Introdução à Programação #21

Henrique Dias
henrique.dias@pplware.com

21 de janeiro de 2015

Depois de vos explicarmos o que são e como se utilizam *strings* na Linguagem C, é importante saber como as podemos ler e imprimir no ecrã. Antes de continuarmos, é importante referir que o "seletor"utilizado para strings é *%s*.

1 Imprimir *Strings*

A impressão de *strings* no ecrã pode ser feita de diversas formas. Iremos abordar duas delas: a função *printf*, já conhecida por todos vós, e a função *puts*.

1.1 Função *printf*

Esta função já é conhecida por todos vós e permite a impressão de diversos tipos de dados de forma bastante simples.

Sintaxe Para a utilizar com *strings* basta escrever da seguinte forma:

```
1 printf("Esta é uma string: %s", nomeDaString);
```

Ou seja, é bastante útil quando necessitamos de imprimir uma *string* variável intercalada com algum outro texto.

1.2 Função *puts*

Por outro lado, temos a função *puts* que quer dizer *put string*. Esta função é bastante útil quando precisamos imprimir apenas uma sequência de caracteres variável.

Sintaxe Ora veja a sintaxe:

```
1 puts(nomeDaString);
```

2 Leitura de *Strings*

Finalmente, abordaremos como se leem *strings* escritas pelo utilizador.

2.1 Função *scanf*

A utilização da função *scanf* para obter sequências textuais é semelhante à forma que utilizamos para ler números, tanto inteiros, como decimais. Porém, existe uma pequena diferença.

Sintaxe Veja a sintaxe e tente descobrir a diferença que existe (além do uso de *%s*):

```
scanf("%s", nomeDaString);
```

Como pode verificar, ao contrário do que acontecia com os restantes tipos de variáveis, neste não colocamos o operador & antes do nome da variável. Porquê? Porque as variáveis que contêm *strings* são *arrays*, logo o seu nome aponta para a primeira posição da memória ocupada por todo o *array*. Daí não ser necessário o uso do operador &.

Exemplo No seguinte exemplo pode ver como utilizamos esta função para obter diversos dados do utilizador que depois são imprimidos recorrendo à função *printf* anteriormente abordada.

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

int main()
{
    char nome[21],
         apelido[21],
         morada[51],
         codigoPostal[11];

    printf("Por favor insira os seus dados conforme pedido:\n\n");
    printf("Primeiro nome: ");
    scanf("%s", nome);

    printf("Último nome: ");
    scanf("%s", apelido);

    printf("Morada: ");
    scanf("%s", morada);

    /* Limpeza do buffer no Windows.
    Usar "_fpurge(stdin)" em sistemas Unix */
    fflush(stdin);

    printf("Código Postal: ");
    scanf("%s", codigoPostal);

    printf("\nO seu Cartão de Identificação:\n");
    printf("Nome: %s, %s\n", apelido, nome);
    printf("Morada: %s\n", morada);
    printf("Código Postal: %s\n", codigoPostal);
    return 0;

}
```

2.2 Função *gets*

Outra função que nos permite ler *strings* é a função *gets*, cujo nome advém de *get string*, ou seja, obter *string*. A utilização desta função é algo bastante simples.

Sintaxe Ora veja a sintaxe:

```
gets(nomeDaString);
```

Onde "nomeDaString" se refere ao nome da variável, do *array* que contém a sequência de caracteres.

Exemplo Aqui pode ver como fica o programa do exemplo anterior utilizando esta função.

```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    char nome[21],
         apelido[21],
         morada[51],
         codigoPostal[11];

    printf("Por favor insira os seus dados conforme pedido:\n\n");
    printf("Primeiro nome: ");
    gets(nome);

    printf("Último nome: ");
    gets(apelido);

    printf("Morada: ");
    gets(morada);

    printf("Código Postal: ");
    gets(codigoPostal);

    printf("\nO seu Cartão de Identificação:\n");
    printf("Nome: %s, %s\n", apelido, nome);
    printf("Morada: %s\n", morada);
    printf("Código Postal: %s\n", codigoPostal);
    return 0;
}
```