Score: 2.500 (=100.0%)
Id: 42253
Napišite klasu `LogProcessor` koja treba imate sljedeći konstruktor i metode:

```
/**
 * Creates object and set file.
 * @param file file name
 */
public LogProcessor(Path file)

/**
 * Load file and populate list of loaded objects (LogEntry).
 * @throws IOException if can not load file
 */
public void load() throws IOException

/**
 * Returns list of log entries.
 * @return list of log entries
 */
public List<LogEntry> getLogs()

/**
 * Extracts elements of line and creates LogEntry.
 * @param line one line og log file
 * @return created log entry object
 */
public static LogEntry parseLog(String line)
```

Potrebno je napisati i klasu `LogEntry` koja ima konstruktor `public LogEntry(String time, String level, String thread, String text)` u kojem se inicijaliziraju svi atributi i gettere za svaki atribut. Klasa treba imati sljedeće atribute: `time`, `level`, `thread` i `text`. Svi atributi su stringovi.

Ako učitavamo sljedeći redak iz datoteke:

```
2020-05-12 05:43:10.370  INFO [AuthenticationAuthorizationManager,,,] 1 --- [           main] s.c.a.AnnotationConfigApplicationContext : Some text ...
```

Dijelovi retka će biti uvijek na istim pozicijama.

iz toga treba stovoriti objekt za sljedećim vrijednostima u atributima:

- time: `"2020-05-12 05:43:10.370"`
- level: `"INFO"`
- thread: `"main"`
- text: `"Some text ..."`

Primjer korištenja:

```
Path inputFile = Path.of("log.txt");
LogProcessor processor = new LogProcessor(inputFile);
processor.load();
List<LogEntry> logs = processor.getLogs();
```

Napomena: Sve klase i sučelja imaju vidljivost postavljenu na *pachoge private*.

Napomena: Prije predaje iz Vašeg koda izbacite sve ispise na standardni izlaz kako bi testovi ispravno radili.

Student's answer:

```
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Path;
import java.util.LinkedList;
import java.util.List;

class LogProcessor {
    private Path file;
    private List<LogEntry> list = new LinkedList<>();
    /**
     * Creates object and set file.
     * @param file file name
     */
    public LogProcessor(Path file) {
        this.file = file;
    }

    /**
     * Load file and populate list of loaded objects (LogEntry).
     * @throws IOException if can not load file
     */
    public void load() throws IOException {
        if(!Files.exists(file)) {
            throw new  IOException();
        }
        List<String> lines = Files.readAllLines(file);
        for(String line : lines) {
            if(line.length() > 0) list.add(parseLog(line));
        }
    }

    /**
     * Returns list of log entries.
     * @return list of log entries
     */
    public List<LogEntry> getLogs() {
        return list;
    }

    /**
     * Extracts elements of line and creates LogEntry.
     * @param line one line og log file
     * @return created log entry object
     */
    public static LogEntry parseLog(String line) {
        String[] data = line.split("\\s+");
        String time = data[0] + " " + data[1];
        String level = data[2];
        // 3 , 4 , 5 , 6 preskacem
        line = line.substring(line.indexOf("]") + 1);
        String thread = line.substring(line.indexOf("[") + 1, line.indexOf("]")).
        // 8 , 9 preskacem

        String text = line.substring(line.lastIndexOf(":") + 1).trim();
        return new LogEntry(time, level, thread, text);
    }
}

class LogEntry {
    private String time;
    private String level;
    private String thread;
    private String text;

    public LogEntry(String time, String level, String thread, String text) {
        this.time = time;
        this.level = level;
        this.thread = thread;
        this.text = text;
    }

    public String getTime() {
        return time;
    }

    public String getLevel() {
        return level;
    }

    public String getThread() {
        return thread;
    }

    public String getText() {
        return text;
    }
}
```

Correct answer:

```
This test does not have "show solutions" option enabled.
```

Hint: Correct. Well done!

Exam results obtained during submission evaluation:

#	stdin	stdout	expected	stderr	percentage	hint	mode	correct?
1	loading	OK	OK	EdgarLibrary v0.11.1	30.00	Correct. Well done!	check elements order : false, case sensitive : false, ignore whitespace : true	true
2	checking	OK	OK	EdgarLibrary v0.11.1	30.00	Correct. Well done!	check elements order : false, case sensitive : true, ignore whitespace : true	true
3	parsing	OK	OK	EdgarLibrary v0.11.1	40.00	Correct. Well done!	check elements order : false, case sensitive : true, ignore whitespace : true	true

Student's result | Correct result

Rerun student's code