Variáveis

Variáveis em C

Variáveis, como o próprio nome já diz, o conteúdo dela pode variar de acordo como você desejar. Elas armazenam valor e guardam esse valor (na memória do computador) para serem usadas na situação em que você solicita. Esta informação está associada com um lugar específico da memória (isso é feito pelo *compilador*). Cada variável tem um tipo associado.

Os tipos primitivos de dados existentes em C são

Tipo de Dado	Bits	Faixa de Valores
char	8	-128 a 127
bool	8	true ou false
int	32	-2.147.483.647 a 2.147.483.647
float	32	7 dígitos significativos
double	64	dígitos significativos

Definição de Variáveis em C

Para criar uma **variável**, usamos um tipo especial de instrução de declaração chamada definição (definição é diferente de declaração).
Definindo uma variável:

```
int x;
```

Ao definir uma variável, mesmo que não atribuirmos valor à ela, a mesma já possuirá uma instância após compilada(*runtime* = tempo de execução), ou seja, já haverá um espaço/endereço reservado para ela na memória RAM.

É possível declarar múltiplas variáveis de mesmo tipo, exemplos:

```
char x, y, z;
```

Para criar um nome de variável deve-se respeitar s seguintes regras:

Inicie suas variáveis somente com letras;
Não use: espaços, acentos, cedilhas, traços… somente letras e/ou underlines, exemplo:

```
int x; // certo
char letra_inicial; // certo
double _y; // certo
int 9x; // errado
int x, int y; // errado
int z, float r; // errado
char word; int z8; // certo
```

Declaração ou Inicialização de Variáveis em C

Quando você define uma variável deve-se declarar/inicializar um valor para ela, o compilador informa um Warning, ou seja, se você definir você precisa declarar um valor.

Observação: É importante você ficar atento algumas nomenclaturas do C se você veio de outra linguagem de programação, exemplos:

Definir = criar uma variável
Declarar ou *Inicializar* = atribuir um valor

Exemplo de declaração de acordo com os exemplos anteriores:

```
int x = 3;
char letra_inicial = 'a';
double _y = 69.03;
```

Outra coisa importante é você não pode declarar uma mesma variável duas vezes, isso nem gera Warning pelo compilador e sim um Error e seu código não compila. Mas você pode alterar o valor da variável.

Tipos de Inicialização de Variáveis em C

Existem 3 formas de você inicializar(declarar ou atribuir valor) uma variável, a mais utilizada principalmente para quem veio de outras linguagens de programação é a INICIALIZAÇÃO DE CÓPIA , mas ela não deve ser usada com frequência, pois você gasta espaço de memória e dependendo do tamanho do seu programa ela pode gerar muita **lentidão** no **tempo de execução.**

Em muitos lugares você obterá informação de variáveis citando: lvalue(lado esquerdo, left da declaração da variável) e rvalue(lado direito, inicialização, right).

Inicialização de Cópia: `int x = 10;`
Inicialização Direta: `int x = ( 10 ); // não tão diferente da cópia, quando o tipo for inteiro int`
Inicialização Uniforme: `int x = { 10 }; // use sempre que possível isso gera ganho de desempenho.`

É possível(e **preferível**) inicializar uma variável sempre que definirmos, caso não haja um valor pré-definido, inicialize de forma vazia, exemplos:

```
double z = {0};
double y = {36.9};
printf("O número z é: %i e y é %.2f\n", x, y);
```

Para as inicializações direta e uniforme devemos sempre usar o sinal de = para separar o lvalue Do rvalue, exemplo: `double z  = { 36.09 };` , isso continua sendo uniforme.

Tipos especificadores para printf

Format Specifier	Type
%c	Used to print a character
%d	Used to print the signed integer
%f	Used to print the float values
%i	Used to print the unsigned integer
%l	Used to print the long integer
%lf	Used to print the double values
%lu	Used to print the unsigned integer or unsigned long integer
%s	Used to print the string
%u	Used to print the unsigned integer

O tipo bool precisa incluir a stdbool.h, veremos na prática!