Day 17: More Exceptions

Objective

Yesterday's challenge taught you to manage exceptional situations by using try and catch blocks. In today's challenge, you will practice throwing and propagating an exception. Check out the Tutorial tab for learning materials and an instructional video.

Task

Write a Calculator class with a single method: int power(int,int). The power method takes two integers, n and p, as parameters and returns the integer result of n^p. If either n or p is negative, then the method must throw an exception with the message: `n and p should be non-negative`.

Note: Do not use an access modifier (e.g.: public) in the declaration for your Calculator class.

Input Format

Input from stdin is handled for you by the locked stub code in your editor. The first line contains an integer, T, the number of test cases. Each of the T subsequent lines describes a test case in 2 space-separated integers that denote n and p, respectively.

Constraints

- No Test Case will result in overflow for correctly written code.

Output Format

Output to stdout is handled for you by the locked stub code in your editor. There are T lines of output, where each line contains the result of n^p as calculated by your Calculator class' power method.

Sample Input

```
4
3 5
2 4
-1 -2
-1 3
```

Sample Output

```
243
16
n and p should be non-negative
n and p should be non-negative
```

Explanation

$T = 4$

T_0: 3 and 5 are positive, so power returns the result of 3^5, which is 243.

T_1: 2 and 4 are positive, so power returns the result of 2^4=, which is 16.