Introduction to Computer Science
Tutorial 12: Time complexity.

1. Order the following complexity bounds in increasing order:

   A. $O(n)$
   B. $O(logn)$
   C. $O(n * logn)$
   D. $O(n^2)$
   E. $O(n^4)$
   F. $O(n!)$
   G. $O(2^n)$
   H. $O(n^n)$

2. Calculate the time bound (in big O notation) of the next time functions:

   A. $T(n) = n(n+1)$
   B. $T(n) = \sum_{i=1}^{n} c$
   C. $T(n) = n * (n + \sum_{i=1}^{n}\sum_{i=1}^{n}\sum_{i=1}^{n}\sum_{i=1}^{n} c)$
   D. $T(n) = n(n+1) + n(n*n) + n(\frac{n}{2})$
   E. $T(n) = \prod_{i=1}^{n} i$
   F. $T(n) = n * (n(logn + n))$

3. Calculate the time function (T(n)) for the programming exercises present in the 10^{th} and 11^{th} tutorials and calculate their respective time bounds. You can improve the time efficiency of your solutions? How?

4. Define and implement a function `int search(int seq[], int lower, int higher, int elem)` that search an element inside of a sequence. This function must have a constant time complexity if `elem` is in the first or last position of `seq`.