Assignment #A: 递归、田忌赛马

Updated 2355 GMT+8 Nov 4, 2025

2025 fall, Complied by 郑文萱 基础医学院

> **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本人头像，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

M018160: 最大连通域面积

dfs similar, http://cs101.openjudge.cn/pctbook/M18160

思路：这个题很像之前做过的晶矿的个数，不过是八个方向的，存储八个方向的题之前也做过，所以最主要练习的就是将他们和dfs结合，弄了两个矩阵一个储存一个标记。写的过程中发现函数很多的时候确实设置一个主函数会很方便（之前从来没设置主函数过），还有列表可以直接元组解包所以可以非常方便的读取方向。

代码

```
def dfs(x,y,panel,visited,n,m):
    if x<0 or x>=n or y<0 or y>=m:
        return 0
    if visited[x][y] or panel[x][y]!='W':
        return 0
    visited[x][y]=True
    area=1
    directions=[[1,1],[1,0],[1,-1],[0,1],[0,-1],[-1,1],[-1,0],[-1,-1]]
    for dx,dy in directions:
        area+=dfs(x+dx,y+dy,panel,visited,n,m)
    return area
def main():
    t=int(input())
    for _ in range(t):
```

```
        n,m=map(int,input().split())
        panel=[]
        for _ in range(n):
            panel.append(list(input().strip()))
        max_area=0
        visited=[[False]*m for _ in range(n)]
        for x in range(n):
            for y in range(m):
                if panel[x][y] and not visited[x][y]:
                    area=dfs(x,y,panel,visited,n,m)
                    max_area=max(area,max_area)
        print(max_area)
if __name__=='__main__':
    main()
```

代码运行截图 （至少包含有"Accepted"）

状态：Accepted

源代码

```
def dfs(x,y,panel,visited,n,m):
    if x<0 or x>=n or y<0 or y>=m:
        return 0
    if visited[x][y] or panel[x][y]!='W':
        return 0
    visited[x][y]=True
    area=1
    directions=[[1,1],[1,0],[1,-1],[0,1],[0,-1],[-1,1],[-1,0],[-1,-1]]
    for dx,dy in directions:
        area+=dfs(x+dx,y+dy,panel,visited,n,m)
    return area
def main():
    t=int(input())
    for _ in range(t):
        n,m=map(int,input().split())
        panel=[]
        for _ in range(n):
            panel.append(list(input().strip()))
        max_area=0
        visited=[[False]*m for _ in range(n)]
        for x in range(n):
            for y in range(m):
                if panel[x][y] and not visited[x][y]:
                    area=dfs(x,y,panel,visited,n,m)
                    max_area=max(area,max_area)
        print(max_area)
if __name__=='__main__':
    main()
```

sy134: 全排列III 中等

https://sunnywhy.com/sfbj/4/3/134

思路：虽然permutations（）就可以直接全排列，但未免过于取巧，并不知道考试时会不会被禁用，况且这道题本身是在练回溯，所以还是手写的（这次用dfs写的），不过排序就直接sorted了，反正是字典序（或者先排序后全排列也能保证字典序），以及去重就用集合了，比较方便。

代码

```
n=int(input())
nums=list(map(int,input().split()))
result=set()
def dfs(path,visited):
    if len(path)==n:
        result.add(tuple(path))
    for i in range(n):
        if not visited[i]:
            visited[i]=True
            dfs(path+[nums[i]],visited)
            visited[i]=False
dfs([],[False]*n)
for j in sorted(result):
    print(' '.join(map(str,j)))
```

代码运行截图 （至少包含有"Accepted"）

```
n=int(input())
nums=list(map(int,input().split()))
result=set()
def dfs(path,visited):
    if len(path)==n:
        result.add(tuple(path))
    for i in range(n):
        if not visited[i]:
            visited[i]=True
            dfs(path+[nums[i]],visited)
            visited[i]=False
dfs([],[False]*n)
for j in sorted(result):
    print(' '.join(map(str,j)))

