Programación orientada a objetos

Capítulo 2

Comprender las definiciones de clase

Tema 2. Comprender las definiciones de clases. Semana 2	
1- El concepto de clase 2- Campos, constructores y métodos 3- Paso de datos mediante parámetros 4- Asignacion de valores 5- Tipos de métodos: a. Métodos de acceso get() b. Métodos de modificación set() c. Método main() 6- Impresión desde métodos 7- Estructuras de control. La sentencia condicional if. 8- Campos, parámetros y variables locales	1- Estudiar el capítulo 2 y leer los apéndices B, C y D del libro base para la Unidad Didáctica I 2- Realizar los ejercicios en el entorno BlueJ sugeridos en el libro base

Orden

```
public class NombreDeClase
{
Campos
Constructores
Métodos
}
```

Ver código MaquinaDeBoletos

Campos

Código 2.3
Los campos de la clase `MaquinaDeBoletos`

```
public class MaquinaDeBoletos
{
    private int precio;
    private int saldo;
    private int total;

    Se omitieron el constructor y los métodos.

}
```

Constructores

```
public class MaquinaDeBoletos
{
Se omitieron los campos
          /**
           * Crea una máquina que vende boletos de un
             determinado precio.
           * Observe que el precio debe ser mayor que cero
             y que no hay
           * controles que aseguren esto.
           */
public MaquinaDeBoletos (int precioDelBoleto)
{
precio = precioDelBoleto;
saldo = 0;
total = 0;
}
   Se omitieron los métodos
}
```

Ejercicio 2.20 *Desafío* ¿Cuál es el error en la siguiente versión del constructor de la clase `MaquinaDeBoletos`?

```
public MaquinaDeBoletos(int precioDelBoleto)
{
   int precio = precioDelBoleto;
   saldo = 0;
   total = 0;
}
```

Método

```
public class MaquinaDeBoletos
{
    Se omitieron los campos.
    Se omitieron los constructores.
/**
* Devuelve el precio de un boleto.
*/
public int obtenerPrecio()
{
    return precio;
}
    Se omitieron los restantes métodos.
}
```

Ejercicio 2.29 ¿Qué elementos del encabezado de `ponerPrecio` nos indican que es un método y no un constructor?

```
public void ponerPrecio (int precioDelBoleto)
```

Imprimir

```
/**
* Imprime un boleto y pone el saldo actual en cero
*/
public void imprimirBoleto()
{
// Simula la impresión de un boleto.
System.out.println("##################");
System.out.println("# Línea BlueJ");
System.out.println("# Boleto");
System.out.println("# " + precio + " cvos.");
System.out.println("##################");
System.out.println();
// Actualiza el total recaudado con el saldo.
total = total + saldo;
// Limpia el saldo.
saldo = 0;
}
```

Ejercicio 2.42 Provea a la clase de dos constructores: uno debe tomar un solo parámetro que especifique el precio del boleto, y el otro no debe tener parámetros y debe establecer el precio como un valor fijo por defecto, el que usted elija. Pruebe su implementación creando máquinas mediante los dos constructores diferentes.

- Hacer ejercicio 2.42

Máquina mejorada

- Ver código maquina mejorada

Un ejemplo más avanzado de sentencia condicional

El método `imprimirBoleto` contiene un ejemplo más avanzado de una sentencia condicional. Aquí está su esquema:

```
if(saldo >= precio) {
Se omitieron los detalles de impresión.
// Actualiza el total recaudado con el precio.
total = total + precio;
// Decrementa el saldo en el valor del precio.
saldo = saldo - precio;
}
        else {
            System.out.println("Debe ingresar como mínimo: "
                                        + (precio - saldo) + "
cvos más.");
        }
```

Variables locales

El método `reintegrarSaldo` contiene tres sentencias y una declaración. La declaración ilustra una nueva clase de variable:

```
public int reintegrarSaldo()
{
int cantidadAReintegrar;
cantidadAReintegrar = saldo;
saldo = 0;
return cantidadAReintegrar;
}
```

Campos, parámetros y variables locales

Con la introducción de `cantidadAReintegrar` en el método `reintegrarSaldo` hemos visto tres tipos diferentes de variables: campos, parámetros formales y variables locales. Es importante comprender las similitudes y diferencias entre estos tipos de variables. A continuación hacemos un resumen de sus características:

- Las tres clases de variables pueden almacenar un valor acorde a su definición de tipo de dato. Por ejemplo, una variable definida como de tipo `int` permite almacenar un valor entero.
- Los campos se definen fuera de los constructores y de los métodos.
- Los campos se usan para almacenar datos que persisten durante la vida del objeto, de esta manera mantienen el estado actual de un objeto. Tienen un tiempo de vida que finaliza cuando termina el objeto.
- El alcance de los campos es la clase: la accesibilidad de los campos se extiende a toda la clase y por este motivo pueden usarse dentro de cualquier constructor o método de clase en la que estén definidos.
- Como son definidos como privados (`private`), los campos no pueden ser accedidos desde el exterior de la clase.

- Los parámetros formales y las variables locales persisten solamente en el lapso durante el cual se ejecuta un constructor o un método. Su tiempo de vida es tan largo como una llamada, por lo que sus valores se pierden entre llamadas. Por este motivo, actúan como lugares de almacenamiento temporales antes que permanentes.
- Los parámetros formales se definen en el encabezado de un constructor o de un método. Reciben sus valores desde el exterior, se inicializan con los valores de los parámetros actuales que forman parte de la llamada al constructor o al método.
- Los parámetros formales tienen un alcance limitado a su definición de constructor o de método.
- Las variables locales se declaran dentro del cuerpo de un constructor o de un método. Pueden ser inicializadas y usadas solamente dentro del cuerpo de las definiciones de constructores o métodos. Las variables locales deben ser inicializadas antes de ser usadas en una expresión, no tienen un valor por defecto.
- Las variables locales tienen un alcance limitado al bloque en el que son declaradas. No son accesibles desde ningún lugar fuera de ese bloque.

Resumen de conceptos

- **campo** Los campos almacenan datos para que un objeto los use. Los campos se conocen como variables de instancia.
- **comentario** Los comentarios se insertan dentro del código de una clase para brindar explicaciones a los lectores humanos. No tienen efecto sobre la funcionalidad de la clase.
- **constructor** Los constructores permiten que cada objeto sea preparado adecuadamente cuando es creado.
- **alcance** El alcance de una variable define la sección de código desde donde la variable puede ser accedida.
- **tiempo de vida** El tiempo de vida de una variable describe el tiempo durante el cual la variable continúa existiendo antes de ser destruida.
- **asignación** Las sentencias de asignación almacenan el valor representado del lado derecho de la sentencia en la variable nombrada en el lado izquierdo.

Resumen de conceptos

- **método** Los métodos están compuestos por dos partes: un encabezado y un cuerpo.
- **método de acceso** Los métodos de acceso devuelven información sobre el estado de un objeto.
- **métodos de modificación** Los métodos de modificación cambian el estado de un objeto.
- **println** El método `System.out.println(...)` imprime su parámetro en la terminal de texto.
- **condicional** Una sentencia condicional realiza una de dos acciones posibles basándose en el resultado de una prueba.
- **expresión booleana** Las expresiones booleanas tienen sólo dos valores posibles: verdadero y falso. Se las encuentra comúnmente controlando la elección entre los dos caminos de una sentencia condicional.
- **variable local** Las variables locales son variables que se declaran y usan dentro de un único método. Su alcance y tiempo de vida están limitados por el método.