1. Consider a data structure that inserts keys in constant time, except that after every 10th key is inserted, the data structure reorganizes itself using a $O(n)$ algorithm, where $n$ is the number of keys currently in the data structure. In other words the cost $c_i$ of the $i^{th}$ operation is as follows.

$$c_i = \begin{cases} 
  i & \text{if } i \mod 10 = 0 \\
  1 & \text{otherwise}
\end{cases}$$

(a) (4 points) Use the aggregate method to show that the amortized cost of each operation is still $O(n)$. 
(b) (4 points) Suppose the following amortized costs are defined in order to apply the accounting method. Are these valid? Why or why not?

\[ \hat{c}_i = \begin{cases} 
1 & \text{if } i \mod 10 = 0 \\
2 & \text{otherwise}
\end{cases} \]

(c) (2 points) Give a potential function \( \Phi(D_i) \) that satisfies the constraints such that the amortized cost is an upper bound on the actual cost (i.e., \( \Phi(D_0) = 0 \) and \( \Phi(D_i) \geq 0 \) for all \( i \)).
2. (4 points) Show any valid binomial heap containing the keys 3, 5, 7, 10, 12, 15.

3. (4 points) Show the Fibonacci heap after executing \texttt{EXTRACTMIN} on the following Fibonacci heap.
4. (4 points) Show the final disjoint-set data structure after executing all of the operations below, using the forest of trees representation with the union by rank and path compression heuristics.

```plaintext
for i = 1 to 10
    MakeSet(i)
Union(1, 2)
Union(3, 4)
Union(5, 6)
Union(7, 8)
Union(9, 10)
Union(1, 3)
Union(4, 6)
Union(7, 9)
FindSet(2)
FindSet(10)
```

5. (3 points) Execute DFS on the following graph by labeling vertices with their discover and finish times, considering vertices in alphabetic order when iterating over a set of vertices.