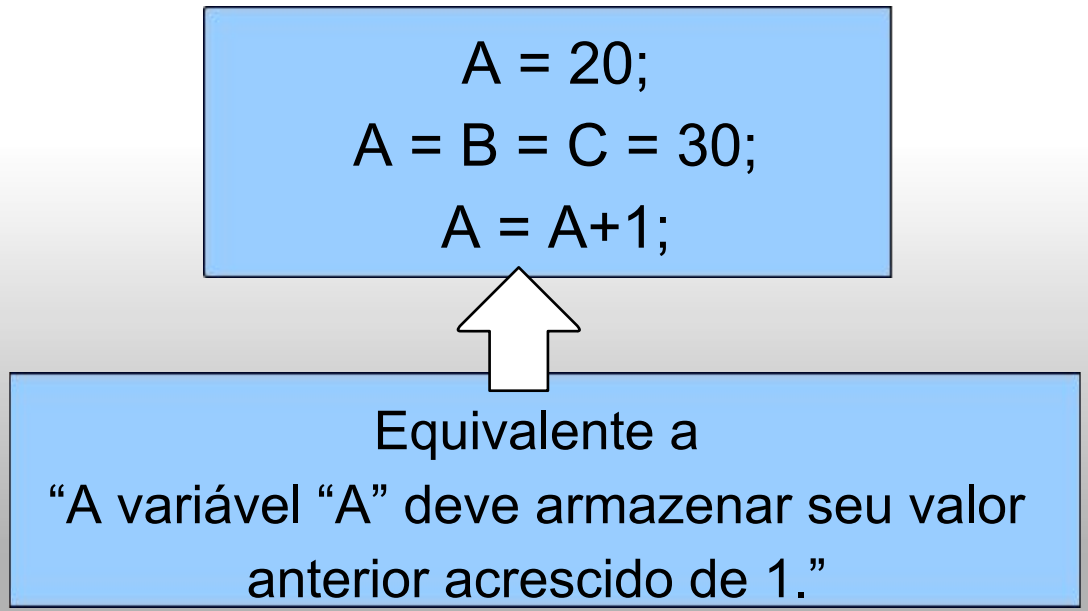
Operadores

- Atribuição
  - Sinal de igual (=)
  - Como visto anteriormente, definimos valores de variáveis utilizando este operador.

```
A = 20;
A = B = C = 30;
A = A+1;
```

Operadores

- Atribuição
  - Sinal de igual (=)
  - Como visto anteriormente, definimos valores de variáveis utilizando este operador.



Operadores

- Aritmética

+	Adição
–	Subtração
*	Multiplicação
/	Divisão
%	Resto da divisão

```
A = 3;
B = A+15;
C = B*A;
D = C/9;
E = D%5;
```

↓

A = 3
B = 18
C = 54
D = 6
E = 1

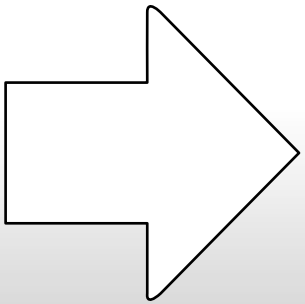
Operadores

- Aritmética

++	Adição simplificada
– –	Subtração simplificada

Servem como prefixos ou sufixos

A = 3;
A++;



A = 4

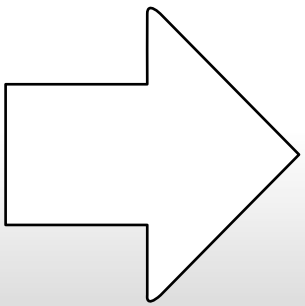
Operadores

- Aritmética

++	Adição simplificada
– –	Subtração simplificada

Servem como prefixos ou sufixos

A = 3;
++A;



