1. IP黑名单

IP黑名单

时间限制： C/C++语言 1000MS；其他语言 3000MS
内存限制： C/C++语言 65536KB；其他语言 589824KB
题目描述：
在软件系统中，经常会使用到IP黑名单功能。例如如下的黑名单样例，条目可能是单个IP，也可能是一个子网，IP黑名单样例：

62.33.12.122
221.58.4.0/24
8.9.88.1
...

请构建数据结构，在加载黑名单后，能判定给定的IP是否命中指定的IP黑名单。

输入

第一行输入指定IP地址，IPv4格式输入；
第二行输入指定的IP黑名单地址，可以是一个IPv4地址，或一个子网地址。

输出

判断指定的IP是否否命中指定的IP黑名单。

样例输入

```
2.33.128.9
2.33.128.0/24
```

样例输出

```
1
```

规则

```
def CheckBlackList(userIP, blackIP):
    userIPs = [int(i) for i in userIP.split('.')]
    userIP_int = userIPs[0] << 24 | userIPs[1] << 16 \
                 | userIPs[2] << 8 | userIPs[3]

    tmpIPs = blackIP.split('/')
    mask = 0xffffffff << (32 - int(tmpIPs[1]))

    blackIPs = [int(i) for i in tmpIPs[0].split('.')]
    blackIP_int = blackIPs[0] << 24 | blackIPs[1] << 16 \
                  | blackIPs[2] << 8 | blackIPs[3]
    return (userIP_int & mask) == (blackIP_int & mask)

# ******************************结束写代码******************************
'''
2.33.128.9
2.33.128.0/24
'''

try:
    _userIP = input()
except:
    _userIP = None

try:
```

```
    _blackIP = input()
except:
    _blackIP = None

res = CheckBlackList(_userIP, _blackIP)
print(str(int(res)) + "\n")
```

> 参考博客：https://blog.csdn.net/jeffleo/article/details/72824240

2. 图论

编程题 | 20.0分 2/3

图论

时间限制： C/C++语言 1000MS；其他语言 3000MS
内存限制： C/C++语言 65536KB；其他语言 589824KB
题目描述：
豚厂办公区里到处摆放着各种各样的零食，人力资源部的调研发现，员工如果可以在自己喜欢的美食旁边工作，工作效率会大大提高，因此，豚厂决定进行一次员工座位大调整。
调整方法如下：
a) 首先将办公区按照各种零食的摆放分成N个不同的区域。（例如：可乐区，饼干区，水果区等等）；
b) 每个员工对不同零食区域有不同的喜好程度（喜好程度的范围为1—100的整数，喜好程度越大表示员工越希望被调整到相应的零食区域）；
c) 由于每个零食区域可以容纳的员工数量有限，人力资源部希望找到一个最优的调整方案使总的喜好程度最大。

输入

前两行包含两个整数N, M, (1 <=N, M<=300),分别表示N个区域和M个员工；
第三行使N个整数构成的数列a，其中a[i] 表示第i个区域可以容纳的员工数，（1<=a[i]<=M, a[1]+a[2]+..+a[N]=M);
最后是一个M X N 的矩阵P，P(i, j)表示第i个员工对j个区域的喜好程度。

输出

对于每个测试数，输出可以达到的最大喜好程度

样例输入

```
3
3
1 1 1
100 50 25
100 50 25
100 50 25
```

样例输出

```
175
```

规则

请尽量在全场考试结束10分钟前调试程序，否则由于密集排队提交，可能查询不到编译结果

```
#include <iostream>
#include <vector>
#include <numeric>
#include <limits>

using namespace std;


/*请完成下面这个函数，实现题目要求的功能
当然，你也可以不按照下面这个模板来作答，完全按照自己的想法来 ^-^
******************************开始写代码******************************/
```

```
bool BestMatch(vector < int > LimitArray, vector < vector < int > > DegMatrix) {


}
/*****************************结束写代码******************************/


int main() {
    bool res;

    int _LimitArray_size = 0;
    int _DegMatrix_rows = 0;
    int _DegMatrix_cols = 0;
    cin >> _DegMatrix_rows >> _DegMatrix_cols;

    _LimitArray_size = _DegMatrix_cols;

    vector<int> _LimitArray;
    int _LimitArray_item;
    for(int _LimitArray_i=0; _LimitArray_i<_LimitArray_size; _LimitArray_i++) {
        cin >> _LimitArray_item;
        cin.ignore (std::numeric_limits<std::streamsize>::max(), '\n');
        _LimitArray.push_back(_LimitArray_item);
    }

    vector< vector < int > > _DegMatrix(_DegMatrix_rows);
    for(int _DegMatrix_i=0; _DegMatrix_i<_DegMatrix_rows; _DegMatrix_i++) {
        for(int _DegMatrix_j=0; _DegMatrix_j<_DegMatrix_cols; _DegMatrix_j++) {
            int _DegMatrix_tmp;
            cin >> _DegMatrix_tmp;
            _DegMatrix[_DegMatrix_i].push_back(_DegMatrix_tmp);
        }
    }

    res = BestMatch(_LimitArray, _DegMatrix);
    cout << res << endl;

    return 0;

}
```

3. 动态规划路径

```

```

> 参考：https://leetcode-cn.com/problems/unique-paths-ii/