Linguagens de Programação

Expressões

Bruno Lopes

Instituto de Computação
uff

Propriedades desejáveis

Legibilidade: A leitura do programa é facilmente compreendida?

Redigibilidade: A implementação reflete o algoritmo? A redação é sucinta?

Confiabilidade: É fácil detectar "enganos" do programador?

Eficiência: Roda rápido?

Facilidade de Aprendizado: É enxuta?

Ortogonalidade: Conceitos podem ser combinados livremente?

Reusabilidade: É possível aproveitar partes em outros programas?

Modificabilidade: É fácil alterar programas?

Portabilidade: Roda da forma esperada em diferentes plataformas?

Instituto de Computação

Expressões

Definição

Uma frase do programa que, ao ser avaliada, produz como resultado um valor.

Elementos:

- Operadores
- Operandos
- Resultado

Instituto de Computação

Expressões

Definição

Uma frase do programa que, ao ser avaliada, produz como resultado um valor.

Elementos:

- Operadores
- Operandos
- Resultado

Instituto de Computação

Expressões

Definição

Uma frase do programa que, ao ser avaliada, produz como resultado um valor.

Elementos:

- Operadores
- Operandos
- Resultado

Instituto de Computação

Expressões

Classificação

- Simples
- Composta

Notação:

- Prefixada
- Infixada
- Posfixada

```
x>y?x:y
```

Instituto de Computação

Expressões

Classificação

- Simples
- Composta

Notação:

- Prefixada
- Infixada
- Posfixada

```
x>y?x:y
```

Instituto de Computação

Expressões

Classificação

- Simples
- Composta

Notação:

- Prefixada
- Infixada
- Posfixada

```
x>y?x:y
```

Instituto de Computação

Aridade dos operadores

```
(+ 1)
(+ 1 2)
(+ 1 2 4)
(+ 3 4 1 2)
...
```

Instituto de Computação

Aridade dos operadores

```
#define JUST3(a, b, c, ...) (a), (b), (c)
#define FUNC(...) func(JUST3(__VA_ARGS__, 0, 0))

FUNC(x) --> func((x), (0), (0))

FUNC(x,y) --> func((x), (y), (0))
```

Instituto de Computação

Aridade dos operadores

```
#define JUST3(a, b, c, ...) (a), (b), (c)
#define FUNC(...) func(JUST3(__VA_ARGS__, 0, 0))

FUNC(x) --> func((x), (0), (0))

FUNC(x,y) --> func((x), (y), (0))
```

Instituto de Computação

Origem dos operadores

- Pré-existentes
- Definidos pelo programados
- Composição de operadores

```
val par = fn (n: int) => (n mod 2 = 0)
val negação = fn (t:bool) => if t then false else true
val impar = negação o par
```

Instituto de Computação

Origem dos operadores

- Pré-existentes
- Definidos pelo programados
- Composição de operadores

```
val par = fn (n: int) => (n mod 2 = 0)
val negação = fn (t:bool) => if t then false else true
val impar = negação o par
```

Instituto de Computação

Tipos de operações

- Literais
- Constantes
- Variáveis

- Estáticas
- Dinâmicas

- Aritméticas
- Reais
- Booleanas

- Binárias
- Condicionais

- Referenciamento
- Derreferenciamento

- Categóricas
- Interativas

Instituto de Computação

Literais

Resultam no valor explícito do texto do programa.

Instituto de Computação

Agregação

```
void f(int i) {
     int a[] = {3 + 5, 2, 16/4};
     int b[] = {3*i, 4*i, 5*i};
     int c[] = {i + 2, 3 + 4, 2*i};
  }
```

Instituto de Computação

Construtores

```
Data d = new Data()
```

Instituto de Computação

Binárias

```
void main() {
     int j = 10;
     char c = 2;
     printf(\%d\n", ~c);        /* imprime -3 */
     printf(\%d\n", j & c);     /* imprime   2 */
     printf(\%d\n", j | c);     /* imprime 10 */
     printf(\%d\n", j ^ c);     /* imprime   8 */
     printf(\%d\n", j << c);    /* imprime 40 */
     printf(\%d\n", j >> c);    /* imprime   2 */
  }
```

Instituto de Computação

Efeitos colaterais

```
x = 3.2 * ++c;
x = 2;
y = 4;
z = (y = 2 * x + 1) + y;
```

Instituto de Computação

Funções e efeitos colaterais

- atualização de variável global
- passagem por referência
- impressão de valores ou gravação em arquivo

Instituto de Computação

Avaliando expressões compostas

Precedência de operadores

```
if a > 5 and b < 10 then
```

Ausência de precedência

Falta de redigibilidade!

Instituto de Computação

Avaliando expressões compostas

Precedência de operadores

```
if a > 5 and b < 10 then
```

Ausência de precedência

Falta de redigibilidade!

Instituto de Computação

```
x = a + b - c;
y = a < b < c;

x = **p;
if (!!x) y = 3;
y = !x++; // Como avaliar?
```

Instituto de Computação

```
i = 2;
a[i] = i++; // Atualiza a[2] ou a[3]?
```

Instituto de Computação

Curto-circuito

```
i = 0;
enquanto ((i < n)  e (a[i] <> valor procurado))
       incrementa i;
fim
```

```
i = 1;
if (b < 2*c || a[ i ++ ] > c) { a[i]++ };
```

Instituto de Computação

Pascal: expressões não são avaliadas em curto circuito, a menos que o compilador implemente por diretiva de compilação.

C, C++: && e ||, operadores booleanos, são avaliadas em curto circuito; & e |, operadores binários, não são avaliadas em curto circuito.

Java: & e | não são avaliadas em curto circuito; && e || são avaliadas em curto circuito.

Instituto de Computação