A = 4

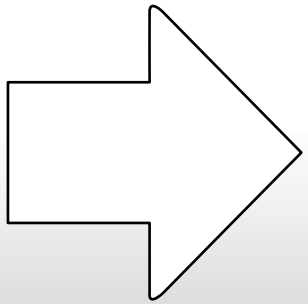
Operadores

- Aritmética

++	Adição simplificada
– –	Subtração simplificada

Servem como prefixos ou sufixos

A = 3;
B = A++;



A = 4
B = 3

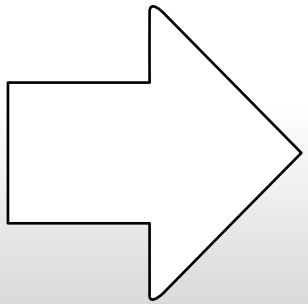
Operadores

- Aritmética

++	Adição simplificada
– –	Subtração simplificada

Servem como prefixos ou sufixos

A = 3;
B = ++A;



A = 4
B = 4

Operadores

- Relações e igualdades

==	Igual a
!=	Diferente de
>	Maior que
<	Menor que
>=	Maior ou igual que
<=	Menor ou igual que

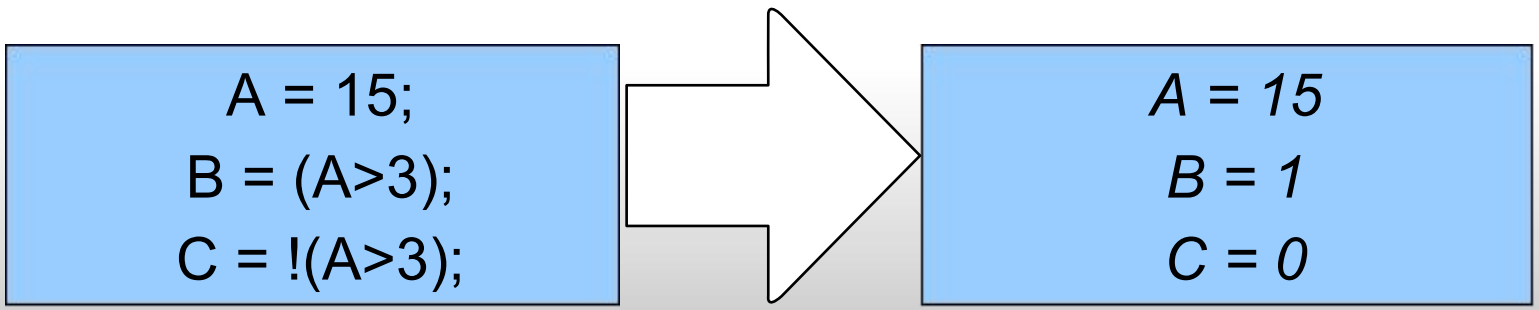
```
A = 3;
B = 15;
C = (A==B);
D = (A!=B);
E = (A>B);
F = (A<B);
G = (A>=B);
H = (A<=B);
```

C = 0
D = 1
E = 0
etc...

Operadores

- Operadores lógicos – negação (!)

A	!A
0	1
1	0

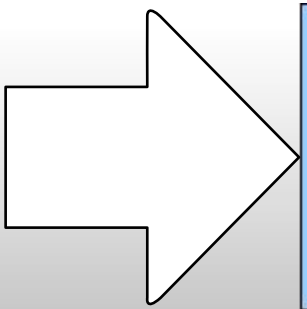


Operadores

- Operadores lógicos – E (&&)

A	B	A&&B
0	0	0
0	1	0
1	0	0
1	1	1

```
A = 15;
B = 3;
C = (A==B) && (B<0);
D = (A>B) && (B>0);
```



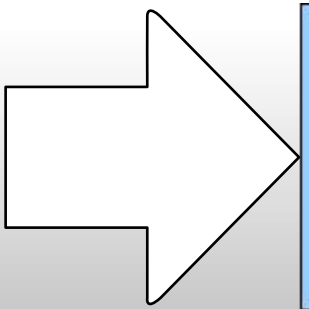
C = 0
D = 1

Operadores

- Operadores lógicos – OU (||)

