



Linguagem C *String*

Instituto Federal de Educação, Ciência e Tecnologia do Triângulo Mineiro
Prof. Edwar Saliba Júnior
Janeiro de 2018



Manipulando Caracteres

- Se queremos ler um caractere, o código a seguir nos atende perfeitamente:

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      char letra;
7
8      printf("Digite um caractere: ");
9      scanf("%c",&letra);
10
11     printf("\n\nA letra digitada foi: %c", letra);
12
13     return 0;
14 }
```

- E se quisermos ler uma sequência de caracteres?

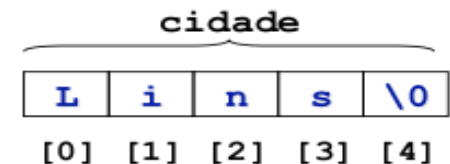


Lendo Sequências de Caracteres

- Uma sequência de caracteres do tipo `char` é denominada `string`;
- Uma `string` nada mais é do que um vetor de caracteres do tipo `char`;
- Toda `string` é terminada pelo caracter `'\0'`;
- Exemplo:

```
char cidade[5];
```

```
...  
cidade[0] = 'L';  
cidade[1] = 'i';  
cidade[2] = 'n';  
cidade[3] = 's';
```





Lendo *Strings*

- Ou você atribui posição por posição:

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      char palavra[5];
7
8      palavra[0] = 'L';
9      palavra[1] = 'i';
10     palavra[2] = 'n';
11     palavra[3] = 's';
12     palavra[4] = '\0';
13
14     printf("\n\nA palavra digitada foi: %s", palavra);
15
16     return 0;
17 }
```



Lendo *Strings*

- Ou você lê do teclado com o comando

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      char palavra[5];
7
8      printf("Digite um caractere: ");
9      gets(palavra);
10
11     printf("\n\nA palavra digitada foi: %s", palavra);
12
13     return 0;
14 }
```



Atribuindo e Copiando Strings

- Exemplo:

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      // Atribuição.
7      char palavra[] = "Lins";
8
9      printf("\n\nA palavra é: %s", palavra);
10
11
12     // Cópia.
13     strcpy(palavra, "Itu");
14
15     printf("\n\nA palavra é: %s", palavra);
16
17     return 0;
18 }
```

- No caso de cópia, é responsabilidade do programador verificar se a variável comporta o tamanho de `string` que está sendo



Exibindo *Strings*

- Comando `printf` e `puts`:

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      char nome[] = "Paullo";
7      char sobrenome[] = "Sampaio Penna";
8
9      // Imprime em uma linha, com uma linha de comando.
10     printf("%s %s \n\n", nome, sobrenome);
11
12     // Imprime em diversas linhas com diversas linhas de comando.
13     puts(nome);
14     puts(sobrenome);
15
16     return 0;
17 }
```



Funções Auxiliares

- `strlen(string)` - retorna o número de caracteres existentes antes do `'\0'`;
- `strcpy(destino, origem)` - atribui a uma variável do tipo `string` uma constante ou o valor de uma outra variável do tipo `string`;
- `strcmp(str1, str2)` - compara duas variáveis ou valores do tipo `string` e retorna um número maior que zero se `(str1 > str2)`, zero se `(str1 == str2)` e menor que zero se `(str1 < str2)`;
- `strcat(str1, str2)` - retorna a concatenação das `strings` passadas como parâmetro.



Exercício

- Implemente programas que simulem cada uma das funções apresentadas no *slide* anterior. Cada programa deverá possuir uma das funções apresentadas a seguir:
 - `int tamanho_string(char str[])` - deverá retornar o tamanho de uma `string` digitada pelo usuário;
 - `void copia_string(char str1[], char str2)` - deverá copiar o valor de `str2` para `str1` e mostrar o valor de `str2`;
 - `void concatena_string(char str1[], char str2[])` - deverá concatenar `str1` e `str2` e mostrar o valor da concatenação.



Bibliografia

- LAUREANO, Marcos. **Programação em C para ambiente Linux**. Disponível em: <http://br-c.org/doku.php>. Acesso em: 06 fev. 2011.
- MURTA, Cristina Duarte. *Slides* da disciplina de Programação de Computadores I. CEFET-MG, 2010.
- SENNE, Edson Luiz França. **Primeiro Curso de Programação em C**. 2. ed. Florianópolis: Visual Books, 2006.
- SOARES, Gabriela Eleutério. *Slides* da disciplina de Programação de Computadores I. CEFET-MG, 2011.