

**ALUNO: PAULO MORAES PESSOA JARDIM**

**PROFESSOR: EDVAR**

**TEMA: ARTIGO SOBRE STRING E O USO DA BIBLIOTECA STRING.H**

**MATÉRIA: TÉCNICAS DE PROGRAMAÇÃO 1**

**ARTIGO SOBRE STRINGS E O USO DA BIBLIOTECA STRING.H**

**PAULO MORAES PESSOA JARDIM**

UNIVERSIDADE FEDERAL RURAL DA AMAZÔNIA

PARÁ-BRASIL

E-mail: paulompj010@gmail.com

**RESUMO:**

Este artigo falará sobre strings, desde a parte teórica, como, por exemplo, o conceito. Até a parte prática, como os comandos, sintaxe, além de conter exemplos minuciosamente resolvidos e explicados, para a melhor compreensão do leitor.

**INTRODUÇÃO:**

Nesta introdução, abordaremos primeiramente a parte teórica:

**Conceito:** O termo string é utilizado na programação para identificar uma sequencia de caracteres, ou seja, textos na prática. Isto é, string também pode ser entendida como uma cadeia de caracteres. Na programação, é muito comum a manipulação de cadeias de caracteres, isso porque, a maioria das informação que nós seres humanos trabalhamos, está no formato de texto. Assim, processamos caracteres a toda hora e rapidamente descobrimos a grande necessidade em tratar e processar textos. Por exemplo, toda linguagem de programação precisa ser interpretada e, para isso, deve haver o reconhecimento de diversos nomes e outros símbolos. Cada parte, ou melhor, cada segmento de texto reconhecido possuíra significado diferente, algumas vezes serão expressões, eai em outras, apenas sinais.

**Características de uma string:** Entramos agora no quesito das características que compõe uma string. Um objeto string, por exemplo, é imutável, ou seja, que o texto que ele carrega nunca é alterado. O comprimento de uma string é o seu número de bytes, sem contar o byte nulo final, além disso, é importante citar também que o endereço de uma string é o endereço do seu primeiro byte.

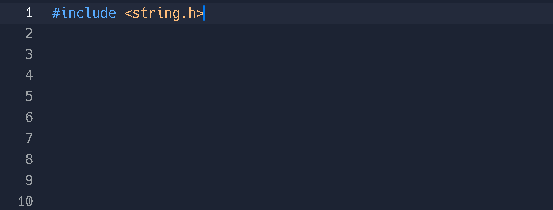
Sendo assim, entendemos que uma string é qualquer sequência de caracteres. Caractere é a menor unidade de todo o texto. Os caracteres são classificados em 3 grupos distintos, são eles: numérico, letras do alfabeto e caracteres especiais. Ao longo desde artigo, será comum ao se referir ao conjunto de números e letras, com o termo alfanumérico.

**DESENVOLVIMENTO**

Tendo já entendido o significado de string e suas derivações, mergulharemos de fato nos **comandos**, **importação** e **sintaxe** de uma string em linguagem C. Todavia, é de suma importância falarmos antes sobre a biblioteca <string.h>.

Ela é uma biblioteca padrão da linguagem C, nela estão contidos protótipos para a manipulação de strings, também conhecidos como array de char. As funções dessa biblioteca são capazes de fazer desde a contagem, cópia e concatenação, comparação e diversas outras utilidades, como veremos mais adiante.

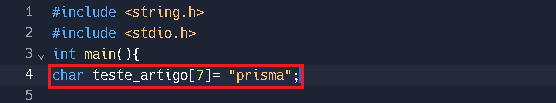
Para importar a biblioteca <string.h> é simples, veja a seguir:



Basta digitar na primeira linha do seu código #include <string.h>

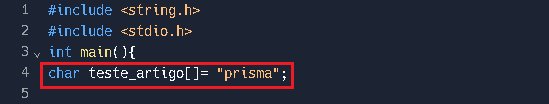
Agora com a biblioteca importada, vamos entender como funciona os **comandos** e a **sintaxe**:

Para fazer uma **declaração de string** em C, basta fazer o seguinte:



Crie a variável com o char, e de um nome a variável. Este comando cria a variável teste\_artigo como um vetor char com capacidade para armazenar 7 caracteres. Entretanto, acontece que o último caractere de uma string, vai ser sempre o caractere nulo “\0” que serve para indicar o final de uma string. Logo, neste exemplo dado, teremos apenas 6 caracteres úteis para armazenamento. Tal como indica a palavra “prisma” ao lado, que tem seis letras.

