A prime number p is a natural number greater than 1 that has only two natural divisors: 1 and p itself. Any natural number n, such that $n > 1$, has a unique decomposition into prime factors, for instance $4 = 2 \times 2$, $5 = 5$, $6 = 2 \times 3$.

Let $sopf(n)$ denote the sum of the prime factors of a natural number n. For instance, $sopf(4) = 2 + 2 = 4$, $sopf(5) = 5$, and $sopf(6) = 2 + 3 = 5$.

If we take the result of this sum, we may compute again the sum of its prime factors, and repeat this *ad nauseam*. However, at some point, we always reach a fix-point, that is a number f such that $f = sopf(f)$. For instance, starting from 8, $sopf(8) = 2 + 2 + 2 = 6$, then $sopf(6) = 2 + 3 = 5$, and $sopf(5) = 5$: applying repetitively sopf from 8 generates the sequence 8, 6, 5, 5, 5, $\ldots$ So, from the initial value 8, it takes 3 applications of sopf to discover that we have reached a fix-point.

Let $lsopf(n)$ denote the number of applications of $sopf$ from n that is required to discover that the fix-point has been reached. For instance, $lsopf(8) = 3$ and $lsopf(4) = 1$.

Your task is, given two natural numbers n and m (with $n > 1$ and $m > 1$), find the largest value the function $lsopf$ takes in the interval between n and m.

Input

The first line of input gives the number of cases, T (with $1 \leq T \leq 150$). T test cases follow. Each test case is on a single line, containing two natural numbers n and m, such that $1 < n, m \leq 500000$.

Output

For each test case first print a line '`Case #`C`:`' (where C is the number of the current test case). Then print another line with the maximum of the function $lsopf$ in the interval bounded by n and m.

Sample Input

```
2
2 10
11 20
```

Sample Output

```
Case #1:
3
Case #2:
4
```