

1 point

1. Suppose a web log is modified to now have a sixth piece of information, a priority, that can be represented as a String.

Which one of the following is the **least** likely change to the LogEntry class to accommodate this new part of a web log?

- ☒ The **toString** method is modified to include a String parameter.
- ☐ A new private variable named **priority** is created.
- ☐ A new String field is initialized in the constructor.
- ☐ A new String parameter is added to the constructor.
- ☐ A new **getPriority** method is created to return the priority.
- ☐ The **toString** method is modified to include the new priority as part of the return String.

1 point

2. Consider the following code for the **readFile** method of the **LogAnalyzer** class.

```
1 public void readFile(String filename) {  
2     FileResource fr = new FileResource(filename);  
3     for (String line : fr.lines()) {  
4         LogEntry le = WebLogParser.parseEntry(line);  
5     }  
6 }
```

In the **Tester** class, **readFile** is called with a correct filename, and then **printAll** is called, but nothing is printed.

Which one of the following is likely the best reason why?

- ☐ In **readFile**, a **System.out.println** statement is missing from the body of the for loop.
- ☒ In **readFile**, a **System.out.println** statement is missing that should be right before the for loop.
- ☐ In **readFile**, the log entries were not stored in **records**.
- ☐ In **readFile**, a **System.out.println** statement is missing that should be right after the for loop.
- ☐ In **readFile**, the wrong parameter is sent to **parseEntry**.

1 point

3. Consider the following code for the method **printAllHigherThanNum** with one integer parameter **num**. This method should print all the logs that have a status code higher than **num**.

Which one of the following would be the best choice for suitable code for this method?

- ☐

```
1 if (le.getStatusCode() > num) {  
2     for (LogEntry le : records) {  
3         System.out.println(le);  
4     }  
5 }
```
- ☒

```
1 for (LogEntry le : records) {  
2     if (le.getStatusCode() > num) {  
3         System.out.println(le);  
4     }  
5     else {  
6         System.out.println();  
7     }  
8 }
```
- ☐

```
1 if (le.getStatusCode() > num) {  
2     for (LogEntry le : records) {  
3         System.out.println(le);  
4     }  
5 }  
6 else {  
7     System.out.println();  
8 }
```
- ☐

```
1 for (LogEntry le : records) {  
2     if (le.getStatusCode() > num) {  
3         System.out.println(le);  
4     }  
5 }
```
- ☐

```
1 if (le.getStatusCode() > num) {  
2     for (LogEntry le : records) {  
3         if (le.getStatusCode() > num) {  
4             System.out.println(le);  
5         }  
6     }  
7 }
```

1 point

4. Run the method **countUniqueIPs** on the file **weblog2_log**.

How many unique IP addresses are in the file?

45

1 point

5. Run the method **uniqueIPVisitsOnDay("Sep 27")** on the file **weblog2_log**.

What size is the ArrayList that is returned?

14

1 point

6. Run the method **countUniqueIPsInRange(400,499)** on the file **weblog2_log**.

What number is returned?

40

1 point

7. Run the method **mostNumberVisitsByIP** after a HashMap has been created from the method **countVisitsPerIP** on the file **weblog2_log**.

What number is returned?

63

1 point

8. Run the method **iPsMostVisits** after a HashMap has been created from the method **countVisitsPerIP** on the file **weblog2_log**.

What single IP address is returned in the ArrayList?

- ☐ 103.57.41.178
- ☒ 188.162.84.63
- ☐ 200.69.213.251
- ☐ 210.4.104.99
- ☐ 212.128.74.248

1 point

9. Run the method **dayWithMostIPVisits** with a HashMap has been created from the method **iPsForDays** on the file **weblog2_log**.

What day is returned?

- ☒ Sep 24
- ☐ Sep 26
- ☐ Sep 28
- ☐ Sep 30

1 point

10. Run the method **iPsWithMostVisitsOnDay** with two parameters—one, a HashMap that has been created from the method **iPsForDays** on the file **weblog2_log** and two, the day "Sep 30".

Two IP addresses are returned in the ArrayList—which are they?

- ☐ 66.67.61.44
- ☐ 103.57.41.178
- ☐ 106.220.155.36
- ☐ 188.162.84.63
- ☐ 210.4.104.99
- ☒ 212.128.74.248

☒ I, **Ning Zheng**, understand that submitting work that isn't my own may result in permanent failure of this course or deactivation of my Coursera account.

[Learn more about Coursera's Honor Code](#)

Submit Quiz