Sumarização de Arquivos de Log de GPS

Atualmente, a maioria dos ciclocomputadores utilizados pelos ciclistas (amadores e profissionais) possuem um GPS. Esses dispositivos também conversam com diferentes sensores, como por exemplo, sensor de frequência cardíaca, velocidade, cadência, potência, etc. Isso permite que o ciclista/treinador possa analisar os dados do pedal e verificar se o treino foi realizado conforme prescrito.

Todas essas informações (GPS e sensores) são armazenadas a cada segundo. No exemplo abaixo, temos o *log* de dois segundos. As duas primeiras linhas contém a informação da bicicleta utilizada e a data da atividade. Na sequência, temos diversos campos com os respectivos valores e unidades registrados a cada segundo. Note que nem todos os campos estarão disponíveis. Por exemplo, se o ciclista não usar um sensor de frequência cardíaca, o campo *heart_rate* não existirá.

gps.log

```
Gear: Cannondale SuperSix
Date: Nov 2, 2020, 10:26:44 AM

altitude: 973.8 m
ascent: 0 m
battery_soc: 100.0 percent
cadence: 62 rpm
calories: 0 kcal
distance: 2.21 m
enhanced_altitude: 973.8 m
enhanced_speed: 3.343 m/s
gps_accuracy: 2 m
grade: 7.9 %
heart_rate: 112 bpm
left_pedal_smoothness: 30.0 percent
left_right_balance: 56
left_torque_effectiveness: 95.0 percent
position_lat: -302650798 semicircles
position_long: -587443105 semicircles
power: 201 watts
right_pedal_smoothness: 28.0 percent
right_torque_effectiveness: 85.5 percent
speed: 3.343 m/s
timestamp: 2020-11-02 10:26:45

altitude: 973.8 m
ascent: 0 m
battery_soc: 100.0 percent
cadence: 63 rpm
calories: 0 kcal
distance: 5.18 m
enhanced_altitude: 973.8 m
enhanced_speed: 3.312 m/s
gps_accuracy: 2 m
grade: 7.2 %
heart_rate: 112 bpm
left_pedal_smoothness: 30.5 percent
left_right_balance: 55
left_torque_effectiveness: 95.0 percent
```

```
position_lat: -302650524 semicircles
position_long: -587442916 semicircles
power: 194 watts
right_pedal_smoothness: 28.0 percent
right_torque_effectiveness: 85.5 percent
speed: 3.312 m/s
timestamp: 2020-11-02 10:26:46
```

Atividade

Para esse trabalho, você terá acesso a diversos arquivos de log (https://www.inf.ufpr.br/lesoliveira/ci1002/) de diferentes bicicletas. Você deve escrever um programa que leia todos os logs de um dado diretório e apresente um resumo para cada bicicleta da seguinte forma:

Bicicleta: Cannodale SuperSix

Data	**Distância (km)**	**Velocidade Média (km/h)**	**Velocidade Máxima (km/h)**	**HR Médio (bpm)**	**HR Máximo (bpm)**	**Cadência Média (rpm)**	**Subida Acumulada (m)**
2/11	62,3	25	50	130	150	75	1200
...	...	...	...	...	...	...	...

Observações:

- No arquivo de log, o valor de uma grandeza é válido até o próximo *timestamp*. No exemplo de log acima, a velocidade de 3.343 m/s é a velocidade praticada no intervalo de 1 (um) segundo que vai do timestamp 2020-11-02 10:26:45 até o timestamp seguinte (2020-11-02 10:26:46).
- Os valores médios (velocidade, cadência e HR) devem ser ponderados em função do tempo, levando em consideração os *timestamps*, ou seja quanto tempo durou cada valor. Este intervalo de tempo é o peso usado no cálculo da média ponderada. No exemplo do item acima, o peso da velocidade 3.343 é 1.
- Valores nulos de velocidade (*speed*), cadência (*cadence*) e HR (*heart rate*) não devem ser considerados no cálculo dos valores médios respectivos. São considerados valores nulos o valor 0 (zero), a palavra **None**, ou a ausência do valor no registro do log.
- Quando há uma parada total do ciclista (por exemplo, o ciclista parou para tomar um café), isto é sinalizado com o valor de velocidade 0 (zero). Nestes casos, o GPS pode ficar alguns minutos sem gravar nada, ou registrar diversas entradas com velocidade nula. A retomada de movimento ocorre quando após um resgistro em que a velocidade é nula, um novo registro é gravado com velocidade não-nula.
- **Subida Acumulada** é quantos metros o ciclista subiu durante a atividade. Nesse cálculo você deve considerar somente o ganho de altimetria, ou seja altitude no tempo $t+1$ > altitude no tempo t.
- Para efeitos de verificação, você pode comparar os valores de saída de seu programa com o seguinte exemplo de saída

  .

Ao fim você deve apresentar um sumário contendo as seguintes informações: Quantidade de Atividades, Total Percorrido em *km*, Pedal mais longo em *km*, Pedal mais curto em *km* e Distância Média em *km*.

