1. Prove $S(n) = \sum_{i=1}^{n} \log i = \Theta(n \log n)$.
   
   (a) Proof of $O()$:

<table>
<thead>
<tr>
<th>GRADE</th>
<th>/20</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>/20</td>
</tr>
<tr>
<td>2</td>
<td>/20</td>
</tr>
<tr>
<td>3</td>
<td>/20</td>
</tr>
<tr>
<td>4</td>
<td>/20</td>
</tr>
<tr>
<td>5</td>
<td>/20</td>
</tr>
<tr>
<td>SUM</td>
<td>/100</td>
</tr>
</tbody>
</table>

   (b) Proof of $\Omega()$:
   
   Hint: One way to prove the lower bound is by considering only the last $\lfloor n/2 \rfloor$ terms in the summation. That is, observe that $S(n) > \sum_{i=\lfloor n/2 \rfloor}^{n} \log i$. 
2. (a) Find the exact solution of the following recurrence. (Assume $n = 2^k$.)

\[
T(1) = 1,
\]

\[
T(n) = 8T(n/2) + n^2, \quad n > 1.
\]

(b) Consider the following recurrence, where $n = 2^k$.

\[
T(1) = 1,
\]

\[
T(n) = 4T(n/2) + n - 3, \quad n > 1.
\]

Prove by induction that the solution is $T(n) = An^2 + Bn + C$, and find the constants $A, B, C$. 
3. The following function finds the min and max of an array of \( n \) elements, where \( n \) is any integer.

```c
void MINMAX(dtype A[], int n, dtype& min, dtype& max)
{
    dtype min1,max1,min2,max2;
    if (n==1) min=max=A[0];
    else if (n==2)
        if (A[0] <= A[1]) {min=A[0]; max=A[1];}
        else {max=A[0]; min=A[1];} }
else{ // case of \( n \geq 2 \):
    MINMAX(A,n-2,min1,max1);
    else {min2=A[n-1]; max2=A[n-2];}
    if (min1 <= min2) min=min1; else min=min2;
    if (max1 <= max2) max=max2; else max=max1;
}
```

(a) Write a recurrence for the number of key-comparisons, \( f(n) \).

(b) Use repeated substitution to find the exact solution.
4. Recall the operation MAKEHEAP(A,n) which starts with an array A[1..n] of n random values and establishes a heap of n elements.

   (a) A poor way of doing this task is by a series of insert operations:
       for (i = 2; i <= n; i++) INSERT(A, i)
       where INSERT(A, i) inserts element A[i] into an existing heap A[1..i - 1] of i - 1 elements.
       • Analyze the worst-case time complexity T(n) for this implementation of MAKEHEAP.

   • Illustrate this MAKEHEAP on the array A[1..5] = (5, 4, 3, 2, 1). Show the resulting heap after each insert, drawn as binary tree.

(b) A more efficient implementation for MAKEHEAP was discussed in class that uses PUSHDOWN.

   • What is the time complexity of this improved MAKEHEAP?

   • Illustrate this efficient MAKEHEAP on the array A[1..5] = (5, 4, 3, 2, 1). Show the result after each pushdown.
5. (a) Insert the following sequence of elements into a Binary-Search-Tree (BST), starting with an empty tree.

20, 25, 10, 15, 30, 7, 5, 28, 18

Is the resulting BST (after the above sequence of inserts) height-balanced? If not, point out which nodes are not balanced.

(b) Delete element 75 in the following BST. (First complete the picture by drawing a line from each node to its children. Be careful as you draw the lines to get a valid BST.)

75
  /   /
60  90
 /   \
55  70  80  95
 /   /   \
52  58  65  85  100
 /   /   /   \
63  67  82  87
 /   \
66  68