月考：AC 5/6

1. (11min)

E28674:《黑神话：悟空》之加密

思路：注意模 26

代码：

```
k=int(input())
k=k%26
code=input()
p=''
for i in range(len(code)):
    if 'a'<=code[i]<='z':
        p+=chr(97+(ord(code[i])-97-k)%26)
    elif 'A'<=code[i]<='Z':
        p+=chr(65+(ord(code[i])-65-k)%26)
print(p)
```

运行：

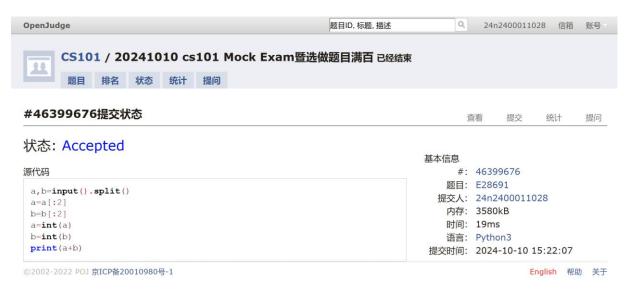## 2. (3min)

E28691: 字符串中的整数求和

http://cs101.openjudge.cn/practice/28691/

思路：简单，直接输入处理

代码：

```python
a,b=input().split()
a=a[:2]
b=b[:2]
a=int(a)
b=int(b)
print(a+b)
```

运行：

3. (9min)

M28664: 验证身份证号

http://cs101.openjudge.cn/practice/28664/

思路：创建字典索引

代码：

```
n=int(input())
lis=[7,9,10,5,8,4,2,1,6,3,7,9,10,5,8,4,2]
dic={0:'1',1:'0',2:'X',3:'9',4:'8',5:'7',6:'6',7:'5
',8:'4',9:'3',10:'2'}
for i in range(n):
    num=0
    id=input()
    for j in range(17):
        num+=int(id[j])*lis[j]
    num=num%11
    if dic[num]!=id[17]:
        print('NO')
    else:
        print('YES')
```

运行：

4. (8min)

M28678: 角谷猜想

http://cs101.openjudge.cn/practice/28678/

思路：可自定义递归函数来做

代码：

```
def jiaogu(a):
    if a==1:
        print('End')
        return
    else:
        if a%2==1:
            b=a*3+1
            print(str(a)+'*3+1='+str(b))
        else:
            b=a//2
            print(str(a)+'/2='+str(b))
        jiaogu(b)
m=int(input())
jiaogu(m)
```

运行:

5. (30min)

M28700: 罗马数字与整数的转换

http://cs101.openjudge.cn/practice/28700/

思路: 我的代码比较粗暴, 类似短除法, 只不过 elif 要写很多

代码:

```python
dic={'I':1,'V':5,'X':10,'L':50,'C':100,'D':500,'M':
1000}
num=input()
if not '0'<=num[0]<='9':
    b=0
    i=0
    while i<=len(num)-1:
        if   i<len(num)-1   and   num[i]=='I'   and
num[i+1]=='V':
            b+=4
            i+=2
        elif   i<len(num)-1   and   num[i]=='I'   and
num[i+1]=='X':
            b+=9
            i+=2
        elif   i<len(num)-1   and   num[i]=='X'   and
num[i+1]=='L':
            b+=40
            i+=2
        elif   i<len(num)-1   and   num[i]=='X'   and
num[i+1]=='C':
            b+=90
            i+=2
        elif   i<len(num)-1   and   num[i]=='C'   and
num[i+1]=='D':
            b+=400
            i+=2
        elif   i<len(num)-1   and   num[i]=='C'   and
num[i+1]=='M':
            b+=900
            i+=2
        else:
            b+=dic[num[i]]
            i+=1
    print(b)
else:
    num=int(num)
    b=''
    while num>0:
        if num>=1000:
            b=b+'M'*(num//1000)
            num-=(num//1000)*1000
        elif 900<=num<1000:
            num-=900
```

```
            b+='CM'
        elif 500<=num<900:
            num-=500
            b+='D'
        elif 400<=num<500:
            num-=400
            b+='CD'
        elif 100<=num<400:
            b+='C'*(num//100)
            num-=(num//100)*100
        elif 90<=num<100:
            num-=90
            b+='XC'
        elif 50<=num<90:
            num-=50
            b+='L'
        elif 40<=num<50:
            num-=40
            b+='XL'
        elif 10<=num<40:
            b+='X'*(num//10)
            num-=(num//10)*10
        elif num==9:
            num-=9
            b+='IX'
        elif 5<=num<9:
            num-=5
            b+='V'
        elif num==4:
            num-=4
            b+='IV'
        elif 1<=num<4:
            b+='I'*num
            num=0
    print(b)
```
运行:

6. (1.5h（考场内没有做出）)

T25353: 排队 （选做）

http://cs101.openjudge.cn/practice/25353/

思路: 参考了郭绍阳学长的贪心优化和上课展示的代码，每次找出所

有的自由节点并排序输出

代码:

```
n,d=map(int,input().split())
line=[0]*n
for i in range(n):
    line[i]=int(input())
check=[False]*n
line_new=[]
while len(line_new)<n:
    buffer=[]
    i=0
    while i<n:
        if check[i]:
```

```
                i+=1
                continue
            if len(buffer)==0:
                buffer.append(line[i])
                maxh=line[i]
                minh=line[i]
                check[i]=True
                continue
            maxh=max(maxh,line[i])
            minh=min(minh,line[i])
            if maxh<=line[i]+d and minh>=line[i]-d:
                buffer.append(line[i])
                check[i]=True
            i+=1
    buffer.sort()
    line_new.extend(buffer)
for i in range(n):
    print(line_new[i])
```

运行:

总结和收获：

1. 贪心算法需要较强的直觉并且证明直觉是正确的，再将想法通过
   数据结构实现，还是很有难度的，需要多加思考

2. 多看题解，压缩代码，学习优秀的算法逻辑，有助于提升解题速度