2. OPERADORES E INSTRUCCIONES

2.1 OPERADORES DE ASIGNACIÓN

En java, la asignación de valores a una variable o de una variable a otra se hace a través de operadores de asignación, el operador de asignación más usado es el operador **=.** Mediante este operador, se le asigna a la variable de la izquierda el valor de la derecha [6].

```
Variable = valor;
Variable1 = Variable2;
Variable1 = Variable2 = Variable3;
```

En el siguiente ejemplo se pueden observar diferentes maneras de hacer uso del operador de asignación.

```
public class OperadorAsignacion
{
      public static void main(String[]args)
            {
            int operador1 = 1000;    //A operador1 se le
                                     asigna el valor//1000
            int operador2 = 2000;    //A operador2 se le
                                     asigna el valor//2000
            int operador3 = 3000;    //A operador3 se le
                                     asigna el valor//3000
            System.out.println("operador1 = "+operador1);
            System.out.println("operador2 = "+operador2);
            System.out.println("operador3 = "+operador3);
            operador1 = operador2;   //A operador1 se le
                                     asigna el valor
                                     //contenido en operador2
            operador3 = operador1;   //A operador3 se le
                                     asigna el valor
                                     //contenido en operador1
            System.out.println("operador1 = "+operador1);
            System.out.println("operador2 = "+operador2);
            System.out.println("operador3 = "+operador3);
```

```
            operador1 = operador2 = operador3 = 5000; // A
                                    operador1,
                                    //operador2 y operador3
                                    se le asigna
                                    //el valor 5000
            System.out.println("operador1 = "+operador1);
            System.out.println("operador2 = "+operador2);
            System.out.println("operador3 = "+operador3);
            }
}
```

En java es posible combinar el operador de asignación **=** con otros símbolos correspondientes a operadores aritméticos, esto permite obtener una simplificación de instrucciones en operaciones acumulativas. La siguiente tabla muestra la equivalencia de estas operaciones:

Operador	Sintaxis de la instrucción	Instrucción equivalente
+=	operador1 += operador2;	operador1 = operador1 + operador2;
-=	operador1 -= operador2;	operador1 = operador1 - operador2;
*=	operador1 *= operador2;	operador1 = operador1 * operador2;
/=	operador1 /= operador2;	operador1 = operador1 / operador2;
%=	operador1 %= operador2;	operador1 = operador1 % operador2;

2.2 OPERADORES ARITMÉTICOS

Los operadores aritméticos tienen la característica de ser operadores binarios, es decir requieren siempre de dos operandos para poderse ejecutar. Estos operadores permiten realizar las operaciones aritméticas usuales así:

Operador	Sintaxis de la instrucción	Operación Aritmética
+	operador1 = operador2 + operador3;	Suma
-	operador1 = operador2 - operador3;	Resta
*	operador1 = operador2 * operador3;	Multiplicación
/	operador1 = operador2 / operador3;	División
%	operador1 = operador2 % operador3;	Residuo de la división

2.3 OPERADORES DE IGUALDAD

Permiten la obtención de un valor lógico (true/false) a partir de la evaluación de dos expresiones.

Caso 1
expIzquierda == expDerecha;
Se obtiene como resultado **true** si *expIzquierda* es igual a *expDerecha*; en caso contrario el resultado será **false**.

Caso 2
expIzquierda != expDerecha;
Se obtiene como resultado **true** si *expIzquieda* es diferente de *expDerecha*; en caso contrario el resultado será **false**.

2.4 OPERADORES RELACIONALES

Permiten evaluar expresiones relacionales obteniendo un valor lógico (true/false) al hacer una comparación entre dos valores de una expresión.

Menor que
expIzquierda < expDerecha;
Se obtiene como resultado **true** si *expIzquierda* es menor que *expDerecha*; en caso contrario el resultado será **false**.

Mayor que
expIzquierda > expDerecha;
Se obtiene como resultado **true** si *expIzquierda* es mayor que *expDerecha*; en caso contrario el resultado será **false**.

Menor o igual que
expIzquierda <= expDerecha;
Se obtiene como resultado **true** si *expIzquierda* es menor o igual que *expDerecha*; en caso contrario el resultado será **false**.

