1. Modern processors have clock speeds in excess of 100MHz. Thus a RISC processor may be executing more than $1 \times 10^8$ machine instructions per second. This means they can process on the order of $1 \times 10^7$ “operations” per second. An “operation” is loosely defined here as something like “one iteration of a very simple loop”. Assuming that your patience allows you to wait:
   (a) one second,
   (b) one minute,
   (c) one hour,
   (d) one day.
Calculate how large a problem you can solve if it is:
   (i) $O(\log_2 n)$
   (ii) $O(n)$
   (iii) $O(\sqrt{n})$
   (iv) $O(n \log n)$
   (v) $O(\log^2 n)$
   (vi) $O(n^2)$
   (vii) $O(2^n)$
   (viii) $O(n!)$
   (ix) $O(n^n)$
Numbers beyond the range of your calculator can simply be reported as “$> 10^x$” or “$< 10^{-x}$”, where $x$ is determined by your calculator.

2. (a) For what values of $n$ is $4 \times 10^6 n^2 > 10 \times 2^n$?
(b) For what values of $n$ is $5 \times n^2 > 3 \times 10^2 \log_2 n$?
(c) Algorithm A requires 200 machine cycles for each iteration and requires $n \log_2 n$ iterations to solve a problem of size $n$. A simpler algorithm, B, requires 25 machine cycles for each iteration and requires $n^2$ iterations to solve a problem of size $n$. Under what conditions will you prefer algorithm A over algorithm B?

3. Insert the following numbers into an ordered binary tree:
   24 9 7 22 0 20 2 43 50 48
   (a) If it takes 50 machine cycles to compare an item in the tree with the search key what is the longest search time (in cycles) that can be experienced?
   (b) Thinking ahead! How would you use the tree to extract a sorted list of items? Your explanation should be sufficient to generate the required code.
   (c) Still thinking ahead!! How would you manipulate the tree to retain its ordered structure when an item (say 9) is deleted?