A	B	A||B
0	0	0
0	1	1
1	0	1
1	1	1

```
A = 15;
B = 3;
C = (A==B) || (B<0);
D = (A<B) || (B>0);
```



C = 0
D = 1

Operadores

- Operadores lógicos *bit* a *bit*

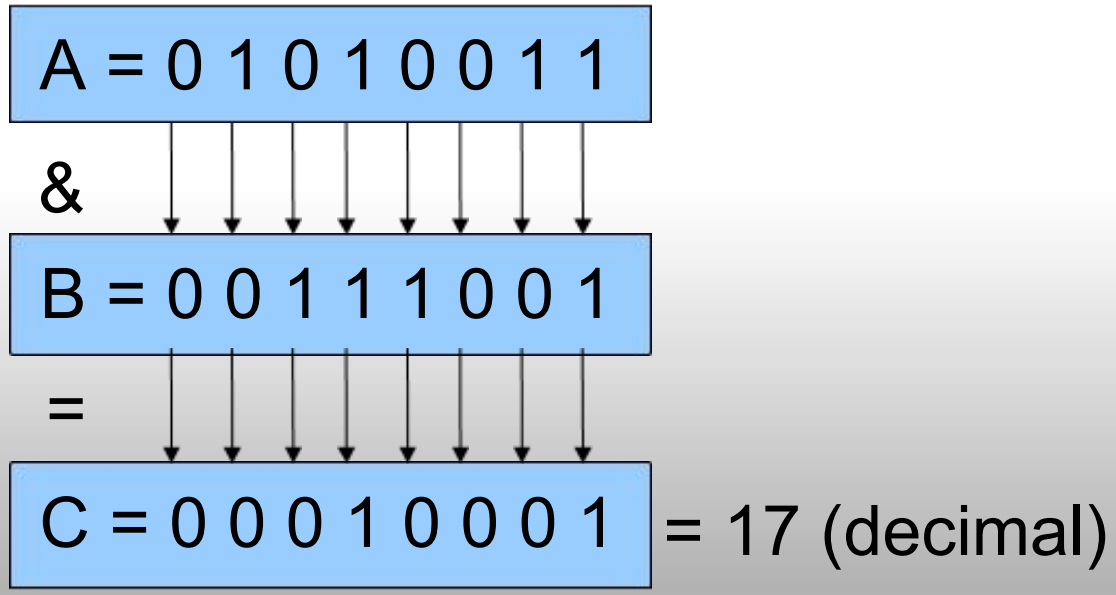
&	E
|	OU
^	XOR (OU EXCLUSIVO)
~	NEGAÇÃO
>>	Deslocamento de bits à direita
<<	Deslocamento de bits à esquerda

Operadores

- Operadores lógicos *bit* a *bit* – exemplo
  - char A = 8 (em base decimal)
  - A = 00001000 (em base binária)
  - A>>1 = 00000100 (binário) = 4 (decimal)
  - A<<3 = 00100000 (binário) = 32 (decimal)
  - A<<6 = 00000001 (binário) = 1 (decimal)

