# Programação 2016/2017

Mestrado em Engenharia Electrotécnica e de Computadores (MEEC)

# BikeStation

# Projeto de Programação – Entrega Final

## 1 Introdução

Com o projeto de programação designado de *BikeStation* pretende-se obter um conjunto de estatísticas da utilização do sistema público de partilha de bicicletas na cidade de Boston nos Estados Unidos da América. O nome deste serviço é *Hubway* e gere aproximadamente 1800 bicicletas e 180 estações em Boston, Brookline, Cambridge e Sommerville. Este tipo de serviço é considerado uma forma de transporte divertida, barata e conveniente para voltas rápidas em cidades e está presente em muitas cidades europeias. Também se prevê que esteja a funcionar em Lisboa brevemente. De uma forma geral, pretende-se descobrir padrões de utilização deste sistema a partir de um conjunto de dados reais, tais como:

- Quais são as estações de bicicleta mais populares?
- Quais são as rotas mais populares neste tipo de serviço?
- Que tipo de utilizadores recorre a este serviço?
- Quanto tempo é que utilizam este serviço?

## 2 Dados de entrada

Para realizar os objetivos deste projeto vão ser disponibilizados dados reais adquiridos pelo serviço de partilha de bicicletas. Os dados encontram-se disponíveis no formato *comma-separated values* (CSV) que é um formato muito habitual para a partilha de dados em forma de tabela. Cada linha do ficheiro é um item da

tabela e cada item possui um ou mais campos, separados por vírgula. Para a realização deste projeto vão ser disponibilizados dois ficheiros, um com informação das viagens e outra com a informação sobre as estações onde as bicicletas estão disponíveis. O ficheiro com informação sobre as viagens (tripdata.csv) contém a seguinte informação:

- Duração da viagem em segundos.
- Hora e data de início da viagem.
- Hora e data do final da viagem.
- ID da estação inicial.
- ID da estação final.
- ID da bicicleta.
- Tipo de usuário:
    - Casual – Utilizador com um passe de 24 horas ou 72 horas.
    - Membro – Membro anual ou mensal.
- Ano de nascimento, no caso de ser membro.
- Género (masculino ou feminino), no caso de ser membro.

Além desta informação também está disponível um ficheiro (stationdata.csv) com informação sobre as estações onde estão disponíveis as bicicletas, mais precisamente:

- Identificador da estação: Corresponde ao ID da estação inicial e da estação final no ficheiro sobre as viagens.
- Nome curto da estação: Nome composto por uma letra e cinco números.
- Descrição da estação: Nome por extenso da estação.
- Latitude: Latitude onde a estação se encontra.
- Longitude: Longitude onde a estação se encontra.
- Município: Nome do município a estação pertence.

# 3 Funcionamento

O programa *BikeStation* deve ter dois modos de funcionamento: o modo gráfico e o modo textual. No modo gráfico deve ser possível visualizar as viagens dos utilizadores deste tipo de serviço e no modo textual deve ser possível calcular um conjunto de estatísticas sobre as viagens recebidas em formato textual.

O programa deve receber como parâmetros (que correspondem a argumentos da função *main*) dois ficheiros: o ficheiro com as viagens e o ficheiro com as estações. Além disso, também é necessário receber como parâmetro uma opção –g ou –t para indicar se o programa vai funcionar em modo textual ou em modo gráfico. Os parâmetros de entrada devem ser indicados pela seguinte ordem: modo de funcionamento, ficheiro com viagens e ficheiro com estações. Por exemplo, para iniciar o programa em modo gráfico deve ser escrito o seguinte comando:

./bikestation –g tripdata.csv stationdata.csv

## 3.1 Modo Textual

No modo textual deve ser apresentado o seguinte menu para o utilizador escolher uma das opções:

1. Seleção de dados
2. Listagem de viagens
3. Listagem de estações
4. Listagem de rotas
5. Listagem de estatísticas

Cada uma destas opções é descrita de seguida.

**Seleção de dados**

Esta opção pergunta ao utilizador se as listagens ou estatísticas a imprimir (opções seguintes) serão sobre todas as viagens carregadas do ficheiro ou apenas a viagens que cumprem um ou mais critérios:

1. Período de tempo: O utilizador escolhe um período de tempo (hora de início e de fim).
2. Dia da semana: O utilizador escolhe um dia da semana.
3. Duração: O utilizador escolhe uma duração máxima para as viagens.

Estes critérios devem ser escolhidos através de um menu. Nesse menu deve ser permitido limpar todos os critérios previamente escolhidos. Note-se que quando o utilizador estabelecer um ou mais critérios, todas as análises devem ser realizadas apenas com os dados que cumprem todos os critérios.

**Listagem de viagens**

Neste modo devem ser impressas todas as viagens que cumprem os critérios definidos na seleção de dados. Devem perguntar ao utilizador quantas viagens devem ser mostradas.

