Lista de Exercícios

6) Strings

a) Escreva uma função que receba uma string str e inteiros positivos i e j e devolva uma string com o mesmo conteúdo que o segmento str[i..j]. Escreva duas versões: na primeira, sua função não deve alocar novo espaço e pode alterar a string str que recebeu; na segunda, sua função deve devolver uma cópia do segmento str[i..j] e não pode alterar a string str que recebeu.

b) Escreva uma função booleana que receba strings s e t e decida se s é um segmento de t. Escreva um programa que use a função para contar o número de ocorrências de uma string s em uma string t.

c) Escreva uma função booleana que decida se uma string ASCII dada é um palíndromo (ou seja, se o inverso da string é igual a ela). Escreva um programa para testar a função.

d) Escreva uma função que receba uma string ASCII e exiba uma tabela com o número de ocorrências de cada um dos caracteres do alfabeto ASCII na string. Escreva um programa para testar a função.

e) O exemplo abaixo conta o número de vogais em uma string. Faça uma função que conte o número de consoantes, excluindo caracteres especiais ('/n', '/t', '.', ‘,’, etc.)

```
int contaVogais (byte s[]) {
  byte *vogais;
  vogais = "aeiouAEIOU";
  int numVogais = 0;
  for (int i = 0; s[i] != '\0'; ++i) {
    byte ch = s[i];
    for (int j = 0; vogais[j] != '\0'; ++j) {
      if (vogais[j] == ch) {
        numVogais += 1;
        break;
      }
    }
  }
  return numVogais;
}
```

f) Escreva uma função que receba uma string ASCII de 0s e 1s, interprete essa string como um número natural em notação binária e devolva o correspondente int. Por exemplo, a string "1111011" deve ser convertida no inteiro 123. (Se a string for longa demais, descarte os *últimos* dígitos.)

g) Escreva uma função que receba um número natural n e devolva a string ASCII de 0s e 1s que represente n em notação binária. Por exemplo, o número 123 deve ser convertido na string “1111011".

h) Escreva uma função que receba um vetor de bytes b[1..n] e um inteiro m e devolva a posição da primeira ocorrência de m espaços (bytes 32) consecutivos em b. (Você pode imaginar que os elementos de b representam caracteres ASCII, embora isso seja irrelevante.) Procure examinar o menor número possível de elementos de b. Escreva um programa para testar sua função.

i) Familiarize-se com a função strstr da biblioteca string que localiza a primeira ocorrência de uma string em outra. Procure descobrir o algoritmo que strstr implementa.

j) Cebolinha é um personagem de história em quadrinhos que quando falava, trocava o "r" pelo "l" (problema conhecido como dislalia). Faça um programa que gera uma versão de um texto fornecido com todos "r" e "rr" trocados por "l", exceto no caso em que o "r" ocorre no final de uma palavra.

l) Busca em **orig** por possíveis candidatos a endereços de e-mail. Copia em **dest** a primeira sequência de caracteres não brancos encontrada que contenha o caractere **'@'**, ou uma string vazia caso contrário. Assuma que o vetor **dest** é grande o suficiente para conter a palavra encontrada.
Exemplo:
Para orig="Enviar mensagem para lucas@ig.com.br ou para outro moderador"
temos "lucas@ig.com.br" que ficará em dest.

void ProcuraEmail(char dest[], char orig[]);