Teste – Monta sentença

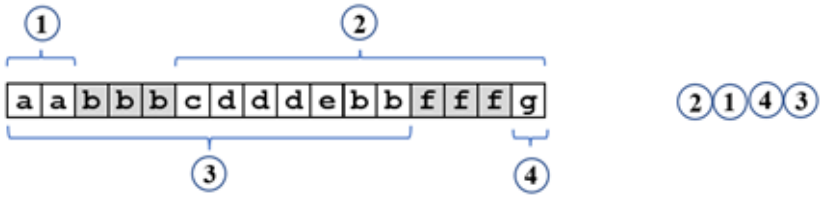
Observações importantes:

i. A única função de `string.h` que você pode usar é `strlen`.
ii. Todas as strings devem ser alocadas dinamicamente (usando `malloc`) do tamanho exato necessário, sem desperdício de memória. Trate do caso de insuficiência de memória e libere memória tão logo possa.
iii. Em todas as funções pedidas, caso não seja possível criar uma nova *string*, a função deve retornar NULL (apenas a *main* imprime mensagens de falta de memória).
iv. Faça todos os testes que achar necessários (a escolha adequada dos testes faz parte da sua formação como programador).

a) Escreva uma função que recebe uma string **não vazia**, quebra essa string em duas novas strings separadas pela maior sequência de caracteres iguais, presentes na string recebida. A função retorna uma nova string como descrita e exemplificada mais abaixo.
Por exemplo:
"aabbbcdddebbfffg" quebra em "aa" e "cdddebbfffg" considerando como maior sequência "bbb".
Porém, também considere a opção de quebrar na última ocorrência da maior sequência. Por exemplo:
"aabbbcdddebbfffg" quebra em "aabbbcdddebb" e "g" considerando como maior sequência "fff".
Uma das novas strings ou até mesmo ambas podem ser vazias. Por exemplo, "hh" resulta em duas strings vazias.

A função deverá alocar espaço para a nova string que será composta pelas quatro partes resultantes das quebras, nessa ordem:

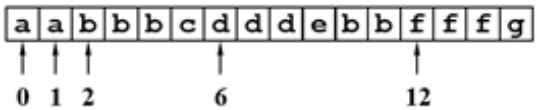
1. Pela parte à direita da sequência localizada da esquerda para a direita (nº 2 na figura: "cdddebbfffg")
2. Pela parte à esquerda da sequência localizada da esquerda para a direita (nº 1 na figura: "aa")
3. Pela parte à direita da sequência localizada da direita para a esquerda (nº 4 na figura: "g")
4. Pela parte à esquerda da sequência localizada da direita para a esquerda (nº 3 na figura: "aabbbcdddebb")



Nesse caso, a sua função deverá devolver "cdddebbfffgaagaabbbcdddebb". Sequência **2143**.

Use obrigatoriamente uma função auxiliar, descrita no item (b), para encontrar a posição da maior sequência de caracteres iguais de uma string.
Use obrigatoriamente uma função auxiliar **recursiva** e desenvolvida por você (descrita no item (c)) para copiar conteúdo de uma string para outra string.

b) Escreva uma função para retornar o tamanho da maior sequência de uma string não vazia e devolver a sua posição. Por exemplo, para "aabbbcdddebbfffg", a função retorna 3 e devolve a posição 2 (i.e. onde começa "bbb").
Considere a existência de um parâmetro nessa função que indica a opção pela primeira ou última ocorrência. Por exemplo, com a opção pela última, temos que para "aabbbcdddebbfffg", a função retorna 3 e devolve a posição 12 (i.e., onde começa "fff").



c) Escreva uma função, obrigatoriamente **recursiva** que receba dois ponteiros para caracteres e um inteiro representando o limite de caracteres e copie a string apontada pelo segundo ponteiro para o local apontado pelo primeiro ponteiro ou até o limite informado. Dica: finalize inserindo o caractere nulo no destino.

d) Faça um programa para testar a sua função do item (a). Faça a quantidade que você achar necessária de testes. Não use os exemplos deste texto nos seus testes (embora você possa testar o seu programa com os exemplos). O seu programa não deverá realizar leitura de dados.

Exemplo 1: String original: "aabbbcdddebbfffg"
Strings quebradas: "aa" e "cdddebbfffg" e também "aabbbcdddebb" e "g"
String devolvida do item (a): "cdddebbfffgaagaabbbcdddebb"

Exemplo 2: String original: "aaccccbnhh"
Strings quebradas: "aa" e "bnhh" e também "aa" e "bnhh"
String devolvida do item (a): "bnhhaabnhhaa "

Exemplo 3: String original: "ppppp"
Strings quebradas: "" e "" e também "" e ""
String devolvida do item (a): ""

Exemplo 4: String original: "asdfg"
Strings quebradas: "" e "sdfg" e também "asdf" e ""
String devolvida do item (a): "sdfgasdf"