Linguagem

C

Professor Alexandre Moreno

A biblioteca string.h

A biblioteca string.h

STRING: é uma sequência de caracteres, geralmente utilizada para representar palavras, frases ou textos de um programa.

Em linguagem C, uma **string** é então um *array* (**conjunto**) de caracteres, que termina sempre com o caracter '\0', para que desta forma seja possível identificar o final da **string**.

Uma string geralmente é delimitada por aspas quando referenciamos ao seu conteúdo.

A biblioteca string.h

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

char sexo;
char nome[40];

int main(){
    system("pause");
    return(0);
}
```

Um único caractere

40 caracteres

O último caractere será o **\0**, ou seja, conseguiremos inserir apenas 39 caracteres

A biblioteca string.h

[*] exemplo string.cpp

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

char sexo;
char nome[40]="professor moreno";

int main(){
    system("pause");
    return(0);
}
```

Inicialização na declaração

Identifica fim de string

	p	r	o	f	e	s	s	o	r		m	o	r	e	n	o	\0
posição	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

tamanho=16

A biblioteca string.h

exemplo string.cpp

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

char sexo;
char nome[40];

int main(){
    nome="professor moreno";
    system("pause");
    return(0);
}
```

Se Liga No Erro

lador | Recursos | Log do Compilador | Debug | Resultados da Busca | Fechar

Unidade	Mensagem
F:\CESUMAR\Compiladores C\exem...	In function `int main()':
F:\CESUMAR\Compiladores C\exem...	incompatible types in assignment of `const char[17]' to `char[40]'

A biblioteca string.h

exemplo string.cpp

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

char sexo;
char nome1[40]="professor moreno";
char nome2[40];

int main(){
    nome2=nome1;
    system("pause");
    return(0);
}
```

Linguagem C não permite fazer atribuição direta de uma *string* a outra

A biblioteca string.h

A biblioteca string.h da linguagem C, contém uma série de funções para manipular strings.

Nesta aula veremos como:

- *Copiar strings em C usando **strcpy** e **strncpy**;*
- *Concatenar strings em linguagem C usando **strcat** e **strncat**;*
- *Descobrir o tamanho de uma string em C usando **strlen**();*
- *Comparar strings em C usando **strcmp**();*

Strcpy - Realiza a cópia do conteúdo de uma variável a outra.

Sintaxe:

strcpy(string_destino, string_origem);

UniCesumar

A biblioteca string.h

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h> //necessário para strcpy

int main ()
{
  char nome[15];

  strcpy(nome, "Fulano de Tal");

  printf("Nome = %s \n ", nome);

  system("pause");
  return 0;
}
```

Strncpy - Realiza a cópia do conteúdo de uma variável a outra, porém, deve ser especificado o tamanho a ser copiado.

Sintaxe:

strncpy(string_destino, string_origem, tamanho);

A biblioteca string.h

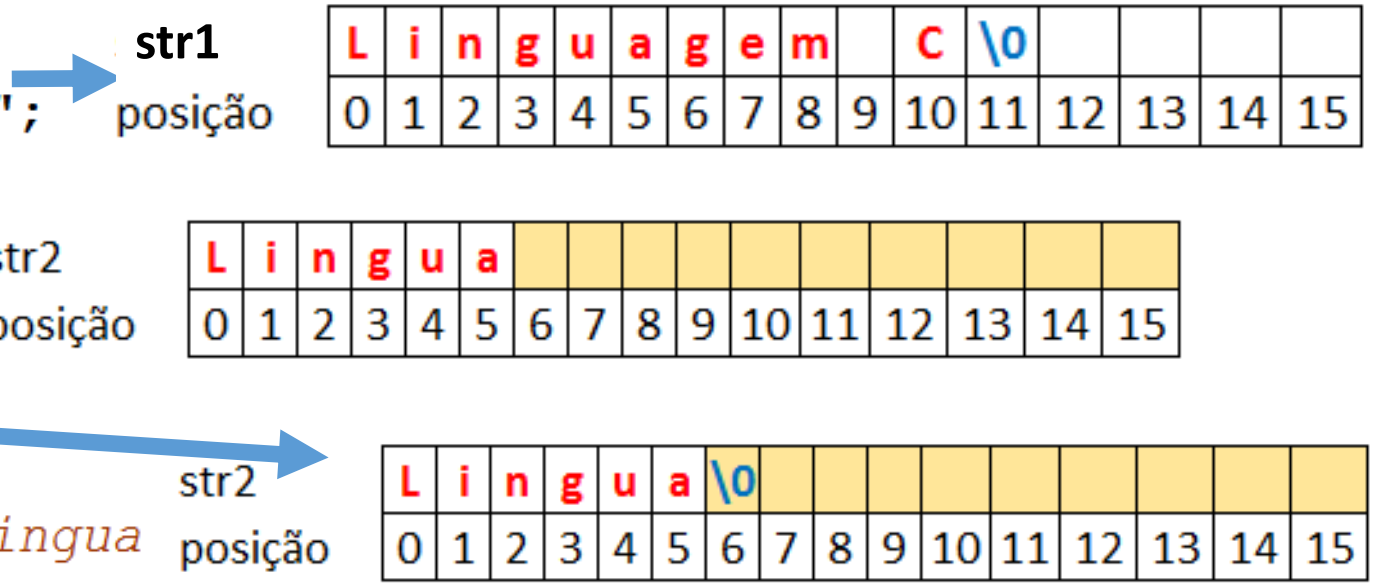
```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h> //necessário para strncpy

