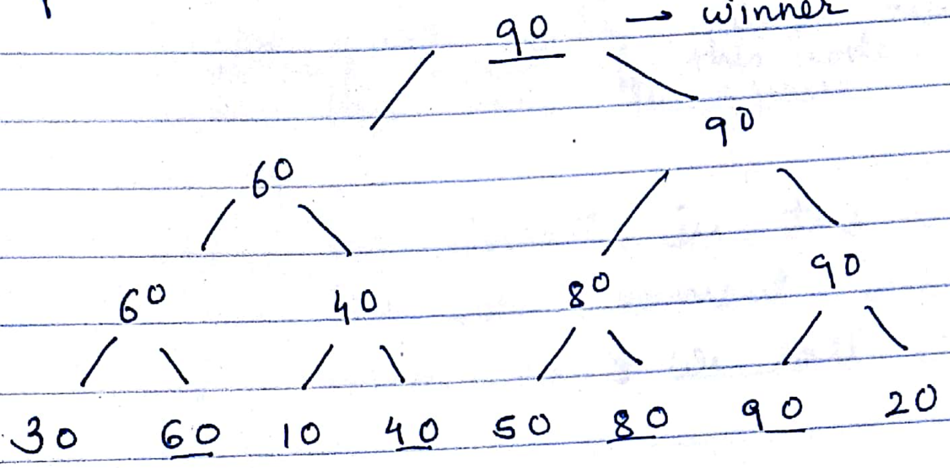
CSE 551: Homework 2

(1.) Example of tournament tree,

[30, 60, 10, 40, 50, 80, 90, 20]

The node with max value will be winner for that match.



In order to get this tree, we need (n-1) comparisons since there are (n-1) matches between n players.

(n-1) represents all the internal nodes in the above tree. ∴ ①

Algo to construct tree:

The logic here is to maintain a list of nodes, remove 2 nodes from the front of list & push their minimum to the end of list.

```
• Push all the nodes in the array to a queue
• while (queue.size != 1)
  {
      pop 2 nodes from queue (x & y)
      create a node with value = max(x & y)
      add x & y as child of this node.
      push this node to queue.
  }
```

The last element in the queue will be the root element with greatest value. or s we call it best player here.

- Algo to find second-best player or second max value in the tree constructed above.

Logic used:

At sometime, second best player must have played match with the best player and lost. We need to find ~~that~~ the max value of ~~the no~~ all the players that played match with best player. Other matches can be ignored. Thus, we need ~~O(1)~~ 1 comparison at every level in tree except the root node. Therefore, the total no. of comparisons = $\underline{\log N - 1}$ ..(2)

& we use recursion here.

```
recur( ~~temp~~ node* temp)
{
      if( temp-> right ==NULL or
          temp-> left == NULL)
          Return;
      if( temp->left-> value==root->value)
       {
          mx = max( temp-> right->value,
                          ~~ro~~  mx)
          recur( temp->left);
       }
       if (temp-> right->value== root->value)
       {
          mx = max( temp-> left->value, mx)
                                    ~~mx~~
          recur( temp-> right)
       }
}
```

From the above algo, we get second best player in mx variable.

From (1) & (2), we can see that the complexity is $N-1 + \log N - 1 = \underline{(N + \log N - 2)}$