



# Introducción a la Algoritmia

---

<b>Curso</b>	Introducción a la Algoritmia (2326)
<b>Formato</b>	Manual de curso
<b>Autor Institucional</b>	Cibertec
<b>Páginas</b>	327 p.
<b>Elaborador</b>	Cibertec
<b>Revisor de Contenidos</b>	Lindo Sanchez, Donna Julissa

# Índice

Presentación	5
Red de contenidos	6
<b>UNIDAD DE APRENDIZAJE 1: INTRODUCCIÓN A LA ALGORITMIA</b>	7
<b>    1.1 Tema 1 : Conceptos Básicos</b>	8
1.1.1 : Algoritmo	8
1.1.2 : Relación Problema – Algoritmo – Programa	9
1.1.3 : Etapas de un algoritmo: Entrada – Proceso – Salida	9
1.1.4 : Variable	10
1.1.5 : Pseudocódigo	12
1.1.6 : Instrucciones algorítmicas básicas	13
1.1.7 : Etapas de desarrollo de un algoritmo computacional	14
1.1.8 : Sentencias de asignación	15
1.1.9 : Operadores aritméticos	16
<b>UNIDAD DE APRENDIZAJE 2: ESTRUCTURAS DE SECUENCIA</b>	20
<b>    2.1 Tema 2 : Estructuras de Secuencia</b>	21
2.2.1 : Estructuras de Secuencia	21
2.2.2 : Algoritmos con fórmulas definidas	21
2.2.3 : Algoritmos de repartos	30
2.2.4 : Algoritmos de ventas	39
2.2.5 : Algoritmos de sueldos	45
<b>UNIDAD DE APRENDIZAJE 3: ESTRUCTURAS DE SELECCIÓN</b>	53
<b>    3.1 Tema 3 : Estructuras de Selección</b>	54
3.3.1 : Operadores lógicos y relacionales	54
3.3.2 : Estructura de selección if, if – else, if – else – if	55
Estructura de selección simple if	55
Estructura de selección doble if – else	86
Estructura de selección encadenada if – else – if	116
3.3.3 : Estructura de selección switch	151
<b>UNIDAD DE APRENDIZAJE 4: MÉTODOS</b>	175
<b>    4.1 Tema 4 : Métodos</b>	176
4.1.1 : Programación modular	176
4.1.2 : Variables locales y globales	177
4.1.3 : Métodos tipo void	177
4.1.4 : Métodos con valor de retorno	197
<b>UNIDAD DE APRENDIZAJE 5: CONTADORES Y ACUMULADORES</b>	226
<b>    5.1 Tema 5 : Contadores y Acumuladores</b>	227
5.1.1 : Operadores de incremento y decremento	228
5.1.2 : Operadores de asignación compleja	228
5.1.3 : Contadores y acumuladores	229

<b>UNIDAD DE APRENDIZAJE 6: ESTRUCTURAS DE REPETICIÓN</b>	<b>249</b>
<b>6.1 Tema 6 : Estructuras de Repetición</b>	<b>250</b>
6.1.1 : Estructura while	250
6.1.2 : Estructura for	271
6.1.3 : Estructuras do – while	294
<b>Bibliografía</b>	<b>327</b>