Mayor o igual que
expIzquierda >= expDerecha;
Se obtiene como resultado **true** si *expIzquierda* es mayor o igual que *expDerecha*; en caso contrario el resultado será **false**.

2.5 OPERADORES LÓGICOS

Estos operadores se utilizan para la construcción de expresiones lógicas.

Operador Y (Conjunción)
expIzquierda **&&** expDerecha;
Se obtiene como resultado true si expIzquierda es verdadera y expDerecha es verdadera; en otros casos el resultado de su evaluación será false.

Operador O (Disyunción)
expIzquierda **||**expDerecha;
Se obtiene como resultado false si las dos expresiones (expIzquierda, expDerecha) son falsas; en los demás casos, el resultado de la evaluación será true.

Operador NO (Negación)
¡expresión;
Se obtiene como resultado el valor lógico (true/false) contrario al que posee la expresión.

2.6 OPERADORES INCREMENTALES

El operador ++ permite incrementar en uno el valor de una variable, mientras que el operador -- permite disminuir en uno el valor de la variable. La tabla siguiente muestra las forma como estos pueden ser utilizados:

Operador	Uso	Descripción
++	operador++;	Utiliza la variable y luego la incrementa.
	++operador;	Incrementa la variable y luego la utiliza.
--	operador--;	Utiliza la variable y luego la decrementa.
	--operador;	Decrementa la variable y luego la utiliza.

2.7 OPERADOR instanceof

El operador ***instanceof*** permite saber si un objeto pertenece o no a una clase dada. Es un operador binario cuya sintaxis de uso es la siguiente:

NombreObjeto instanceof NombreClase

Su uso está orientado hacia la evaluación de expresiones (verifica que el primer operando, que debe ser un objeto, pertenezca al segundo operando, que debe ser una clase) y da como resultado un valor true/ false.

En el siguiente ejemplo se puede observar su uso:

```
public class OperadorInstanceof
{
      public static void main(String[]args)
      {
            Muebles Escritorio = new Muebles();
            System.out.println(Escritorio.cuotaCredito());
            if(Escritorio instanceof Muebles)
            {
                  System.out.println("Escritorio es
                  instancia de la clase Muebles");
            }
            else
            {
                  System.out.println("Escritorio NO es
                  instancia de la clase Muebles");
            }
      }
}

class Muebles
{
      double valor = 300000;
      boolean credito = true;
      double cuotaCredito()
      {
            double valorcuota;
            final int CUOTAS = 12;
```

```
            valorcuota = valor/CUOTAS;
            return valorcuota;
      }
}
```

EJERCICIOS PROPUESTOS PARA ESTA UNIDAD

1. Cuál es el resultado final de las variables a,b,c,d,e,f,g y h al ejecutar el siguiente programa?

```
public class Operadores
{
      public static void main(String[]args)
      {
            Registro Registro1 = new Registro();
            Registro1.cambiarValores();
            Registro1.imprimirValores();
      }
}
class Registro
{
      int a,b,c;
      double d,e,f=10;
      char g,h='i';
      void cambiarValores()
      {
            a=b=c=300;
            d=5.5;
            e=28*d;
            f*=e;
            g='b';
            h++;
      }
      void imprimirValores()
      {
            System.out.println("a: "+a+" b: "+b+" c: "+c+"
            d: "+d+" e: "+e+" f: "+f+" g: "+g+" h: "+h);
      }
}
```

2. Si del método main() del anterior programa quitamos la instrucción `Registro1.cambiarValores()`;, cuál será el valor final para las variables a,b,c,d,e,f,g y h?

3. La administración de un edificio de apartamentos liquida la cuota mensual de administración de los copropietarios de la siguiente forma: Apartamentos tipo 1 cuota = 30000, tipo 2 = 40000, tipo 3 = 50000 y tipo 4 = 70000. Además para los apartamentos que tienen garaje, dicha cuota tiene un adicional de 20000. Elabore un programa en java, donde además de la clase principal, se tengan dos clases adicionales, una para Apartamentos y otra para Mensualidades. El programa debe permitir la liquidación de la cuota mensual de administración para un apartamento determinado, de un tipo dado y que posea garaje. La salida debe estar dada en dos líneas. Ejemplo:

```
CUOTA DE AMINISTRACION
APARTAMENTO: 302 VALOR: 60000
```