int main ()
{
  char str1[15] = "Linguagem C";
  char str2 [7];

  strncpy(str2, str1, 6);

  str2[6] = '\0';
  printf("str2 = %s\n", str2);
  // será apresentado str2 = Lingua

  system("pause");
  return 0;
}
```



Strcat - Realiza a concatenação do conteúdo de uma variável a outra. Ambas devem ser strings.

Sintaxe:

strcat(string_destino, string_origem);

UniCesumar

A biblioteca string.h

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h> //necessário para strcat

int main ()
{
  char str[30] = "Curso";
  strcat(str, " de linguagem C");
  //Concatena a string " de C" com o conteúdo da string str

  printf("str = %s\n", str);
  //será apresentado: str = Curso de linguagem C

  system("pause");
  return 0;
}
```

Strncat - Realiza a concatenação do conteúdo de uma variável a outra, porém, deve ser especificado o tamanho a ser concatenado. Ambas devem ser strings.

Sintaxe:

strncat(string_destino, string_origem, tamanho);

A biblioteca string.h

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h> //necessário para strcat

int main (void)
{
  char str1[20] = "Curso";
  char str2[20] = " de linguagem C";

  strncat(str1, str2, 13);
  //concatena a string1 com 15 posições da string2

  printf("str1 = %s\n", str1);
  //Será exibido Curso de Linguagem


  system("pause");
  return 0;
}
```

Strlen – retorna a quantidade de caracteres contidas na string

Sintaxe:

variável do tipo inteiro = **strlen**(string);

A biblioteca string.h

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h> //necessário para strcat

int main ()
{
  char str[50] = "Curso de linguagem C";
  int tamanho;

  tamanho = strlen(str);

  printf("O tamanho da string %s = %d \n \n", str, tamanho);
  //será apresentado O tamanho da string Curso de linguagem C = 20


  system("pause");
  return 0;
}
```

Strcmp – Compara o conteúdo de duas strings;

Possíveis valores de retorno:

0: conteúdo das strings são iguais

-1: conteúdo da string1 é menor do que string2

1: conteúdo da string1 é maior do que string2

Sintaxe:

variável do tipo inteiro = **strcmp**(string1, string2);

UniCesumar

A biblioteca string.h

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h> //necessário para strcat

int main ()
{
  char str1[4] = "abc";
  char str2[4] = "abd";
  int retorno;

  retorno = strcmp(str1, str2);
  printf("retorno = %d\n", retorno);
  //mostra o retorno da função strcmp

  system("pause");
  return 0;
}
```

Strncmp – Também faz a comparação do conteúdo de duas *strings*, porém, deve ser especificado o tamanho a ser comparado.

Possíveis valores de retorno:

=0: conteúdo das *strings* são iguais
-1: conteúdo da string1 é menor do que string2
1: conteúdo da string1 é maior do que string2

Sintaxe:

variável do tipo inteiro = **strcmp**(string1, string2, tamanho);

Strncmp – Também faz a comparação do conteúdo de duas strings, porém, deve ser especificado o tamanho a ser comparado.

Possíveis valores de retorno:

0: conteúdo das strings são iguais
-1: conteúdo da string1 é menor do que string2
1: conteúdo da string1 é maior do que string2

Sintaxe:

variável do tipo inteiro = **strcmp**(string1, string2, tamanho);

UniCesumar

A biblioteca string.h

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h> //necessário para strncmp

int main ()
{
  char str1[15] = "Curso de C";
  char str2[15] = "Curso de Java";
  int retorno;

  retorno = strncmp(str1, str2, 5);

  printf("retorno = %d\n", retorno);

  system("pause");
  return 0;
}
```

A biblioteca ctype.h

É possível **converter** um caractere em **maiúsculo** utilizando comando da biblioteca **ctype.h**

Sintaxe:

variável do tipo caractere = **toupper**(caractere);

A biblioteca ctype.h

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

char nome1[40]="professor moreno";
                012345
int main(){

    nome1[0]=toupper(nome1[0]);

    printf("%s \n",nome1);
    system("pause");
    return(0);
}
```

Será convertido APENAS o primeiro caractere (posição zero) da string

A biblioteca ctype.h

exemplo string.cpp

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

char nome1[40]="professor moreno";
int x;
int main(){

    for (x=0; x<strlen(nome1); x++)
       nome1[x]=toupper(nome1[x]);

    printf("%s \n",nome1);
    system("pause");
    return(0);
}
```

012345

Para converter uma sequencia de caracteres (*string*), é necessário executar a conversão caractere a caractere, dentro de um laço de repetição

strlen=**16**

caracteres de **0** a **15**

Também é possível **converter** um caractere em **minúsculo** utilizando comando da biblioteca **ctype.h**

Sintaxe:

variável do tipo caractere = **tolower**(caractere);

Também é possível **converter** uma string em **maiúsculo** ou **minúsculo** utilizando comando da biblioteca **string.h**

Sintaxes:

strupr(variável); //maiúsculo

strlwr(variável); //minúsculo

A biblioteca ctype.h

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

int main(){
 char pala[50];
 printf("Informe uma palavra: ");
 gets(pala);
 printf("Palavra maiusculo: %s \n",strupr(pala));
 printf("Palavra minúsculo: %s \n",strlwr(pala));
 system("pause");
 return 0;
}
```