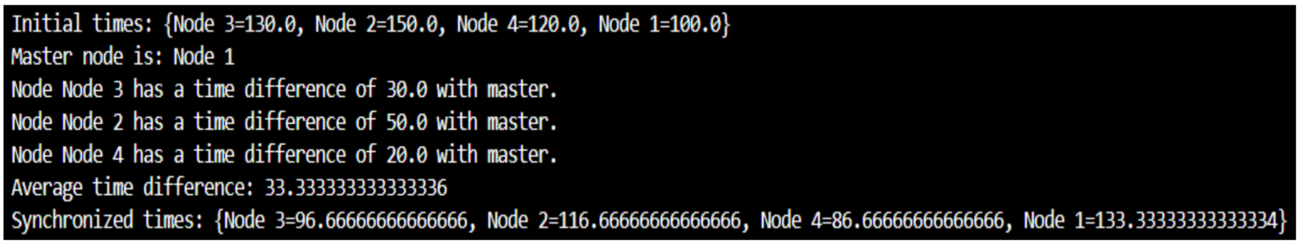
NAME: SHASHWAT SHAH
SAP ID: 60004220126
DIV/BATCH: C22

DISTRIBUTED COMPUTING (DC)
EXPERIMENT 08

AIM: To demonstrate clock synchronization algorithm using JAVA.

CODE:
```
import java.util.HashMap;
import java.util.Map;
import java.util.Random;

public class BerkeleyClockSync {

   // Function to adjust time based on average difference
   public static double adjustTime(double timeDiff, double nodeTime) {
      return nodeTime + timeDiff;
   }

   // Berkeley Clock Synchronization function
   public static Map<String, Double> berkeleyAlgorithm(Map<String, Double> nodeTimes) {
      // Randomly select a master node
      Random rand = new Random();
      Object[] nodes = nodeTimes.keySet().toArray();
      String masterNode = (String) nodes[rand.nextInt(nodes.length)];

      System.out.println("Master node is: " + masterNode);

      // Step 1: Master node calculates time differences
      Map<String, Double> timeDiffs = new HashMap<>();
      double masterTime = nodeTimes.get(masterNode);

      for (Map.Entry<String, Double> entry : nodeTimes.entrySet()) {
         String node = entry.getKey();
         double time = entry.getValue();
         if (!node.equals(masterNode)) {
            double timeDiff = time - masterTime;
            timeDiffs.put(node, timeDiff);
            System.out.println("Node " + node + " has a time difference of " + timeDiff + " with
master.");
         }
      }

      // Step 2: Calculate average time difference
      double avgDiff = timeDiffs.values().stream().mapToDouble(Double::doubleValue).sum() /
timeDiffs.size();
      System.out.println("Average time difference: " + avgDiff);

      // Step 3: Adjust time for all nodes
      for (String node : nodeTimes.keySet()) {
            if (!node.equals(masterNode)) {
                nodeTimes.put(node, adjustTime(-avgDiff, nodeTimes.get(node)));
            } else {
                // Master node adjusts its own time
                nodeTimes.put(node, masterTime + avgDiff);
            }
        }

        return nodeTimes;
    }

    public static void main(String[] args) {
        // Sample node times in seconds
        Map<String, Double> nodeTimes = new HashMap<>();
        nodeTimes.put("Node 1", 100.0);
        nodeTimes.put("Node 2", 150.0);
        nodeTimes.put("Node 3", 130.0);
        nodeTimes.put("Node 4", 120.0);

        System.out.println("Initial times: " + nodeTimes);

        // Synchronize using Berkeley Algorithm
        Map<String, Double> synchronizedTimes = berkeleyAlgorithm(nodeTimes);

        System.out.println("Synchronized times: " + synchronizedTimes);
    }
}
```

OUTPUT:



```
Initial times: {Node 3=130.0, Node 2=150.0, Node 4=120.0, Node 1=100.0}
Master node is: Node 1
Node Node 3 has a time difference of 30.0 with master.
Node Node 2 has a time difference of 50.0 with master.
Node Node 4 has a time difference of 20.0 with master.
Average time difference: 33.333333333333336
Synchronized times: {Node 3=96.66666666666666, Node 2=116.66666666666666, Node 4=86.66666666666666, Node 1=133.33333333333334}
```