Tendo entendido essa lógica, podemos fazer a declaração de outra maneira:



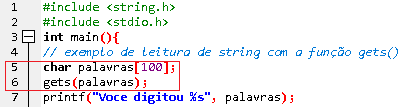
Neste caso, não colocamos dentro dos colchetes o tamanho do vetor. Logo, a quantidade de caracteres para armazenamento é calculada de forma a ter o tamanho exato para conter a string que está sendo atribuída. No caso, a palavra prisma, 6 letras mais o espaço para o terminador “\0”, ficamos com 7 espaços úteis, mas apenas 6 que podem ser realmente usados.

Então, se você declarar 100 espaços uteis dentro do colchete, o usuário poderá inserir 99 caracteres.

**COMANDO GETS:**

É uma função que serve para a leitura de strings a partir do teclado. Sua sintaxe é bem simples, basta usar como parâmetro o nome do vetor onde a string será armazenada, veja a seguir:

SINTAXE DO GETS : gets( )

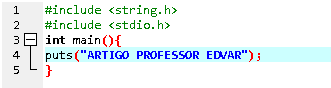


Primeiro declaramos o vetor onde a string será salva (Linha cinco) depois bastar utilizar o comando gets e colocar dentro dele o nome do vetor onde a string ficará guardada (Linha seis), pronto, agora quando o programa for executado, o nome digitado pelo usuário será lido pelo comando gets e armazenado no vetor declarado, o gets funciona parecido com o comando scanf. Após isso, se quiser, pode imprimir na tela o nome digitado pelo usuário e que ficou armazenado no vetor, utilizando o comando printf, como mostra a linha 7.

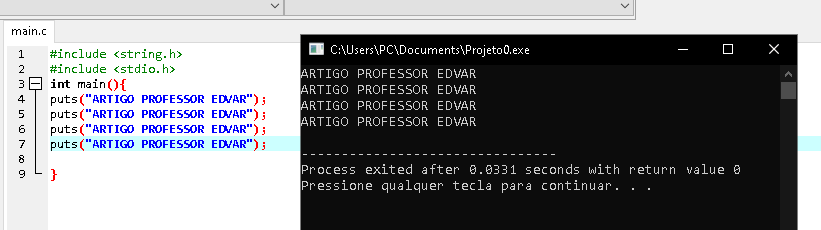
**COMANDO PUTS:**

Sua sintaxe também é bem simples, esse comando pode receber um vetor de caracteres ou uma string literal. Tem o funcionamento parecido com do printf, a diferença é que o comando puts já pula uma linha, diferente do printf que precisaria usar o “\n”. Veja os exemplos a seguir:

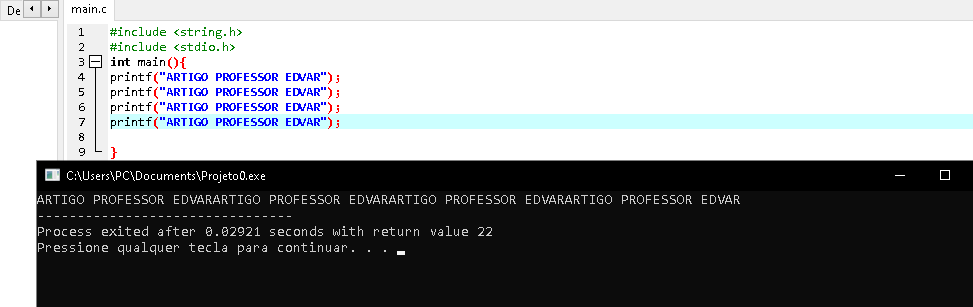
SINTAXE DO PUTS: puts( )



No caso acima, estamos usando o comando puts apenas para imprimir uma mensagem literal (string literal). Para usar corretamente, basta digitar o comando puts, abrir parênteses, depois abrir aspas e digitar a mensagem desejada (linha quatro). Entretanto, observe nesse próximo exemplo que diferente do printf o comando puts já pula uma linha depois da mensagem, veja:

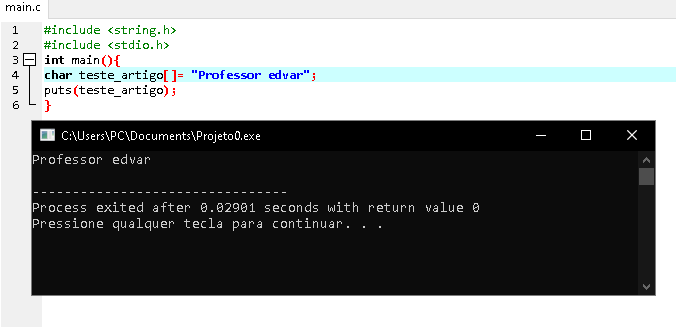


Se tivéssemos usado o printf, veja como ficaria:



Observe que as mensagens ficaram imprensadas uma na outra, se tivéssemos colocado o \n no final de cada mensagem, o programa teria ficado igual ao executado pelo comando puts.

