

GUIA DE REFERÊNCIA AULA 1 a 6

Estrutura básica

```
// Síntese
// Objetivo:
// Entrada :
// Saída  :
#include <stdio.h>

int main(void) {
    // Declarações

    // Instruções

    return 0;
}
```

Variável

Tipo	Máscara	Armazena
<i>string</i> char[]	%s	Palavras, frases...
char	%c	UMA letra, símbolo...
float	%f	Número com ponto flutuante (vírgula)
int	%d ou %i	Número sem ponto flutuante
int	-	0 (falso) ou ≠0 (verdadeiro)
void	-	nada

```
#include <stdio.h>
// Variável global
tipo nome;

int main(void) {
    // Variável local
    tipo nome;
    tipo nome=valor; // atribuição em tempo de declaração
}
```

Variável composta homogênea

```
int main(void) {
    tipo nome[tamanho]; // vetor

    nome[posição]; // acesso/uso de uma posição do vetor - 1ª posição = 0
}
```

String

```
int main(void) {
    char nome[tamanho]; // string = vetor de char
}
```

Constante

```
#include <stdio.h>
#define NOME valor
```

Entrada

```
// Instruções
scanf("mascara", &variável);
gets(variável string); // exclusivo para string
```

Saída

```
// Instruções
printf("Texto mascara texto", coisa(s));
puts(variável string); // exclusivo para string
// coisa = variável; valor; expressão; subprograma.
```

Atribuição

```
// Instruções
variável = coisa;
variável = variável ... = coisa; // atribuição em cascata
variável operador= coisa; // atribuição por operação
// coisa = variável; valor; expressão; subprograma.
```

Operadores aritméticos

Operador	Operação
+	Soma
-	Subtração
*	Multiplicação
/	Divisão
%	Resto da divisão
()	Precedência
++	Incremento (pré / pós)
--	Decremento (pré / pós)

Operadores relacionais

Operador	Operação
==	Igual
!=	Diferente
>	Maior
<	Menor
>=	Maior ou igual
<=	Menor ou igual

Operadores lógicos

Operador	Operação
&&	E
	OU
!	Não

Estrutura de seleção: if..else

```
if (condição)
    comando/bloco
else
    comando/bloco
```

Estrutura de seleção: switch case

```
switch (variável) {  
    case valor1:  
        comando(s)  
        break;  
    case valor2:  
        comando(s)  
        break;  
    default:  
        comando(s)  
}
```

Estrutura de repetição: for

```
for (var=inicio; condição; expressão)  
    comando/bloco
```

Estrutura de repetição: while

```
while (condição)  
    comando/bloco
```

Estrutura de repetição: do..while

```
do {  
    comando(s)  
} while (condição);
```

Função

```
#include <stdio.h>  
  
// Protótipo da função  
tipoRetorno nomeFuncao(tipoParâmetro(s));  
  
int main(void) {  
    nomeFuncao(parâmetro(s));  
    return 0;  
}  
  
// Função  
tipoRetorno nomeFuncao(parâmetro(s)) {  
    comando(s)  
    return coisa; // coisa = variável; valor; expressão; subprograma.  
}
```

Passagem de parâmetro

```
// Protótipo da função
tipoRetorno nomeFuncao(tipoValor, tipoReferencia*);

int main(void) {
    nomeFuncao(valor, &referencia);
    return 0;
}

// Função
tipoRetorno nomeFuncao(tipoValor nomeValor, tipoReferencia *nomeRef) {
    nomeValor = coisa; // exemplo de uso do parâmetro
    *nomeRef = coisa; // exemplo de uso do parâmetro
    return coisa;
}
// coisa = variável; valor; expressão; subprograma.
```

Funções para manipular string

Função	Operação
strlen(str1)	Retorna o tamanho da string
strcpy(str1, str2)	Copia a segunda string para a primeira
strcmp(str1, str2)	Retorna o valor da comparação das duas strings. Zero se forem iguais, maior que zero quando a primeira é maior que a segunda e menor que zero quando a primeira é menor que a segunda.
strcat(str1, str2)	Concatena a primeira string com a segunda.