1. An array is passed by pointer. Time = Θ(1).
2. An array is passed by copying. Time = Θ(N), where N is the size of the array.
3. An array is passed by copying only the subrange that might be accessed by the
called procedure. Time = Θ(q - p + 1) if the subarray A[p..q] is passed.

a. Consider the recursive binary search algorithm for finding a number in a sorted
array (see Exercise 2.3-5). Give recurrences for the worst-case running times
of binary search when arrays are passed using each of the three methods above,
and give good upper bounds on the solutions of the recurrences. Let N be the
size of the original problem and n be the size of a subproblem.

b. Redo part (a) for the MERGE-SORT algorithm from Section 2.3.1.

4.4 More recurrence examples
Give asymptotic upper and lower bounds for T(n) in each of the following recurrences. Assume that T(n) is constant for sufficiently small n. Make your bounds
as tight as possible, and justify your answers.

a. T(n) = 3T(n/2) + n lg n.

b. T(n) = 5T(n/5) + n/lg n.

c. T(n) = 4T(n/2) + n^2 / lg n.

d. T(n) = 3T(n/3 + 5) + n/2.

e. T(n) = 2T(n/2) + n/lg n.

f. T(n) = T(n/2) + T(n/4) + T(n/8) + n.

g. T(n) = T(n - 1) + 1/n.

h. T(n) = T(n - 1) + lg n.

i. T(n) = T(n - 2) + lg n.

j. T(n) = \sqrt{T(\sqrt{n})} + n.

4.5 Fibonacci numbers
This problem develops properties of the Fibonacci numbers, which are defined
by recurrence (3.21). We shall use the technique of generating functions to solve