Apostila de Programação em C

Sumário

1. Estrutura Básica de um Programa C

Um programa em C é composto por diretivas de pré-processamento, uma função principal `main` e declarações de variáveis e comandos.

Exemplo básico de um programa em C:

```c
#include <stdio.h>

int main() {
    printf("Olá, mundo!\n");
    return 0;
}
```

Neste exemplo, `#include <stdio.h>` inclui a biblioteca padrão de entrada e saída e `printf` é usado para imprimir na tela.

2. Tipos de Dados

C oferece uma variedade de tipos de dados, que são usados para declarar variáveis.

Tipos de dados primitivos:

- `int`: Números inteiros.
- `float`: Números de ponto flutuante.
- `double`: Números de ponto flutuante com precisão dupla.
- `char`: Caracteres individuais.

Exemplo:

```c
int idade = 25;
float altura = 1.75;
char inicial = 'A';
```

3. Operadores

Operadores são símbolos que dizem ao compilador para realizar operações específicas.

Operadores aritméticos:

- `+`, `-`, `*`, `/`, `%` (adição, subtração, multiplicação, divisão e módulo).

Exemplo:

```c
int soma = 10 + 5;
int produto = 4 * 7;
```

Operadores relacionais:

- `==`, `!=`, `>`, `<`, `>=`, `<=` (igual, diferente, maior que, menor que, maior ou igual, menor ou igual).

4. Estruturas de Controle

As estruturas de controle determinam o fluxo de execução do programa.

Exemplo de `if`:

```c
int a = 10;
if (a > 5) {
    printf("A é maior que 5\n");
} else {
    printf("A é menor ou igual a 5\n");
}
```

Exemplo de `for`:

```c
for (int i = 0; i < 10; i++) {
    printf("%d\n", i);
}
```

5. Funções

Funções são blocos de código que executam uma tarefa específica e podem retornar valores.

Exemplo:

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

int main() {
    int resultado = soma(3, 4);
    printf("Soma: %d\n", resultado);
    return 0;
}
```

6. Ponteiros

Ponteiros são variáveis que armazenam endereços de memória.

Exemplo:

```c
int a = 10;
int *p = &a;
printf("Valor de a: %d\n", *p);
```

Neste exemplo, `*p` é usado para acessar o valor armazenado no endereço ao qual `p` aponta.

7. Estruturas (Structs)

Structs permitem agrupar diferentes tipos de dados em uma única unidade.

Exemplo:

```c
struct Pessoa {
    char nome[50];
    int idade;
};

int main() {
    struct Pessoa pessoa1;
    pessoa1.idade = 30;
    printf("Idade: %d\n", pessoa1.idade);
    return 0;
}
```

8. Entrada e Saída

C oferece funções para entrada e saída de dados.

Exemplo de entrada com `scanf`:

```c
int a;
scanf("%d", &a);
printf("Valor lido: %d\n", a);
```

9. Compilação e Linkagem

O processo de compilação transforma o código fonte em um executável.

Comandos típicos:

```bash
gcc programa.c -o programa
```

10. Gerenciamento de Memória

C permite alocação dinâmica de memória usando funções como `malloc` e `free`.

Exemplo:

```c
int *p = (int*) malloc(sizeof(int));
*p = 10;
printf("Valor alocado: %d\n", *p);
free(p);
```