Programación Orientada a Objetos

Resolución de Problemas Por paso por Valor

Todos los tipos de datos primitivos (int, long, float, boolean) se pasan en los métodos por valor. Sus valores se copian en nuevas posiciones, que se pasan al método; como consecuencia de esto, si un argumento se cambia dentro de un método, no se cambiará en el programa llamador original.

Ejemplo 1:

Calcular e imprimir el precio de venta de un artículo. Se tienen los datos descripción del artículo y costo de producción. El precio de ventas se calcula añadiendo al costo el 12% como utilidad y el 15% de impuesto.

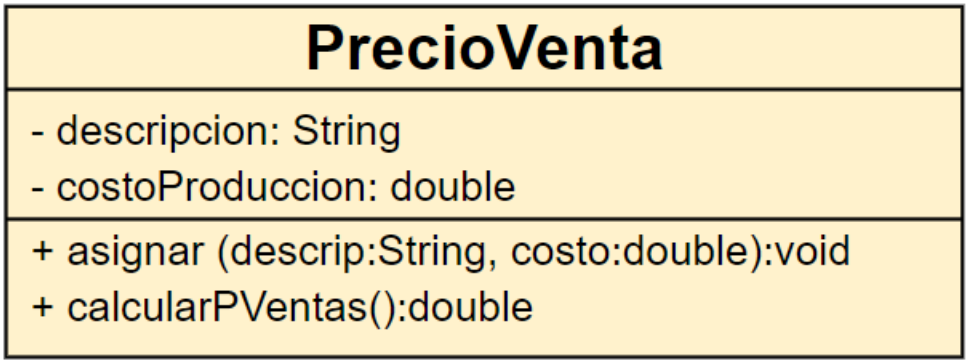


Diagrama de clase Ejemplo 1

Clase secundaria

```
package Ejemplo1; // Importacion de Paquetes

public class PrecioVenta { //Nombre de la clase
    1 usage
    private String descripcion;//Declaración de atributos
    3 usages
    private double costoProduccion;//Declaración de atributos

    1 usage
    public void asignar(String descrip, double costo){//método asignar  redibe dos parametros
        descripcion=descrip;//Asignacion
        costoProduccion=costo;//Asignacion
    }
    1 usage
    public double calcularPventas(){//método para calcular el costos de venta
        double resp;//Declaracion de una variable local al métodos
        resp=costoProduccion*0.12+costoProduccion;//Asignacion
        resp=resp+resp*0.15;//Asignacion
        return resp;// retorna el resultado
    }
}
```

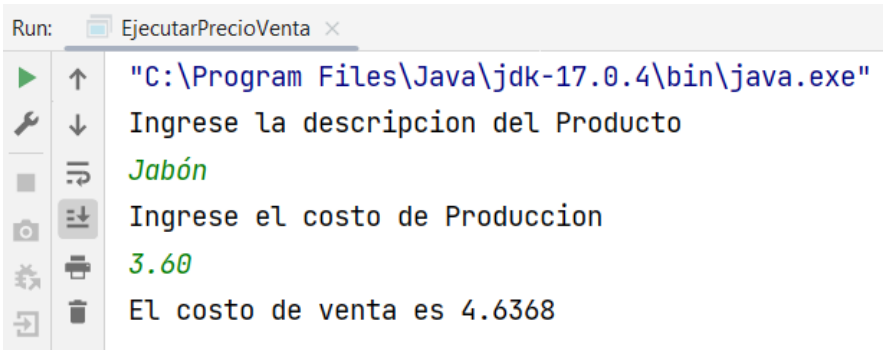
Clase Principal

```java
//Importacion de paquetes y libreias
package Ejemplo1;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class Principal {
    //Método principal de Java. Es el identificador que la JVM busca como punto de partida del programa java
    public static void main(String[] args) throws IOException {
        //Declaracion de Variables
        String descripcion;
        double costo;
        //Instancia a la clase  BufferedReader se crea el objeto br
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        //Instancia a la clase  PrecioVenta se crea el objeto objPrecioVenta
        PrecioVenta objPrecioVenta = new PrecioVenta();
        //Impresion [Solicitud de datos de entrada al usuario]
        System.out.println("Ingrese la descripcion ");
        //Asignacion de lectura realizada mediante el objeto al metodo reaLine
        descripcion= br.readLine();
        //Impresion [Solicitud de datos de entrada al usuario]
        System.out.println("Ingrese el costo de Produccion ");
        /*Asignacion de lectura realizada mediante el objeto al metodo reaLine Como la lectura es un
          String se realiza una conversion de datos
          Se utilza Double.Double.parseDouble(br.readLine()) , para realizar la conversiém
         */
        costo=Double.parseDouble(br.readLine());
        //Mediante el objeto objPrecioVenta se invoca al metodo asignar() se envia los valores
        objPrecioVenta.asignar(descripcion,costo);
        //Mediante el objeto objPrecioVenta se invoca al metodo calcularPventas(), Se imprime el resultado retornado
        System.out.println("El costo de venta "+objPrecioVenta.calcularPventas());
    }
}
```

