Prática Strings

Coloque #include <string.h> e #include <ctype.h> e em todos os programas

1) Utilizando strings, crie um programa que leia o nome de uma pessoa e, em seguida, mostre o nome lido. O nome pode ter no máximo 50 caracteres. Use a função gets() para efetuar a leitura

   Exemplo de saída:

```
<< Vetor de Char>>

Digite um nome: Bruno

O nome digitado é:  Bruno
```

2) Faça um programa semelhante ao anterior, só que o nome digitado deve ser mostrado em letras maiúsculas e também em minúsculas. Procure em string.h uma função que faça a conversão

   Exemplo de saída:

```
<< Vetor de Char>>

Digite um nome: BruNo

O nome digitado é:  BRUNO
O nome digitado é:  bruno
```

3) Faça um programa em que o usuário digita dois nomes, e o programa informa se os nomes são iguais. Use strcmp.

   Exemplo de saída:

```
<< Compara nomes>>

Digite o nome 1: Bruno
Digite o nome 1: Bruno

Os nomes digitados são iguais
```

   Exemplo de saída:

```
<< Compara nomes>>

Digite o nome 1: Bruno
Digite o nome 1: BrunO

Os nomes digitados são diferentes
```

4) Refaça o exercício anterior, para que agora não haja diferenciação entre caracteres maiúsculos e minúsculos.

Exemplo de saída:

```
<< Compara nomes>>

Digite o nome 1: Bruno
Digite o nome 1: BrunO

Os nomes digitados são iguais.
```

5) Faça um programa que recebe o nome e sobrenome de uma pessoa (máximo 40 caracteres), copie o nome para um vetor e o sobrenome para outro. Mostre os dois vetores copiados.

```
<< Vetor de Char>>

Digite um nome e sobrenome: Bruno Travençolo

Nome: Bruno
Sobrenome: Travençolo
```

6) Faça um programa que mostre as letras de uma string separadamente.
Exemplo de saída:

```
<< Separa letras>>

Digite o nome : Bruno


1a letra: B
2a Letra: r
3a Letra: u
4a Letra: n
5a letra: o
```

7) Faça um programa que receba do usuario uma string. O programa imprime a string sem suas vogais.
   Exemplo de saída:

```
<<Sem vogais>>

Digite a string : Correios

Saída sem vogais: Crrs
```

8) Faça um programa em que a pessoa entra com o seu nome, CPF, RG, endereço, CEP, Cidade, Estado e o programa crie, automaticamente, uma carta de declaração (veja o modelo abaixo)
   Exemplo de saída:

```
Digite seu nome: Maria de Lurdes
Digite seu CPF: 356.874.875-98
Digite seu RG: 24.565.789-5
Digite seu endereço: Rua Major José, 45
Digite a cidade: Uberlândia
Digite o estado: MG
Digite o CEP: 38400-890


                    D E C L A R A Ç Ã O


Eu, Maria de Lurdes, RG 24.565.789-5, CPF 356.874.875-98,
 residente a Rua Major José, 45, Uberlândia-MG,
CEP 38400-890 venho por meio desta declarar que vou estudar para a prova
```

9) Receba uma palavra e imprima de trás-para-frente

```
<<invertendo>>

Digite a string : Correios

Saída invertida: soierroC
```

10) Faça um programa que dado uma string, imprima se ela for palíndromo ou não. Lembrando que: Um palíndromo é uma palavra que tem a propriedade de poder ser lida tanto da direita para a esquerda como da esquerda para a direita.

    Ex: ovo, arara, Ana, Bob

11) Considere o seguinte programa

```
#include <stdio.h>
#include <string.h>

int main(void)
{
    char buff[5];
    int pass = 0;

    printf("\n Entre com a senha : \n");
    gets(buff);

    if (strcmp(buff, "1234"))  {
        printf ("\n Senha Errada \n");
    }  else   {
        printf ("\n Senha Correta \n");
        pass = 1;
    }

    if (pass) {
       /* O usuário acertou a senha, poderá continuar*/
        printf ("\n Acesso liberado \n");
    } else {
        printf ("\n Acesso negado \n");
    }

    return 0;
}
```

a) Qual é a função do programa?

b) Qual a saída quando a senha correta é digitada (senha correta: 1234)

c) Qual a saída para as seguintes senhas

   a. 123
   b. 12345
   c. 123456

d) Em quais casos o programa obteve um comportamento inválido. Explique o motivo.

12) A função fgets() também permite a leitura de strings do teclado. A diferença para a função gets() é que é possível limitar o número de caracteres que serão lidos do teclado (no caso representado por stdin – standard input).
Exemplo:

```
char nome[100];
fgets(nome,100,stdin);
```

Com esse comando é possível ler um nome com até 99 caracteres (o centésimo caractere é '\0', colocado automaticamente pelo fgets()).

Uma observação importante é que o fgets armazena o caractere `\n` (newline). Assim, no exemplo acima, se o nome digitado for Maria, a string nome fica assim:

'M'	'a'	'r'	'i'	'a'	**'\n'**	'\0'	lixo	lixo	Lixo

O seguinte comando pode ser usado para substituir o `\n` por `\0`

```
nome[strcspn(nome, "\n")] = '\0';
```

a) Pense em uma solução para o problema do exercício 11 que utiliza fgets. Considere que o tamanho máximo de uma senha são 4 dígitos.
b) Teste com os seguintes valores
    a. 123
    b. 12345
    c. 123456