**Listagem de estações**

Neste modo devem ser impressas uma lista das estações, representadas pelo seu ID, nome e local (latitude e longitude). Além disso, devem ser impressas o número máximo, mínimo e médio do número de bicicletas a chegar por hora bem como o número máximo, mínimo e médio do número de bicicletas a sair por hora. Os dados a mostrar nesta listagem devem tomar em consideração os critérios da seleção de dados.

**Listagem de rotas**

Neste modo o utilizador escolhe uma estação através do seu identificador e o programa deve imprimir todas as rotas entre a estação escolhida e as restantes estações (independentemente da estação escolhida ter sido de origem ou de chegada). O número de viagens entre a estação escolhida e as restantes (rotas) deve ser impresso, bem como o ID e nome da estação de destino. Além disso, a lista a imprimir deve estar ordenada por ordem decrescente de número de viagens e distinguir as rotas que começaram e as que terminaram na estação escolhida. Devem perguntar ao utilizador quantas rotas devem ser mostradas.

**Listagem de estatísticas**

Este modo tem quatro opções:

- **Estatísticas por género:** Percentagem de viagens efetuadas por utilizadores do género feminino e masculino para os dois tipos de utilizadores (casual e registrado).
- **Estatísticas da idade:** Percentagem de viagens efetuadas pelos utilizadores em função da idade. Escolha intervalos de 5 anos.
- **Estatísticas da duração:** Percentagem de viagens efetuadas pelos utilizadores em função da duração. Escolha intervalos de 15 minutos com a duração máxima de 6 horas.
- **Estatísticas da velocidade:** Indique a velocidade média por hora por tipo de utilizador, género ou idade. Pergunte ao utilizador qual das 3 opções pretende.

## 3.2 Modo Gráfico

O objetivo do modo gráfico é mostrar as rotas que ocorreram num dado dia da semana graficamente, o que corresponde ao modo listagem (textual) de rotas acima definido. No início deste modo gráfico devem perguntar qual ou quais são os dias da semana ao utilizador e mostrar informação apenas referentes a esses dias. A interface gráfica deve obedecer ao seguinte cenário:

- Uma roda com pontos (ou outras entidades gráficas) a representarem as estações.
- Desenhar uma recta (ou arco) por cada rota (isto é, viagens entre duas estações). A grossura de cada recta (e/ou cor) deve ser proporcional ao número de viagens que ocorreram em cada rota. Também devem distinguir viagens para a estação (ponto de chegada) e viagens da estação (ponto de partida)
- Em vez dos dados de todas as estações permitir mostrar as rotas de uma só estação permitindo ao utilizador escolher a estação através do teclado ou do rato, usando a biblioteca SDL2.

Opcionalmente, em vez de mostrar as estações em cima de uma roda, podem imprimir as estações em cima de um mapa de Boston. Um exemplo duma aplicação deste tipo é: http://bostonography.com/hubwaymap/. Obviamente, não se espera que a vossa tenha a mesma funcionalidade (por exemplo, zoom no mapa). Eventuais melhorias ao modo gráfico poderão contribuir com uma bonificação na nota do projeto.

# 4 Estruturas de Dados

No desenvolvimento deste jogo, é obrigatório obedecerem às seguintes regras:

- Os dados das estações e das viagens têm de ser representados através de estrutura de dados.
- Devem usar uma lista (escolham o tipo de lista mais adequado) para representarem as estações e as viagens. Opcionalmente, podem usar um vetor de estruturas para representarem a lista de estações.
- Devem usar alocação dinâmica de memória para reservar a memória referente às estações e viagens. Sempre que precisarem de reservar espaço de memória para vetores (por exemplo, para guardarem o número de viagens em cada rota) devem também alocar a memória dinamicamente.
- A seleção de dados deve ser implementada apagando nós da lista, de forma a poupar memória e tornar o processo de percorrer a lista mais rápido.

- Sempre que possível as listas devem estar ordenadas, por identificador da estação ou por data/hora da viagem.
- Também se recomenda, utilizar ponteiros em cada nó da lista das viagens para as estações de partida e de chegada ou utilizar uma lista de listas para facilitar a extração da informação necessária!

Também é importante salientar que o programa desenvolvido deverá ser estruturado em múltiplos ficheiros que permitam uma organização adequada do código.

# 5 Desenvolvimento do Programa

Também é fundamental que os alunos cumpram as regras que se seguem no desenvolvimento do jogo.

## 5.1 Desenvolvimento Faseado

O desenvolvimento deste projeto deverá ser feito de uma forma faseada, devendo os alunos garantir que todas as funcionalidades codificadas até ao momento estão a funcionar corretamente.

