Metric Parsing Code Challenge

1.1. Background

This challenge is based in the metric ingress interface for Arden where metrics are received via a TCP network connection using a simple text based protocol / format. The metrics are accepted via the TCP connection, parsed and then validated against a set of rules that apply checks and restrictions on the metric name and / or tag names and values based on the sender of the metric. Valid metrics are converted to JSON and streamed on to a Kafka topic. All of the metric formats described are parsed into a common metric object format so that they will encode to a common consistent JSON structure.

This challenge is simplified to the extent of accepting metrics from a file and restricts validation to only accepting well formed metrics.

1.2. The Challenge

Write an application that can read text string metrics from a file and parse them into a metric "object" - this could be an object or a dictionary depending on language. The input file may contain a mixture of the metric formats defined below.

Each metric line will be terminated with a linefeed character (0x0a), and optionally preceded by a carriage return character(0x0d).

The metrics must conform to one of the formats defined below. Metrics not conforming with the rules must be rejected.

Metric names, tag names and tag values must all be converted to lowercase.

Metric strings will be encoded using ISO-8859-1 (Latin-1).

1.2.1. Basic Graphite Format

The metric format consists of 3 fields separated from each other by a single space character.

The first field is the metric name which consists of one or more "words" separated by the dot (.) character.

Each "word" may only consist of the following characters: upper and lowercase letters, digits and the following punctuation characters '_', '+', '%' and '-'. The metric name must start with a letter.

The second field is the metric value and may be an signed or unsigned long or floating point (64 bit) value; floating point values may be represented using exponents (e.g. 1.65e4) and the values "inf", "+inf", "-inf", "infinity", "+infinity", "-infinity" and "NaN" are valid double values, and should be interpreted as case-insensitive, as should the 'e' in double exponential notation.

The third field is the timestamp represented as the integer number of seconds or milliseconds since January 1, 1970 (midnight UTC/GMT) - Unix epoch time with an extension for milliseconds format.

Example:

some.metric.name 1234 1562763195

another.metric.value.cpu% 1.23e-1 1562763195000

1.2.2. Tagged Graphite Format

This format extends the previous format by permitting the metric name to be composed of a basic name and a number of "tags" represented as name / value pairs.

Tags are separated from the base metric name and from each other by a ';' character. The value of a tag is separated from the tag name by the '=' character.

Tag names may include the following characters: upper and lowercase letters, digits and the following punctuation characters '_', '+', '%' and '-' and must start with a letter.

Tag values may include the following characters: upper and lowercase letters, digits and the following punctuation characters '_', '+', '%' and '-'.

The basic metric name, value and timestamp as as per the Basic Graphite Format.

Examples:

some.metric.value;host=somehost;application=apache 1234 1562763195

1.2.3. Carbon 2.0 Format

The Carbon 2.0 format changes the format to consist of 4 fields.

The first field is a set of intrinsic tags (those that would be used by the metric system to identify the metric) "tags", name value pairs with each tag separated from the next by a single space; tag names and values are separated by the '=' character.

The second field consists of "extrinsic" tags in the same format as the previous tags. These tags can provide meta-data about the metric; they wouldn't become part of the metric name, but convey additional useful information. This field must be preceded by two consecutive space characters. This field is optional and may not be present.

In both cases, the tag name consists of the following characters: upper and lowercase letters, digits and the following punctuation characters '_', '+', '%' and '-' and must start with a letter.

In both cases, the tag value consists of the following characters: upper and lowercase letters, digits and the following punctuation characters '_', '+', '%', '-' and '/'.

The metric value and timestamp are in the same format as the graphite metrics, with the metric value preceded by a single space.

Carbon 2.0 format metrics must include the intrinsic tags "unit" and "mtype".

site=mydomain mtype=rate unit=Req/s host=web12 agent=statsdaemon1 234 1560852124

1.2.4. Examples

1.2.4.1. Valid Metric Examples - the trailing newline is not depicted

```
test_metric.value 100 1560852124
test_metric.value -100.25 1560852124
test_metric.value 0.25 1560852124
test_metric.value;label1=value1;label2=value2 Inf 1560852124000
site=mydomain mtype=rate unit=Req/s host=web12  agent=statsdaemon1 234 1560852124
site=mydomain mtype=rate unit=Req/s host=web12 234 1560852124
```

1.2.4.2. Invalid Examples - the trailing newline is not depicted

```
test_metric<.value 100.25 1560852124
test_metric.value; 10.25 1560852124
test_metric.value 100.25
test_metric.value;7label=wibble 100.25 1560852124
site=mydomain mtype==rate unit=Req/s host=web12  agent=statsdaemon1 234 1560852124
site=mydom|ain mtype=rate unit=Req/s host=web12  agent=statsdaemon1 234 1560852124
```