

Ministério da Educação
Universidade Federal de Santa Maria
Curso Técnico em Informática para Internet Integrado ao Ensino Médio
Colégio Técnico Industrial de Santa Maria - CTISM



Algoritmos e Programação

luciana.lourega@ufsm.br





Strings

- Como definimos que uma matriz é um conjunto de dados homogêneos e que podem ser armazenados em uma única variável, não é errado dizer que uma string, que é um conjunto de caracteres, é uma matriz.
- Em relação ao armazenamento de informações do tipo literal na memória, deve-se lembrar que um dado deste tipo possui um certo comprimento dado pelo número de caracteres nele contido.

String



- Portanto, para guardar um dado do tipo literal, devemos alocar (reservar) um espaço contíguo de memória igual ao comprimento do mesmo, destinando um byte para cada caractere da informação.
- Usando a informação ALGORITMO, que possui 9 caracteres, consequentemente, 9 bytes serão necessários para reter a referida informação na memória, mas é conveniente que estejam juntos (posição contíguas).

String

- A primeira posição deste conjunto de bytes é absolutamente arbitrária e sua escolha, geralmente, é feita automaticamente pelo compilador.
- Bem, tendo assumido que uma string (literal) é um conjunto de caracteres e sabendo que conjuntos em algoritmos são tratados como vetores e matrizes, fica fácil tratarmos uma string como matriz. Veja o exemplo:

0	1	2	3	4	5	6	7	8
A	L	G	O	R	I	T	M	O

String

A hand is shown pointing at a digital interface. The background is blue with a pattern of hexagons. Some hexagons contain white icons: a person, a shopping cart, and a group of three people. The word 'String' is written in large, black, sans-serif font across the top of the image.

- Salvetti (1998) define que `string` é uma cadeia de caracteres e que a mesma é uma sequencia de letras, algarismos ou símbolos.
- Cada caractere é uma informação e uma cadeia de caracteres é um conjunto de informações.
- Portanto, uma cadeia de caracteres é um vetor em que cada elemento é um caractere.



String

- Exemplo em sala de aula
- Deve ser observado que o tipo de dados **String** não existe como tipo de dados primitivos na linguagem C, porém, esse tipo pode ser “montado” através de uma matriz de char, afinal, uma palavra é um conjunto de letras.
- Sempre que for obter uma string através da função **scanf()** fique atento para não usar o **operador de endereço de memória**, o **&**, pois em C, uma matriz



String

- já representa diretamente um endereço de memória.
- Uma outra observação diz respeito ao que é armazenado na variável.
- Caso seja digitado **Visual Books**, a variável armazenará apenas a palavra **Visual**, ou seja, é guardado os valores até que um espaço em branco seja encontrado.

The background of the slide features a blue and white hexagonal pattern. A hand is visible in the upper right corner, with the index finger pointing towards a hexagon containing a group of three people icon. Other hexagons contain icons for a shopping cart, a person, and a document.

String

- Caso seja de interesse armazenar uma frase ou palavras separadas por espaço, faça uso da função **gets()**.
- Usando gets() é possível inserir caracteres pelo teclado até que o Enter seja pressionado. No final da string é colocado um terminador nulo e então a função retorna.

The background of the slide features a blue and white hexagonal pattern. A hand is visible in the upper right corner, with the index finger pointing towards a hexagon containing a group of three people icon. Other hexagons contain icons for a shopping cart, a person, and a document.

String

- A função `puts()` pode receber os códigos de barra invertida e é considerada muito mais rápida do que o `printf()`.
- O único problema é que `puts()` trabalha apenas com string de caracteres enquanto que o `printf()` trabalha com todos os tipos de dados.
- Exemplo em sala de aula.




String

- Existem vários tipos de entrada e saída de dados pelo console. Sendo que os mais comuns são as funções `scanf()` e `printf()`, pois podem ser usadas para qualquer tipo de dados existentes em C, além dos dados serem formatados com facilidade.
- **As funções `getch()` e `getche()`**
- As funções `getch()` e `getcher()` retornam imediatamente após uma tecla ser pressionada.