Ejecución



Ejemplo 2:

Calcular e imprimir el precio de venta de un artículo. Se tienen los datos descripción del artículo y costo de producción. El precio de ventas se calcula añadiendo al costo el 12% como utilidad y el 15% de impuesto. Resolver mediante **getters** y **setters.**

Los getters (de la palabra inglés get - obtener) indica que podemos tomar algún valor de un atributo y los **setters** (de la palabra ingles set-poner/fijar) podemos guardar algún valor sobre un atributo.

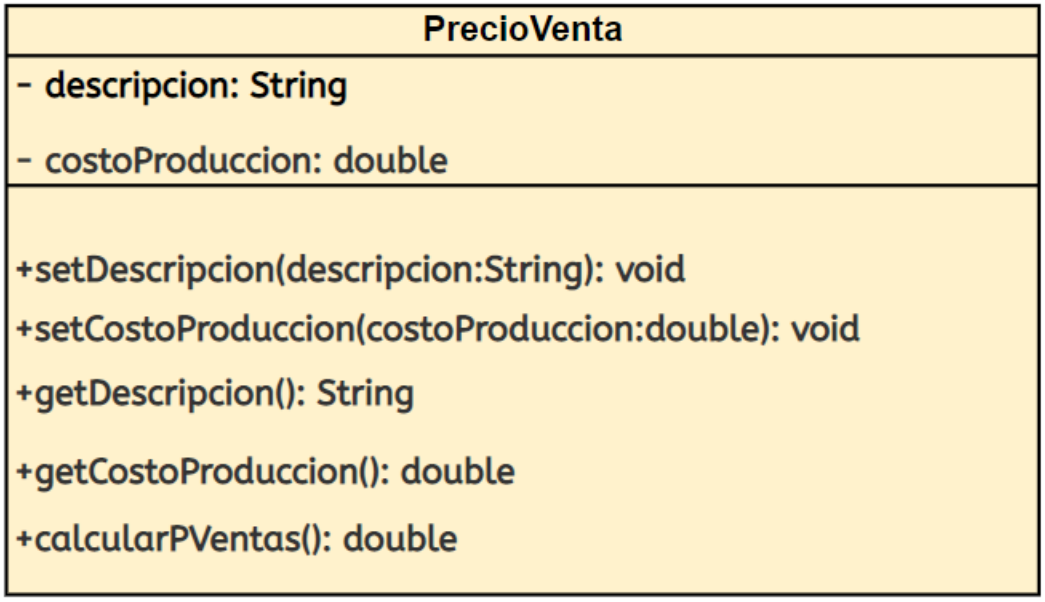


Diagrama de clase Ejemplo 2

Clase secundaria

```
package Ejemplo1B;// Importacion de Paquetes

public class PrecioVenta {//Nombre de la clase
    //Declaración de atributos de la clase
    2 usages
    private String descripcion;
    4 usages
    private double costoProduccion;

    //método get del atributo descripcion
    public String getDescripcion() {
        return descripcion;
    }

    //método set del atributo descripcion
    1 usage
    public void setDescripcion(String descripcion) {
        this.descripcion = descripcion;
    }

    //método get del atributo costoProduccion
    public double getCostoProduccion() {
        return costoProduccion;
    }
     //método get del atributo costoProduccion
     1 usage
     public void setCostoProduccion(double costoProduccion) {
         this.costoProduccion = costoProduccion;
     }

     //método para calcular el costo de venta
     1 usage
     public double calcularPVentas(){
         double resp=0;
         resp=costoProduccion*0.12+costoProduccion;
         resp=resp+resp*0.15;
         return  resp;
     }

}
```