```

测试输入 提交结果 历史提交

完美通过

100% 数据通过测试 详情

运行时长: 0 ms

sy136: 组合II 中等

https://sunnywhy.com/sfbj/4/3/136

给定一个长度为的序列，其中有n个互不相同的正整数，再给定一个正整数k，求从序列中任选k个的所有可能结果。

思路：这个用回溯做的，跟全排列思路是一样的，代码也差不多就改个条件。

代码

```
n, k = map(int, input().split())
nums = list(map(int, input().split()))
result = []
def backtrack(start, path):
    if len(path) == k:
        result.append(path[:])
        return
    for i in range(start, n):
        path.append(nums[i])
        backtrack(i + 1, path)
        path.pop()
backtrack(0, [])
for comb in result:
    print(' '.join(map(str, comb)))
```

代码运行截图 （至少包含有"Accepted"）

```python
n, k = map(int, input().split())
nums = list(map(int, input().split()))
result = []
def backtrack(start, path):
    if len(path) == k:
        result.append(path[:])
        return
    for i in range(start, n):
        path.append(nums[i])
        backtrack(i + 1, path)
        path.pop()
backtrack(0, [])
for comb in result:
    print(' '.join(map(str, comb)))
```

测试输入 提交结果 历史提交

完美通过

100% 数据通过测试 详情

运行时长: 0 ms

sy137: 组合III 中等

https://sunnywhy.com/sfbj/4/3/137

思路：前一个组合是元素互不相同的，这个会有重合，其实跟前面的全排列2一样排序后用集合去重就可以了。代码和上一题几乎无差别，除了把接收改成集合以及把中间过程的列表改成元组。

代码

```
n, k = map(int, input().split())
nums = list(map(int, input().split()))
result=set()
def backtrack(start, path):
    if len(path) == k:
        result.add(tuple(path))
        return
    for i in range(start, n):
        path.append(nums[i])
        backtrack(i + 1, path)
        path.pop()
backtrack(0, [])
for item in result:
    print(' '.join(map(str,item)))
```

代码运行截图 （至少包含有"Accepted"）

```python
n, k = map(int, input().split())
nums = list(map(int, input().split()))
result=set()
def backtrack(start, path):
    if len(path) == k:
        result.add(tuple(path))
        return
    for i in range(start, n):
        path.append(nums[i])
        backtrack(i + 1, path)
        path.pop()
backtrack(0, [])
for item in result:
    print(' '.join(map(str,item)))

```

测试输入 提交结果 历史提交

完美通过 查看题解

100% 数据通过测试 详情

运行时长: 0 ms

M04123: 马走日

dfs, http://cs101.openjudge.cn/pctbook/M04123

思路：这个题也是一个回溯，结束条件是步数等于格数。学到了声明nonlocal变量，可以修改外层函数中的变量。m和n都比较小，并未剪枝也AC了。

代码

```
def horse(n,m,x,y):
    directions=[(-2,1),(-1,2),(1,2),(2,1),(2,-1),(1,-2),(-1,-2),(-2,-1)]
    count=0
    visited=[[False]*m for _ in range(n)]

    def dfs(x,y,step):
        nonlocal count
        if step==n*m:
            count+=1
            return
        for dx,dy in directions:
            nx,ny=x+dx,y+dy
            if 0<=nx<n and 0<=ny<m and not visited[nx][ny]:
                visited[nx][ny]=True
                dfs(nx,ny,step+1)
                visited[nx][ny]=False
    visited[x][y]=True
    dfs(x,y,1)
    return count
t=int(input())
for _ in range(t):
    n,m,x,y=map(int,input().split())
    print(horse(n,m,x,y))
```

