HW#D

经济学院 刘安澜 1700015495

OJ2981：大整数加法（送分题目）
http://cs101.openjudge.cn/practice/2981/
解题思路：这题确实是送分题🤣🤣🤣。

```
m=int(input())
n=int(input())
print(m+n)
```

OJ18161：矩阵运算(matrix)，cs101-2017 期末机考备选
http://cs101.openjudge.cn/practice/18161/
解题思路：因为学过线性代数，所以对这道题的运算比较熟悉，也能很快写出判断条件。

```
rowA,colA=map(int,input().split())
A=[[int(x) for x in input().split()] for i in range(rowA)]
rowB,colB=map(int,input().split())
B=[[int(x) for x in input().split()] for i in range(rowB)]
rowC,colC=map(int,input().split())
C=[[int(x) for x in input().split()] for i in range(rowC)]
if colA!=rowB or rowA!=rowC or colB!=colC:
    print('Error!')
else:
    ans=[[0]*colB for i in range(rowA)]
    for i in range(rowA):
        for j in range(colB):
            for k in range(colA):
                ans[i][j]+=A[i][k]*B[k][j]
            ans[i][j]+=C[i][j]
    for i in range(rowC):
        print(' '.join([str(x) for x in ans[i]]))
```

OJ12559：最大最小整数 v0.2(string,sorting)，cs101 2016 final exam
http://cs101.openjudge.cn/practice/12559/
解题思路：是这次作业中最没有思路的题目，一开始认为是简单的字符串排序，后来发现对于 9 和 9123 会给出错误的排序。后来查了冒泡排序，但还是看到了群里同学提供的冒泡排序筛选思路才明白了（否则暴力循环一定会超时），这样时间复杂度可以迅速降低。这个排序方式非常巧妙，因为两个数前后组合的位数是相等的，就不会出现不同长度字符串排序的干扰。

```
def bubble_sort(array):
    for i in range(1, len(array)):
```

```
        for j in range(0, len(array)-i):
            if array[j]+array[j+1] > array[j+1]+array[j]:
                array[j], array[j+1] = array[j+1], array[j]
    return array

n=int(input())
l=input().split()
sortl=bubble_sort(l)
print(''.join([str(x) for x in reversed(sortl)]),''.join([str(x) for x in sortl]))
```

OJ19963：买学区房 v0.3(sort,math), cs101 2019 Final Exam
http://cs101.openjudge.cn/practice/19963/
解题思路：感觉是一道比较中规中矩的排序题目，难点主要在坐标的处理上，我先把字符串两头去掉再按照括号分割，之后取数字即可。我觉得自己的写法开了很多列表，其实有些冗余，但好在本题样本量不大。中位数的提取，因为不能使用 numpy 库，所以只能相对麻烦写一个判断。
PS：这题最开始困惑我的是，房子距离越近，价格越低对购买者越有利，但是为啥性价比是越高越好（？？？？），这样房子距离越远价格越低性价比越高……感觉这个经济意义有点问题😅

```
n=int(input())
l=input()
price=[int(x) for x in input().split()]
l1=l[1:len(l)-2].split(') (')
l2=[[0,0] for i in range(n)]
distance=[0]*n
ratio=[0]*n
for i in range(n):
    l2[i][0],l2[i][1]=map(int,l1[i].split(','))
    distance[i]=l2[i][0]+l2[i][1]
    ratio[i]=distance[i]/price[i]
if n%2==0:
    ratiom=(sorted(ratio)[n//2]+sorted(ratio)[n//2-1])/2
    pricem=(sorted(price)[n//2]+sorted(price)[n//2-1])/2
else:
    ratiom=sorted(ratio)[n//2]
    pricem=sorted(price)[n//2]
ans=0
for i in range(n):
    if price[i]<pricem and ratio[i]>ratiom:
        ans+=1
print(ans)
```

OJ19948：因材施教 (greedy), cs101 2019 Final Exam
http://cs101.openjudge.cn/practice/19948/
解题思路：这题一开始没有太多思路，能想到的就是要开一个差值的列表。但后来忽然明白本题的本质就是去掉 m-1 个最大的差值，所以最后的程序很简单。感觉这种题目思路通了就非常顺了。

```
n,m=map(int,input().split())
l=sorted([int(x) for x in input().split()])
diff=[0]*(n-1)
for i in range(n-1):
    diff[i]=l[i+1]-l[i]
print(sum(sorted(diff,reverse=True)[m-1:]))
```

