

# Aula 04 – Linguagem C

---

Disciplina: Algoritmos

Prof. Allbert Velleniche de Aquino Almeida

E-mail: [allbert.almeida@fatec.sp.gov.br](mailto:allbert.almeida@fatec.sp.gov.br)

Site: <http://www.allbert.com.br>



/allbert.almeida

# Acentuação na Linguagem C

---

- A Linguagem C utiliza a localização padrão "C", está localização é neutra e contém informações mínimas para possibilitar execução;
- Podemos utilizar a função **setlocale** para modificar essa localização, utilizando o idioma que seja adequada ao programa.

# Acentuação na Linguagem C

---

```
#include<stdio.h>
#include<stdlib.h>
#include<locale.h>
main(){
    printf("Não é possível usar acentuação ou ç
corretamente...\n\n");
    printf("\n***** Alterando para a
localidade do sistema *****\n\n");
    setlocale(LC_ALL,"");
    printf("Já posso usar acentuação e também o
character ç...\n\n\n");
    system("pause");
}
```

# Caracteres

---

- Os caracteres são um tipo de dado: o char. O C trata os caracteres ('a', 'b', 'x', etc ...). Na linguagem C, também podemos usar um char para armazenar valores numéricos inteiros, além de usá-lo para armazenar caracteres de texto. Para indicar um caractere de texto usamos apóstrofes. Vejamos um exemplo:

# Caracteres

```
#include <stdio.h>
#include <stdlib.h>
main(){
    char ch;
    ch='d';
    printf ("Tecla %c",ch);
    system("pause");
}
```

- No programa acima, **%c** indica que **printf()** deve colocar um caractere na tela. Um **char** também é usado para armazenar um número inteiro, conhecido como o código ASCII correspondente ao caractere;
- Troque **%c** por **%i** e veja o que ocorrerá;

# Caracteres

---

- Muitas vezes queremos ler um caractere fornecido pelo usuário. Para isto a função mais usada é:
  - **scanf()** leitura da variável;

# Exemplo

---

```
#include <stdio.h>
#include <stdlib.h>
main(){
    char ch;
    scanf("%c",&ch);
    printf ("Tecla %c",ch);
    system("pause");
}
```

# Strings

---

- No C uma string é um **vetor de caracteres** terminado com um caractere nulo. O terminador nulo também pode ser escrito usando a convenção de barra invertida do C como sendo '\0'. Embora o assunto vetores seja discutido posteriormente, veremos aqui os fundamentos necessários para que possamos utilizar as strings. Para declarar uma string, podemos usar o seguinte formato geral:
  - `char nome_da_string[tamanho];`



# Strings

---

- Isto declara um **vetor de caracteres** (uma string) com número de posições igual a **tamanho**. Note que, como temos que reservar um caractere para ser o terminador nulo, temos que declarar o comprimento da string como sendo, no mínimo, um caractere maior que a maior string que pretendemos armazenar. Vamos supor que declaremos uma string de 7 posições e coloquemos a palavra João nela. Teremos:

J	o	a	o	\0	...	...
---	---	---	---	----	-----	-----

# Strings

---

- Se quisermos ler uma string fornecida pelo usuário podemos usar a função **scanf()**. A função **scanf()** coloca o terminador nulo na string, quando você aperta a tecla "Enter" ou ao inserir um espaço.

```
#include <stdio.h>
#include <stdlib.h>
main (){
    char nome[100];
    printf ("Digite uma string: ");
    scanf("%s",&nome);
    printf ("\n\nVoce digitou %s",nome);
    system("PAUSE");
}
```

# Strings

---

- Também podemos ler uma string fornecida pelo usuário usando a função **gets()**. A função **gets()** coloca o terminador nulo na string, quando você aperta a tecla "Enter".

```
#include <stdio.h>
#include <stdlib.h>
main (){
    char nome[100];
    printf ("Digite uma string: ");
    gets (nome);
    printf ("\n\nVoce digitou %s",nome);
    system("PAUSE");
}
```

# Strings

---

- Como as strings são **vetores de caracteres**, para se acessar um determinado caracter de uma string, basta "indexarmos", ou seja, usarmos um **índice** para acessarmos o caracter desejado dentro da string;

```
#include <stdio.h>
#include <stdlib.h>
main (){
    char nome[10];
    printf ("Digite uma string: ");
    gets (nome);
    printf ("\n\nVoce digitou %c",nome[1]);
    system("PAUSE");
}
```

# Strings

---

- **Fique atento:**

- "teste" é na realidade uma **constante string**.
- Isto implica, por exemplo, no fato de que 't' é diferente de "t", pois 't' é um **char** enquanto que "t" é uma constante string com dois **chars** onde o primeiro é 't' e o segundo é '\0'.

# Constantes Strings

Código	Significado
<code>\n</code>	Nova linha ("new line")
<code>\t</code>	Tabulação horizontal ("tab")
<code>\"</code>	Aspas
<code>\'</code>	Apóstrofo
<code>\o</code>	Nulo (o em decimal)
<code>\\</code>	Barra invertida
<code>\v</code>	Tabulação vertical
<code>\a</code>	Sinal sonoro ("beep")
<code>\r</code>	Retrocesso

# Expressões que podem ser abreviadas

Expressão Original	Expressão Equivalente
$x = x + k;$	$x += k;$
$x = x - k;$	$x -= k;$
$x = x * k;$	$x *= k;$
$x = x / k;$	$x /= k;$