

Fundação Hermínio Ometto

Bacharelado em Sistemas de Informação

SIF009 - Linguagem de Programação I

Prof. Dr. Sérgio Luis Antonello

Aula 11

13/10/2020

Plano de ensino

1. Unidade I - Programação estruturada e Linguagem C (objetivos b, c e d).

- 1.1. Conceitos de programação estruturada.
- 1.2. Estrutura de um programa de computador.
- 1.3. Códigos fonte, objeto e executável.
- 1.4. Biblioteca de códigos.
- 1.5. Compiladores e Interpretadores.
- 1.6. Processos de compilação e link edição.
- 1.7. Identificação dos tipos de erros e alertas (léxicos, sintáticos e semânticos).
- 1.8. Depuração de código.
- 1.9. Palavras reservadas.
- 1.10. Tipos de dados.
- 1.11. Constantes. Variáveis simples e estruturadas. Escopo de variáveis.
- 1.12. Operadores e precedência.
- 1.13. Expressões aritméticas, lógicas e relacionais.
- 1.14. Comandos.
- 1.15. Ambientes de desenvolvimento e programação.

2. Unidade II - Estruturas de controle (sequência, decisão e repetição), registro e arquivo (objetivos a, c, d).

- 2.1. Comandos if e switch.
- 2.2. Comandos for, while e do while.
- 2.3. Blocos de comandos e aninhamento.
- 2.4. Definição de tipos.
- 2.5. Registro.
- 2.6. Arquivo: leitura e gravação de dados em disco.

3. Unidade III - Ponteiros e Funções (objetivos a, c, d, e).

- 3.1. Ponteiros.
- 3.2. Funções.
- 3.3. Passagem de parâmetro por valor.
- 3.4. Passagem de parâmetro por referência.

4. Unidade IV - Strings e Variáveis indexadas (objetivos a, c, d).

- 4.1. Manipulação de strings.
- 4.2. Manipulação de caracteres.
- 4.3. Declaração e manipulação de vetores.
- 4.4. Declaração e manipulação de matrizes.

Plano de ensino

Data	Atividade
04/08	Aula 01
11/08	Aula 02
18/08	Aula 03
25/08	Aula 04
01/09	Aula 05
08/09	Aula 06
15/09	Prova 1
22/09	Aula 08
29/09	Aula 09
06/10	Aula 10

Data	Atividade
13/10	Aula 11
20/10	Maratona FHO de Programação
27/10	Aula 13
03/11	Aula 14
10/11	Aula 15
17/11	Prova 2 Entrega Trabalho
24/11	Prova SUB
01/12	Semana Científica
08/12	Aula 19
15/12	Aula 20

Sumário da aula

Primeiro momento (revisão)

- ✓ Conceitos de funções
- ✓ Escopo de variáveis
- ✓ Passagem de parâmetro por valor e por referência

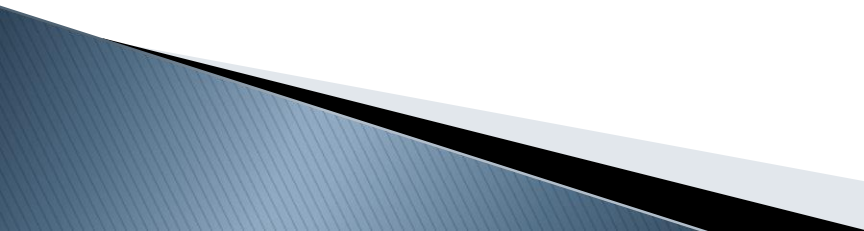
Segundo momento (conteúdo)

- ✓ Conceitos de string
- ✓ Manipulação de string
- ✓ Manipulação de caractere

Terceiro momento (síntese)

- ✓ Retome pontos importantes da aula

1. Primeiro momento: revisão

- ✓ Funções constituem-se de serem um conjunto de comandos agrupados em um bloco, com objetivos específicos, que recebe um nome e através deste nome pode ser evocado a qualquer momento.
 - ✓ Variáveis declaradas dentro de uma função apresentam escopo local.
 - ✓ Variáveis que são declaradas fora das funções tem escopo global.
 - ✓ A passagem de parâmetros é a forma de comunicação de dados entre programas e funções, ou entre duas funções.
 - ✓ Elas ocorrem de duas forma: por valor e por referência.
- 

1. Primeiro momento: revisão

Sempre que a passagem de parâmetros é realizada por valor, não é a variável ou expressão que é enviada para a função, mas sim uma cópia do seu valor.