Clase Principal

```java
// Importacion de Paquetes y clases
package Ejemplo1B;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

//Nombre de la clase principal
public class EjecutarPrecioVenta {
    //Método principal de Java. Es el identificador que la JVM busca como punto de partida del programa java
    public static void main(String[] args) throws IOException {
        //Instancia a la clase  BufferedReader se crea el objeto br
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        //Instancia a la clase  PrecioVenta se crea el objeto objPrecio
        PrecioVenta objPrecio= new PrecioVenta();
        //Declaracion de Variables
        String descrip;
        double costo;
        //Impresion [Solicitud de datos de entrada al usuario]
        System.out.println("Ingrese la descripcion del Producto ");
        //Asignacion de lectura realizada mediante el objeto al metodo reaLine
        descrip= br.readLine();

        //Impresion [Solicitud de datos de entrada al usuario]
        System.out.println("Ingrese el costo de Produccion ");
        //Asignacion de lectura realizada mediante el objeto al metodo reaLine
        costo=Double.parseDouble(br.readLine());
        //Mediante el objeto objPrecio invoca al metodo setCostoProduccion se envia los valores
        objPrecio.setCostoProduccion(costo);
        //Mediante el objeto objPrecio invoca al metodo setDescripcion se envia los valores
        objPrecio.setDescripcion(descrip);
        //Mediante el objeto objPrecioVenta se invoca al metodo calcularPventas(), Se imprime el resultado retornado
        System.out.println("El costo de venta es "+objPrecio.calcularPVentas());

    }
}
```

Ejecución

```
Run:  EjecutarPrecioVenta
"C:\Program Files\Java\jdk-17.0.4\bin\java.exe" "
Ingrese la descripcion del Producto
Jabon
Ingrese el costo de Produccion
3.50
El costo de venta es 4.508

Process finished with exit code 0
```

Constructores

Constructores son métodos cuyo nombre coincide con el nombre de la clase y que nunca devuelven ningún tipo de dato, no siendo necesario indicar que el tipo de dato devuelto es void. Los constructores se emplean para inicializar los valores de los objetos y realizar las operaciones que sean necesarias para la generación de este objeto.

Ejemplo 3

Una persona desea cambiar pesos a dólares y requiere un sistema orientado a objetos. Para ello, define una clase con sus respectivos componentes. Cree un objeto e inicialice el tipo de cambio (utilice el constructor con parámetro), imprima la conversión a dólares

1 peso colombiano = 0.00023 dólar

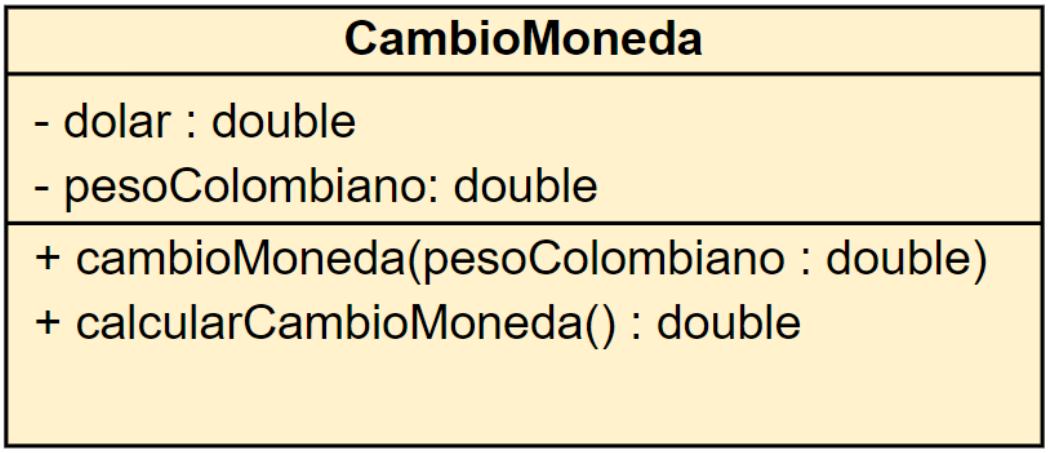


Diagrama de clase Ejemplo 3

Clase secundaria

```
package Ejemplo6; // Importacion de Paquetes

public class CambioMoneda {//Nombre de la clase
    //Declaración de atributos de la clase
    2 usages
    private double dolar;
    2 usages
    private double pesoColombiano;

    //Constructor de la Clase, recibe un parametro
    1 usage
    public CambioMoneda(double pesoColombiano) {
        //Asignaciones de datos
        this.dolar = 0.00023;
        this.pesoColombiano=pesoColombiano;
    }

    //Método donde calcula la conversión de moneda
    1 usage
    public double calcularCambioMoneda(){
        return dolar*pesoColombiano;
    }
}
```

