



PROGRAMAÇÃO EM C



INTRODUÇÃO

A linguagem C é uma linguagem de programação foi criada nos anos 1972 para desenvolvimento de sistemas operacionais (SOs). Ela é amplamente utilizada em sistemas embarcados. Utilizando linguagem C podemos criar SOs, aplicativos, drivers e outros controladores de dispositivos, programar microcontroladores, etc. A linguagem C é conhecida por gerar programas que rodam muito rápido, principalmente quando comparado com outras linguagens de programação como Java e Python.

DENNIS RITCHIE

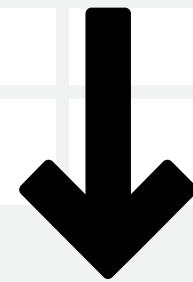


A linguagem C foi usada como referência por várias outras linguagens de programação, ela foi desenvolvida por Dennis Ritchie nos laboratórios Bell em 1972, Dennis é também pai do Unix que acabou sendo base para outros sistemas operacionais.

COMPARAÇÃO



```
graph TD; A[COMPARAÇÃO] --> B[C]; A --> C[PYTHON]; B --> D["• Linguagem compilada<br/>• Mais próximo do hardware<br/>• Execução mais rápida<br/>• Desenvolvimento mais lento."]; C --> E["• Linguagem interpretada<br/>• Mais abstrato<br/>• Execução mais lenta<br/>• Desenvolvimento mais rápido"]
```



C

- Linguagem compilada
- Mais próximo do hardware
- Execução mais rápida
- Desenvolvimento mais lento.



PYTHON

- Linguagem interpretada
- Mais abstrato
- Execução mais lenta
- Desenvolvimento mais rápido

COMPILAR VS INTERPRETAR

COMPILAÇÃO

Código Fonte
(C)

COMPILADOR

Código de
máquina

EXECUTAR

Hello!

INTERPRETAÇÃO

Código Fonte
(Python)

INTERPRETADOR & EXECUTAR

Hello!

COMANDOS

ENTRADA

`scanf()` é o comando que faz a entrada de dados.

SAÍDA

`printf()` é o comando que faz impressão de informações na tela

EXEMPLO

```
#include <stdio.h>

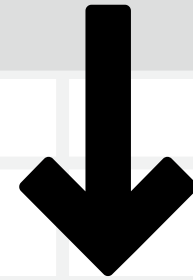
int main() {
    char nome[20];
    int idade;
    float altura;

    printf("Digite seu primeiro nome: ");
    scanf("%s", nome);
    printf("Digite sua idade: ");
    scanf("%d", &idade);
    printf("Digite sua altura: ");
    scanf("%f", &altura);

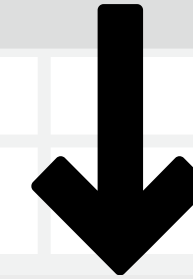
    printf("%s tem %d anos de idade e mede %.2fm.", nome, idade, altura);

    return 0;
}
```


OPERADORES ARITMÉTICOS E RELACIONAIS



	Operador	Significado
Aritméticos	+	Adição
	-	Subtração
	*	Multiplicação
	/	Divisão
	%	Resto/Módulo
	++	Incremento
	--	Decremento
Atribuição composta	+=	Atribuição composta/Adição
	-=	Atribuição composta/Subtração
	*=	Atribuição composta/Multiplicação
	/=	Atribuição composta/Divisão
	%=	Atribuição composta/Módulo
Relacionais	==	Igualdade
	!=	Desigualdade
	<	Menor que
	>	Maior que
	<=	Menor ou igual
	>=	Maior ou igual




	Operador	Significado
Aritméticos	+	Adição
	-	Subtração
	*	Multiplicação
	/	Divisão
	%	Resto/Módulo
	//	Quociente
	**	Exponenciação
Atribuição composta	+=	Atribuição composta/Adição
	-=	Atribuição composta/Subtração
	*=	Atribuição composta/Multiplicação
	/=	Atribuição composta/Divisão
	%=	Atribuição composta/Módulo
	//=	Atribuição composta/Quociente
	**=	Atribuição composta/Exponenciação
Relacionais	==	Igualdade
	!=	Desigualdade
	<	Menor que
	>	Maior que
	<=	Menor ou igual
	>=	Maior ou igual

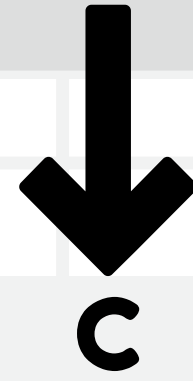


COMPARAÇÃO

Vamos comparar o C com o Python em diversos aspectos:

- Estrutura básica
 - Declaração de variáveis
 - Condicionais (if-else)
 - Loops (for, while)
 - Funções
 - Execução do software
- 

ESTRUTURA BÁSICA



C

```
#include <stdio.h>

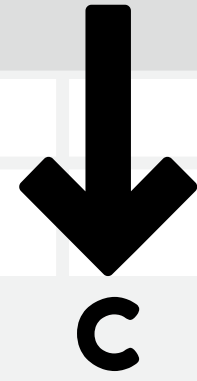
int main() {
    printf("Hello world");
    return 0;
}
```