**É preferível um programa que implementa poucas funcionalidades, mas que funcionam corretamente, do que um programa totalmente desenvolvido mas que faz muito pouco.**

Assim sugerem-se os seguintes passos, pela ordem apresentada, para realização do projeto. Note-se que esta lista não é exaustiva. Desta forma, é necessário realizar as seguintes tarefas:

1. Criação das estruturas de dados necessárias para representar as estações e as viagens.
2. Leitura parâmetros do *main* e dos ficheiros necessários ao funcionamento do programa.
3. Construção dos menus e introdução dos dados.
4. Construção das listas de viagens e de estações, inserindo os elementos de uma forma ordenada.
5. Implementação do modo de listagem de viagens.
6. Implementação da seleção de dados, removendo elementos da lista de viagens.
7. Implementação do modo de listagem das estações.
8. Implementação do modo de listagem de rotas.
9. Implementação do modo gráfico.
10. Implementação da análise estatística de dados.
11. Eventuais melhorias ao modo gráfico da aplicação, recorrendo a um mapa real.

No final de cada passo devem verificar a execução correcta do programa. Os alunos deverão garantir a robustez da aplicação verificando todos os casos de erro (por exemplo quando um parâmetro de entrada não seja válido).

## 5.2 Documentação

O código produzido pelos alunos deverá ser comentado. Os comentários presentes no código deverão explicitar e explicar o funcionamento da aplicação assim como as decisões tomadas. As seguintes regras devem ser cumpridas:

- O código deve ser comentado sempre que realize alguma operação não óbvia.
- Os comentários devem ser claros, gramaticalmente corretos e usando frases simples.
- A declaração de todas as variáveis e constantes deve ser acompanhada de um comentário com uma breve descrição de para que servem.
- Cada bloco de código (seleção ou repetição) deve ser precedido de um breve comentário explicativo.
- Todos os programas devem ter um comentário inicial que identifique, no mínimo, o trabalho, os seus autores, o fim a que se destina e a data de realização.

## 5.3 Indentação

Um ponto fundamental na organização de escrita de código é a indentação, isto é, organização hierárquica das linhas de código, de acordo com âmbito onde elas se encontram. A indentação deixa o código fonte do programa mais organizado, mais legível, fácil de entender e de modificar, sendo uma parte essencial do trabalho.

## 5.4 Estrutura do Código

Todos os programas em C devem possuir a mesma estrutura genérica, composta pelas seguintes secções:

- Bloco de comentários.
- Diretivas #include.
- Constantes globais e variáveis globais (caso sejam necessárias).
- Declaração de funções.
- Função main().
- Definição de funções.

Como regra geral deve considerar que as funções devem caber num único ecrã, isto é, devem ter no máximo cerca de 30 linhas. Também deve cumprir as seguintes regras:

- Inicialize sempre as variáveis na sua declaração.
- Teste a validade dos parâmetros recebidos por uma função.
- Declare constantes e evite usar números no corpo das funções.
- Evite repetições de código, use funções, ciclos, etc.
- Evite o uso de variáveis globais.
- Não use **goto.**
- Escreva código simples e claro que um colega seu possa perceber!

## 5.5 Compilação

O compilador a usar na execução do projeto é o gcc em ambiente Linux. **Os projetos que não compilem, i.e. que tenham erros de sintaxe, não serão avaliados.** A existência de avisos durante a fase de compilação poderá ser indício da existência de problemas no código. Estes deverão ser eliminados corretamente ou ignorados com cuidado extremo.

## 5.6 Decisões do Projeto

Como em qualquer projeto de informática, o funcionamento do programa não está totalmente definido no enunciado, existindo algumas ambiguidades e omissões. Para resolver essas omissões os alunos deverão tomar algumas decisões aquando do desenvolvimento do projeto. Estas decisões devem ser fundamentadas, sem nunca ir contra o definido no enunciado.

## 5.7 Biblioteca SDL

Durante o desenvolvimento deste projeto devera ser usada a biblioteca SDL2. A aplicação deverá ser compilada usando as bibliotecas SDL2, SDL2_image, SDL2_ttf. Mais informação disponível aqui:

- http://wiki.libsdl.org/APIByCategory
- https://wiki.libsdl.org/FrontPage

# 6 Submissão

Os alunos deverão submeter o código desenvolvido através do sistema FENIX até 25 de Maio. Devem entregar o código correspondente ao programa desenvolvido, nomeadamente os ficheiros de texto com extensão **.c** e **.h** e outros ficheiros necessários ao funcionamento do programa (por exemplo, fontes e imagens). Não utilize um processador de texto (e.g. Word) para formatar o seu programa.

# 7 Plágio

Os trabalhos serão objeto de um sistema de deteção de plágio. Os alunos podem conversar entre si para discutir possíveis soluções para algum problema que tenham mas não podem partilhar código fonte. Nesta entrega final, o projeto deve ser realizado em grupos de dois alunos! Se uma cópia for detetada todos os alunos envolvidos na cópia serão penalizados.

# 8 Avaliação

A avaliação do projeto terá em conta diversos parâmetros:

- Funcionalidades implementadas
- Uso correto de estruturas de dados
- Criação de listas, filas ou pilhas de forma adequada aos objetivos do projeto
- Eficiência e qualidade da implementação das funcionalidades do projeto
- Qualidade do aspecto gráfico
- Qualidade do código produzido
- Estruturação da aplicação
- Comentários e legibilidade do código
- Tratamento de erros