代码运行截图 （至少包含有"Accepted"）

状态：Accepted

源代码

```
def horse(n,m,x,y):
    directions=[(-2,1),(-1,2),(1,2),(2,1),(2,-1),(1,-2),(-1,-2),(-2,-1)]
    count=0
    visited=[[False]*m for _ in range(n)]

    def dfs(x,y,step):
        nonlocal count
        if step==n*m:
            count+=1
            return
        for dx,dy in directions:
            nx,ny=x+dx,y+dy
            if 0<=nx<n and 0<=ny<m and not visited[nx][ny]:
                visited[nx][ny]=True
                dfs(nx,ny,step+1)
                visited[nx][ny]=False
    visited[x][y]=True
    dfs(x,y,1)
    return count
t=int(input())
for _ in range(t):
    n,m,x,y=map(int,input().split())
    print(horse(n,m,x,y))
```

T02287: Tian Ji -- The Horse Racing

greedy, dfs http://cs101.openjudge.cn/pctbook/T02287

思路：这是一个贪心题，需要各两个指针，一快一慢。操作上就是不要浪费快马：当田忌快马快于王的快马，赢200；当田忌快马慢于王快马，用慢马输200；当田忌快马等于王快马，看慢马，如果田忌慢马快于王慢马，赢200；如果田忌慢马慢于王慢马，用慢马抵王快马，输200，又回到循环。思路上由于对这个故事非常熟悉，还是能想到的。但是，遗漏了最后田忌的慢马有可能并不比王的快马慢的判断，导致调了一个小时才发现。。。

代码

```
import sys
def tianji(t_horse,k_horse,n):
    t_horse.sort(reverse=True)
    k_horse.sort(reverse=True)
    money=0
    t_left,t_right=0,n-1
    k_left,k_right=0,n-1
```

```
    for _ in range(n):
        if t_horse[t_left] >k_horse[k_left]:
            t_left += 1
            k_left += 1
            money += 200
        elif t_horse[t_left] <k_horse[k_left]:
            t_right -= 1
            k_left += 1
            money -= 200
        else:
            if t_horse[t_right]>k_horse[k_right]:
                t_right -= 1
                k_right -= 1
                money += 200
            else:
                if t_horse[t_right] < k_horse[k_left]:
                    money-=200
                t_right -= 1
                k_left += 1

    return money
data=sys.stdin.read().strip().split()
idx=0
result=[]
while idx<len(data):
    n=int(data[idx])
    idx+=1
    if n==0:
        break
    t_horse=list(map(int,data[idx:idx+n]))
    idx+=n
    k_horse=list(map(int,data[idx:idx+n]))
    idx+=n
    result.append(str(tianji(t_horse,k_horse,n)))
print('\n'.join(result))
```

代码运行截图 （至少包含有"Accepted"）

状态：Accepted

源代码

```
import sys
def tianji(t_horse,k_horse,n):
    t_horse.sort(reverse=True)
    k_horse.sort(reverse=True)
    money=0
    t_left,t_right=0,n-1
    k_left,k_right=0,n-1
    for _ in range(n):
        if t_horse[t_left] >k_horse[k_left]:
            t_left += 1
            k_left += 1
            money += 200
        elif t_horse[t_left] <k_horse[k_left]:
            t_right -= 1
            k_left += 1
            money -= 200
        else:
            if t_horse[t_right]>k_horse[k_right]:
                t_right -= 1
                k_right -= 1
                money += 200
            else:
                if t_horse[t_right] < k_horse[k_left]:
                    money-=200
                t_right -= 1
                k_left += 1

    return money
data=sys.stdin.read().strip().split()
idx=0
result=[]
while idx<len(data):
    n=int(data[idx])
    idx+=1
    if n==0:
        break
    t_horse=list(map(int,data[idx:idx+n]))
    idx+=n
    k_horse=list(map(int,data[idx:idx+n]))
    idx+=n
    result.append(str(tianji(t_horse,k_horse,n)))
print('\n'.join(result))
```

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“计概2025fall每日选做”、CF、LeetCode、洛谷等网站题目。 这次作业的马走日，田忌赛马都比较难，中间三个题有点像回溯小练习，没有使用剪枝因为并不太会（写完火速看题解）。主要练习了递归，回溯，dfs。本周几乎每天都在做每日选做，希望能早点赶上进度。