# Presentación

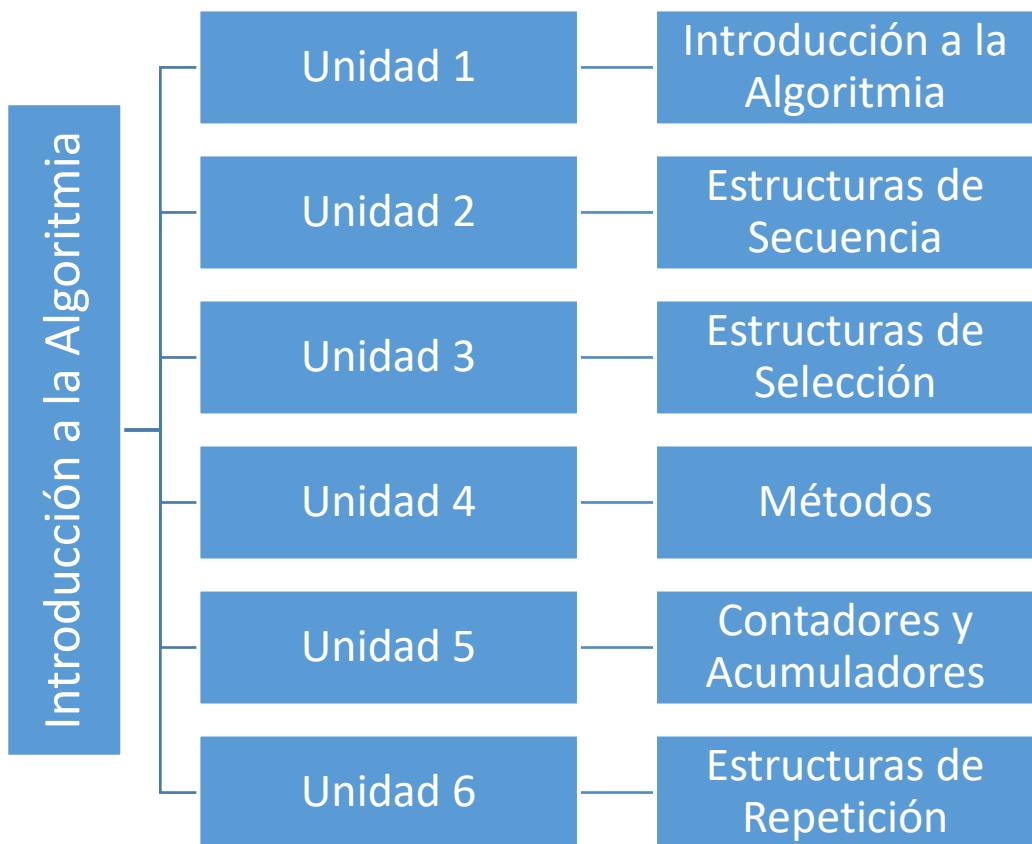
Un algoritmo es una secuencia ordenada y finita de pasos que permite resolver un problema. El término algoritmia proviene del nombre del gran matemático árabe Al-Juarismi, el cual escribió sobre los años 800 y 825 su obra *Quitad Al Mugabala*, donde se recogía el sistema de numeración hindú y el concepto del cero.

La algoritmia es un pilar fundamental de las ciencias de la computación puesto que provee métodos de solución de problemas, que serán implementados en los programas de computadora. En este sentido, un programa de computadora es la implementación de un algoritmo en un determinado lenguaje de programación. Este curso es una introducción a la algoritmia y a la programación en Java.

Este manual consta de ocho temas, los cuales serán desarrollados en 16 semanas. Se ha contemplado para ello objetivos concretos y un conjunto de actividades que serán desarrolladas en clase bajo la guía del profesor.

Finalmente, se espera que el alumno valore el material que tiene en sus manos y pueda probar los programas en la máquina.

# Red de contenidos





# INTRODUCCIÓN A LA ALGORITMIA

## LOGRO DE LA UNIDAD DE APRENDIZAJE

Al finalizar la unidad, el alumno define las etapas de desarrollo de un algoritmo y diseña algoritmos básicos en pseudocódigo.

## TEMARIO

- 1.1 Tema 1 : Conceptos Básicos**
  - 1.1.1 : Algoritmo
  - 1.1.2 : Relación Problema – Algoritmo – Programa
  - 1.1.3 : Etapas de un algoritmo: Entrada – Proceso – Salida
  - 1.1.4 : Variable
  - 1.1.5 : El pseudocódigo
  - 1.1.6 : Sentencias de asignación
  - 1.1.7 : Operadores aritméticos

## ACTIVIDADES PROPUESTAS

- Los alumnos desarrollan algoritmos básicos con fórmulas definidas.

