1 Estás a desenvolver um programa que recebe e processa informação referente às pontuações de uma temporada de Fórmula 1. A informação recebida é relativa a cada piloto e inclui 3 campos com o seguinte formato: `nome equipa pontos`. O nome do piloto (duas palavras, máximo de 50 caracteres) e a equipa (uma única palavra, máximo de 12 caracteres) indicam o nome do piloto e da respetiva equipa; o último campo indica o número de pontos desse piloto no final da temporada. Considera que é introduzida informação, no máximo, sobre 10 equipas.

O teu programa deve determinar qual a equipa com a melhor pontuação, considerando todos os pilotos que fazem parte dessa equipa. Neste problema deves usar o ficheiro `prob1.c`, que contém uma implementação incompleta. Analisa-o com atenção antes de implementar as seguintes funções:

```
int lerEquipas(char nomes[][TSIZE], int totalPontos[])
/* Guarda nomes das equipas no vetor de strings nomes e o total de
 * pontos conquistados de cada equipa na mesma posição no vetor
 * totalPontos. Retorna o número de equipas lidas, sem repetições.
 * Nota: os nomes dos pilotos são lidos, mas não são guardados.
 */

int melhorPontuacao(int pontos[], int n)
/* Determina qual a equipa com maior pontuação.
 * Retorna a posição dessa equipa no vetor.
 */
```

O teu programa pode ser testado com o ficheiro `teams.txt` [exemplo de utilização: `./prob1 < teams.txt`]. Para esse ficheiro de teste o resultado deverá ser:

```
Total de equipas: 10
Equipa com maior pontuacao: MERCEDES (655 pontos)
```

2 Estás a desenvolver um programa para analisar dados de jogadores ao longo de uma época da NBA. Para isso, considera que a informação referente a cada jogador inclui o seu nome (duas palavras), equipa em que joga (uma palavra), jogos, média de minutos jogados e média de pontos marcados. Neste problema deves usar o ficheiro `prob2.c`, que contém uma implementação incompleta. Analisa-o com atenção antes de implementar as alíneas seguintes.

2.1 Implementa a função `mvp` que determina o jogador com melhor média de pontos marcados.

```
float mvp(jogador statsJogadores[], int n, jogador *mvPlayer)
```

A função retorna a média de pontos marcados e o melhor jogador no parâmetro `mvPlayer`. Os parâmetros `statsJogadores` e `n` são o vetor com todos os jogadores e o respetivo tamanho.

Depois de implementada a função, o programa deverá apresentar:

```
*** Most Valuable Player (media de pontos 35.4) ***

James Harden    HOU 41 37.3 35.4
```

2.2 Implementa a função `gravaEstatisticas` que armazena num ficheiro de texto a informação de todos os jogadores de uma determinada equipa (um jogador por linha).

```
int gravaEstatisticas(jogador statsJogadores[], int n, char *equipa,
                      char *nomeFicheiro)
```

A função retorna o número de jogadores gravados no ficheiro, ou -1 em caso de erro. Os parâmetros `statsJogadores` e `n` são o vetor com todos os jogadores e o respetivo tamanho. As

strings `equipa` e `nomeFicheiro` são o nome da equipa à qual os jogadores devem pertencer e do ficheiro onde a informação deve ser gravada.

Depois de implementada a função, o programa deverá apresentar:

```
*** A gravar jogadores da equipa GSW para o ficheiro statsJogadores.txt ***

Informacao de 16 jogadores gravada com sucesso
```

2.3 Implementa o procedimento `top10Jogadores` que determina os 10 jogadores com melhor média de pontos marcados.

```
void top10Jogadores(jogador statsJogadores[], int n, jogador topJogadores[])
```

O resultado é devolvido no vetor (ordenado) `topJogadores`. Os parâmetros `statsJogadores` e `n` são o vetor com todas as equipas e o respetivo tamanho.

Depois de implementado o procedimento, o programa deverá apresentar:

```
*** Top 10 jogadores (media de pontos) ***

James Harden    HOU 41 37.3 35.4
Stephen Curry   GSW 35 34.4 29.8
Anthony Davis   NOP 41 37.1 29.3
...
Paul George     OKC 43 35.9 26.6
Giannis Antetokounmpo   MIL 41 33.0 26.4
Damian Lillard  POR 47 35.3 26.1
```