5.1 Escreva um programa que lê repetidamente carateres até encontrar uma mudança de linha (`\n`) e contabiliza o *número total de letras* (isto é, carateres de `'A'` a `'Z'` e de `'a'` a `'z'`). Exemplo (em sublinhado o texto introduzido pelo utilizador):

```
Ola, Mundo!
A frase contém 8 letra(s)
```

Sugestão: usar `getchar()` para ler um carater de cada vez.

5.2 Escreva um programa para contar palavras da entrada-padrão. Considere que as palavras são sequências de "carateres normais" (e.g. letras, algarismos ou sinais de pontuação) separados por "carateres brancos" (espaços, tabulação ou mudanças de linha, ou seja, `' '`, `'\t'` ou `'\n'`).
Por exemplo, o texto seguinte contém 13 palavras (note que a sequência `---` é considerada uma palavra):

```
To be or   not to be,  that is the question.
--- William Shakespeare
```

Sugestão: Utilize duas variáveis para guardar os dois últimos carateres lidos; podemos então contar o número de vezes que um carater "normal" (i.e., não branco) é seguido de um carater branco. No exemplo seguinte assinalamos as transições relevantes com o simbolo `^`:

```
To␣be␣or␣␣␣not␣to␣be␣
  ^  ^  ^     ^  ^  ^
```

5.3 Escreva um programa que lê carateres e contabiliza o número de vezes de ocorreu cada letra (A a Z) ignorando a distinção entre maiúsculas e minúsculas. No final deve imprimir uma tabela com as contagens:

```
A: 3    B: 2    C: 2    D: 1    E: 7    ...
```

Sugestão: utilize uma variável indexada com 26 elementos para contabilizar a contagem das letras. Pode ainda converter cada carater para maíuscula para evitar ter de tratar letras minúsculas.

5.4 Escreva uma função `void capitalizar(char str[])` que transforma todas as letras duma cadeia em maiúsculas; outros carateres devem ficar inalterados. O argumento é uma cadeia de carateres (não necessariamente letras) terminada por `\0`.

Sugestão: use a função `toupper` da biblioteca padrão para transformar cada carater em maíscula.

5.5 Escreva uma função `int palindromo(char str[])` que testa se uma cadeia de carateres é um palíndromo, isto é, se tem a mesma sequência de carateres da esquerda para a direita e vice-versa.

5.6 Escreva uma função `int todos_letras(char str[])` que testa se uma cadeia contém apenas carateres letras (maísculas ou minúsculas). O resultado deve ser 1 em caso afirmativo e 0 em caso negativo.

Sugestão: utilize a função `isalpha` da biblioteca-padrão.

5.7 Escreva uma função `int algum_digito(char str[])` que testa se uma cadeia é contém algum carater de dígito decimal (`'0'` e `'9'`). O resultado deve ser 1 em caso afirmativo e 0 em caso negativo.

Sugestão: utilize a função `isdigit` da biblioteca-padrão.

5.8 Escreva uma função `int forte(char str[])` que verifica se uma cadeia de carateres é uma *palavra passe forte* usando o seguinte critério:

1. deve ter pelo menos 6 carateres;
2. deve conter pelo menos uma letra maíscula, uma letra minúscula e um algarismo.

O resultado da função deve ser 1 se ambos os critérios se verificam e 0 caso contrário. Por exemplo: `"Abr4cadabra"` e `"Apric0t"` são palavras-passe fortes, mas `"Ub40"` não é (porque o comprimento é inferior a 6) e `"POLICE"` também não (porque só tem letras maísculas).

Sugestão: pode usar a função `strlen` da biblioteca-padrão para calcular o comprimento da cadeia e as funções `islower`, `isupper`, `isdigit` para testar os três tipos de carateres.

5.9 Escreva uma função `int decimal(char str[])` que converte uma cadeia de carateres com algarismos de 0 a 9 no valor inteiro decimal correspondente. Por exemplo: `decimal("1234")` deve dar retornar o inteiro 1234.

5.10 Escreva uma função `int calc(char str[])` que implementa uma mini-calculadora: a cadeia de carateres dada tem sempre 3 carateres correspondentes a dois algarismos decimais (`'0'` até `'9'`) e um sinal de operação no meio (`'+'`, `'-'`, `'*'`). A função deve calcular o valor da expressão e retornar o inteiro correspondente.

Exemplos: `calc("5-3")` dá 2; `calc("2*3")` dá 6.

5.11 Defina uma função `void eliminar(char str[], char ch)` que elimina a primeira ocorrência de um carater `ch` de uma cadeia de carateres. Exemplo: se `str = "ABBA"`, então depois de executar `eliminar(str, 'B')` devemos ter `str = "ABA"`.

Tenha o cuidado de colocar corretamente o terminador `'\0'`.

5.12 Escreva uma função `int anagramas(char str1[], char str2[])` que determina se duas cadeias de carateres são *anagramas*, isto é, se se escrevem com os mesmos carateres. O resultado deve ser 1 em caso afirmativo e 0 caso contrário. Por exemplo, "deposit" e "topside" são anagramas:

```
char str1[] = "deposit";
char str2[] = "topside";
int r = anagramas(str1,str2); // resultado 1
```

Sugestão: para cada string, defina um vetor de contadores (em que os índices corresponderão aos códigos ASCII), que utilizará para contar o número de ocorrências de cada caracter de cada uma das strings.

Apêndice: funções da biblioteca-padrão

```
#include <ctype.h>

int isupper(int ch); // testes
int islower(int ch);
int isdigit(int ch);

int toupper(int ch); // converter em maiúsculas
int tolower(int ch); // converter em minúsculas

#include <string.h>

size_t strlen(char str[]); /* comprimento */
```