## 1.1. TEMA 1: CONCEPTOS BÁSICOS

### 1.1.1. Algoritmo

Un algoritmo es un método de solución de un problema expresado a través de un conjunto de pasos, procedimientos o acciones. Son algoritmos los siguientes:

- Las instrucciones para instalar un equipo de sonido
- Una receta para preparar un plato de comida
- Las instrucciones para hallar el máximo común divisor de dos números
- Las instrucciones para convertir una cantidad dada en soles a dólares

#### Ejemplo 1

El siguiente algoritmo determina el índice de masa corporal de una persona (IMC)

```
Inicio
    Obtener peso y estatura
    Calcular: IMC = peso / (estatura*estatura)
    Mostrar IMC
Fin
```

#### Ejemplo 2

El siguiente algoritmo determine el área y el precio de un terreno rectangular cuyo costo por metro cuadrado es S/. 750

```
Inicio
    Obtener largo y ancho
    Calcular: areaterreno = largo*ancho
    Calcular: precioterreno = areaterreno*750
    Mostrar areaterreno y precioterreno
Fin
```

### Clasificación de los algoritmos

Los algoritmos pueden clasificarse en *algoritmos computacionales* y *algoritmos no computacionales*.

Un algoritmo es no computacional cuando no puede ser ejecutado en una computadora. El ejecutor de este tipo de algoritmo es un ser humano.

Por ejemplo:

- Las instrucciones para instalar un equipo de sonido
- Una receta para preparar un plato de comida

Un algoritmo es computacional cuando puede ser ejecutado en una computadora.

Por ejemplo:

- Las instrucciones para hallar el máximo común divisor de dos números
- Las instrucciones para convertir una cantidad dada en soles a dólares

### 1.1.2. Relación Problema – Algoritmo – Programa

Un programa es un conjunto de instrucciones expresadas mediante un lenguaje de programación como Java, C, C++, etc. Las instrucciones del programa se obtienen escribiendo las instrucciones del algoritmo mediante el lenguaje de programación elegido. A las instrucciones de un programa expresadas mediante un lenguaje de programación se denomina también código fuente.

Un programa surge ante la necesidad de automatizar la solución de un problema. La Figura 1 muestra que antes de la escritura de un programa, se necesita diseñar un algoritmo. De esta manera, podemos decir, que un programa es la implementación de un algoritmo mediante un lenguaje de programación de forma que sea entendible por el procesador.

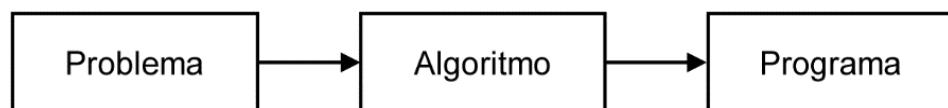


Figura 1 Problema, Algoritmo y Programa

### 1.1.3. Etapas de un algoritmo: Entrada – Proceso – Salida

Todo algoritmo tiene tres etapas claramente diferenciadas: entrada de datos, proceso de cálculo y salida de resultados.

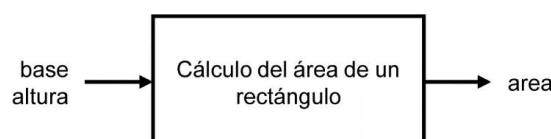
Generalmente, como es el caso de este curso, los datos de entrada se toman de la pantalla del programa y los datos de salida se muestran en la pantalla del mismo programa.



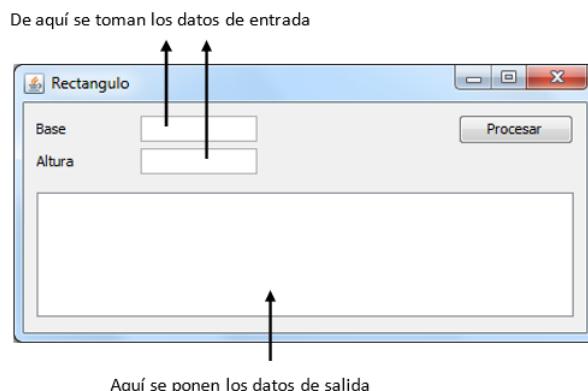
Figura 2 Etapas de un algoritmo

#### Ejemplo 3

En el siguiente gráfico, se especifican los datos de entrada y de salida para el cálculo del área de un rectángulo.

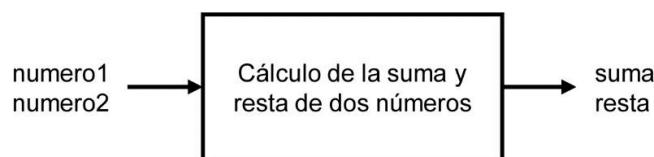


En la figura que sigue, se muestra de dónde se toman los datos de entrada y dónde se ponen los datos de salida.

