, e = graph.vertices[path[i]], graph.vertices[path[i + 1]]
        print(f"{s.name}->({s.neighbour[e]})->", end="")
    print(path[-1])
```

#49039646提交状态

查看 提交 统计 提问

状态: Accepted

源代码

```
from heapq import heappush, heappop

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

class Graph:
    def __init__(self):
        self.vertices = {}
    def edge(self, start : str, end : str, dis):
        start = self.vertices[start]
        end = self.vertices[end]
        start.neighbour[end] = dis
        end.neighbour[start] = dis

graph = Graph()

def dijkstra(start, end):
    que = [(0, [start], start)]
    while que:
        length, path, s = heappop(que)
        if s == end:
            return path
        for node, dis in graph.vertices[s].neighbour.items():
            heappush(que, (length + dis, path + [node.name], node.name))

V = int(input())
for _ in range(V):
    s = input()
    graph.vertices[s] = Vertex(s)

E = int(input())
for _ in range(E):
    start, end, dis = input().split()
    graph.edge(start, end, int(dis))

T = int(input())
for _ in range(T):
    start, end = input().split()
    path = dijkstra(start, end)
    for i in range(len(path) - 1):
        s, e = graph.vertices[path[i]], graph.vertices[path[i + 1]]
        print(f"{s.name}->({s.neighbour[e]})->", end="")
    print(path[-1])
```

基本信息

#: 49039646
题目: 05443
提交人: 24n2400010766
内存: 3584kB
时间: 20ms
语言: Python3
提交时间: 2025-04-30 11:04:59

English 帮助 关于

T28050: 骑士周游

dfs, http://cs101.openjudge.cn/practice/28050/

思路:

代码:

```
dir = [(2, 1), (1, 2), (-1, 2), (-2, 1),
       (-2, -1), (-1, -2), (1, -2), (2, -1)]

def isValid(r, c):
    return 0 <= r < n and 0 <= c < n

def knight_tour(n, sr, sc):
    board = [[-1]*n for _ in range(n)]
```

```
    board[sr][sc] = 0
    def dfs(step, r, c):
        if step == n*n - 1:
            return True
        candidates = []
        for dr, dc in dir:
            nr, nc = r + dr, c + dc
            if isValid(nr, nc) and board[nr][nc] == -1:
                cnt = 0
                for dr2, dc2 in dir:
                    tr, tc = nr + dr2, nc + dc2
                    if isValid(tr, tc) and board[tr][tc] == -1:
                        cnt += 1
                candidates.append((cnt, nr, nc))
        candidates.sort()
        for _, nr, nc in candidates:
            board[nr][nc] = step + 1
            if dfs(step + 1, nr, nc):
                return True
            board[nr][nc] = -1
        return False
    return dfs(0, sr, sc)

n = int(input())
sr, sc = map(int, input().split())
print("success" if knight_tour(n, sr, sc) else "fail")
```

#49046518提交状态

查看 提交 统计 提问

状态: Accepted

基本信息

#: 49046518
题目: 28050
提交人: 24n2400010766
内存: 3972kB
时间: 27ms
语言: Python3
提交时间: 2025-05-01 21:38:45

源代码

```
dir = [(2, 1), (1, 2), (-1, 2), (-2, 1),
       (-2, -1), (-1, -2), (1, -2), (2, -1)]

def isValid(r, c):
    return 0 <= r < n and 0 <= c < n

def knight_tour(n, sr, sc):
    board = [[-1]*n for _ in range(n)]
    board[sr][sc] = 0
    def dfs(step, r, c):
        if step == n*n - 1:
            return True
        candidates = []
        for dr, dc in dir:
            nr, nc = r + dr, c + dc
            if isValid(nr, nc) and board[nr][nc] == -1:
                cnt = 0
                for dr2, dc2 in dir:
                    tr, tc = nr + dr2, nc + dc2
                    if isValid(tr, tc) and board[tr][tc] == -1:
                        cnt += 1
                candidates.append((cnt, nr, nc))
        candidates.sort()
        for _, nr, nc in candidates:
            board[nr][nc] = step + 1
            if dfs(step + 1, nr, nc):
                return True
            board[nr][nc] = -1
        return False
    return dfs(0, sr, sc)

n = int(input())
sr, sc = map(int, input().split())
print("success" if knight_tour(n, sr, sc) else "fail")
```

2. 学习总结和收获

1. `set`本身 *不可哈希*，所以 **不能**当作键；如果真要用集合作键，可用 `frozenset`（不可变集合）。
2. 感觉Msy382: 有向图判环 的数据太弱了，于是做了[2608. 图中的最短环 - 力扣（LeetCode）]