Lab 2: Primality test

The objective of this lab is to test different methods for the same problem and also to learn how to measure the execution time of a program.
This involves testing whether a natural number is prime.
Four algorithms are proposed that must be implemented in C language, then compared using the time management functions which are provided in the time.h library.

Reminder: A natural integer N is prime if it has only 2 divisors: the number 1 and the number N itself.

Algorithm 1 (A1) : Naive approach

1. This solution includes a loop in which we will test whether the number N is divisible by 1, 2, 3, …, N. Write the corresponding algorithm.
2. Calculate the best and worst case theoretical complexity of this algorithm in Big Oh.
3. Write the corresponding program.
   a. Check that the numbers N proposed in the table below (1000003, 2000003, …) are prime.
   b. Measure the execution times T for the numbers N below and complete the table:

N	1000003	2000003	4000037	8000009	16000057	32000011	64000031
T							

N	128000003	256000001	512000009	1024000009	2048000011
T					

   c. What do we notice about the data and the measurements obtained? (Hint: compare each number N with the next and each measurement of time with the next.)
   d. Compare theoretical complexity and experimental measurements.

Are the theoretical predictions compatible with the experimental measurements?

   e. Represent (in the same graph) with 2 curves the variations of the experimental and theoretical execution time T(N) for best case and 1 curve for experimental worst case. To do this, use graphics software such as Excel.

Algorithm 2 (A2): Improvement of the naive approach

We know that any divisor i of the number N satisfies the relation: $i \leq N/2$, with $i \neq N$.

1. Develop a 2nd algorithm taking this property into account and repeat the same previous questions.
2. Compare algorithms A1 and A2 (represent the curves of the 2 algorithms in the same graph). Which of the 2 algorithms is better (or more efficient)?

Algorithm 3 (A3):

There is a mathematical property about integers:

Property: The divisors of an integer N are half ≤ $N^{\frac{1}{2}}$(=$\sqrt{N}$) and the other half > $N^{\frac{1}{2}}$

1. Develop a 3rd A3 algorithm taking this property into account and repeat the same previous questions.
2. Compare the 3 algorithms. Which of the 3 algorithms is better (or more efficient)?

Algorithm 4 (A4):

Another possible improvement is to test if N is odd and in this case in the loop, you should only test the divisibility of N by odd numbers.

3. Develop a 4th A4 algorithm taking this proposal into account and repeat the same previous questions.
4. Compare the 4 algorithms. Which of the 4 algorithms is better (or more efficient)?