PYTHON

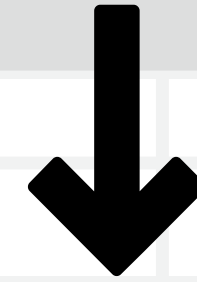
```
print("Hello world")
```

DECLARAÇÃO DE VARIÁVEIS



C

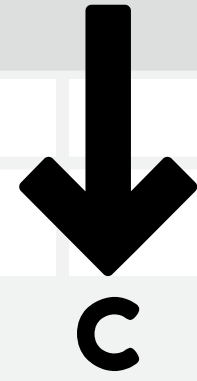
```
int main() {  
    int x = 0;  
    float y = 3.14;  
    char z[10] = "Olá!";  
    long w = 123456789000000;  
  
    return 0;  
}
```



PYTHON

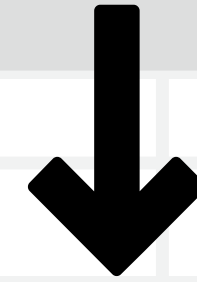
```
x = 0  
y = 3.14  
z = "Olá!"  
w = 123456789000000
```

CONDICIONAIS



C

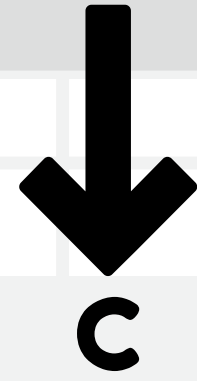
```
if (x > 10) {  
    printf("Olá");  
}  
else if (x < 10) {  
    printf("Mundo");  
}  
else {  
    printf("Tchau!");  
}
```



PYTHON

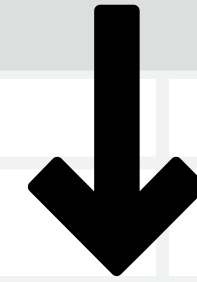
```
if x > 10:  
    print("Olá")  
elif x < 10:  
    print("Mundo")  
else:  
    print("Tchau!")
```

LOOPS - FOR



C

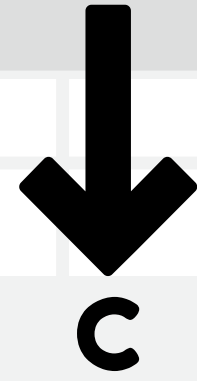
```
int i;  
for (i = 0; i < 10; i++) {  
    printf("Hello\n");  
}
```



PYTHON

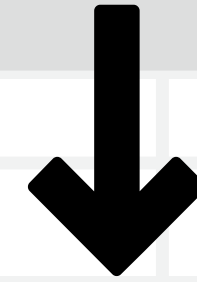
```
for i in range(10):  
    print("Hello")
```

LOOPS - WHILE



C

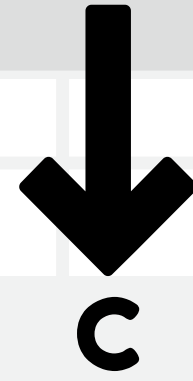
```
int i = 0;
while (i < 10) {
    printf("Hello");
    i += 1;
}
```



PYTHON

```
i = 0
while i < 10:
    print("Hello")
    i += 1
```

LOOPS - DO WHILE



C

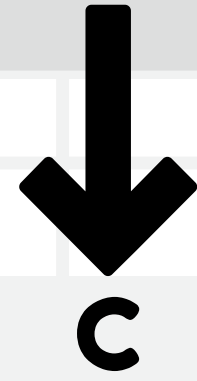
```
int i = 0;  
do {  
    printf("Hello");  
    i += 1;  
} while (i < 10);
```



PYTHON

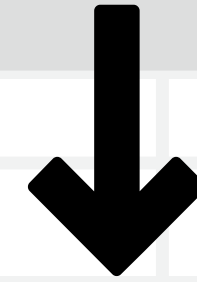
????

FUNÇÕES



C

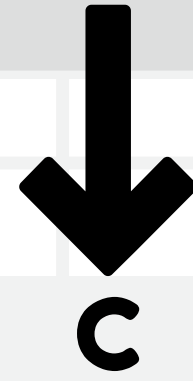
```
int sum(int a, int b) {  
    return a + b;  
}
```



PYTHON

```
def sum(a, b):  
    return a + b
```

EXECUÇÃO



C

```
# Compilar o código  
gcc main.c -o main.exe  
  
# Executar o programa  
./main.exe
```



PYTHON

```
# Interpretar e executar o código  
python main.py
```



EXEMPLOS PRÁTICOS



EXEMPLO 1

- **Vamos implementar passo a passo o Hello World**
 - Ver a estrutura básica do programa em C.
 - Mostrar na tela a mensagem.
 - Compilar
 - Executar o programa.



EXEMPLO 2

- **Cálculo do valor máximo/mínimos entre dois números.**
 - Estrutura básica
 - Definir e implementar as funções max e min.
 - Leitura de dados do usuário.
 - Compilar.
 - Executar o programa.