Forma de chamada:

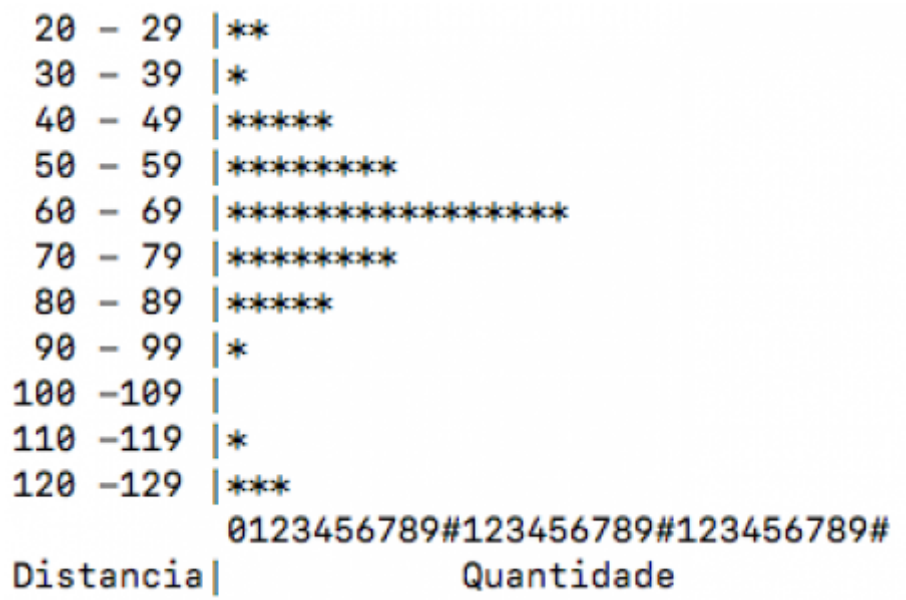
```
./gps -d <diretório de arquivos log>
```

Com essa chamada, seu programa deve ler todos os arquivos de *log* encontrados no diretório passado como parâmetro e armazenar as informações relevantes em memória.

Enquanto o programa estiver lendo os arquivos de *log*, o usuário deve ser informado que o programa está em execução.

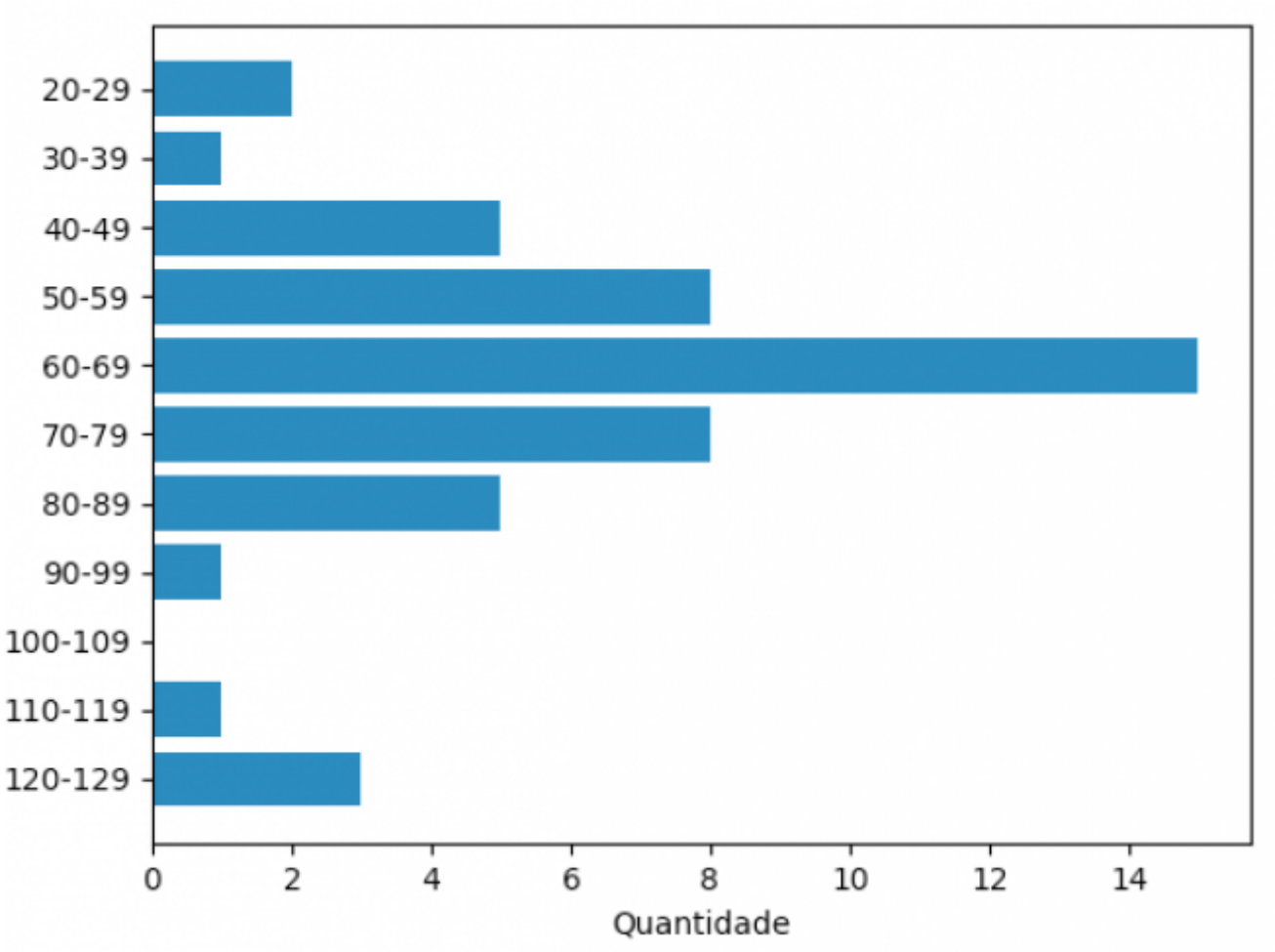
Uma vez feito isso, o programa deverá apresentar as seguintes opções ao usuário:

