Miniteste 2 - Exemplo

1 Escreva um programa que leia e processe os dados de acesso de um carro a um parque de estacionamento. A informação de acesso ao parque é composta por um número arbitrário de pares de valores - tempo de entrada e tempo de saída, em segundos. A leitura termina quando não for possível ler os 2 valores.

Considere que no máximo existem 100 acessos e que os tempos são sempre crescentes. Adicionalmente, quando a diferença entre o tempo de entrada e o tempo de saída anterior for inferior a 120 segundos, deverá ser considerado um único acesso com tempo total igual a: segundo tempo de saída - primeiro tempo de entrada.

Pretende-se com este programa determinar o tempo mínimo, máximo e médio do carro no estacionamento. Implemente e utilize no seu programa as seguintes funções:

```
int ler_acessos(int *tempos);
/* Lê os tempos correspondentes às estadias do carro no estacionamento
 * (i.e., tempo de saída - tempo de entrada) para o vetor "tempos".
 * Retorna o número de acessos lidos */

void stats(int *tempos, int n, int *min, int *max, float *med);
 /* Determina os tempos mínimo, máximo e médio de estadias no parque. */
```

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

```
Numero de acessos: 19
Min: 665 | Max: 28818 | Med: 7955.21
```

2 Pretende-se desenvolver um programa para gerir as submissões num concurso de programação. Para cada equipa, o programa guarda a seguinte informação: nome da equipa, tempo de submissão (em segundos, desde o início da competição) e número de respostas certas. Neste problema deverá usar o ficheiro prob2.c fornecido, que contém uma implementação incompleta. Analise-o com atenção antes de implementar as alíneas seguintes.

2.1 Desenvolva a função `ler_equipas` que preenche o vetor `resultados` com a informação introduzida pelo utilizador e retorna o número de submissões lidas.

```
int ler_equipas(equipa *resultados);
 /* lê e guarda no vetor resultados a informação das equipas;
 retorna o número de valores lidos */
```

Inicialmente deverão ser lidos 4 valores (valores inteiros), que correspondem aos resultados corretos para cada problema da competição de programação. Posteriormente, para cada equipa, devem ser lidos 6 valores: nome (uma palavra), tempo de submissão e as 4 respostas para cada problema (valores inteiros). A leitura deverá terminar quando não for possível ler todos os dados de uma equipa. Esta função deve ainda garantir que não são ultrapassados os limites do vetor.

O seu programa pode ser testado com o ficheiro `submissoes.txt` [exemplo de utilização: `./prob21 < submissoes.txt`]. Para esse ficheiro o resultado deverá ser:

```
Numero de equipas: 10
FEUPAllStars 7119 1
PowerString 9267 0
...
FooBarQux 11575 1
```

2.2 Implemente a função `filtra_resultados` que cria um vetor com todas as equipas com um número mínimo de respostas certas.

```
equipa* filtra_resultados(equipa *resultados, int N, int nMin, int *Nr);
/* retorna um vetor contendo todas as equipas que têm pelo menos nMin
respostas certas. O número de elementos nesse vetor é devolvido por
referência no parâmetro Nr. Os parâmetros resultados e N são o vetor com
todas as equipas e o respetivo tamanho. O parâmetro nMin indica o número
mínimo de repostas certas. */
```

O seu programa pode ser testado com o ficheiro `submissoes.txt` [exemplo de utilização: `./prob22 < submissoes.txt`]. Para esse ficheiro o resultado deverá ser:

```
Numero de equipas: 10
...
Numero de equipas com pelo menos 2 respostas certas: 4
n00bz 10247 4
Alpacas 7431 2
SegFault 10621 2
GrrrlGang 5405 3
```

2.3 Altere a função `ler_equipas` de forma a ler os dados diretamente do ficheiro. O nome do ficheiro deverá ser indicado como parâmetro da função.

```
int ler_equipas(equipa *resultados, char *nomeFicheiro);
 /* lê a informação das equipas a partir do ficheiro com o nomeFicheiro e
guarda-a no vetor resultados; retorna o número de valores lidos */
```

2.4 Descreva um algoritmo que permita determinar os 3 primeiros classificados na competição (maior número de respostas certas, no menor tempo de submissão). Descreva sucintamente, e de forma clara, o algoritmo não incluindo código na sua resposta.