**¿Cuál es la dificultad computacional del problema, por qué es difícil de computar?**

* El problema es difícil de computar puesto que pide determinar si determinados números son primos o no. La manera de determinar esta respuesta tiene una complejidad de raíz(n), siendo n el número el cual se desea determinar si es primo o no. Por otro lado, se debe determinar sobre una cantidad m de “consultas”, por lo que si ya es difícil el determinar si un número es o no primo, el problema se hace más difícil de computar si son varios números primos.

**Complejidad del algoritmo más básico que se puede hacer**

* El algoritmo más básico que se puede hacer para comprobar si un número es primo o no, es comprobar sus divisores a partir de 1, hasta el número. Si el número tiene solamente dos divisores (1 y si mismo), entonces es un número primo. Por último, se debe iterar por sobre cada número que se quiere consultar, siendo este Q.

**Complejidad que se necesita para resolver el problema**

* En primer lugar, se tiene que la cantidad de “consultas” que se puede hacer es de 200. A este número de consultas le llamaremos N. Por otro lado, cada “consulta” puede pedir determinar si un número es primo o no, siendo el máximo posible de este número , a este número le llamaremos X. La respuesta debe ser generada en a lo más 1 segundo, por lo que, en el peor caso, se debe tener una complejidad de Q\*√n. Este valor resuelve el problema.

**Explicar la solución implementada**

* La solución implementada, itera sobre la cantidad total de números a consultar. Por cada número que se quiere determinar si es primo o no, se aplica una función que revisa sus divisores hasta la raíz cuadrada del número. Esto debido a una propiedad matemática que indica que los divisores de un número se encuentran en misma cantidad a la izquierda de la raíz, como a la derecha de esta (pensando en una recta numérica). Esto nos lleva a tener una complejidad de N\*√X.