String

- A única diferença é que a `getch()` não mostra o caractere na tela enquanto que a `getche()` mostra.
- Na maioria dos compiladores C, a biblioteca a ser inserida no programa para utilizar essas duas funções é a “`conio.h`”. Essas funções serão utilizadas nos programas.
- Exemplo em sala de aula.

A hand is shown pointing at a digital interface. The interface features a blue background with a pattern of hexagons. Some hexagons contain white icons: a shopping cart, a padlock, and a group of three people. The hand is positioned on the right side of the frame, with the index finger pointing towards the group of people icon.

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
int main()
{
    int i;
    char nome[7];
    printf("\n Informe o nome de uma pessoa com no maximo 7 letras");
    scanf("%s", nome);
    for(i=0;i<7;i++)
        printf("%c", nome[6-i]);
    getch();
}
```

String

- Função **gets()**
 - Lê a string até o primeiro enter
 - Sintaxe: `gets (nome_da_string);`
 - **Exemplo em sala de aula.**
- Função **scanf()**
 - Lê a string até o primeiro espaço em branco.
- `char curso[15] = "Engenharia"; // válido somente na declaração`
- `char curso [15];`
- `strcpy(curso, "Engenharia"); // requer a biblioteca string.h`

String



- Copia a string-origem para a string-destino
- Sintaxe: `strcpy(string_destino, string_origem);`
- **Exemplo em sala de aula.**
- A linguagem C possui funções especiais para análise e manipulação de strings.
- Tais funções estão definidas na biblioteca **string.h**.
- A biblioteca `string.h` possibilita a manipulação de strings completas (sem considerar caractere a caractere).

String



- `strcat (str1, str2)` = concatena `str2` ao final de `str1`
- A string de origem permanecerá inalterada e será anexada ao fim da string de destino.
- Sintaxe: `strcat (string_destino, string_origem);`
- **Exemplo em sala de aula.**
- `int tam = strlen (str1);`
 - Retorna o tamanho de `str1`
 - Sintaxe: `strlen (string);`
 - Exemplo em sala de aula.

String

- `int valor = strcmp (str1, str2);`
 - `valor =0`, se `str1` e `str2` são iguais;
 - `valor < 0`, se `str1 < str2`;
 - `valor > 0`, se `str1 > str2`;
- Compara a `string1` com a `string2`. Se as duas forem idênticas a função retorna zero. Se elas forem diferentes a função retorna não-zero.
- Sintaxe: `strcmp(string1, string2);`
- **Exemplo em sala de aula.**



String

- `strupr (str)`
 - Converte uma string para maiúsculas.
 - **Exemplo em sala de aula.**
- `strlwr (str)`
 - Converte uma string para minúsculas.
 - **Exemplo em sala de aula.**



String

- `strrev (str)`
 - Inverte o conteúdo de uma string.
 - **Exemplo em sala de aula.**
- `strset(str, char)`
 - Substitui todos os caracteres de uma string pelo caractere especificado.
 - **Exemplo em sala de aula.**
- Estas funções são úteis pois não é possível, por exemplo, igualar duas strings:
`string1 = string2;`

String

- **Matrizes de Strings**

- Como definido anteriormente, uma string pode ser uma cadeia de caracteres, mas o que fazer quando se precisa armazenar em uma mesma variável uma série de strings? Faz-se necessário uma matriz de duas dimensões.

	0	1	2	3	4	5	6	7
0	P	R	O	G	R	A	M	A
1	J	O	Ã	O				
2	M	A	R	I	A			
3	R	E	C	E	I	T	A	



String

- Note que cada palavra fica armazenada em uma linha e que cada coluna representa uma letra de cada palavra.
- Sendo assim, assume-se que, sempre que houver a necessidade de se armazenar várias strings (palavras) em uma única variável, ela precisa ser matriz de strings.
- Uma atenção especial deve ser dada quando se aplicar este conceito em linguagens, pois neste caso, deve-se verificar a documentação da mesma para identificar como ela trata a matriz de strings.