Operadores

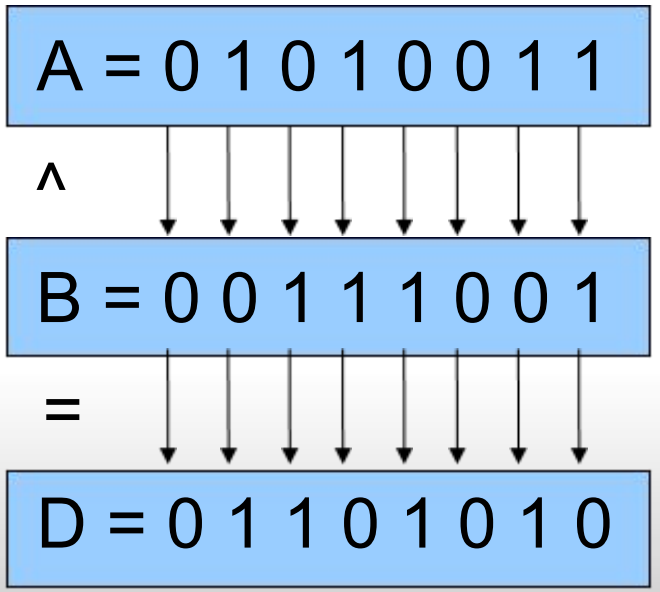
- Operadores lógicos *bit* a *bit* – exemplo
  - char A = 83 (decimal) = 01010011 (binário)
  - char B = 57 (decimal) = 00111001 (binário)
  - char C = A & B = (01010011) & (00111001)



Operadores

- Operadores lógicos *bit* a *bit* – exemplo
  - char D = A ^ B = (01010011) ^ (00111001)

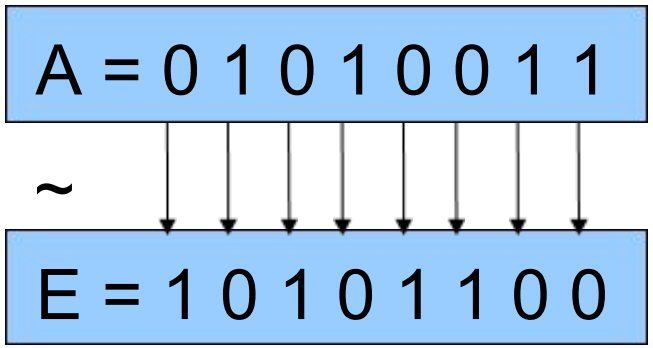
A	B	A^B
0	0	0
0	1	1
1	0	1
1	1	0



= 106 (decimal)

Operadores

- Operadores lógicos *bit* a *bit* – exemplo
  - char E = ~A = ~ (01010011)



= 172 (decimal)

Operadores

- Expressões abreviadas
  - A = 3;
  - A expressão “A = A+10;” pode ser escrita como “A+=10;”
  - A mesma técnica é válida para os operadores **–**, *****, ***/***, **%**, **>>**, **<<**, **&**, **|** e **^**
  - Por exemplo:

    f = 10; f %= 9;

    resulta em *f = 10%9 = 1*

Operadores

- Operador condicional
  - *CondiçãoX ? Resultado1 : Resultado*
  - Significa “Se a CondiçãoX for verdadeira, retorne o valor Resultado1. Caso contrário, retorne o valor Resultado2.”
  - Por exemplo:

```
a = 10;
b = (a>20) ? 2 : 5;
c = (a==10) ? a+5 : a+7
```

    resulta em *a = 10, b = 5* e *c = 15*

Operadores

- Modeladores
  - A existência de diferentes tipos de variáveis pode causar erros.
  - Por exemplo:

    unsigned char A;

    A = 256;
  - O número 256 em decimal é escrito 100000000 em binário. São 9 *bits*. Como a variável *A* só tem 8 *bits* (por ser *unsigned char*), ela receberá o valor 00000000 (os 8 *bits* menos significativos).

Operadores

- Modeladores
  - Caso a variável seja *signed*, isto poderá gerar ainda mais problemas.
  - Problemas semelhantes podem surgir quando misturamos diferentes variáveis.
  - Por exemplo, se temos três valores inteiros, e queremos saber a média aritmética dos três, é bem provável que esta terá um valor de ponto flutuante.

Operadores

- Modeladores
  - Para evitar erros na conta devido a conflitos na representação dos valores, utilizamos modeladores.

```
int a = 30, b = 10, c = 6;
float media;
media = (float)a;
media += (float)b;
media += (float)c;
media /= 3.0;
```