

UNIVERSIDAD INTERNACIONAL DEL ECUADOR

ALUMNO: WILSON ARMIJOS

DOCENTE: ING. PABLO QUEZADA.

MATERIA: PROGRAMACION ORIENTADA A OBJETOS.

FACULTAD: INFORMATICA Y MULTIMEDIA

RESUMEN DEL LIBRO "COMO PROGRAMAR EN JAVA"

2. ESTRUCTURAS DE CONTROL

BUCLE FOR

Este bucle esta basado en una condición si dicha condición se cumple pasara a cumplir las otras instrucciones repetitivas caso contrario pasara a evaluar otra condición hasta que se cumpla, su sintaxis es.

```
for (i = 1; i <= 4; i++) {
```

```
}
```

BUCLE WHILE

Permite ejecutar una serie de instrucciones mientras se cumpla una condición de continuidad

```
while ( condición ) {
```

```
    instrucciones
```

```
}
```

CONDICIONAL IF

Permite ejecutar varias instrucciones en base a la evaluación de la condición.

```
if ( condición ) {
```

```
    instrucción
```

```
} else {
```

```
    instrucción
```

```
}
```

CONDICIONAL SWITCH

Permite definir un número ilimitado de ramas basadas en una misma condición, y su sintaxis es:

```
switch (expresion) {  
    caso valor1:  
        instrucciones;  
    break;  
}
```

3. MÉTODOS Y ESTRUCTURA DE DATOS

MÉTODOS

Permite agrupar una serie de instrucciones de manera que puedan ser ejecutadas desde diferentes puntos de la aplicación.

```
static double Hipotenusa (double cateto1, double cateto2) {  
    double Auxiliar;  
    Auxiliar = (Cateto1 * Cateto1) + (Cateto2 * Cateto2);  
    return Math.sqrt (Auxiliar);  
}
```

STRING

También llamados literales o cadenas de caracteres, nos permiten declarar, definir y operar con palabras y frases, su sintaxis es:

```
string InstanciodeString = "Palabra o frase";
```

MATRICES

Permite declarar con un solo nombre un conjunto de variables ordenadas de un mismo tipo, su sintaxis es:

```
tipo [] Variable = new tipo [Elementos];
```

PAQUETES Y ATRIBUTOS DE ACCESO

Los paquetes sirven para agrupar clases relacionadas, de esta manera cada paquete contiene un conjunto de clases.

Las clases existentes dentro de un paquete deben tener nombre distintos para poder diferenciarse entre sí.

DEFINICIÓN DE PAQUETES

Definir el paquete al que pertenece una clase es muy sencillo, basta con incluir la sentencia package.

```
package Terminal:  
public class Teléfono {  
  
}
```

Las clases definidas como public son accesibles desde fuera del paquete, las que no presentan este atributo son accesibles desde dentro del paquete.

UTILIZACIÓN DE LAS CLASES DE UN PAQUETE

Cuando se necesita hacer uso de todas o algunas de las clases de un paquete, debemos indicarlo de una manera explícita para que el compilador sepa donde se encuentran las clases. Para ello se utiliza la sentencia import:

```
import Nombre_Paquete.Nombre_Clase
```

ATRIBUTOS DE ACCESO A LOS MIEMBROS DE UNA CLASE

Existen cuatro posibles atributos de acceso, que listamos a continuación según el nivel de restricción que imponen.

- | | | | |
|-------------------|---------------------|-------------|-----------|
| • Privado | private | • Protegido | protected |
| • Sin Especificar | int PSinEspecificar | • Público | public |

4. PROGRAMACIÓN ORIENTADA A OBJETOS

USANDO CLASES

DEFINICIÓN DE CLASES

Son objetos que utiliza JAVA para soportar la programación orientada a objetos, constituyen la estructura básica sobre la cual se desarrollan las aplicaciones. Una clase permite definir propiedades y métodos relacionados entre sí.

```
public class Semáforo {  
    private String EstadoSemáforo = "Rojo";
```

SOBRECARGA DE MÉTODOS Y CONSTRUCTORES

SOBRECARGA DE MÉTODOS

Es un mecanismo muy útil que permite definir en una clase varios métodos con el mismo nombre.

```
Dimensiones (double Ancho, double Alto, double Profundo, String Medida);  
Dimensiones (String Medida, double Ancho, double Alto, double Profundo).
```

CONSTRUCTORES

Son objetos que sirven para iniciar los objetos al definirse las instancias de los mismos, los constructores permiten a la vez crear instancias y establecer el estado inicial de cada objeto.

