# Day 25: Running Time and Complexity

**Objective**

Today we will learn about running time, also known as time complexity. Check out the Tutorial tab for learning materials and an instructional video.

**Task**

A *prime* is a natural number greater than $1$ that has no positive divisors other than $1$ and itself. Given a number, $n$, determine and print whether it is `Prime` or `Not prime`.

**Note:** If possible, try to come up with a $O(\sqrt{n})$ primality algorithm, or see what sort of optimizations you come up with for an $O(n)$ algorithm. Be sure to check out the *Editorial* after submitting your code.

**Input Format**

The first line contains an integer, $T$, the number of test cases.
Each of the $T$ subsequent lines contains an integer, $n$, to be tested for primality.

**Constraints**

- $1 \leq T \leq 30$
- $1 \leq n \leq 2 \times 10^9$

**Output Format**

For each test case, print whether $n$ is `Prime` or `Not prime` on a new line.

**Sample Input**

```
3
12
5
7
```

**Sample Output**

```
Not prime
Prime
Prime
```

**Explanation**

*Test Case 0:* $n = 12$.

$12$ is divisible by numbers other than $1$ and itself (i.e.: $2$, $3$, $4$, $6$), so we print `Not prime` on a new line.

*Test Case 1:* $n = 5$.

$5$ is only divisible $1$ and itself, so we print `Prime` on a new line.

*Test Case 2:* $n = 7$.

$7$ is only divisible $1$ and itself, so we print `Prime` on a new line.