# Problem Statement: Log Parsing

Let’s build a basic request log parsing application. You will be given CSV files of the following structure:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| timestamp | url | method | response\_time | response\_code |
| 1581333404 | /person/all | GET | 124 | 200 |
| 1581333441 | /person/add | POST | 283 | 201 |
| 1581333450 | /book/2 | GET | 90 | 200 |
| … | | | | |

## Implement the following

* Using a single input file
  1. URL with the Top 5 highest throughput (highest number of times called) along with the frequency (Example output: Refer output #1 in the next section)
  2. Max, Min and average response times for each distinct api (Example output: Refer output #2 in the next section)

### Notes

* You should mask all integers in the url as {id}. Examples for masked URL are as follows:
  + Actual URL: “/book/2”

Masked URL: “/book/{id}”

* + Actual URL: “/v2/person/223/details”

Masked URL: “/v2/person/{id}/details”

* Output can be written to a file, or STDOUT

### 

### Reference output format:

1. Top 5 highest throughput URLs:

|  |  |  |
| --- | --- | --- |
| Method | URL | Frequency |
| PUT | /book/{id} | 7 |
| GET | /book/{id} | 5 |
| POST | /person/add | 4 |
| GET | /book/{id}/return | 4 |
| GET | /person/all | 3 |

1. Time taken for each endpoint:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Method | URL | Min Time | Max Time | Average Time |
| PUT | /book/{id} | 20 | 234 | 98.57 |
| GET | /book/{id} | 37 | 110 | 65.4 |
| POST | /person/add | 60 | 140 | 97.25 |
| GET | /book/{id}/return | 45 | 78 | 63 |
| GET | /person/all | 60 | 102 | 86.67 |
| GET | /person/{id}/details | 35 | 87 | 66.67 |
| Note: Results for all URLs have to be shown here... | | | | |

### What are we looking for:

(in order of priority)

* No compilation errors (irrelevant if you’re coding in an interpreted language)
* Functionally correct code
* OOPs concepts. Separation of concerns
* Extensibility
* Speed of execution
* Unit tests