CLASES UTILIZADAS COMO PARÁMETROS

Utilizar una clase como parámetro en un método hace posible que el método utilice toda la potencia de los objetos, independizando las acciones realizadas de los objetos sobre las que las realice.

PROPIEDADES Y MÉTODOS DE CLASE Y DE INSTANCIA

En una clase, las propiedades y los métodos pueden definirse como:

- De instancia

- De clase.

PROPIEDADES DE INSTANCIA

Se caracterizan porque cuando se define una instancia de la clase, se crean nuevas variables que contendrán los valores de dichas propiedades en la instancia creada.

PROPIEDADES DE CLASE

Una propiedad de clase se declara con el atributo `static`:

```
class SenalloEstatico {  
    static public int PropiedaddeClase;  
}
```

MÉTODOS DE INSTANCIA

Solo pueden ser utilizados a través de una instancia de la clase.

Cualquier intento de acceder a un método de instancia a través del nombre de la clase nos dará un error de compilación.

MÉTODOS DE CLASE

Un método estático puede ser utilizado sin necesidad de definir previamente instancias de la clase que contiene el método.

5. INSTRUCCIONES DE CONTROL

PARTE 2

1. FUNDAMENTOS DE LA REPETICIÓN CONTROLADA

POR CONTADOR

Esta sección utiliza la instrucción de repetición `while`, para formalizar los elementos requeridos y llevar a cabo la repetición controlada por un contador. Este tipo de repetición requiere:

- * Una variable de control.
- * El valor inicial de la variable de control.
- * El incremento con el que se modifica la variable de control cada vez que pasa por el ciclo.
- * La condición de continuación del ciclo, que determina si el ciclo debe continuar o no.

INSTRUCCIÓN DE REPETICIÓN DO-WHILE

Es similar a la instrucción `while`, ya que el programa evalúa la condición, si la condición es verdadera ejecuta las instrucciones y de ser falsa la condición no se ejecuta.

INSTRUCCIÓN DE REPETICIÓN FOR

Java cuenta con la instrucción de repetición `for` que especifica los detalles de la repetición controlada por contador en una sola línea.

```
for (int contador = 1; contador <= 10; contador++) {  
    instrucciones  
}
```


INSTRUCCIÓN DE SELECCIÓN MULTIPLE SWITCH

Java cuenta con la instrucción switch de selección múltiple para realizar distintas acciones, con base en los posibles valores de una variable o expresión entera.

```
switch (color){  
    case Red;  
        System.out.println("Red");  
        break;  
    case Blue;  
        System.out.println("Blue");  
        break;  
    case
```

INSTRUCCIONES BREAK Y CONTINUE

Las instrucciones break y continue etiquetadas, se usa en los casos que es conveniente alterar el flujo de control en las instrucciones de control anidados.

OPERADORES LÓGICOS

Los operadores lógicos son:

- * AND Condicional (&&)
- * OR Condicional (||)
- * AND Lógico Booleano (&)
- * OR Inclusivo Lógico Booleano (|)
- * OR Exclusivo Lógico Booleano (^)
- * NOT LÓGICO (!)

OPERADOR AND (&) CONDICIONAL

Este tipo de operador se utiliza para evaluar 2 condiciones.

```
if (genero == femenino && edad >= 65)
    ++mujeresMayores;
```

OPERADOR OR CONDICIONAL (||)

Se usa para evaluar dos condiciones que pueden ser verdaderas o falsas.

```
if ((promedioSemestre >= 90) || (examenFinal >= 90))
    System.out.println("La calificación del estudiante es A");
```

OPERADORES AND LÓGICO BOOLEANO (&) Y OR INCLUSIVO LÓGICO BOOLEANO (|)

Este tipo de operadores funcionan de la misma manera que los operadores AND Condicional y OR Condicional, con una excepción los operadores lógicos booleanos siempre evalúan ambos operandos

```
(genero == 1) & (edad >= 65)
```

OR EXCLUSIVO LÓGICO BOOLEANO (^)

Este tipo de operador es complejo ya que si uno de sus operandos es verdadero su resultado es verdadero caso contrario si sus dos operandos son verdaderas o falsas toda la condición es falsa.

OPERADOR LÓGICO DE NEGACIÓN (!)

Este operador invierte el significado de la condición.

expresión	!expresión
false	true
true	false