Outra finalidade do comando puts é que ele também pode receber um vetor de caracteres, veja no exemplo a seguir:



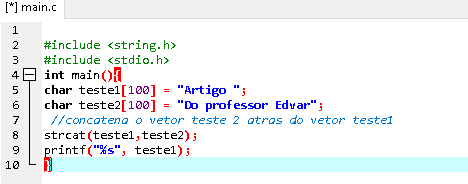
Primeiramente, declaramos o vetor teste\_artigo e atribuímos a ele o valor “professor edvar” (LINHA QUATRO), após isso, mandamos o comando puts imprimir o vetor teste\_artigo na tela (LINHA5), para isso basta digitar puts, abrir parênteses e colocar dentro dos parênteses o nome do vetor.

**COMANDO STRCAT**

Este comando serve para concatenar dois strings, ou seja, juntar uma string ao final da outra. Veremos a sintaxe dela e como funciona, com exemplos, a seguir.

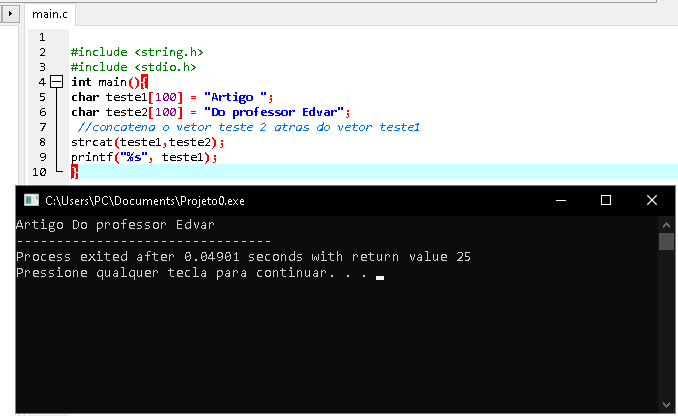
SINTAXE DO STRCAT: strcat (char destino, char fonte).

A sintaxe será mais bem explicada com o exemplo e a explicação abaixo:



Primeiramente, neste exemplo, declaramos duas strings, a char teste1 e a char teste2(LINHAS 5 E 6) cada uma com um valor diferente atribuído. Digamos que queremos juntar o valor do char teste2 logo atrás do valor char teste1, pra ficar do seguinte jeito( char teste1 char teste 2) ou seja, “Artigo do professor Edvar”. É bem simples de fazer, digitamos o comando strcat, abrimos parênteses, e colocamos na ordem em que queremos. No exemplo acima, colocamos o char teste1 pra ficar na frente e o char teste 2 para ficar atrás.

Daí, o valor do char teste 1 automaticamente se torna char teste1 char teste2, ou seja, ( Artigo do professor Edvar), isto é, o valor char teste2 “cola” no valor do char testes1. Então, basta imprimir na tela o valor do char teste 1, usando o comando printf. Vejamos como fica o programa executado:



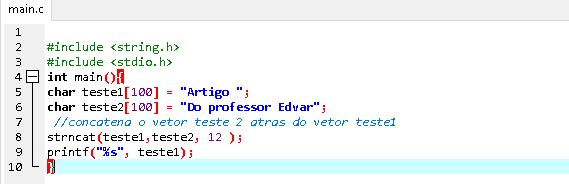
Observe como é importante deixar um espaço entre o final da palavra artigo e a ultima aspas da string teste1, pois se não tivesse, ficaria ( Artigodo Professor Edvar).

**COMANDO STRNCAT**

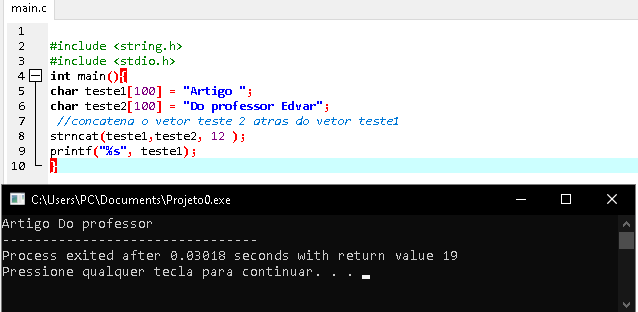
É uma função parecida com a anterior (strcat), a diferença é que com a strncat nós temos a possibilidade de escolher qual a quantidade de caracteres iremos concatenar na string da frente. Vamos explicar minuciosamente a sintaxe e o funcionamento, com o exemplo a seguir:

SINTAXE DO STRNCAT: strncat(char destino, char fonte, num)

A sintaxe será mais bem explicada com o exemplo e a explicação abaixo:



