Práctica 2 - Análisis sintáctico

Módulo I – Lenguaje Estructurado

Procesamiento de Lenguaje Escrito

Grado en Ciencia e Ingeniería de Datos

2022/2023

Profesores: Carlos Dafonte – carlos.dafonte@udc.es
Daniel Garabato – daniel.garabato@udc.es

Departamento de Ciencias de la Computación y Tecnologías de la Información
Universidade da Coruña

Diseño e implementación mediante la herramienta `SLY` de un analizador sintáctico para el tratamiento de ficheros GPX. Se trata de un fichero en formato XML ampliamente utilizado para registrar información asociada a rutas GPS. El objetivo de la práctica consiste en llevar a cabo su conversión a formato JSON y, además, calcular algunos datos estadísticos básicos.

A continuación se muestra, a modo de ejemplo, un fragmento de este tipo de ficheros:

```
<?xml version="1.0" encoding="UTF-8"?>
<gpx creator="Garmin Connect" version="1.1"
    xmlns="http://www.topografix.com/GPX/1/1">
  <metadata>
    <link href="connect.garmin.com">
      <text>Garmin Connect</text>
    </link>
    <time>2019-05-17T09:07:55.000Z</time>
  </metadata>
  <trk>
    <name>Duatlon Baio</name>
    <type>mountain_biking</type>
    <trkseg>
      <trkpt lat="43.14609627239406" lon="-8.956256993114948">
        <ele>166</ele>
        <time>2019-05-17T09:39:16.000Z</time>
        <extensions>
          <ns3:TrackPointExtension>
            <ns3:atemp>fadsf</ns3:atemp>
            <ns3:hr>173</ns3:hr>
            <ns3:cad>73</ns3:cad>
          </ns3:TrackPointExtension>
        </extensions>
      </trkpt>
      <trkpt lat="43.14616559073329" lon="-8.956147525459528">
        <ele>166</ele>
        <time>2019-05-17T09:39:19.000Z</time>
        <extensions>
          <ns3:TrackPointExtension>
            <ns3:atemp>11.0</ns3:atemp>
            <ns3:hr>zz</ns3:hr>
            <ns3:cad>80</ns3:cad>
          </ns3:TrackPointExtension>
        </extensions>
      </trkpt>
      <trkpt lat="43.14621252939105" lon="-8.956068065017462">
        <ele>166</ele>
        <time>2019-05-17T09:39:21.000Z</time>
        <extensions>
          <ns3:TrackPointExtension>
            <ns3:atemp>11.0</ns3:atemp>
```

```
            <ns3:hr>182</ns3:hr>
            <ns3:cad>xx</ns3:cad>
          </ns3:TrackPointExtension>
        </extensions>
      </trkpt>
    </trkseg>
  </trk>
</gpx>
```

En primer lugar, debéis identificar los *tokens* y, en base a ellos, definir una gramática que permita representar el formato de estos documentos. Así mismo, tendréis que implementar las acciones semánticas que os permitan generar el JSON correspondiente y calcular una estadística simple (contador, valor mínimo, valor máximo y valor medio) para los parámetros: altitud, ritmo cardíaco, cadencia y temperatura. Por último, es necesario verificar, en la medida de lo posible, la validez de los campos. Por ejemplo, no estaría permitido un valor textual para la altitud o un valor negativo para el ritmo cardíaco. En caso de detectar un error en el fichero de entrada (p.e. ausencia de un *tag* de apertura o formato no permitido para un determinado campo), éste deberá reportarse adecuadamente, indicando la línea en que se ha producido, junto con información acerca del tipo de error.

A continuación, se muestra el formato que debe tener la salida generada:

```
{
  "name": "Duatlon Baio",
  "type": "mountain_biking",
  "trackpoints": [
  {
    "latitude": 43.14609627239406,
    "longitude": -8.956256993114948,
    "elevation": 166,
    "datetime": "2019-05-17T09:39:16.000Z",
    "heart-rate": 173,
    "cadence": 73
  },
  {
    "latitude": 43.14616559073329,
    "longitude": -8.956147525459528,
    "elevation": 166,
    "datetime": "2019-05-17T09:39:19.000Z",
    "temperature": 11.0,
    "cadence": 80
  },
  {
    "latitude": 43.14621252939105,
    "longitude": -8.956068065017462,
    "elevation": 166,
```

```
    "datetime": "2019-05-17T09:39:21.000Z",
    "temperature": 11.0,
    "heart-rate": 182
  }
  ],
  "stats": {
    "elevation": {
      "count": 3,
      "min": 166,
      "max": 166,
      "avg": 166.0
    },
    "heart-rate": {
      "count": 2,
      "min": 173,
      "max": 182,
      "avg": 177.5
    },
    "cadence": {
      "count": 2,
      "min": 73,
      "max": 80,
      "avg": 76.5
    },
    "temperature": {
      "count": 2,
      "min": 11.0,
      "max": 11.0,
      "avg": 11.0
    }
  },
  "errors": [
  "Valor de temperatura no válido ('fadsf') en la línea 22",
  "Valor de ritmo cardíaco no válido ('zz') en la línea 34",
  "Valor de cadencia no válido ('xx') en la línea 46"
  ]
}
```


Este formato debe respetarse estrictamente, incluyendo los nombres de los elementos. Lo único que debe variar es el contenido, que deberá ajustarse al fichero de entrada proporcionado. Por otro lado, los mensajes de error incluídos arriba son únicamente a modo de ejemplo y podéis definirlos como mejor consideréis.

Junto con el enunciado, se proporciona un esqueleto en Python (`p2_base.py`) del que tendréis que partir y cuya sección `__main__` **NO debéis modificar**. En ella se lee íntegramente el fichero GPX proporcionado como entrada por el sistema estándar. Así mismo, procesa la entrada con el analizador léxico y, posteriormente, se le propor-

ciona al analizador sintáctico. La impresión del mensaje debe realizarse directamente desde el axioman de la gramática o, en su caso, desde la función de error.

Nota: los errores deben incluirse siempre como parte del JSON de salida y nunca como mensajes aislados.

Para el desarrollo de esta práctica emplearse cualquier módulo estándar en Python, como `re`, `sys` o `json`. Adicionalmente también podéis hacer uso de `pandas` si lo consideráis necesario.