Quando um parâmetro é passado por referência, o que é fornecido para a função é o endereço de memória da variável correspondente ao parâmetro.

pass by reference



fillCup()

pass by value



fillCup()

1. Primeiro momento: revisão

Ponteiro: declaração, atribuição de valor e aritmética.

```
int var;
```

```
int *p;
```

```
var = 10;
```

```
p = &var;
```

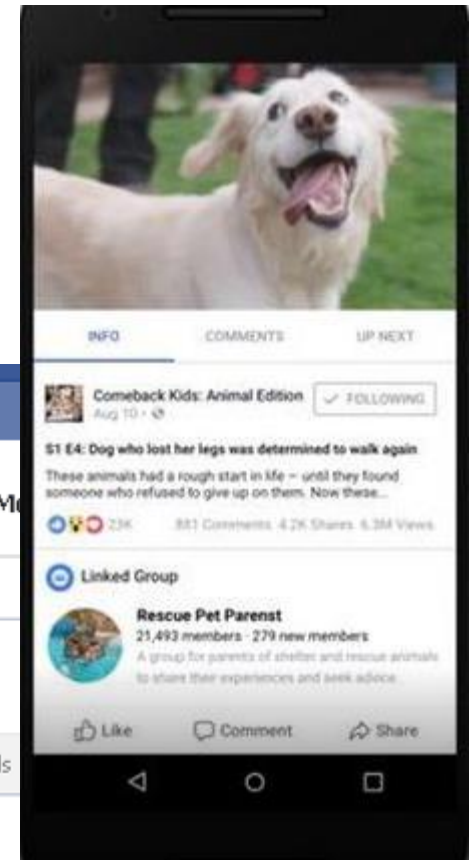
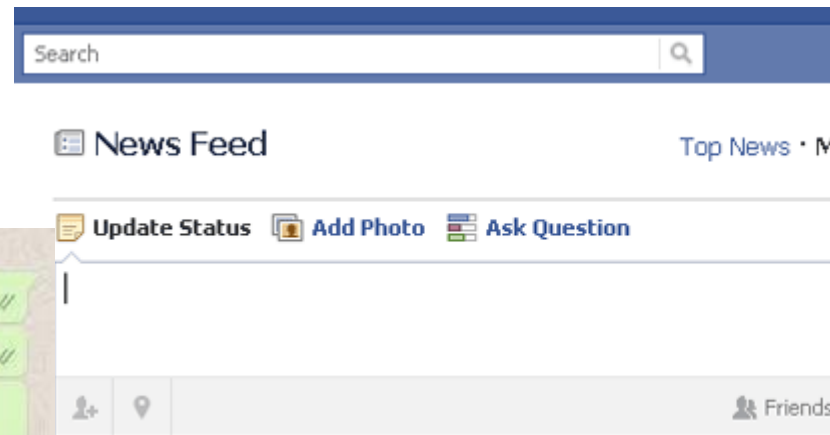
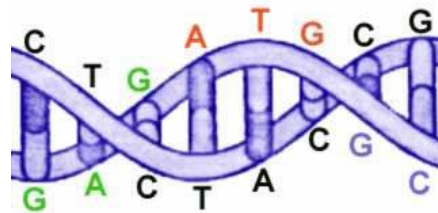
← Declaração do ponteiro de nome “p”

← O ponteiro “p” recebe o endereço de memória da variável “var”, ou seja, agora o ponteiro “p” está apontando para “var”.

- **p++**; passa a apontar para a próxima posição de memória a partir da posição que p apontava.
- **p+=2**; avança duas posições de memória a partir da posição que p apontava.
- **p--**; retroage uma posição de memória a partir da posição que p aponta.

2. Segundo momento: motivação

Em que formato estão grande parte dos dados disponíveis?



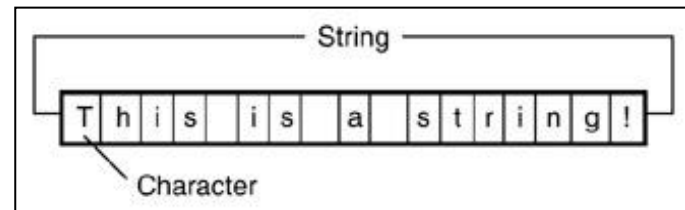
3. String

É uma importante forma de dados, composta por caracteres alfanuméricos e caracteres especiais. Exemplos:

- Nome de uma pessoa
- Parágrafo de um livro
- Conjunto de dígitos como um CPF ou CNPJ.

Em C ANSI, consiste em um vetor de caracteres (tipo char), exemplo:

- `char nome[30];`



Cada caractere pode ser acessado individualmente, como em qualquer vetor, através do uso de colchetes. Exemplo:

- `letra = nome[7];`

3. String

Os compiladores C estão preparados para tratar como string constante, qualquer coisa que esteja entre aspas.

'\0' - o caractere nulo e termina uma string.

Exemplo: `char nome[30];`


Neste exemplo nota-se que:

- declaração de uma variável “nome”
- permite armazenar até trinta caracteres.
- vale lembrar que o último caractere de uma string deve ser o caractere nulo, ou seja, são 29 caracteres de trabalho mais o '\0'.


3. String

É possível trabalhar individualmente cada caractere da string, usando o índice de acesso a ele.

- `char nome[30];`
- `nome[0] = 'A';`
- `nome[1] = 'N';`
- `nome[2] = 'T';`
- `nome[3] = 'E';`
- `nome[4] = 'R';`
- `nome[5] = 'O';`
- `nome[6] = '\0';`



O primeiro caractere da string ocupa a posição de índice zero.