Primeiramente, começamos declarando os vetores de caracteres (char) onde as strings serão armazenadas (LINHAS 5 E 6). Após isso, entramos com a função strncat, que tem a sintaxe parecida com a da função anterior (strcat). Com a diferença que a strncat nós da à opção de escolher quantos caracteres nos iremos concatenar no string da frente, ou seja, nesse exemplo, selecionamos apenas 12 caracteres do char teste2 para concatenar no char teste1, devendo ficar dessa maneira(Artigo do professor), sem o “Edvar”. Lembrando que os espaços em branco contam como caractere. Veremos agora como fica o exemplo sendo executado:



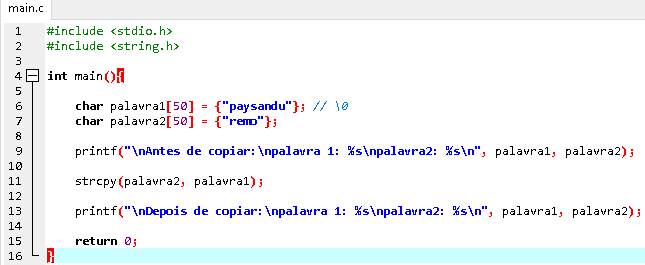
Observe como ficou o resultado do programa. Apenas os 12 primeiros caracteres, incluindo os espaços, do char teste2 concatenaram no char teste1.

**COMANDO STRCPY**

Este comando serve para fazer cópia de string. Iremos ver a sintaxe e como utiliza-lo, a partir do exemplo abaixo:

SINTAXE DO STRCPY: strcpy(char destino, char fonte)

A sintaxe será mais bem explicada com o exemplo e a explicação abaixo;



Primeiro, nas linhas 6 e 7, declaramos os vetores de caracteres, onde as strings ficaram armazenadas. Para usar o comando strcpy, é bem simples, digitamos o comando strcpy, abrimos parênteses, e dentro os parênteses colocaremos primeiro o nome do vetor que receberá o vetor que estiver em segundo lugar, por exemplo, no exemplo acima o vetor palavra 2 veio primeiro, pois receberá o valor do vetor palavra1, ou seja, o vetor palavra2 que antes tinha a string “remo” dentro, receberá o valor do vetor palavra1 que tem a string “paysandu”. Concluindo, o vetor palavra2 perderá a sua string “remo” e receberá a string “paysandu”.

**CONSIDERAÇÔES FINAIS**

Ao final deste artigo, percebe-se que o manuseio de strings em linguagem C não é tão complicado, mas requer um árduo estudo do mesmo. Portanto, o conhecimento da linguagem C como um todo, principalmente ter sapiência com o manejo de strings é de suma importância para qualquer pessoa que queira ser um bom programador.

**REFERÊNCIAS:**

[**https://wagnergaspar.com/como-ler-uma-string-com-a-funcao-gets-na-linguagem-de-programacao-c/**](https://wagnergaspar.com/como-ler-uma-string-com-a-funcao-gets-na-linguagem-de-programacao-c/)

[**https://wagnergaspar.com/como-copiar-uma-string-em-c-com-a-funcao-strcpy/**](https://wagnergaspar.com/como-copiar-uma-string-em-c-com-a-funcao-strcpy/)

[**https://www.unicamp.br/fea/ortega/info/aula09.htm**](https://www.unicamp.br/fea/ortega/info/aula09.htm)

[**https://linguagemc.com.br/a-biblioteca-string-h/**](https://linguagemc.com.br/a-biblioteca-string-h/)

[**https://www.vivaolinux.com.br/script/Comando-strncat**](https://www.vivaolinux.com.br/script/Comando-strncat)

[**https://petbcc.ufscar.br/stringfuncoes/#strncat**](https://petbcc.ufscar.br/stringfuncoes/#strncat)

[**http://www.cmaismais.com.br/referencia/cstdio/puts/#:~:text=puts-,puts%20fun%C3%A7%C3%A3o,)'%20%C3%A9%20impresso%20na%20tela.&text=Em%20caso%20de%20sucesso%2C%20a,um%20n%C3%BAmero%20inteiro%20n%C3%A3o%2Dnegativo**](http://www.cmaismais.com.br/referencia/cstdio/puts/#:~:text=puts-,puts%20fun%C3%A7%C3%A3o,)'%20%C3%A9%20impresso%20na%20tela.&text=Em%20caso%20de%20sucesso%2C%20a,um%20n%C3%BAmero%20inteiro%20n%C3%A3o%2Dnegativo)**.**

[**https://acervolima.com/funcao-strncat-em-c-c/**](https://acervolima.com/funcao-strncat-em-c-c/)