Prática 06 – *Strings*

Desenvolva os seguintes algoritmos em linguagem de programação C:

1. Escrever um programa para ler uma *string* dada pelo usuário e mostrá-la invertida.

2. Escreva um programa para ler um *array* de caracteres dado pelo usuário e substituir todas as vogais pelo caractere '#'. Exemplo: se usuário digita "carro", mostrar "c#rr#".

3. Uma das formas mais importantes de se preservar a segurança dos dados em computadores é a criptografia. Na criptografia por substituição, o alfabeto normal é trocado por outro, com as letras fora de ordem. Na codificação da mensagem, cada caracter é trocado pelo seu correspondente no novo alfabeto; na decodificação, é feito o processo inverso. Escreva um programa de criptografia com as seguintes especificações:
   - Ler, via teclado, o tipo de operação (codificação/decodificação);
   - Ler, via teclado, a mensagem a ser codificada/decodificada;
   - Codificar/decodificar a mensagem lida segundo o código secreto abaixo;
   - Imprimir a mensagem criptografada/descriptografada resultante.

Alfabeto normal:	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	
Novo alfabeto:	p	x	c	r	b	t	y	j	m	z	c	i	s	w		l	a	f	u	d	n	h	o	g	q	e	k

4. Escreva um programa simples de adivinhação. Crie uma palavra qualquer no programa, relacionada a um tema. Por exemplo: se você escolher o tema "times de futebol", crie uma *string* com o nome de um time qualquer. Indique ao usuário o tema e a 1ª letra da palavra e peça a ele que adivinhe a palavra. Dê a ele 3 chances de acertar, ou seja, ele pode digitar até 3 vezes. A cada digitação, verificar se acertou.

5. Escreva um programa para verificação de senhas criptografadas. O programa deve inicializar um *array* com uma senha criptografada, ler do usuário uma senha e criptografá-la segundo a tabela da questão 3. Em seguida, verificar se a senha está correta. Mostrar o resultado (afirmativo ou negativo). A senha pode conter números, mas estes não serão criptografados.

6. Escreva um programa para ler uma frase dada pelo usuário e contar quantas vezes cada uma das vogais aparecem na frase. Mostrar esses valores e qual vogal aparece mais.

Observações:

strlen(*string*);

A função devolve o comprimento da *string* x. Em outras palavras, devolve o número de caracteres de x (sem contar o '\0' final). Uso típico: k = strlen(*string*);

strcmp(*string*1,*string*2);

Compara lexicograficamente as *strings* apontadas por x e y. Devolve um número negativo se *string*1 vem antes de *string*2, devolve 0 se *string*1 é igual a *string*2 e devolve um número positivo se *string*1 vem depois de *string*2. Uso típico: if (strcmp(*string*1,*string*2) == 0)... ;

strcpy(*string*1,*string*2);

Copia a *string*2 (inclusive o '\0' final) no espaço alocado para a *string*1. Antes de chamar a função, certifique-se de que o espaço alocado a *string*1 tem pelo menos strlen(*string*2) + 1 bytes. A função devolve *string*1. Uso típico: strcpy(*string*1, *string*2);

strcat(*string*1,*string*2);

Concatena as duas *strings*, isto é, acrescenta *string*2 ao final de *string*1. Devolve o endereço da *string* resultante, ou seja, devolve *string*1. Antes de chamar a função, certifique-se de que o espaço alocado a *string*1 é suficiente para comportar strlen(*string*2) bytes adicionais (após o '\0' que marca o fim de *string*1). Uso típico: strcat(*string*1, *string*2);

EXEMPLO

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define TAM 15

int main(int argc, char *argv[]) {
      char nome[TAM],texto[TAM],copia[TAM];
      int i;
      puts("Insira um nome: ");
      gets(nome);
      printf("\nNome invertido: ");
      for(i=TAM-1;i>=0;i--) {
            printf("%c", nome[i]); /*imprime resíduo da string!!!  verificar!*/
      }
      puts("\nInsira um texto: ");
      gets(texto);
      printf("\nResultado da comparacao: %d\n", strcmp(nome,texto));
      printf("\nCopia do vetor TEXTO para vetor COPIA: %s\n", strcpy(copia,texto));
      printf("\nConcatenacao dos vetores: %s\n", strcat(nome,texto));
      printf("\n%s possui %d letras\n", nome, strlen(nome));
      printf("\nComparando as 3 primeiras letras do vetor NOME: %d\n",
strncmp(nome,"raf",3));
      system("PAUSE");
      return 0;
}
```