COT 3100 Program #2 Spring 2014
Assigned: 2/1/2014
Check WebCourses for due date

Note: This program (and all programs for the course) are ONLY for students who signed up for the programming option. If you signed up for this option, you MUST DO ALL FIVE programs assigned.

General Program Directions (to be followed for all five programs)

Turn in a single source file, either C or Java, with the name designated in the note at the bottom that solves the problem described below. Please read all of your input from standard in and output to standard out. To test your program with input files, please pipe the input file into your program and pipe the output to another file. If you don't know how to do this, please see a TA to describe this process to you.

The Problem: Fast Modular Matrix Exponentiation

Raising exponents to large powers is used to solve several types of problems. In some cases, it makes sense to look at the values of the resultant matrix mod a particular value. (For regular integers, fast modular exponentiation is typically used for encryption purposes for secure communication via public key cryptosystems.) For this program, you will be given input matrices, an exponent and a modulus and be asked to compute the given matrix raised to the given exponent mod the modulus.

Problem Solving Restrictions

You must use two dimensional arrays to perform matrix multiplication and write an exponentiation function similar to the posted fast modular exponentiation for integers. You may not use any built in matrix type in any API.

Input Format

The first line of the input file will contain a single positive integer, *n*, representing the number of matrix exponentiations to calculate.

Each input case will follow. The first line of each input case will contain three space separated positive integers: *s* ($s \leq 10$), *e* ($e \leq 10^{18}$), and *m* ($m \leq 2x10^{9}$), and representing the number of rows in the matrix, the exponent to which to raise the matrix, and the modulus for the calculation. The next *s* lines will each contain *s* non-negative integers less than one billion, representing the values in the input matrix. The j^{th} value on the i^{th} of these lines represents the entry in row i, column j of the input matrix.

Output Format

For each test case, output a single header line with the format

```
Case k
```

where k is the number of the test case, starting with 1.

For each test case, output *s* lines, each with *s* values, each followed by a space, representing the resultant matrix.

Sample Input

```
2
2 3 10
1 1
1 0
3 1000000000 1000000007
8 1 6
3 5 7
4 9 2
```

Sample Output

```
Case 1
3 2
2 1
Case 2
28734370 690120370 690120370
690120370 28734370 690120370
690120370 690120370 28734370
```

Deliverables

A single source file, named either ***fastmatexpo.c*** or ***fastmatexpo.java*** that solves the program stated above, using the input and output formats stated above, using standard input and standard output. The file should be submitted via WebCourses by the due date and time stated in WebCourses.