Clase Principal

```
// Importacion de Paquetes y clases
package Ejemplo6;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.text.DecimalFormat;

//Nombre de la clase principal
public class EjecutarCambioMoneda {
    //Método principal de Java. Es el identificador que la JVM busca como punto de partida del programa java
    public static void main(String[] args) throws IOException {
        //Instancia a la clase  BufferedReader se crea el objeto br
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        //Instancia a la clase DecimalFormat se crea el objeto df
        DecimalFormat df = new DecimalFormat( pattern: "####.##");
        //Declaración de variables
        double pesos;
        //Impresion [Solicitud de datos de entrada al usuario]
        System.out.println("Ingrese la cantidad de pesos colombianos");
        /*Asignacion de lectura realizada mediante el objeto al metodo reaLine Como la lectura es un
          String se realiza una conversion de datos
          Se utilza Double.Double.parseDouble(br.readLine()) , para realizar la conversión
         */
        pesos=Double.parseDouble(br.readLine());
        //Instancia a la clase  CambioMoned se crea el objeto objCambio
        CambioMoneda objCambio= new CambioMoneda(pesos);
        //Mediante el objeto objCambio invoca al metodo calcularCambioMoneda(), Se imprime el resultado retornado
        System.out.println("El cambio a Dolares es $"+df.format(objCambio.calcularCambioMoneda()));
    }
}
```

Ejecución

```
Run:  EjecutarPrecioVenta  EjecutarCambioMoneda
"C:\Program Files\Java\jdk-17.0.4\bin\java.exe"
Ingrese la cantidad de pesos colombianos
5000
El cambio a Dolares es $1.15

Process finished with exit code 0
```

Sobrecarga de métodos

La sobrecarga de métodos es la creación de varios métodos con el mismo nombre, pero con diferente lista de tipos de parámetros. Java utiliza el número y tipo de parámetros para seleccionar cuál definición de método ejecutar. Java diferencia los métodos sobrecargados con base en el número y tipo de parámetros o argumentos que tiene el método y no por el tipo que devuelve.

Ejemplo 4

Realice la suma de dos valores, luego realice la suma de tres valores, emplee sobrecarga para resolver.

Clase Secundaria

```
package Suma;// Importacion de Paquetes

public class SumarNumeros {//Nombre de la clase
    /*
        En la clase SumarNumeros, hay dos métodos que se llaman igual
        cuandos dos métodos se llaman igual estamos hablando de
        métodos sobrecargados
     */
    //Método suma , recibe dos parametros
    public double suma (double num1, double num2){
        return num1+num2;
    }
    //Método suma , recibe tres parametros
    public double suma (double num1, double num2,double num3){
        return num1+num2+num3;
    }
}
```

Clase Principal

```java
//Importacion de paquetes y libreias
package Suma;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

//Nombre de la clase principal
public class EjecutarSuma {
    //Método principal de Java. Es el identificador que la JVM busca como punto de partida del programa java
    public static void main(String[] args) throws IOException {
        //Declaracion de Variables
        double valor_a , valor_b, valor_c , res, respuesta;
        //Instancia a la clase  BufferedReader se crea el objeto br
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        //Instancia a la clase   SumarNumeros se crea el objeto obj
        SumarNumeros obj = new SumarNumeros();
        //Impresion [Solicitud de datos de entrada al usuario]
        System.out.println("Ingrese un numero Decimal ");
         /*Asignacion de lectura realizada mediante el objeto al metodo reaLine Como la lectura es un
          String se realiza una conversion de datos
          Se utilza Double.Double.parseDouble(br.readLine()) , para realizar la conversión
         */

        valor_a=Double.parseDouble(br.readLine());
        //Impresion [Solicitud de datos de entrada al usuario]
        System.out.println("Ingrese un numero Decimal ");
        valor_b=Double.parseDouble(br.readLine());
        //Impresion [Solicitud de datos de entrada al usuario]
        System.out.println("Ingrese un numero Decimal ");
        valor_c=Double.parseDouble(br.readLine());
        //Mediante el objeto objPrecioVenta se invoca al metodo suma(), se aginar el valor retornado en res
        res=obj.suma(valor_a, valor_b);
        //Mediante el objeto objPrecioVenta se invoca al metodo suma(), se aginar el valor retornado en respuesta
        respuesta=obj.suma (valor_a, valor_b, valor_c);
        //Se imprime el resultado retornado
        System.out.println("El Resultado del Metodo SobreCarga con dos parametros "+res);
        System.out.println("El Resultado del Metodo SobreCarga con tre parametros  "+respuesta);
    }
}
```

Ejecución