Programação 2

Enunciado do trabalho — Ano lectivo 2017-18

Grupos de **3** alunos — Cotação: 4 valores

Data-limite de entrega: 30 de Maio às 18h00m

Instruções para a entrega:

- Entregar apenas este ficheiro: **myPyGPX.py**
  [com informação extra no nome do ficheiro, explicado abaixo]
- O ficheiro deve estar identificado nas linhas iniciais com o número do grupo e com o número e nome de cada um dos elementos do grupo; essa identificação deve ser feita usando comentários com o símbolo **#**.
- Antes de entregar o ficheiro, mudar o seu nome para
  myPyGPX_XXX_fcXXXXX.py
  onde **XXX** é o *número do grupo* e **fcXXXXX** é a referência ao *número do aluno que submete o trabalho no moodle*.
  - exemplo: **`myPyGPX_009_fc90900.py`** se for o aluno 90900 a submeter o trabalho no moodle, e o seu grupo for o nº 9
- Fazer upload do ficheiro no moodle, quando a página de entrega estiver aberta.

Requisitos do programa:

Assegure-se de que o módulo **myPyGPX** não contém erros sintácticos e de que o programa de teste **client_for_myPyGPX** executa correctamente com o Python 3.6 no ambiente Windows dos laboratórios.

Os programas serão avaliados pela correcção, mas também pela estrutura do código e pela formatação. Procurem não ultrapassar as 80 colunas por linha. Comentem o vosso código onde isso for útil.

Tema

Neste trabalho concretiza-se um conjunto de ferramentas para processar dados obtidos de ficheiros **GPX**. O acrónimo **GPX** é a versão simplificada de **GP**S e**X**change format. É um esquema **XML** que estabelece um padrão de formato de dados com componentes geográficas e outras. Os dados são sobretudo oriundos de aparelhos GPS, ou destinados a estes aparelhos. O formato padrão facilita a codificação de *software* para processar os dados.

As entidades ou conceitos de dados presentes nos ficheiros GPX não são tratados de maneira completamente uniforme pelos fabricantes de aparelhos GPS, nem pelos produtores de *software*. Assim, assumimos algumas convenções para efeitos de processamento, as quais são usadas neste trabalho. Em particular, mapeamos os conceitos relevantes na seguinte estrutura de *classes Python*:

`GPXDocument` — guarda toda a informação relevante, lida de um ficheiro GPX

`Time` — um tipo de dados que exprime *tempo*, com definições próprias deste trabalho

`Point` — um ponto com coordenadas (latitude, longitude) e "elevação" (ou seja, altitude)

- `RoutePoint` — um ponto especializado, com informação descritiva adicional
  - `WayPoint` — um RoutePoint especializado
- `TrackPoint` — um ponto especializado, com informação adicional útil para processar *tracks*

`ListPoints` — um simples contentor de pontos (ou seja, objectos Point) em estrutura de lista

- `Route` — uma lista de RoutePoint
- `TrackSeg` — uma lista de TrackPoint

`Track` — uma lista de TrackSegment; um *track* contém pelo menos 1 *track segment*

`Analyse` — oferece um serviço com operações extra para analisar dados

`Plot` — oferece um serviço de visualização de dados em gráficos

O foco do nosso trabalho é a análise e processamento de *tracks* lidos de ficheiros GPX. Esses ficheiros também podem conter *routes* e *waypoints*, mas não lhes damos importância.

A classe **Track** está a negrito pois é aquela onde estão definidos quase todos os métodos solicitados aos alunos.

Adicionalmente, os alunos devem implementar 2 métodos da classe **Analyse**.

Uma etapa crucial na análise dos dados é o *parsing* (interpretação) do texto contido nos ficheiros GPX, usando esse texto para instanciar os objectos do nosso domínio (cada um dos TrackPoint, um Track, etc.). Os alunos não têm de se preocupar em definir o *parser*, pois esse serviço é fornecido pelo módulo externo **GPXparser** já concretizado.

Outros aspectos conceptuais e de implementação serão discutidos com os alunos em 4 aulas teóricas de apresentação e apoio ao trabalho, a partir de 14 de Maio.

Trabalho a realizar pelos alunos

O trabalho consiste em definir ("implementar") os seguintes métodos no ficheiro **myPyGPX.py**:

- da classe Track:
  - produceXYdata
  - _computeSpeedForEachTrackPoint
  - hidePartOfTrack
  - totalTime
  - totalDistance
  - totalAccumulatedElevation
  - averageSpeed
- da classe Analyse:
  - secondsToHoursMinSec
  - paceDecimalMinutesToMinSec

Não é preciso re-implementar os métodos que são fornecidos aos alunos.

Aliás, ***não é mesmo permitido alterar o código que é fornecido e que está finalizado***.

Pode definir outros métodos e funções auxiliares caso considere útil.

Todos os métodos a definir devem estar de acordo com a respectiva documentação, ou seja, ***é preciso respeitar os contratos.***

O código-fonte dos métodos pedidos deve ser inserido no ficheiro **myPyGPX.py** onde actualmente está a instrução `pass`.

Se os métodos estiverem correctamente definidos, a execução do módulo de teste **client_for_myPyGPX** fornecido aos alunos deve

- completar-se sem levantar qualquer excepção;
- enviar para o output standard texto igual ao que está registado no ficheiro **output.txt.**

Como obter documentação sobre os métodos a implementar?

No ficheiro **myPyGPX.py**, ler as *docstrings* bem como alguns detalhes de implementação indicados em comentários a seguir ao símbolo **#**.

A análise do módulo de teste **client_for_myPyGPX** permite ganhar alguma compreensão adicional sobre as classes e métodos, pois mostra exemplos de *uso* dos mesmos.

Como saber se o meu programa está correcto?

Os alunos são responsáveis por fazer os testes necessários para verificar que os métodos obedecem aos contratos.

Um desses testes (mas não o único) consiste em comparar o output de **client_for_myPyGPX** que importa as definições do módulo **myPyGPX** editado pelos alunos, com o output fornecido pelos docentes.

Por exemplo, nas fases finais do trabalho, pode-se guardar num ficheiro de texto o output obtido e verificar se o mesmo já é igual ao pretendido, usando o comando **FC** (**File Compare**) numa janela de comandos do Windows. A sintaxe do comando é aqui ilustrada. Neste caso, **output.txt** e **output_2.txt** são iguais, mas **output_3.txt** difere numa linha (**86:03:09** em vez de **86:03:10**).

```
C:\Users\csl\Desktop\pasta_de_teste>FC output.txt output_2.txt
Comparing files output.txt and OUTPUT_2.TXT
FC: no differences encountered

C:\Users\csl\Desktop\pasta_de_teste>FC output.txt output_3.txt
Comparing files output.txt and OUTPUT_3.TXT
***** output.txt
309789.4 segundos corresponde a 86:03:09
309789.9 segundos corresponde a 86:03:10

***** OUTPUT_3.TXT
309789.4 segundos corresponde a 86:03:09
309789.9 segundos corresponde a 86:03:09

*****
```

Poderão ainda ser fornecidos outros módulos de teste com testes adicionais.

Nota sobre o formato do output

Uma vez que é examinado o *output* de um programa que depende de código escrito pelos alunos, e esse *output* é textual, determina-se que

> *a resolução apresentada pelos alunos deve obedecer exactamente ao exemplo dado no ficheiro **output.txt** (ou outros que venham a ser fornecidos aos alunos com exemplos extra de execução).*