

Teodor Samoila - Alexandru Cotofan

teodor.samoila@solera.com ; alexandru.cotofan@solera.com

```
private boolean checkDimensions(CLIQUEUnit other, int e) {
    for(int i = 0, j = 0; i < e; i++, j += 2) {
        if (dims[i] != other.dims[i]
            || bounds[j] != other.bounds[j]
            || bounds[j + 1] != bounds[j + 1]) {
            return false;
        }
    }
    return true;
}
```

other.bounds[j+1]

Problem 1: 2 points

```
@Override
public double[] computeMean() {
    // Not supported except for singletons.
    return points.size() == 1 ? points.get(1) : null;
}
```

points.get[0]

Problem 2: 2 points

```
protected PreDeConModel computeLocalModel(DoubleDBIDList neighbors, ....) {
    final int referenceSetSize = neighbors.size();
    ....
    // Shouldn't happen:
    if(referenceSetSize < 0) {
        LOG.warning("Empty reference set –
            should at least include the query point!");
        return new PreDeConModel(Integer.MAX_VALUE, DBIDUtil.EMPTYDBIDS);
    }
    ....
    for(int d = 0; d < dim; d++) {
        s[d] /= referenceSetSize;
        mvVar.put(s[d]);
    }
    ....
}
```

referenceSetSize < 1

Problem 3: 2 points

```
public String getStrippedSubstring() {
    int sstart = start, ssend = end;
    while(sstart < ssend) {
        char c = input.charAt(sstart);
        if(c != ' ' || c != '\n' || c != '\r' || c != '\t') {
            break;
        }
        ++sstart;
    }
    ....
}
```

replace || with &&

Problem 4: 2 points

```

public static final ByteSequence prefixEnd0f(ByteSequence prefix) {
    byte[] endKey = prefix.getBytes().clone();
    for (int i = endKey.length - 1; i >= 0; i--) {
        if (endKey[i] < 0xff) {
            endKey[i] = (byte) (endKey[i] + 1);
            return ByteSequence.from(Arrays.copyOf(endKey, i + 1));
        }
    }
    return ByteSequence.from(NO_PREFIX_END);
}

```

Problem 5: 3 points

```

private static byte char64(char x) {
    if ((int)x < 0 || (int)x > index_64.length)
        return -1;
    return index_64[(int)x];
}

```

(int)x >= index_64.length.

Problem 6: 2 points

```

@Override
public void putToCache(PutRecordsRequest putRecordsRequest)
{
    ....
    if (dataTmpFile == null || !dataTmpFile.exists())
    {
        try
        {
            dataTmpFile.createNewFile();
        }
        catch (IOException e)
        {
            LOGGER.error("Failed to create cache tmp file, return.", e);
            return;
        }
    }
    ....
}

```

if (dataTmpFile != null && !
dataTmpFile.exists())

Problem 7: 2 points

```
private void shiftRightDestructive(int wordShifts,
                                   int bitShiftsInWord,
                                   boolean roundUp)
{
    if (wordShifts == 0 && bitShiftsInWord == 0) {
        return;
    }

    assert (wordShifts >= 0);
    assert (bitShiftsInWord >= 0);
    assert (bitShiftsInWord < 32);
    if (wordShifts >= 4) {
        zeroClear();
        return;
    }

    final int shiftRestore = 32 - bitShiftsInWord;

    // check this because "123 << 32" will be 123.
    final boolean noRestore = bitShiftsInWord == 0;
    final int roundCarryNoRestoreMask = 1 << 31;
    final int roundCarryMask = (1 << (bitShiftsInWord - 1));
    ....
}
```

Problem 8: 5 points

```
public void logSargResult(int stripeIx, boolean[] rgsToRead)
{
    ....
    for (int i = 0, valOffset = 0; i < elements; ++i, valOffset += 64) {
        long val = 0;
        for (int j = 0; j < 64; ++j) {
            int ix = valOffset + j;
            if (rgsToRead.length == ix) break;
            if (!rgsToRead[ix]) continue;
            val = val | (1 << j);
        }
        ....
    }
    ....
}
```

long(1)<<j

Problem 9: 5 points

```
public void testSignSHA256CompleteEvenHeight2() {
    ....
    int height = 10;
    ....
    for (int i = 0; i < (1 << height); i++) {
        byte[] signature = xmss.sign(new byte[1024]);
        switch (i) {
            case 0x005b:
                assertEquals(signatures[0], Hex.toHexString(signature));
                break;
            case 0x0822:
                assertEquals(signatures[1], Hex.toHexString(signature));
                break;
            ....
        }
    }
}
```

is greater than 1024, won't work

Problem 10: 3 points

```

@Override
public boolean equals(Object o)
{
    ....
    CheckpointStatistics that = (CheckpointStatistics) o;
    return id == that.id &&
        savepoint == that.savepoint &&
        triggerTimestamp == that.triggerTimestamp &&
        latestAckTimestamp == that.latestAckTimestamp &&
        stateSize == that.stateSize &&
        duration == that.duration &&
        alignmentBuffered == that.alignmentBuffered &&
        processedData == processedData &&
        persistedData == that.persistedData &&
        numSubtasks == that.numSubtasks &&
        numAckSubtasks == that.numAckSubtasks &&
        status == that.status &&
        Objects.equals(checkpointType, that.checkpointType) &&
        Objects.equals(
            checkpointStatisticsPerTask,
            that.checkpointStatisticsPerTask);
}

```

proccssedData == processedData,
should have been that.processedData

Extra problem: The team that does not find it loses 2 points