Projeto da disciplina Programação Imperativa

2018.2 T06

O objetivo é construir um programa que processe dados similares aos gerados por um relógio de corrida.
O programa receberá um arquivo de entrada referente aos dados de uma corrida. O arquivo é de tipo texto e conterá uma sequência de mensagens, cada uma contendo dados sobre a corrida. Cada mensagem contém um *timestamp* (marca de tempo). As mensagens são classificadas em três categorias:

- **Eventos.** Um evento podem ser início, fim, pausa ou reinício.
- **Registros.** Provêm dados com resolução relativamente precisa sobre a corrida. Os dados podem ser: longitude e latitude (posição geográfica), altitude (geográfica), batimentos cardíacos por minuto e número de passos desde o início da atividade. Mensagens do tipo registro sempre estão em ordem cronológico.
- **Lap.** Laps permitem quebrar uma corrida em segmentos de interesse. Na prática, eles podem ser baseados em distância, tempo, ações do usuário (aperto de um botão), etc.

Descrição do arquivo de entrada

Dentro de uma linha, os dados estão separados por caractere branco.

O arquivo de entrada contém um cabeçalho e uma sequência de mensagens. O cabeçalho é uma linha contendo dois *timestamps*, o primeiro indicando o início e o outro indicando o fim da atividade.
Cada mensagem é descrita em várias linhas. Na primeira linha vem o tipo da mensagem e o *timestamp* da mensagem. O tipo da mensagem é descrito com um caractere: **'e'**, **'r'**, ou **'l'**, significando que a mensagem é um evento, registo, ou lap, respectivamente.
No caso da mensagem ser um evento, a próxima linha terá um caractere **'i'**, **'f'**, **'p'** ou **'r'** para indicar início, fim, pausa ou reinício, respectivamente.
No caso da mensagem ser um registro, cada uma das seguintes linhas conterá um caractere e um dado segundo a seguinte tabela

caractere	**descrição**
'n'	longitude
'l'	latitude
'a'	altitude
'b'	batimentos cardíacos por minuto (bpm)

'p'	número de passos desde o início da atividade

O final de um registro é indicado por uma linha contendo **“#”.**

Funcionalidade do programa

O programa deve oferecer funcionalidades para:

1. Apresentar a data, horário e duração da corrida.
2. Apresentar um resumo da corrida contendo: distância total percorrida, tempo total, ritmo (*pace*) médio em mins/km, média de batimentos por minuto (bpm) ponderada pelo tempo, bpm máxima, bpm mínima, cadência média de passos (número de passos por minuto) e as altitudes máxima e mínima, relativas a altitude inicial.
3. Apresentar um resumo de cada quilômetro da corrida contendo: tempo, ritmo médio em mins/km, cadência média, média ponderada de bpm e o ganho (ou perda) de altitude.
4. Para os dados apresentados em 2 e 3 permitir considerar ou desconsiderar períodos de pausa.
5. Se a atividade contiver laps, apresentar um resumo de cada lap contendo: tempo, ritmo médio, média de bpm, bpm máxima e bpm mínima, ambas junto com o momento em que foi alcançado (em horas e minutos). Ignore as mensagens relacionadas com pausas.
6. Apresentar gráfico de percurso, marcando início, fim e cada quilômetro percorrido.
7. Apresentar gráfico de ritmo ao longo do tempo
8. Apresentar gráfico de altitude ao longo do tempo.
9. Apresentar gráfico de bpms ao longo do tempo.
10. Apresentar um gráfico de zonas de bpm durante a corrida ou treino. Utilize a definição de zonas dadas em https://support.polar.com/e_manuals/M450/Polar_M450_user_manual_Portugues/Content/Heart_rate_Zones.htm.
11. Permitir a junção dos gráficos 7, 8 e 9.

Marcas de tempo

Timestamps (marcas de tempo) são dados em segundos. *Timestamps* são usados para determinar data e horários. Python usa a convenção de Linux de que um *timestamp* representa a quantidade de de segundos transcurridos desde as 0 horas do 1 de janeiro de 1970, UTC. A bibloteca time oferece várias funções relacionadas com tempo.

Latitud, Longitude e cálculo da distância

Latitudes e longitudes são dados em graus. Para calcular a distância entre dois pontos podem usar a biblioteca geopy. Para instalar, executem na linha de comandos:

pip install geopy

Para calcular a distância entre dois pontos geográficos use:

```
from geopy import distance
...
... distance.distance((latitude1, longitude1), (latitude2, longitude2)) ...
```