EXEMPLO 3

- **Cálculo da Tabuada**
 - Estrutura básica
 - Definir e implementar a função tabuada.
 - Leitura de dados do usuário.
 - Compilar.
 - Executar o programa.




EXERCÍCIOS!!





EXERCÍCIO 1

- **Cálcule a n-ésimo número da sequência de fibonacci.**
 - Leitura do usuário
 - Exemplos:
 - $\text{fibonacci}(0) = 0$
 - $\text{fibonacci}(4) = 3$
 - $\text{fibonacci}(5) = 5$
 - $\text{fibonacci}(9) = 34$
 - $\text{fibonacci}(11) = 89$
- 




EXERCÍCIO 2

- **Cálculo da média de 3 números.**
 - Leitura de cada número;
 - Cálculo da média
 - Apresentar para o usuário.



EXERCÍCIO 3

- O número 135 satisfaz a propriedade $135 = 1^1 + 3^2 + 5^3$
 - As potências na soma correspondem à posição do respectivo algarismo no número 135 (ex.: o 5 é o terceiro algarismo no número 135).
 - Encontre todos os números menores que 1000 que satisfazem a mesma propriedade.
- 



EXERCÍCIO 4

- **Soma dos dígitos de um número:**
 - Leitura do número;
 - Cálculo da soma;
 - Apresentar para o usuário.
- 135 -> 9
- 263 -> 11
- 11 -> 2



PROGRAMAÇÃO EM C

PARTE II





REVISÃO

- **Linguagens compiladas x interpretadas**
- **Comparação entre estruturas em Python e C:**
 - Declaração de variáveis.
 - Condicionais
 - Loops
 - Funções

SWITCH

- Quando há muitas opções no IF para uma mesma verificação, usa-se switch

```
int dia = 3;
switch (dia) {
    case 1:
        printf("Domingo");
        break;
    case 2:
        printf("Segunda-feira");
        break;
    case 3:
        printf("Terça-feira");
        break;
    case 4:
        printf("Quarta-feira");
        break;
    // Continua.
    default:
        printf("Dia inválido");
        break;
}
```

ARRAYS

- Um array é uma **coleção** de variáveis do **mesmo tipo** que são acessadas por um **índice**.

```
// Num A
int num_a[3];

num_a[0] = 1;
num_a[1] = 2;
num_a[2] = 3;

// Num B
int num_b[] = {1, 2, 3};
```

ARRAYS

- Cálculo do tamanho do array.

```
int numeros[] = {1, 2, 3};  
  
int tamanho = sizeof(numeros)/sizeof(numeros[0]);
```

EXEMPLO

- Encontrar o maior número em um array

```
#include <stdio.h>

int main() {
    int numeros[] = {3, 1, 4, 1, 5, 9, 2, 6, 5, 3};
    int tamanho = sizeof(numeros) / sizeof(numeros[0]);
    int maior = numeros[0];

    for (int i = 1; i < tamanho; i++) {
        if (numeros[i] > maior) {
            maior = numeros[i];
        }
    }

    printf("O maior elemento é: %d\n", maior);

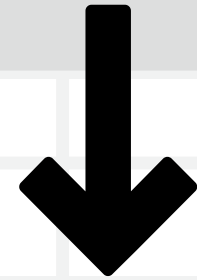
    return 0;
}
```

STRINGS

- Em C, strings são arrays de caracteres terminados por um caractere nulo ('\0').

```
char nome[20];  
  
printf("Digite o nome: ");  
gets(nome)
```

STRINGS



DECLARAÇÃO & INICIALIZAÇÃO

```
char str1[20];           // Declaração de array com tamanho 20;  
char str2[] = "Olá";    // Inicialização de um array de caracteres
```



LEITURA & ESCRITA

```
// Declaração  
char nome[50];  
  
// Leitura  
printf("Digite seu nome: ");  
scanf("%s", nome);  
  
// Escrita  
printf("Seu nome: %s\n", nome);
```

STRING.H



COMPRIMENTO

```
#include<string.h>

char nome[] = "João";
size_t comprimento = strlen(nome);
```

CÓPIA

```
char destino[20];
strcpy(destino, nome);
```

CONCATENAÇÃO

```
char saudacao[50] = "Olá , ";
strcat(saudacao, nome);
```

COMPARAÇÃO

```
int comparacao = strcmp(str1, str2);
```


EXEMPLO I

- Leitura e concatenação de strings

```
#include <stdio.h>
#include <string.h>

int main() {
    char nome[50];
    char saudacao[50] = "Olá, ";

    printf("Digite seu nome: ");
    scanf("%s", nome);

    strcat(saudacao, nome);
    printf("%s\n", saudacao);

    return 0;
}
```

EXEMPLO II

- Comparação de Strings

```
#include <stdio.h>
#include <string.h>

int main() {
    char str1[20];
    char str2[20];

    printf("Digite a primeira string: ");
    scanf("%s", str1);
    printf("Digite a segunda string: ");
    scanf("%s", str2);

    if (strcmp(str1, str2) == 0) {
        printf("As strings são iguais.\n");
    } else {
        printf("As strings são diferentes.\n");
    }

    return 0;
}
```

