1 Numa linha de enchimento de garrafas é mantido um registo ao longo do dia do volume colocado em cada garrafa. Sabendo que por dia não são enchidas mais do que 1000 garrafas, escreva um programa que lê e guarda num vector os volumes de enchimento realizados ao longo de um dia. O número de valores lidos é especificado pelo utilizador. Pretende-se determinar o volume médio de enchimento, sendo que deve ser excluído do cálculo do valor médio os valores que se encontram acima de um valor máximo e abaixo de um valor mínimo (ambos a especificar pelo utilizador). Implemente e utilize no seu programa as seguintes funções:

```
int ler_volumes(float *volumes);
 /* lê e guarda no vetor volumes os valores de enchimento de um dia;
retorna o número de valores lidos */

float volume_medio(float *volumes, int nvolumes, float max, float min);
/* determina o volume médio de enchimento */
```

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

```
Foram lidos 100 valores.
Indique valor mínimo: 0.2
Indique valor máximo: 0.8
O volume medio de enchimento foi de 0.504 litros.
```

2 Pretende-se desenvolver um programa para gerir o registo de pacientes (animais) numa clínica veterinária. O registo animal electrónico (RAE) de cada paciente consiste no nome do animal, tipo de animal (cão, gato, ...), nome e telefone do dono.

2.1 Desenvolva a função `ler_animais` que preenche o vetor clinica com a informação introduzida pelo utilizador e retorna o número de pacientes lidos.

```
int ler_animais(animal *clinica);
 /* lê e guarda no vetor clinica a informação de pacientes;
 retorna o número de valores lidos */
```

Para cada animal devem ser lidos 4 campos: nome (composto por uma palavra), tipo de animal (uma só palavra), nome do dono (composto por duas palavras) e telemóvel do dono. A leitura deverá terminar quando não for possível ler todos os dados de um animal. Sugestão: verifique o valor de retorno da função `scanf`. Esta função deve ainda garantir que não são introduzidos mais do que `NUMANIMAIS`.

O seu programa pode ser testado com o ficheiro `animais1.txt` [exemplo de utilização: `./prob2a < animais1.txt`]. Para esse ficheiro o resultado deverá ser:

```
Num de registos lidos: 65
```

2.2 [20 valores] Implemente a função `lista_animais` que cria uma lista com todos os animais de um determinado tipo. Considere o programa apresentado anteriormente para testar a função que desenvolveu e determinar o número de pacientes da clínica que são tartarugas ou coelhos.

```
animal* lista_animais(animal clinica[], int Na, char *tipo, int *Nl);
/* retorna um vetor contendo a lista de animais de um dado tipo.
O número de elementos nesse vetor é devolvido por referência no
parâmetro Nl. Os parâmetros clinica e Na são o vetor com o registo dos
animais e o respetivo tamanho. O parâmetro tipo indica o animal a
incluir na lista. */
```

O seu programa pode ser testado com o ficheiro `animais1.txt` [exemplo de utilização: `./prob2b < animais1.txt`]. Para esse ficheiro o resultado deverá ser:

```
Num de registos lidos: 65
Numero de tipo == "coelho": 13
```

2.3 [10 valores] Implemente a função `guarda_lista` que grava num ficheiro a lista de animais de um dado tipo.

```
void guarda_lista(animal lista[], int n, char *nomeFicheiro);
/* armazena no ficheiro com nome nomeFicheiro a lista de animais.
Os parâmetros lista e n são o vetor com a lista de animais e o tamanho. */
```

2.4 [5 valores] Descreva um algoritmo que permita ordenar todos os pacientes do vector clinica por ordem alfabética do seu nome e do nome do dono. Descreva sucintamente, e de forma clara, o algoritmo não incluindo código na sua resposta.