

## STRINGS - RESUMO

`fgets / puts / strcat / strstr / strcpy / strcmp / strlen`

- **fgets (s, n, stdin)**

Lê uma cadeia de caracteres do teclado (até o ENTER) e armazena na variável `s` (acrescentando do `'\0'` no final); `n` é o tamanho de `s` e `stdin` refere-se à entrada padrão (no caso, o teclado).

A função **fgets**, no entanto, copia o ENTER para a cadeia digitada no teclado, caso a mesma termine com ENTER. Assim, a função **strlen** retornará sempre o sucessor do número de caracteres que compõe o string. A função **gets** não tinha esse problema, porém ela foi descontinuada. Para corrigir o problema, deve-se usar o comando:

```
s[strlen (s)-1] = '\0';
```

- **puts (s)**

Imprime o conteúdo da variável `s` na tela (até o `'\0'`);

- **strcat (s1, s2)**

Acrescenta a cadeia `s2` no final da cadeia `s1`; deve-se garantir que `s1` possua espaço suficiente para acomodar `s2`;

- **strstr (s1, s2)**

Determina se `s2` é subcadeia de `s1`; em caso afirmativo retorna um valor diferente de zero, em caso negativo retorna zero;

- **strcpy (s1, s2)**

Copia o conteúdo da variável `s2` para a variável `s1`;

- **strcmp (s1, s2)**

Compara os conteúdos das variáveis `s1` e `s2` segundo o critério da ordenação lexicográfica; retorna um valor negativo se `s1 < s2`, zero se `s1 = s2` ou um valor positivo se `s1 > s2`;

- **strlen (s)**

Retorna um número inteiro (maior ou igual a zero) que representa a quantidade de caracteres contidos na variável `s` (ou seja, o comprimento da cadeia contida em `s`); no **printf**, deve-se usar o formato `%ld`.

As funções **fgets** e **puts** estão definidas na biblioteca `<stdio.h>`. As demais funções estão definidas na biblioteca `<string.h>`. Alternativamente, pode-se manipular cadeias de caracteres como vetores apenas. Também, pode-se usar o formato `"%s"` para imprimir cadeias de caracteres na função **printf**. Outras funções estão disponíveis na biblioteca `<string.h>`.

## STRINGS – EXERCÍCIOS

1. (\*) Ler uma cadeia de caracteres e determinar quantas vogais essa cadeia contém.
2. (\*) Ler uma cadeia de caracteres e, a partir dela, obter uma outra cadeia em que as letras minúsculas sejam substituídas por maiúsculas e vice-versa.
3. (\*) Ler uma cadeia de caracteres e, a partir dela, obter uma outra cadeia em que os caracteres estão rebatidos (primeiro caracter torna-se o último e assim por diante) e, além disso, todas as vogais estão substituídas por "#".
4. Ler uma seqüência indeterminada de cadeias de caracteres e indicar:
  - Qual cadeia tem o maior comprimento e qual o seu comprimento;
  - Qual cadeia tem o menor comprimento e qual o seu comprimento;
  - Qual cadeia é a primeira na ordenação alfabética;
  - Qual cadeia é a última na ordenação alfabética.

A cadeia vazia deverá ser usada para sinalizar o final da seqüência.

```
#include <stdio.h>
#include <string.h>
#define N 20
int main () {
    char s [N];
    char maior [N];
    char menor [N];
    char primeira [N];
    char ultima [N];
    fgets (s,N,stdin);
    s[strlen(s)-1]='\0';
    if (strlen (s)!=0) {
        strcpy (maior,s);
        strcpy (menor,s);
        strcpy (primeira,s);
        strcpy (ultima,s);
        fgets (s,N,stdin);
        s[strlen(s)-1]='\0';
        while (strlen (s)!=0) {
            if (strlen (s)>strlen (maior)) strcpy (maior,
s);
            if (strlen (s)<strlen (menor)) strcpy (menor,
s);
            if (strcmp (s,primeira)<0) strcpy (primeira,
s);
            if (strcmp (s,ultima)>0) strcpy (ultima, s);
            fgets (s,N,stdin);
        }
    }
}
```

```

        s[strlen(s)-1]='\0';
    }
    printf ("A maior é %s.\n",maior);
    printf ("A menor é %s.\n",menor);
    printf ("A primeira é %s.\n",primeira);
    printf ("A última é %s.\n",ultima);
}
return 0;
}

```

5. Ler uma seqüência indeterminada de cadeias de caracteres e gerar uma nova cadeia de caracteres composta pela concatenação de todas as cadeias previamente lidas, exceto as que contêm a subcadeia "abc" ou o caracter "d".

```

#include <stdio.h>
#include <string.h>
#define NS 20
#define NR 200
int main () {
    char s [NS];
    char r [NR];
    strcpy (r,"");
    fgets (s,NS,stdin);
    s[strlen(s)-1]='\0';
    while (strlen (s)!=0) {
        if ((strstr (s,"abc")==0)&&(strstr (s,"d")==0))
    strcat (r,s);
        fgets (s,NS,stdin);
        s[strlen(s)-1]='\0';
    }
    printf ("%s\n",r);
    return 0;
}

```

6. Ler e armazenar numa matriz NS cadeias de caracteres, cada uma com o máximo de CS-1 caracteres. Em seguida, copiar para uma nova matriz apenas as cadeias de caracteres que possuem comprimento variando entre a metade do maior comprimento e o maior comprimento verificado entre todas as cadeias previamente lidas.

```

#include <stdio.h>
#include <string.h>
#define NS 5
#define CS 20
int main () {
    char m [NS][CS];
    char n [NS][CS];
    int i, j, maior;
    for (i=0;i<NS;i++) {
        strcpy (m[i],"");
    }
}

```

```

        strcpy (n[i], "");
    }
    maior = 0;
    for (i=0;i<NS;i++) {
        fgets (m[i],CS,stdin);
        m[i][strlen(m[i])-1]='\0';
        if (strlen(m[i])>maior) maior = strlen (m[i]);
    }
    printf ("---");
    j=0;
    for (i=0;i<NS;i++) {
        if (strlen(m[i])>=maior/2) strcpy (n[j],m[i]);
        j++;
    }
    for (i=0;i<NS;i++)
        printf ("%s\n", n[i]);
    return 0;
}

```

(\*) devem ser feitos tratando strings como vetores.