UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
DEPARTAMENTO DE INFORMÁTICA E MATEMÁTICA APLICADA

Exercícios Algoritmos e Linguagem C
◁ Exercícios - parte 7 ▷

**Observações:**

1. as strings que admitem espaço são referidas como **frase**(x), onde x é a quantidade máxima de caracteres (incluindo \0) (leia com gets)
2. as strings que não admitem espaço são referidas como **palavra**(x) (leia com scanf)
3. assuma que a entrada respeita os limites de tamanho da string/palavra especificada, com isso não há risco de buffer overflow
4. assuma que não há caractere *multi-character* ('é', 'à', etc) nas strings
5. as questões com ⊵ não possui casos de teste. Edite o arquivo lista7.sh, adicionando os casos de teste no seguinte formato:
   avaliar 'qx' 'entrada' 'saida esperada'
   Por exemplo, o caso de teste do exemplo 3 da questão 11 deve ser colocado como:
   avaliar 'q11' '-gm12' '-12'
6. os seus casos de teste devem gerar erros em códigos que estão realmente errados
7. **Envie também o seu lista7.sh**

**Questões:**

1. ▷ Você está implementando um jogo revolucionário chamado campo minado... 1D. A entrada do programa consiste em uma palavra(20) com "." e "*" , onde o primeiro representa um espaço vazio e o segundo representa uma bomba. Depois o programa deve ler um inteiro representando um índice na palavra (começando de 0) e escrever na tela "bum!" caso nesse índice haja uma bomba ou escrever na tela um inteiro representando quantas bombas há na adjacência do índice em questão.

   Exemplo 1:
   ```
   ..**..*..*.. 0
   0
   ```

   Exemplo 2:
   ```
   ..**..*..*.. 1
   1
   ```

   Exemplo 3:
   ```
   ..**..*..*.. 2
   bum!
   ```

2. ▷ Escreva um programa em C que leia uma frase(100) e escreva na tela qual(is) letra(s) minúscula(s) **não** aparece(m) na string.

Exemplo 1:

**Tres pratos de trigo para tres tigres tristes**
b c f h j k l m n q u v w x y z

3. ▷ Escreva um programa que verifica se uma string contém um código válido. Para ser tal, a string deve estar exatamente no formato: AB-X. Ou seja 4 caracteres, sendo nessa ordem: 2 dígitos, traço e 1 dígito verificador. Um dígito verificador têm como objetivo evitar fraudes ou erros de digitação/transmissão. O dígito X deve ser o resto da divisão de $2 * A + 3 * B$ por 7 (lembre-se de que é o valor inteiro do dígito e não o valor pela tabela ASCII). O programa deve ler uma palavra(20) e escrever na tela "sim" caso seja um código válido ou "nao" caso contrário.

Exemplo 1:
**35-0**
sim

Exemplo 2:
**35-00**
nao

Exemplo 3:
**74-4**
nao

4. ▷ Escreva um programa que leia uma frase(100). Em seguida deve escrever na tela a mesma string invertida.

Exemplo 1:
**O seguro morreu de velho**
ohlev ed uerrom oruges O

5. ▷ Um dos princípios utilizados para compactação de arquivos é que símbolos que ocorrem com mais frequência podem ser codificados com menos bits enquanto que os que ocorrem com menos frequência podem ser codificados com mais bits. Escreva um programa em C que leia uma frase(200) e escreva na tela, em ordem alfabética, quantas vezes aparece cada uma das letras minúsculas e maiúsculas, desde que haja pelo menos uma ocorrência (maiúsculas primeiro).

Exemplo:
**Ando devagar porque ja tive pressa**
A: 1
a: 4
d: 2
e: 4
g: 1
i: 1
j: 1
n: 1
o: 2
p: 2
q: 1
r: 3
s: 2
t: 1
u: 1
v: 2

6. ⊵ Escreva um programa em C que leia uma frase(200) e escreva na tela a quantidade de palavras da string. Define-se aqui palavra como qualquer sequência de caracteres diferentes de espaços.

   Exemplo 1:
   **Quantas palavras ha nessa frase?**
   5

   Exemplo 2:
   **Este eh um ponto .**
   5

7. ⊵ Escreva um programa em C que leia uma frase(200), altere todos os caracteres de acordo com essa escrita *leet*: a → 4, e → 3, i → 1, s → 5, t → 7 e, em seguida, escreva-a na tela.

   Exemplo:
   **Esta eh uma string**
   E574 3h um4 57r1ng

8. ⊵ Palíndromos são palavras ou frases que são as mesmas se lidas no sentido contrário, possivelmente ignorando os espaços. Alguns exemplos:

   - arara
   - osso
   - o galo no lago
   - anotaram a data da maratona

   Escreva um programa que leia uma frase(100) e escreva na tela "sim" caso a string seja um palíndromo ou "não" caso contrário.

   Exemplo:
   **anotaram a data da maratona**
   sim

9. ▷ Escreva um programa em C que leia uma palavra(20) e em seguida escreva se essa string representa uma placa válida ("sim" ou "não" ). Uma string é uma placa válida se contiver somente e na sequência: 3 letras maiúsculas, hífen e 4 dígitos numéricos.

   Exemplo 1:
   **AGH-3201**
   sim

   Exemplo 2:
   **AGh-3201**
   não

   Exemplo 3:
   **AG3201**
   não

10. ▷ Escreva um programa em C que leia uma **palavra(20)** e em seguida escreva se essa string representa uma data (dia/mês) válida ("sim" ou "nao" ). Uma string é uma data válida se contiver exatamente 5 caracteres, sendo nessa ordem: 2 dígitos numéricos (para o dia), o caractere '/', 2 dígitos numéricos (para o mês). Além disso, o número de dias deve estar nos limites do respectivo mês e os meses entre 1 e 12. Os meses que contêm 30 dias são: 4, 6, 9 e 11. Assuma que o ano não é bissexto, portanto o mês 2 contém 28 dias. Os demais meses contêm 31 dias.

    Exemplo 1:
    **03/05**
    sim

    Exemplo 2:
    **03/05/2018**
    nao

    Exemplo 3:
    **01/13**
    nao

    Exemplo 4:
    **1/02**
    nao

11. $\trianglerighteq$ Atoi é uma função em C que transforma uma string em número inteiro. Seu objetivo aqui será de implementar uma funcionalidade parecida. Escreva um programa em C que leia uma palavra(30), **transforme em inteiro** de acordo com as seguintes regras e escreva na tela esse número inteiro.

    Regras:

    - pode haver caracteres diferentes de dígitos numéricos, mas eles são ignorados
    - o número pode ser negativo, desde que o - seja o **primeiro** caractere

    Exemplo 1: **3491**
    3491

    Exemplo 2: **0001k-jt3**
    13

    Exemplo 3: **-gm12**
    -12