



2.4 Expresiones y Asignaciones

Expresiones aritméticas y lógicas,
sobrecarga de operadores,
conversiones de tipo, evaluación con
corto-circuito




Introducción

- Lenguajes imperativos se caracterizan por el uso dominante de expresiones y asignaciones
- El valor de las expresiones depende del orden de evaluación de operadores y operandos
- Ambigüedades en el orden de la evaluación puede conducir a diferentes resultados



Expresiones Aritméticas

- Orden de evaluación está principalmente definida por las reglas de precedencia y asociatividad
- Paréntesis fuerzan determinado orden
- Representación finita de números puede tener efectos no deseados en el orden de la asociatividad



Departamento de Informática
Universidad Técnica Federico Santa María

Lenguajes de Programación


Reglas de Precedencia de Operadores Aritméticos

| Fortran | Pascal | C |
|---------|----------------|----------------|
| ** | *, /, div, mod | Postfix ++, -- |
| *, / | +, - | Prefix ++, -- |
| +, - | | +, - (unario) |
| | | *, /, % |
| | | +, - (binario) |

Alta
↓
Baja

RMA/2003

II-7-
4



Departamento de Informática
Universidad Técnica Federico Santa María


Lenguajes de Programación

Reglas de Asociatividad

| | |
|-----------------|--|
| FORTTRAN | Izq: *, /, +, - Der: ** |
| Pascal | Todos por la izquierda |
| C | Izq: ++ y -- postfijo; *, /, %, + y - binario Der: ++ y -- prefijo; + y - unarios |
| C++ | Izq: *, /, %, + y - binario Der: ++, --, - y + unario |

RMA/2003

II-7-
5



Departamento de Informática
Universidad Técnica Federico Santa María

Lenguajes de Programación

Expresión Condicional

```


/* version N°1 */
int abs(int n)
{
    if (n>=0)
        return n;
    else return -n;
}

/* version N°2 */
int abs(int n)
{
    return (n>=0)? n: -n;
}

```

RMA/2003

II-7-
6



Departamento de Informática
Universidad Técnica Federico Santa María

Lenguajes de Programación

Efectos Laterales Funcionales

- Orden de evaluación de los operandos puede producir resultados diferentes si existen efectos laterales
- Ejemplo:

```

int a = 2;

int f1() {
    return a++;
}


int f2 (int i) {
    return (--a * i);
}

void main {
    printf("%i\n", f1()*f2(3));
}

```

RMA/2003

II-7-
7



Departamento de Informática
Universidad Técnica Federico Santa María


Lenguajes de Programación

¿Cómo evitar efectos laterales de funciones?

- Deshabilitar efectos laterales en la evaluación de funciones
 - Quita flexibilidad, por ejemplo habría que negar acceso a variables globales*
- Imponer un orden de evaluación a las funciones
 - Evita que el compilador puede realizar optimizaciones*
 - Enfoque seguido en Java (evaluación de izquierda a derecha).*

RMA/2003

II-7-
8



Departamento de Informática
Universidad Técnica Federico Santa María


Lenguajes de Programación

Sobrecarga de Operadores

- Un mismo operadores puede ser usado con diferentes tipos de operandos y para diferentes fines.
- Ejemplo:
 - +** para sumar enteros y reales (tb. strings)
 - &** en C es AND al bit y operador de dirección
- Ayuda a mejorar la lectura, pero puede que errores de escritura no sean detectados.

RMA/2003

II-7-
9



Departamento de Informática
Universidad Técnica Federico Santa María

Lenguajes de Programación

Sobrecarga de Operadores definidas por el Programador

- Algunos lenguajes con soporte de TDA permiten al programador sobrecargar símbolos de operadores (e.g. ADA, Fortran 90 y C++)

```

matrix A, B, C, D;


D = A + (B*C);

/* es mas conveniente, pero requiere de
sobrecarga de operadores; sino debiera
escribirse en un estilo como: */
MatrixAssign(D, MatrixAdd(A, MatrixMult(B, C)));

```

RMA/2003

II-7-
10



Departamento de Informática
Universidad Técnica Federico Santa María


Lenguajes de Programación

Conversiones de Tipo

- Clases: **extensión** o **estrangulamiento**
 - Extensión**: paso de entero a punto flotante o subrango de enteros a entero
 - Estrangulamiento**: paso de real a entero, de tipo base a subrango en general
- En general, **extensión** es más segura, pero puede tener algunos problemas (e.g. pérdida de precisión en la mantisa de entero a real)

RMA/2003

II-7-
11



Departamento de Informática
Universidad Técnica Federico Santa María

Lenguajes de Programación


Coerción en Expresiones

- Coerción** es cuando la conversión de tipo es implícitamente asumida por el compilador.
- Da más flexibilidad al uso de operadores, pero reduce la posibilidad de detectar errores y introduce código adicional.
- Ada y Modula-2 admiten pocos casos de coerción
- Java enteros cortos (byte, short y char) se convierten a int.
- Conversión explícita se denomina **casting**