//选做：
OJ4101：晶矿的个数 (dfs)
http://cs101.openjudge.cn/practice/4101/
解题思路：两道 dfs 题目我都是直接借用上次池塘题目的代码改的，感觉这类题目非常类似，所以也很好借鉴。这道题看了一下答案，我觉得我写的有些麻烦，应该在 dfs 函数里面再加一个晶矿类型的参数就更方便。

```
dx=[0,0,1,-1]
dy=[1,-1,0,0]
def dfsr(i,j):
    if l[i][j]!='r':
        return
    l[i][j]='R'
    for s in range(4):
        dfsr(i+dx[s],j+dy[s])

def dfsb(i,j):
    if l[i][j]!='b':
        return
    l[i][j]='B'
    for s in range(4):
        dfsb(i+dx[s],j+dy[s])

T=int(input())
for i in range(T):
    N=int(input())
    l=[[0]*(N+2)]+[[0]+list(input())+[0] for i in range(N)]+[[0]*(N+2)]
    countr=0
    countb=0
```

```
    for i in range(1,N+1):
        for j in range(1,N+1):
            if l[i][j]=='r':
                countr+=1
                dfsr(i,j)
            if l[i][j]=='b':
                countb+=1
                dfsb(i,j)
    print(countr,countb)
```

OJ18160：最大连通面积（dfs），cs101-2017 期末机考备选
http://cs101.openjudge.cn/practice/18160/
解题思路：代码整体上还是参考池塘的 dfs。本题是我第一次使用全局变量声明，所以对这类需要在循环内调用外部的变量的形式有了更多的认识。不过貌似 OJ 平台上对于先函数定义的 global 变量不友好，必须在出现之前先赋值一下变量才能过……
https://blog.csdn.net/qq_28888837/article/details/88060376

```
dx=[0,0,1,1,1,-1,-1,-1]
dy=[1,-1,1,0,-1,1,0,-1]
count=0
def dfs(i,j):
    global count
    if l[i][j]!='W':
        return
    l[i][j]='M'
    count+=1
    for s in range(8):
        dfs(i+dx[s],j+dy[s])

T=int(input())
for i in range(T):
    N,M=map(int,input().split())
    l=[[0]*(M+2)]+[[0]+list(input())+[0] for i in range(N)]+[[0]*(M+2)]
    ans=0
    for i in range(1,N+1):
        for j in range(1,M+1):
            count=0
            if l[i][j]=='W':
                dfs(i,j)
                ans=max(ans,count)
    print(ans)
```

OJ21608：你和你比较熟悉的同学 (dfs, bfs)，2020fall 沉浸式教学样例
http://cs101.openjudge.cn/practice/21608/
解题思路：这题我参考了老师提供的 dfs 思路，在自己复现的时候又改进了一下细节：一个是如果节点为-1 代表这个节点后面没有数据，所以直接在函数内处理一下判断条件；还有一个是按照' : '分割可以去掉 id 后面的空格，就不需要再使用 rstrip 函数了。本题的关键是首先生成一个完整的图，然后对图里的元素进行遍历索引，标记索引过的元素。个人认为这道题 dfs 思路比 bfs 更加容易明白，这道题算是提供了一个对于图类问题的解决模板。

```
def dfs(graph,node,visited):
    if node!=-1 and node not in visited:
        visited.append(node)
        if node not in graph:
            return visited
        for nei in graph[node]:
            dfs(graph,nei,visited)
    return visited

graph={}
ids=set()
n=int(input())
for i in range(n):
    l=input().split(' : ')
    ids.add(int(l[0]))
    if int(l[0]) not in graph:
        graph[int(l[0])]=[int(x) for x in l[1].split()]

maxp=0
for i in ids:
    dfs_path=dfs(graph,i,[])
    maxp=max(maxp,len(dfs_path))
print(maxp)
```

PS：这学期学完计概之后，本周在写 MATLAB 的时候也将计概的一些循环技巧套用在了里面，觉得自己编程水平确实提高了🤣还是挺有收获的！