CS 311: Discrete Structures  
Spring 2003  

Homework 10. Due Thursday, April 22, 2004  

1) Give the best possible bound for each function below using big-O notation and justify your answer.  
(a) \( t(n) = 2n^2 + 3n - 5 \).  
(b) \( t(n) = 8 \cdot \log_2(n) + n - 5 \)  
(c) \( t(n) = 4n \log_2 n + 4n + 10 \)  
(d) \( t(n) = \sum_{i=1}^{2n} i \)  
Extra Credit: \( t(n) = \sum_{i=1}^{\log_2 n} 2^i \)  

2) Show that \( t(n) = 2n + 1 \) is \( \Theta(n) \).  

3) Show that \( t(n) = 2n + 1 \) is not \( O(\log_2 n) \).  

4) Prove: If \( f(n) \) is \( O(n) \) and \( g(n) = f(n) + 100 \) then \( g(n) \) is \( O(n) \).  

5) Consider the following code:  
\[
\text{read } n  
i = 1  
\text{while } (i \leq n) \{  
i = 2 \cdot i  
\text{print } i  
\}\]  
(a) If \( n = 50 \) what does the code print out?  
(b) As a function of \( n \), how many times is the loop executed?  
(i.e., how many numbers does the code print out?) Do some small examples to figure this out.  

6) Give recurrence relation for running time of the following functions.  
For part (a) count the number of multiplications. For part (b) count the number of comparisons \( (\text{tmp} < \text{A[start]}) \). For part (c) count the number of comparisons \( \text{if (start == end)} \).
(a) int myst(int n) 
    { 
      if (n == 0) return 1;
      if (n == 1) return 2;
      else return myst(n-2)*4;
    }

(b) int find_min(int A[], int start, int end) 
    { 
      int tmp;

      if (start == end) return A[start];
      else { 
        tmp = find_min(A, start + 1, end);
        if (tmp < A[start]) return tmp;
        else return A[start];
      }
    }

(c) int process(int A[], int start, int end) 
    { 
      int mid;

      if (start == end) {
        printf("%d\n", A[start]);
        return;
      }
      else { 
        mid = (start + end)/2;
        process(A, start, mid);
        process(A, mid+1, end);
      }
    }

You may assume for part (c) that n is a power of 2.