1. Bicicletas Encontradas: Mostra todas as bicicletas encontradas durante o processamento dos arquivos de log.
2. Pede para o usuário informar uma das bicicletas encontradas e apresenta a lista de atividades, resumo conforme descrito acima.
3. Lista todas atividades agrupadas por bicicleta e ordenadas pela data
4. Lista todas atividades agrupadas por bicicleta e ordenadas pela distância
5. Lista todas atividades ordenadas pela subida acumulada
6. *Histograma*: O usuário deve escolher uma bicicleta e apresentar um histograma da seguinte forma:
    - O histograma deve mostrar a distribuição da distância das atividades da bicicleta escolhida como no exemplo abaixo. Para facilitar a comparação, o gráfico deve conter um número fixo de colunas no qual cada coluna contém os intervalos de 10km. Para definir o número de colunas, você deve encontrar a atividade mais curta (C) e a mais longa (L). Por exemplo, suponha C = 25 e L = 124. Nesse caso, a primeira linha do histograma vai de 20 a 29, a segunda de 30 a 39 e assim por diante sendo que a última linha deve conter a atividade mais longa. O histograma pode ser apresentado no formato ASCII como no exemplo abaixo, com as distâncias no eixo **y** e a quantidade de atividades no eixo **x**:



Atividade Extra:

Os alunos que apresentarem o mesmo histograma no formato gráfico (vide exemplo abaixo) receberão 25 pontos extras na nota final do trabalho. Para gerar o gráfico, você pode utilizar qualquer biblioteca (por exemplo, gnuplot), desde que a implementação seja na linguagem C.



ATENÇÃO

- Sempre que possível, as informações necessárias às funções devem ser transferidas como parâmetros (por valor ou por referência, dependendo da situação). Minimizar o uso de variáveis globais.
- Use alocação dinâmica de memória para leitura e processamento dos arquivos de log.
- Para testar seu programa, use os arquivos de log disponíveis aqui

Estrutura do código-fonte

O código-fonte deve ser devidamente modularizado e estruturado em arquivos `.c` e `.h` que agrupem as diversas funcionalidades do programa: leitura/escrita em arquivos, alocação de memória, geração de tabelas, etc. Deve haver pelo menos 3 arquivos: um contendo o programa principal (apenas) e os demais contendo os diversos módulos que compõem o programa.

O que deve ser entregue

- Deve ser entregue ao professor um arquivo *.tar* ou *.zip* contendo:
  - arquivos `.c` e `.h`
  - arquivo `Makefile`
  - Por favor, NÃO ENVIE OS ARQUIVOS DE LOG!
- O `Makefile` para o projeto deve ter pelo menos:
  - Os alvos `all` (default), `clean` e `purge`.
  - CFLAGS = `-std=c99 -Wall`

        - ATENÇÃO: Deve ser OBRIGATORIAMENTE usada a opção de compilação `-std=c99`
    - Compilar e ligar separadamente (gerar arquivos `.o` intermediários)
- Os trabalhos devem ser entregues através do Moodle C3SL:
    - Turma BCC1 (Prof. David Menotti)
    - Turma BCC2 (Prof. Armando Delgado)
    - Turma BCC3 (Prof. Luiz Oliveira)

Avaliação

Os itens de avaliação do trabalho e respectivas pontuações são:

- Modularização e organização do código-fonte (15 pontos)
- Funcionamento: corretude das respostas nos testes executados (40 pontos)
- Eficiência: algoritmos e estruturas de dados utilizados para obter um melhor desempenho e uso eficiente de alocação dinâmica de memória (45 pontos)
- **Atividade Extra**: Histograma em formato gráfico (+25 pontos)

ATENÇÃO: programas que tiverem erros de compilação ou terminarem a execução de forma abrupta sem que tenha havido processamento adequado receberão nota **ZERO**