Q1假设某系统上有一些进程在运行，请编写程序算出给定时间里同时在运行的最大进程数和最小进程数 (Python/C/C++/Java)。

要求：

编写程序：

1. 处理如下输入：
   1. 两个整数，分别代表需要计算进程数的开始时间和结束时间
   2. 进程的数目
   3. 每个进程的运行时间 （两个整数，分别代表这个进程运行的开始时间和结束时间）。
2. 算出给定时间内同时运行的最小进程数和最大进程数

输入（标准输入）：

第一行 两个以空格分隔的整数：b e, 0 <= b <= e <= 1000000000, 分别代表需要计算进程数的开始时间和结束时间（包含e）。

第二行 一个整数n, 1 <= n <= 10000， 代表系统上运行的所有的进程数。

第i + 2行(1 <= i <= n) 两个以空格分隔的整数：pi ki, 0 <= pi <= ki <= 1000000000, 分别代表第i个进程运行的开始时间和结束时间，表示第i个进程从pi开始运行，到ki结束运行（运行时间包含ki）。

输出（标准输出）：

程序输出两行。

第一行输出在[b,e]时间内同时运行的最小进程数。

第二行输出在[b,e]时间内同时运行的最大进程数。

程序运行时间限制：1 s

示例：

输入：

8 20

3

2 10

10 12

8 30

输出：

1

3

Q2 斗地主是一个喜闻乐见的牌类游戏，在这个游戏中地主会随机抓取一副扑克牌中20张牌，两个农民会分别随机抓取其中17张牌。作为其中一个农民你每次抓完牌都要考虑怎么组合这些牌，来用最少的次数把牌出完 (Python/C/C++/Java)。

那么，在只考虑自己手中牌的情况下，出完这17张牌最少要出多少次牌呢？

斗地主出牌规则：

单牌：单个牌（如红桃5）。

对牌：数值相同的两张牌（如梅花4+方块4）。

三张牌：数值相同的三张牌（如三个J）。

四张牌：数值相同的四张牌（如四个J）。

三带一/三带二：数值相同的三张牌 + 一张单牌或一对牌。例如：333+6 或 444+99。

单顺：五张或更多的连续单牌（如：45678 或 78910JQK）。不包括 2 点和双王。

双顺：三对或更多的连续对牌（如：334455 、778899 1010 JJ）。不包括 2 点和双王。

三顺：二个或更多的连续三张牌（如：333444 、 555 666 777 888）。不包括 2 点和双王。

飞机带翅膀：三顺+同数量的单牌（或同数量的对牌）。如： 444555+79 或 333444555+7799JJ。

四带二：四张牌+两手牌（注意：四带二不是炸弹）。如： 5555+3+8 或 4444+55+77。

请实现如下接口：

int PlayCard(char cards[17]);

输入：长度17的字符数组（10用S表示，大小王用W表示）

输出：

输出有多行，打印最少出牌的次数，对于最少出牌的次数可能有多种出牌方案，打印每一种出牌方案中每一次出的牌。

第一行输出最少出牌的次数。

后面每一行打印每一种出牌方案所出的牌，以空格分隔每一次出的牌

示例：（10用S表示，王用W表示）

输入：

3 3 3 4 4 4 5 5 6 7 8 9 S J Q K A

输出：

2

33344455    6789SJQKA

3334445A    56789SJQK

Q3 用Python开发VNFM RESTful API的框架，具体函数如下：

1. 获取所有的VNF数据

Path: GET /vnfs

返回：HTTP code 200和json 格式的VNF数据（数据如示例所示）。

示例：

[{“vnf\_id”: 1, “vnf\_name”: “vnf01”, “vnf\_desc”: “test1”}, {“vnf\_id”: 2, “vnf\_name”: “vnf02”, “vnf\_desc”: “test2”}, {“vnf\_id”: 3, “vnf\_name”: “vnf03”, “vnf\_desc”: “test3”}]

1. 获取某一个VNF的详细数据

Path: GET /vnfs/{vnf\_id}

返回：HTTP code 200和json 格式的VNF详细数据（数据如示例所示）。

示例：

{“vnf\_id”: 1, “vnf\_name”: “vnf01”, “vnf\_desc”: “test1”, “vnf\_load”: “R17”}

1. 创建一个VNF

Path: POST /vnfs

Body参数如下：

{“vnf\_id”: 1, “vnf\_name”: “vnf01”, “vnf\_desc”: “test1”}

在API里打印Body参数的值

返回：HTTP code 201

1. 删除一个VNF

Path: DELETE /vnfs/{vnf\_id}

返回：HTTP code 204

通用规则：

1. 如有疑问请与各部门coordinator联系

Klaus Zhang’s department – Daniel Zhang ([zhang.daniel@alcatel-lucent.com](mailto:zhang.daniel@alcatel-lucent.com))

Klaus Zhang’s department – Chris Chen (Chris.a.Chen@alcatel-lucent.com)

Ellen Li’s department – Henry He ([Henry.He@alcatel-lucent.com](mailto:Henry.He@alcatel-lucent.com))

Ellen Li’s department –Wang Albin ([Albin.Wang@alcatel-lucent.com](mailto:Albin.Wang@alcatel-lucent.com))

Jim Wang’s department – Joshua Zhao (joshua.zhao@alcatel-lucent.com)

Grace Zhao’s department – Kumar Zhang (kumar.zhang@alcatel-lucent.com)

Grace Zhao’s department – Jeffrey Li (jeffrey.li@alcatel-lucent.com)

Seaman Chen’s department – Winter Wang (winter.wang@alcatel-lucent.com)

Tracy Zhang’s department – Rush Song (Jiqing.Song@alcatel-lucent.com)

Robbin Zhou’s department - Sid Li (Sid.Li@alcatel-lucent.com)

1. 时间： 08/15/2016 09:00 AM - 08/22/2016 00:00 AM

非Java语言编程规则：

1. 文件命名规则：

${csl}\_${fullName}\_${questionNumber}.${suffix}

${csl} is your csl.

${fullName} is your full English name.

${questionNumber} is the number of the question.

示例:

zsan\_SanZhang\_Q1.c

zsan\_SanZhang\_Q2.cc

2. 按照要求编写程序，完成编译、测试， 如果认为没有问题就可以提交代码。

1） 一个题目对应一个源文件；

2） 最后提交的程序，不要打印log或者任何要求外的输出；

3） 请把源文件通过邮件发送至各部门的coordinator。

Java 编程规则：

1. 文件打包规则（所有在目录src下面的源文件打包成zip格式的压缩文件，文件命名规则如下，不区分大小写）

${email\_addr}\_${department}\_${question\_no}\_${submit\_id}

${邮箱地址}\_${部门名称}\_${命题编号}\_${流水号（表示第几次提交）}

示例:

Jiqing.song@alcatel-lucent.com\_com\_q1\_1

1. 每个题目均提供基础工程，请下载后导入到eclipse中，工程提供基础的UT测试案例。
2. 除给定的类库和用户自定义类外不允许引用额外的类库。
3. 评测标准：1）编译通过；2）通过所有测试用例（比赛期间会开放一部分辅助开发）；3）通过静态代码扫描工具扫描；4）人工评比（满足前三条的情况下）
4. 题目完成后请将源文件发送到：jiqing.song@alcatel-lucent.com