Tema 4

Operadores y Expresiones

Escuela Técnica Superior de Ingeniería del Diseño

DSIC

DEPARTAMENTO DE SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

Contenidos

1. Conceptos Básicos.
2. Operadores Aritméticos.
3. Operadores de Relación, de Igualdad y Lógicos.
4. Operadores de Incremento y Decremento.
5. Operadores y Expresiones de Asignación.
6. Expresión Condicional.
7. Precedencia y Orden de Evaluación.
8. Conversiones de Tipos.

1. Conceptos Básicos

- ***Operador:*** **Símbolo que se asocia a una determinada operación básica que se realiza con los datos en algún punto del programa.**

- ***Expresión:*** **Combinación de variables, constantes, operadores, paréntesis y nombres de función, escritos en un determinado orden que tiene la propiedad de ser evaluada para obtener un valor.**

- **Tipos de expresiones:**
  - ***Aritméticas:*** Operandos y resultado de tipo numérico.
  - ***Relacionales:*** Operandos de cualquier tipo y resultado lógico (tipo entero significando verdadero, si distinto de cero; o falso, cuando es cero).
  - ***Lógicas:*** Los operandos y el resultado tienen valores lógicos (verdadero o falso).

2. Operadores Aritméticos

Operador	Acción	Tipos de Datos
+	Suma	char, int, float, double
-	Resta	char, int, float, double
*	Multiplicación	char, int, float, double
/	División entera o real	char, int, float, double
%	Resto de la división entera	char, int

- **Reglas de prioridad:**
  - **Los paréntesis permiten cambiar el orden de evaluación predeterminado. Si existen varios paréntesis anidados, las expresiones de los más internos se evalúan primero.**
  - **El orden de prioridad de los operadores aritméticos es:**
    - Primero: `/`, `*`, `%` (tienen igual prioridad entre ellos).
    - Después: `+`, `–` (tienen igual prioridad entre ellos).
  - **Dentro del mismo orden de prioridad se sigue la asociatividad de izquierda a derecha.**

Ejercicio

- Calcular el valor que tomaría la variable **a** tras ejecutar cada una de las líneas del siguiente código:

```
int a=1,b=5,c=2,d=3,e=6;
 a=a+b%d;
 a=d*b/2;
 a=d*(b/2);
 a=d+b*c-e;
 a=d+e*b/2;
 a=(d+e)*b/2;
```

3.1. Operadores de Relación y de Igualdad

- *Operadores de relación e igualdad:* **Se utilizan para realizar comparaciones, devolviendo verdadero o falso.**

Operador	Significado
>	mayor que
>=	mayor o igual que
<	menor que
<=	menor o igual que
==	igual a
!=	distinto de

- **Reglas de prioridad:**
  - **Tienen menos prioridad que los aritméticos.**
  - **El orden de prioridad de los operadores de relación es:**
    - Primero: **>**, **>=**, **<**, **<=** (tienen igual prioridad entre ellos).
    - Después: **==**, **!=** (tienen igual prioridad entre ellos).
  - **Dentro del mismo orden de prioridad se sigue la asociatividad de izquierda a derecha.**

Ejercicio

- Calcular el valor de las siguientes expresiones, sabiendo que i=2:

Expresiones de relación	Resultado
`i<2`	
`i<=2`	
`i*3>=(1+3)*i`	
`(2+i)!=(1+i)`	

3.2. Operadores Lógicos

- ***Operadores lógicos:*** **Combinan expresiones que devuelven valores lógicos, devolviendo valores lógicos también.**

Operador	Acción	
&&	And	"Y" lógico. La expresión es cierta si ambos operandos son ciertos.
||	Or	"O" lógico. La expresión es cierta si cualquier operando es cierto.
!	Not	"NO" lógico. Devuelve lo contrario del operando.

- **Reglas de prioridad:**
  - **Primero: !**
  - **Luego: &&**
  - **Después: ||**

Ejercicio

- Calcular el valor de las siguientes expresiones, sabiendo que i=2:

Operación	Resultado
`(i<2) || (i>10)`	
`(i>=1) && (i<=20)`	
`!(i>1)`	
`(i>0) && !(10<i)`	
`(i<1) || (i*2>10) && (i<15)`	

4. Operadores de Incremento y Decremento

Operador	Acción
++	Incrementa en 1 el valor de una variable
--	Decrementa en 1 el valor de una variable

- ***Operador prefijo:*** precede al operando (`--x`, `++x`).
- ***Operador sufijo:*** detrás del operando (`x--`, `x++`).
- Cuando estos operadores se usan en una instrucción aislada tienen el mismo efecto (`++x;` es lo mismo que `x++;` y `--x;` es lo mismo que `x--;`).
- Cuando se usan en una expresión:
  - ***Operador prefijo:*** incrementa o decrementa el valor de la variable antes de que se use su valor.

    Ejemplo: Si a vale 1, `b=++a;` deja a=2 y b=2.
  - ***Operador sufijo:*** usa el valor de la variable y después la incrementa o decrementa.

    Ejemplo: Si a vale 1, `b=a++;` deja a=2 y b=1.

5. Operadores y Expresiones de Asignación

- **Ya hemos usado múltiples veces el operador de asignación: la sentencia *variable = expresión* evalúa la expresión de la derecha y la asigna a la variable de la izquierda.**
- **Sin embargo el operador de asignación (=) se puede combinar con los operadores aritméticos (*op*).**
- ***De este modo:***
  - *variable op= expresión* equivale a *variable = variable op (expresión)*

5. Operadores y Expresiones de Asignación

Oper.	Significado
+=	Incrementar el valor de la variable en un valor determinado
-=	Decrementar el valor de la variable en un valor determinado
*=	Multiplicar el valor de la variable por un valor determinado
/=	Dividir el valor de la variable por un valor determinado
%=	Calcular el resto de la división de la variable por un valor determinado

Ejercicio

- Calcular el valor que van tomando las variables en el siguiente código:

```
int a=3, b=7;
a++;
b+=b*2;
b=++a;
b=a++;
b--;
a=b=2;
a=b-=2;
```

6. Expresión Condicional

expr1 ? expr2 : expr3

- **Se evalúa expr1:**
  - Si *expr1* es verdadero entonces se devuelve *expr2*
  - Si *expr1* es falso entonces se devuelve *expr3*

- ***Ejercicio:* Desarrollar código para asignar en la variable `a` el máximo de `b` y `c`.**

7. Precedencia y Orden de Evaluación

Máxima

! ++ --
* / %
+ -
> >= < <=
== !=
&&
||
?:
= += -= *= /= %=

Mínima

- Para cambiar esta precedencia se usan los paréntesis.
- Los operadores de igual precedencia se van evaluando de izda. a dcha., exceptuando los de asignación que van de dcha. a izda.

8. Conversiones de Tipos

- **Si en las operaciones aritméticas se combinan operandos de distintos tipos, se trabaja con el tipo del operando con más precisión y se devuelve ese tipo.**
- **Si a una variable entera se le asigna un resultado real, la parte decimal se pierde (se trunca). El caso contrario se asigna correctamente.**
- **Si a una variable entera se le asigna un valor que la sobrepasa en rango, el resultado no será correcto.**
- ***Conversiones de tipo forzadas*, *"operador de cast"*, o *"casting"*: Una variable o el resultado de una expresión se fuerzan a un tipo diferente del inicial.**
  - (tipo) expresión
  - Ejemplo: `(int)3.5f+7.5f`
    (es diferente de `(int)(3.5f+7.5f)`)