O caractere `\0` inicia o final da string.

4. Manipulação de string

“
String manipulation
”



4. Manipulação de string

Na linguagem C, dentre os comandos de entrada e saída de dados estão:

➤ `scanf("%s", nome);`

- Aceita a entrada de uma única palavra.
- Se encarrega de incluir o caractere nulo no final da string.

➤ `gets(nome);`

➤ `printf("%s", nome);`

- Aceita várias palavras
- Se encarrega de incluir o caractere nulo no final da string.

4. Manipulação de string

Para manipulação de strings são fornecidas diversas funções que estão na biblioteca string.h.

```
#include <stdio.h>
#include <string.h>
int main() {
    char nome[30];
    printf("Digite seu nome: ");
    gets(nome);
    printf("\nOlá %s, tudo bem?\n", nome);
}
```

Biblioteca onde estão funções de manipulação de strings.

4. Manipulação de string

A função `fgets()` possibilita a entrada de dados controlando o número de caracteres a serem armazenados na string correspondente.

Também possibilita entrada de dados armazenados em disco.

O exemplo abaixo mostra a entrada de dados limitada a 30 (trinta) caracteres na string `nome`, a partir da entrada padrão (`stdin`).

```
fgets(nome, 30, stdin);
```


4. Manipulação de string

A função `strlen()` retorna a quantidade de caracteres armazenados na string correspondente.

Ela é conhecida por retornar o tamanho da string.

Não necessariamente esse tamanho corresponde a mesma quantidade de caracteres de quando ela foi declarada e sim quantos caracteres estão armazenados naquele momento.

```
quantidade = strlen(nome);
```

4. Manipulação de string

A função `strcmp()` compara duas strings e retorna um valor numérico correspondente a:

- um valor menor que zero se a primeira string for menor que a segunda;
- um valor igual a zero se as strings forem iguais; e
- um valor maior que zero se a primeira string for maior que a segunda.

Não se trata de comparar qual das duas tem maior quantidade de caracteres.

Consiste em verificar a ordem alfabética de cada caracteres das duas strings, comparação essa baseada na tabela ASCII.

```
x = strcmp(str1, str2);
```

4. Manipulação de string

Atenção com o uso da `strcmp()`:

- A comparação de strings consiste em verificar a ordem alfabética baseada na tabela ASCII.
- Fique atento a questão dos caracteres maiúsculos e minúsculos.

Por exemplo, para `str1` igual a “Abelha” e `str2` igual a “abelha”, o valor retornado em “X” será negativo, porque o valor ASCII de “A” é menor que o valor ASCII de “a”.

```
x = strcmp(str1, str2);
```

4. Manipulação de string

A função `strcpy()` faz a cópia de conteúdo entre duas strings.

- A string `str1` recebe o conteúdo da string `str2`, ou seja, O conteúdo da `str2` é sobreposto sobre o conteúdo da `str1`.

```
strcpy(str1, str2);
```

A função `strcat()` permite concatenar (juntar) o conteúdo de duas strings.

- O conteúdo da `str2` será acrescentado ao final do conteúdo da `str1`, ou seja a `str1` terá juntado os dois conteúdos.

```
strcat(str1, str2);
```

5. Manipulação de caracteres

Uma string é formada por um conjunto de caracteres que podem ser manipulados individualmente.

Ainda assim, existem as variáveis “char” de um único caractere.

Em ambos os casos, a linguagem C oferece um conjunto de funções para manipulação deles.

Essas funções estão na biblioteca [ctype.h](#).



5. Manipulação de caracteres

Dentre as várias funções de manipulação de caracteres, serão apresentadas 3 delas.

A função `tolower()` converte um caractere em minúsculo.

```
str1[3] = tolower(str1[3]);
```

A função `toupper()` converte um caractere em maiúsculo.

```
str1[0] = toupper(str1[0]);
```

A função `isdigit()` verifica se o caractere é um dígito decimal.

```
x = isdigit(CPF[8]);
```

6. Exercícios

**Vamos
Programar!**



6. Exercícios

- 1) Desenvolver um programa que dada a entrada de uma string, mostre seu conteúdo totalmente em maiúsculo, depois totalmente em minúsculo e por último mostre a string original.
- 2) Desenvolver um programa que dada uma determinada string, conte quantas vogais existem.
- 3) Resolva o problema URI 1238 – Combinador.

<https://www.urionlinejudge.com.br/judge/pt/problems/view/1238>

- 4) Resolva o problema URI 1235 - De dentro para fora.

<https://www.urionlinejudge.com.br/judge/pt/problems/view/1235>



7. Terceiro momento: síntese

As strings são formadas por conjunto de caracteres.

Existem funções específicas para manuseio de strings.

Essas funções são encontradas na biblioteca `string.h`.

Existem funções específicas para manuseio de caracteres.

Essas funções são encontradas na biblioteca `ctype.h`.

