1. (Graded by: Tomoko Sakurayama)
   Sequence of answers: c b b e a c d b a b

1.1. **Answer: c**
   PUE=Power Usage Effectiveness. Total facility power / IT Equipment power
   \[ \frac{800 \text{kWh}}{(800 - 200) \text{kWh}} = 1.33 \]

1.2. **Answer: b**
   The new heartbeat’s counter (89) is older than the current counter (95), hence
   the entry is not updated.

1.3. **Answer: b**
   Mapreduce attempts to schedule a map task on 1. a machine that contains a
   replica of corresponding input data, or failing that, 2. on the same rack as a
   machine containing the input, or failing that, 3. Anywhere
   The only machine on the same rack as a replica is S21.

1.4. **Answer: e**
   BitTorrent has a policy to download the local rarest block first. 4 is the rarest
   block in this case because it is only held by C and D.

1.5. **Answer: a**
   Let L(e) be the lamport timestamp of e. Let V(e) be the vector timestamp of e.
   Additionally, let V(e1) = V(e2) mean V(e1) is equal or incomparable to V(e1). We
   can now prove each of the statements to be correct or incorrect.
   a. We know L(e1) < L(e2). This means either e1 → e2 or e || e2. So, either V(e1) < V(e2) or V(e1) = V(e2). So, the statement is false.
   b. We know V(e1) < V(e2). So, e1 → e2. So, L(e1) < L(e2). As desired.
   c. We know V(e1) is equal to V(e2). WLOG, suppose e1 is on p1 and e2 is on p2.
   From the vector timestamp algorithm, V(e1)[p1] is equivalent to L(e1), and
   V(e2)[p2] is equivalent to L(e2). Since V(e1)[p1] is equal to V(e2)[p2], L(e1) = L(e2). As desired.
   d. We know L(e1) = L(e2). So e1 || e2. So, V(e1) = V(e2). As desired.

1.6. **Answer: c**
   Remember the rules of comparing two Vector timestamps. \[ \text{VectorTS(e2)} < \text{VectorTS(e1)} \].

1.7. **Answer: d**
   Because C100 is using Consistency Level ONE for reads, it may not receive the
   latest written value by itself or any other client written with CL QUORUM.

1.8. **Answer: b**
   C1 sends a push request to C45 containing the address at which C1 can accept
   incoming connections.
1.9. **Answer: a**
Increasing gossip timeout (Tfail, Tcleanup) increases detection time and reduces false positive rate, but bandwidth only depends on gossip frequency (which is not changed).

1.10. **Answer: b**
2. (Graded by: Ritwik Deshpande)

a.

In typed form (this is how we expected the answer written):

- **P0**: Snd(M01): 1, Snd(M02): 2, Rcv(M21): 3, Snd(M03): 4, Snd(M04): 5, Rcv(M24): 8, Rcv(M22): 9
- **P1**: E11: 1, Snd(M11): 2, RcvM02: 3

Common mistakes included: not starting with 0 timestamps, marking vector timestamps instead of Lamport timestamps (gulp!).

b. Concurrent with same Timestamp (P0, M01, 1) and (P3, E31, 1)
   Concurrent with different Timestamp (P0, M02, 2) and (P2, M21, 1)
3. (Graded by: Pengyu Lu)
   (a) Successors: 1->2->5->14->42->132->429->1430->1
   (b) 2048 points ⇒ m = 11
   Finger table entries for peer id 42:

<table>
<thead>
<tr>
<th>i</th>
<th>ft[i]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>132</td>
</tr>
<tr>
<td>1</td>
<td>132</td>
</tr>
<tr>
<td>2</td>
<td>132</td>
</tr>
<tr>
<td>3</td>
<td>132</td>
</tr>
<tr>
<td>4</td>
<td>132</td>
</tr>
<tr>
<td>5</td>
<td>132</td>
</tr>
<tr>
<td>6</td>
<td>132</td>
</tr>
<tr>
<td>7</td>
<td>429</td>
</tr>
<tr>
<td>8</td>
<td>429</td>
</tr>
<tr>
<td>9</td>
<td>1430</td>
</tr>
<tr>
<td>10</td>
<td>1430</td>
</tr>
</tbody>
</table>

   (c) Route a message for key (file id) 6 starting from peer id 42:
   42->1430->1->5->14 (see steps 1-4 below for details)
   1) Look for the largest successor/finger entry <= 6 in peer id 42’s FT: **hop to peer id 1430**
   2) Look for the largest successor/finger entry <= 6 in peer id 1430’s FT: (FT shown below): **hop to peer id 1**
   FTE for peer id 1430:

<table>
<thead>
<tr>
<th>i</th>
<th>ft[i]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
</tr>
</tbody>
</table>
3) Look for the largest successor/finger entry <= 6 in peer id 1’s FT: (FT shown below): **hop to peer id 5**
   FTE for peer id 1:

<table>
<thead>
<tr>
<th>i</th>
<th>ft[i]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>14</td>
</tr>
<tr>
<td>4</td>
<td>42</td>
</tr>
<tr>
<td>5</td>
<td>42</td>
</tr>
<tr>
<td>6</td>
<td>132</td>
</tr>
<tr>
<td>7</td>
<td>132</td>
</tr>
<tr>
<td>8</td>
<td>429</td>
</tr>
<tr>
<td>9</td>
<td>1430</td>
</tr>
<tr>
<td>10</td>
<td>1430</td>
</tr>
</tbody>
</table>

4) The largest successor/finger entry <= 6 does not exist. Hop to successor(5). That is, **hop to peer id 14**. File with key=6 is stored here.
4. (Graded by: Samarth Aggarwal)

**M1** (a, b):

```
// a: _
// b: \{timestamp, IP, URL\}

Emit (b.URL, (b.timestamp, b.IP))
```

**R1** (key, value):

```
// key: URL
// value: \{(timestamp1, IP1), (timestamp2, IP2), \ldots\}

Emit ( _ , ( |value|, key, value) )
```

**M2** (key, value):

```
// key: _
// value: \{ (visit_count, URL, visits = \{(timestamp1, IP1), (timestamp2, IP2), \ldots\}) \}

// Identity Map
Emit ( key , value )
```

**R2** (key, value):

```
// key: _
// value: \{(count1, URL1, visits1), (count2, URL2, visits2), \ldots\}

// sort in descending order by value.count
sorted_visits = sorted(value, key = lambda x: x.count, reverse=True)

// Emit top 10 from sorted list
for count, URL, visits in sorted_visits[:10]:
    Emit ( count, (URL, visits) )
```

**M3** (key, value):

```
// key: count of visits
// value: \{ (URL, \{(timestamp1, IP1), (timestamp2, IP2), \ldots\}) \}

URL, visits = value
```
for (time, ip) in visits:
    Emit ( ip, URL )

R3 (key, value):
    // key: IP
    // value: { URL1, URL2, ... }

    if | unique(value) | == 10:
        Emit ( __, key )
5. (Graded by: Matt Hokinson)
   A) All of the processes in an intersecting row/column can detect one another. Since there are \( M+K-1 \) unique processes in this ‘cross’, if all but one of these processes were to die, the remaining process would be able to detect the failure. Thus, \( L=M+K-1 \), and \( L-1=M+K-2 \) is the maximum number of failures allowed for completeness.
   B) \( M=4 \). \( K+M-2\geq 10 \). This gives us \( K+2\geq 10, K\geq 8 \)
   C) This is a trick question, as it’s impossible to guarantee accuracy in an asynchronous system with a SWIM/ping based protocol or a heartbeating protocol, because these protocols guarantee Completeness.