

Fundamentos de Programação

Linguagem C++: arrays de caracteres (strings)

Bruno Emerson Gurgel Gomes¹

¹Instituto Federal de Educação, Ciência e Tecnologia do RN (IFRN)

2011

1 Introdução

2 Cadeias de caracteres

- Lendo e imprimindo strings
- Algumas funções da biblioteca C++
- Biblioteca `<cstring>`

- *Arrays* são capazes de armazenar dados de qualquer tipo
- Um tipo especial de *array* é o de caracteres
 - Também denominado de *string*
- Um *string* é um array do tipo **char** terminado com o caractere '`\0`' (nulo)

Cadeias de caracteres (strings)

- O tamanho de um *string* = quantidade de caracteres + caractere nulo.
- Declaração:

String constante:

```
char str1[] = "exemplo";
```

```
char str2[] = {'e', 'x', 'e', 'm', 'p', 'l', 'o', '\0'};
```

String variável(determinada em tempo de execução):

```
char str3[8];
```

Lendo e imprimindo strings com cin e cout

- O comando **cin** >> **str** lê uma *string* da entrada padrão e a armazena na variável *str*
 - A entrada termina quando é encontrado um espaço em branco
 - O caractere nulo ('\0') é automaticamente inserido no final da string após o término da entrada
 - **str** deve ter tamanho suficiente para armazenar os caracteres
 - Sujeito a perda de dados ou comportamento imprevisível
- **cout** << **str** imprime os caracteres da *string* até que o caractere nulo seja encontrado
- **cout** e **cin** não verificam se o tamanho da *string* foi atingido

Exemplo: lendo com cin

trecho de programa

```
char nome[80];  
  
cout << "Digite o seu nome: ";  
cin >> nome;  
cout << "O nome digitado foi: " << nome;
```

Resultado

```
Digite o seu nome: Bruno Gomes  
O nome digitado foi: Bruno
```

Algumas funções da biblioteca C++ para leitura de string

<cstdio>

`gets(str)` : Lê uma string da entrada até que seja digitada a tecla ENTER.
EVITE UTILIZAR: Não verifica o tamanho da string

<iostream>

`cin.getline(str, tam, delim)` : Lê uma *string* (removendo o delimitador da entrada) até que:

- *tam - 1* seja atingido
- o indicador de fim de arquivo seja inserido
- o caracter delimitador (*delim*) seja encontrado.

`cin.getline(str, tam)` : Mesmo que o anterior, mas sem especificar um caractere delimitador

`cin.get(str, tam, delim)` : Mesmo que *getline*

`cin.get()` : lê apenas um caracter por vez. Retorna 0 quando o fim de arquivo é encontrado.

Exemplo: lendo com cin.getline

trecho de programa

```
char nome[80];  
  
cout << "Digite o seu nome: ";  
cin.getline(nome, 80);  
cout << "O nome digitado foi:" << nome;
```

Resultado

```
Digite o seu nome: Bruno Gomes  
O nome digitado foi: Bruno Gomes
```

Funções para manipulação de string <cstring>

`char *strcpy(char *s1, const char *s2)` : Copia a *string* *s2* na *string* *s1*.
Retorna *s1*.

`char *strncpy(char *s1, const char *s2, int n)` : Copia *n* caracteres da *string* *s2* na *string* *s1*. Retorna *s1*.

`char *strcat(char *s1, const char *s2)` : Concatena a *string* *s2* na *string* *s1*. O primeiro caracter de *s2* é gravado em cima do caractere nulo de *s1*. Retorna *s1*.

Funções para manipulação de string <cstring>

`int strcmp(const char *s1, const char *s2)` : Compara *s1* com *s2*.

- Se $s1 == s2$, retorna 0
- Se $s1 < s2$, retorna < 0 (menor que 0)
- Se $s1 > s2$, retorna > 0 (maior que 0)

`char *strtok(char *s1, const char *s2)` : Quebra a *string* *s1* em *tokens* separados de acordo com os caracteres contidos na *string* *s2*.

`size_t strlen(const char *s)` : Determina o tamanho da *string*, sem contar com o caracter nulo.

Exemplo: utilizando algumas funções de <cstring>

```
#include <cstring >...
char x[] = "Programando em C++";
char y[25], z[15];

cout << "O string no array x é: " << x
      << "\nO string no array y é: " << strcpy(y, x) << endl;

strncpy(z, x, 14);
z[14] = '\0';

cout << "O string no array z é: " << z << endl;
cout << "O tamanho do array z é: " << strlen(z) << endl;
if (strcmp(x, z)) cout << "x é maior que z\n";
```

```
A string no array x é: Programando em C++
A string no array y é: Programando em C++
A string no array z é: Programando em
A tamanho do array z é: 14
x é maior que z
```

Exemplo: retorna o dia da semana a partir de uma data

```
#include <iostream>

using namespace std;

int main() {
    int dia;

    char dsemana[8][14] = {
        "Dia inválido",
        "Domingo",
        "Segunda-feira",
        "Terça-feira",
        "Quarta-feira",
        "Quinta-feira",
        "Sexta-feira",
        "Sábado"
    };

    cout << "Digite o número do dia da semana:\n";
    cin >> dia;
    cout << "O dia por extenso é: ";
        (dia < 1 || dia > 7) ?
        cout << dsemana[0] :
        cout << dsemana[dia];

    return 0;
}
```