

MC-102 — Aula 04

Atribuições e Operações Aritméticas

Instituto de Computação – Unicamp

Segundo Semestre de 2007

Roteiro

- 1 Atribuição
- 2 Expressões aritméticas
- 3 Conversão de tipos

Atribuição

- Atribuir um valor de uma expressão a uma variável significa calcular o valor daquela expressão e copiar aquele valor para uma determinada variável.
- O operador de atribuição é o sinal de igual (=)

À esquerda do operador de atribuição deve existir somente o nome de uma **variável**.

=

À direita, deve haver uma **expressão** cujo valor será calculado e armazenado na variável

Expressão

- Já vimos que constantes, variáveis e endereços de variáveis são expressões.
- Uma expressão também pode ser um conjunto de operações aritméticas, lógicas ou relacionais utilizadas para fazer “cálculos” sobre os valores das variáveis.

Exemplo

$a + b$

Calcula a soma de a e b

Expressões

- $\langle \textit{expressao} \rangle + \langle \textit{expressao} \rangle$: Calcula a soma de duas expressões.
Ex: $a = a + b$;
- $\langle \textit{expressao} \rangle - \langle \textit{expressao} \rangle$: Calcula a subtração de duas expressões.
Ex: $a = a - b$;
- $\langle \textit{expressao} \rangle * \langle \textit{expressao} \rangle$: Calcula o produto de duas expressões.
Ex: $a = a * b$;

Expressões

- $< \text{expressao} > / < \text{expressao} >$: Calcula o quociente de duas expressões.
Ex: $a = a / b;$
- $< \text{expressao} > \% < \text{expressao} >$: Calcula o resto da divisão (inteira) de duas expressões.
Ex: $a = a \% b;$
- $- < \text{expressao} >$: Inverte o sinal da expressão.
Ex: $a = -b;$

Expressões

- As expressões aritméticas (e todas as expressões) operam sobre outras expressões.
- É possível compor expressões complexas como por exemplo:
 $a = b + 2 + c$

Qual o valor da expressão **$5 + 10 \% 3$** ?

E da expressão **$5 * 10 \% 3$** ?

Precedência

- Precedência é a ordem na qual os operadores serão calculados quando o programa for executado. Em C, os operadores são calculados na seguinte ordem:
 - $*$ e $/$, na ordem em que aparecerem na expressão.
 - $\%$
 - $+$ e $-$, na ordem em que aparecerem na expressão.

Alterando a precedência

- (*< expressao >*) também é uma expressão, que calcula o resultado da expressão dentro dela para só então permitir que as outras expressões executem. Deve ser utilizada quando a ordem da precedência não atende aos requisitos do programa.
Ex: $5 + 10 \% 3$ retorna 6, enquanto $(5 + 10) \% 3$ retorna 0
- Você pode usar quantos parênteses desejar dentro de uma expressão, contanto que utilize o mesmo número de parênteses para abrir e fechar expressões.

Incremento(++) e Decremento(--)

- Operadores de incremento e decremento tem duas funções: servem como uma expressão e incrementam ou decrementam o valor da variável ao qual estão associados em uma unidade. Ex: `c++` — incrementa o valor da variável `c` em uma unidade
- Dependendo da posição do operador de incremento e decremento, uma função é executada antes da outra.

Incremento(++) e Decremento(--)

- **Operador à esquerda da variável:** Primeiro a variável é incrementada, depois a expressão retorna o valor da expressão. Ex:

```
#include <stdio.h>
main () {
    int a = 10;
    printf ("%d", ++a);
}
```

Imprime 11

Incremento(++) e Decremento(--)

- **Operador à direita da variável:** Primeiro a expressão retorna o valor da variável, e depois a variável é incrementada. Ex:

```
#include <stdio.h>
int main (void) {
    int a = 10;
    printf ("%d", a++);
}
```

Imprime 10

Incremento(++) e Decremento(--)

- Em uma expressão, os operadores de incremento e decremento são sempre calculados primeiro (tem maior precedência)

```
#include <stdio.h>
int main (void) {
    int a = 10;
    printf ("%d", a * ++a);
}
```

Imprime 121

Atribuições simplificadas

Uma expressão da forma

$$a = a + b$$

onde ocorre uma atribuição a uma das variáveis da expressão pode ser simplificada como

$$a += b$$

Atribuições simplificadas

Comando	Exemplo	Corresponde a:
<code>+=</code>	<code>a += b</code>	<code>a = a + b;</code>
<code>-=</code>	<code>a -= b</code>	<code>a = a - b;</code>
<code>*=</code>	<code>a *= b;</code>	<code>a = a * b;</code>
<code>/=</code>	<code>a /= b;</code>	<code>a = a / b;</code>
<code>%=</code>	<code>a %= b;</code>	<code>a = a % b;</code>

Conversão de tipos

- É possível converter alguns tipos entre si.
- Existem duas formas de fazê-lo: implícita e explícita:
- Implícita
 - Capacidade (tamanho) do destino deve ser maior que a origem
Ex.: `int a; short b; a = b;`
 - Operações entre `int` e `float` **sempre** convertem para `float`
- Explícita:
 - Aplicável a variáveis e expressões
Ex. `a = (int)( (float)b / (float)c );`
 - Não modifica o tipo “real” da variável, só o valor de uma expressão.
Ex. `int a; (float)a=1.0; ← Errado`

Um uso da conversão de tipos

A operação de divisão (/) possui dois modos de operação de acordo com os seus argumentos: inteira ou de ponto flutuante.

- Se os dois argumentos forem inteiros, acontece a divisão inteira. A expressão $10 / 3$ tem como valor 3.
- Se **um** dos dois argumentos for de ponto flutuante, acontece a divisão de ponto flutuante. A expressão $1.5 / 3$ tem como valor 0.5.

Quando se deseja obter o valor de ponto flutuante de uma divisão (não-exata) de dois inteiros, basta converter um deles para ponto flutuante:

Exemplo

A expressão $10 / (\text{float}) 3$ tem como valor 3.33333333