RMA/2003

II-7-
12

4



Departamento de Informática
Universidad Técnica Federico Santa María


Lenguajes de Programación

Expresiones Relacionales

| Operación | Pascal | C | Ada | Fortran |
|-------------------|--------|----|-----|---------|
| Igual | = | == | = | .EQ. |
| No es igual | <> | != | /= | .NE. |
| Mayor que | > | > | > | .GT. |
| Menor que | < | < | < | .LT. |
| Mayor o igual que | >= | >= | >= | .GE. |
| Menor o igual que | <= | <= | <= | .LE. |

RMA/2003

II-7-
13



Departamento de Informática
Universidad Técnica Federico Santa María


Lenguajes de Programación

Operadores Booleanos

- Incluye: **AND**, **OR**, **NOT** y, a veces, **XOR**.
- La precedencia está generalmente definida de mayor a menor: NOT, OR y AND (excepto ADA, que todos tienen igual sin considerar NOT).
- Operadores aritméticos tienen mayor precedencia que relacionales y, generalmente, booleanos menor que relacionales (excepto Pascal).
- La asignación en C, C++ y Java es el operador que tiene la menor precedencia

RMA/2003

II-7-
14



Departamento de Informática
Universidad Técnica Federico Santa María

Lenguajes de Programación

Ejemplos: Operadores relacionales y lógicos

```

C:      b + 1 > b*2    /* aritmeticos primero */


C:      a > 0 || a < 5  /* relacional primero */

Pascal: a > 0 OR a < 5  {es ilegal}

```

RMA/2003

II-7-
15



Departamento de Informática
Universidad Técnica Federico Santa María

Lenguajes de Programación

Corto Circuito o Término Anticipado de Evaluación

C:

```
(13*a) * (b/13 -1)

(a >=0) || (b < 10)


while ( (c = getchar()) != EOF && c != '\n') {
    procesar línea;
}
```

PASCAL:

```
i := 1;
WHILE (i <= listlen) AND (list[i] <> key) DO
    i := i+1;
```

RMA/2003

II-7-
16



Departamento de Informática
Universidad Técnica Federico Santa María


Lenguajes de Programación

Corto Circuitos en los Lenguajes Imperativos

- **C, C++ y Java** definen corto circuito para: && y || (AND y OR)
- **Modula-2** tb. lo define para: AND y OR
- **Pascal** no lo especifica (algunas implementaciones los tienen, otras no); algo parecido sucede con **Fortran**
- **ADA** permite especificar explícitamente con los operadores: **and then** y **or else**

RMA/2003

II-7-
17



Departamento de Informática
Universidad Técnica Federico Santa María


Lenguajes de Programación

Sentencias de Asignación

- Permite cambiar dinámicamente el valor ligado a una variable
- *Fortran, Basic, PL/I, C, C++ y Java* usan **=**
- *Algol, Pascal y ADA* usan **:=**
- *C, C++ y Java* permiten incrustar una asignación en una expresión (actúa como cualquier operador binario)

RMA/2003

II-7-
18



Departamento de Informática
Universidad Técnica Federico Santa María


Lenguajes de Programación

Asignación Múltiple

- PL/I permite:
SUM, TOTAL = 0
- C, C++ y Java permiten:
SUM = TOTAL = 0

RMA/2003

II-7-
19



Departamento de Informática
Universidad Técnica Federico Santa María


Lenguajes de Programación

Asignación Condicional

- C, C++ y Java permiten:
(a > 0) ? cuenta1 : cuenta2 = 0;
- Equivale a:
if (a > 0)
 cuenta1 = 0;
else cuenta2 = 0;

RMA/2003

II-7-
20



Departamento de Informática
Universidad Técnica Federico Santa María

Lenguajes de Programación


Operadores Compuestos y Unarios de Asignación

- Sólo presentes en C, C++ y Java

| | |
|-------------------|---|
| sum += A[i]; | sum = ++contador; |
| equivale a: | equivale a: |
| sum = sum + A[i]; | contador = contador + 1; sum = contador; |

RMA/2003

II-7-
21



Departamento de Informática
Universidad Técnica Federico Santa María


Lenguajes de Programación

Asignación en Expresiones

- *C*, *C++* y *Java* permiten incrustar asignaciones en cualquier expresión
- Permite codificar en forma más compacta
- La desventaja de esta facilidad es que puede provocar efectos laterales, siendo fuente de error en la programación
- Es fácil equivocarse confundiendo `==` y `=`

RMA/2003

II-7-
22



Departamento de Informática
Universidad Técnica Federico Santa María

Lenguajes de Programación

Coerción en la Asignación

- Habilitar asignación en modo mixto requiere de reglas de conversión de tipo
- *C*, *C++* y *Fortran* permiten aplicación libre de coerción, aumentando riesgo de errores
- *Pascal* limita coerción, por ejemplo entero a real, pero no viceversa
- *Java* se diferencia de *C* y *C++* permitiendo sólo coerción en extensión.

RMA/2003

II-7-
23

8