Fibonacci numbers are the numbers in the Fibonacci sequence, as shown below:
1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ...

By definition, the first two numbers in the Fibonacci sequence are 1 and 1, and each subsequent number is the sum of the previous two.
The sequence \( F_n \) of Fibonacci numbers is defined by the recurrence relation:
\[
Fib(n) = Fib(n - 1) + Fib(n - 2)
\]

Given that,
\[
Fib(1) = 1 \text{ and } Fib(2) = 1
\]

1) Write a recursive algorithm to compute a Fibonacci number \( Fib(n) \).

2) Write an iterative algorithm to compute a Fibonacci number \( Fib(n) \).

3) What is the worst case running time complexity of both algorithms?