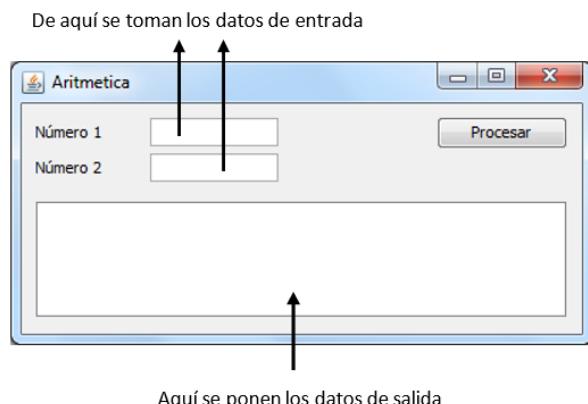


#### Ejemplo 4

En el siguiente gráfico, se especifican los datos de entrada y de salida para el cálculo de la suma y la resta de dos números.

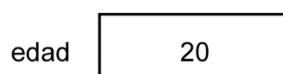


En la figura que sigue, se muestra de dónde se toman los datos de entrada y dónde se ponen los datos de salida.



#### 1.1.4. Variable

Una variable es el nombre asignado a una zona (casilla) de la memoria RAM durante la ejecución de un programa con la finalidad de almacenar un dato o valor. En la Figura 3, se muestra una variable llamada **edad** que almacena un valor **entero** igual a 20.



**Figura 3** Variable

Todas las variables deben ser declaradas antes de ser utilizadas. Declarar una variable consiste en especificar su **nombre** y el **tipo de dato** que almacenará, de acuerdo con una de las siguientes reglas de declaración:

Para una sola variable:

```
tipo nombre;
```

Para varias variables del mismo tipo:

```
tipo nombre1, nombre2, ..., nombreN;
```

En el nombre de una variable deben tenerse en cuenta las siguientes reglas:

- Debe empezar con una letra, un símbolo de subrayado (\_) o un símbolo de dólar (\$). Los siguientes caracteres pueden ser letras, dígitos, símbolos de subrayado (\_) o símbolos de dólar (\$).
- No puede ser una palabra reservada del lenguaje.
- Las mayúsculas y minúsculas se consideran diferentes. El tipo de dato de la variable puede ser uno de la siguiente tabla:

**Tabla 1** Tipos de datos básicos del lenguaje Java

Tipo de dato	Naturaleza del dato
int	Número entero
double	Número decimal
String	Conjunto de caracteres

### Ejemplo 5

La siguiente instrucción declara la variable **numHijos** para almacenar el número de hijos de una persona. El número de hijos es un valor entero, por lo que su tipo de dato es **int**.

```
int numHijos;
```

Esto crea la siguiente casilla de memoria sin contenido.

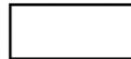
numHijos 

### Ejemplo 6

La siguiente instrucción declara la variable **peso** para almacenar el peso de una persona. El peso es un valor puede ser decimal, por lo que su tipo de dato es **double**.

```
double peso;
```

Esto crea la siguiente casilla de memoria sin contenido.

peso 

### Ejemplo 7

La siguiente instrucción declara la variable **nomApe** para almacenar los nombres y apellidos de una persona. Los nombres y apellidos de una persona están formados por un conjunto de caracteres, por lo que el tipo de dato es **String**.

```
String nomApe;
```

Esto crea la siguiente casilla de memoria sin contenido.

nomApe 

## 1.1.5. Pseudocódigo

### Definición

A las instrucciones de un programa escritas en un lenguaje de programación se denomina **código fuente**. Antes de escribir el **código fuente**, el programador escribe un **pseudocódigo**, esto es, una **imitación del código**. Como imitación del código, el pseudocódigo no tiene reglas formales, varía de un programador a otro.

Un pseudocódigo puede contener símbolos (+, -, \*, /, =, etc.), términos (leer, imprimir, abrir, cerrar, etc.) y estructuras de programación (Si, Si...sino, Hacer...mientras, Mientras...hacer, Para...mientras).

La ventaja del pseudocódigo es que, su uso, en la planificación de un programa, permite al programador concentrarse en la lógica y en las estructuras de control y no preocuparse de las reglas de un lenguaje específico.

En la Figura 4, puede verse la relación entre pseudocódigo y código. Por ejemplo, en el código se usa el tipo **int**, que corresponde al lenguaje Java, en el pseudocódigo se usa el tipo **entero**.

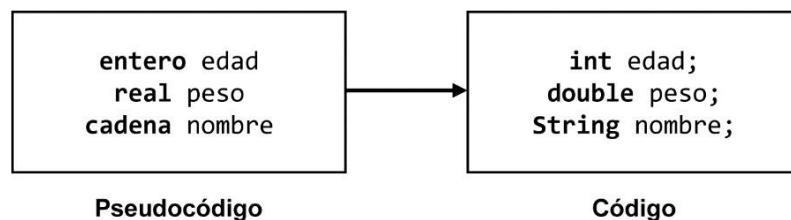


Figura 4 Pseudocódigo y Código

### 1.1.6. Instrucciones algorítmicas básicas

#### Declaración de variables

Para declarar variables en el algoritmo, imitaremos la forma de declarar variables del lenguaje Java usando los *tipos de datos algorítmicos* mostrados en la siguiente tabla:

Tabla 2 Tipos de datos algorítmicos

Java	Algoritmo
int	entero
double	real
char	carácter
String	cadena
boolean	lógico

#### Entrada de datos

La entrada consiste en obtener un dato desde algún dispositivo de entrada y trasladarlo a la memoria para ser almacenada en una variable (Figura 5). En general, la entrada de una variable se escribe en el pseudocódigo de la siguiente forma:

**Ler** variable

Por ejemplo:

**Ler** edad

Ingrasa un valor para la variable edad.

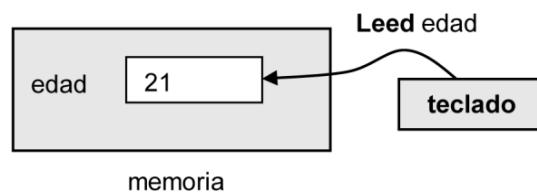


Figura 5 Entrada

## Salida de resultados

La salida consiste en trasladar a algún dispositivo de salida el valor de una variable (Figura 6). En general, la salida de una variable a la pantalla se escribe en el pseudocódigo de la siguiente forma:

```
Imprimir variable
```

Por ejemplo:

```
Imprimir sueldo
```

Imprime el valor de la variable sueldo.

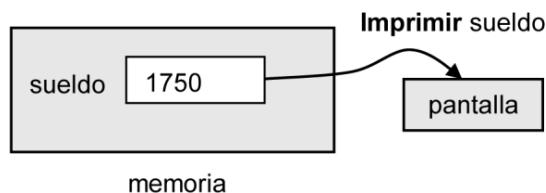


Figura 6 Salida

### 1.1.7. Etapas de desarrollo de un algoritmo computacional

Todo algoritmo computacional puede dividirse en cuatro etapas:

- Declaración de variables
- Entrada de datos
- Proceso de cálculo
- Salida de resultados

### Ejemplo 8

A continuación, se muestra el pseudocódigo del algoritmo del Ejemplo 1 incluyendo todas las etapas de desarrollo de un algoritmo computacional.

```

Inicio
    // Declaración de variables
    real peso, estatura, IMC

    // Entrada de datos
    Leer peso, estatura

    // Proceso de cálculo
    IMC = peso / (estatura * estatura)

    // Salida de resultados
    Imprimir IMC
Fin
  
```

## Ejemplo 9

A continuación, se muestra el pseudocódigo del algoritmo del Ejemplo 2 incluyendo todas las etapas de desarrollo de un algoritmo computacional.

```

Inicio
    // Declaración de variables
    real largo, ancho, areaterreno, precioterreno

    // Entrada de datos
    Ler largo, ancho

    // Proceso de cálculo
    areaterreno = largo*ancho
    precioterreno = areaterreno*750

    // Salida de resultados
    Imprimir areaterreno, precioterreno
Fin
```

## Ejemplo 10

Diseñe un algoritmo que determine el área de un círculo y la longitud de su circunferencia conociendo su radio. Considere las siguientes fórmulas:

$$A = 3.1416 \times r^2$$

$$C = 3.1416 \times r$$

Siendo  $A$  el área y  $C$  la longitud de la circunferencia.

## Solución

```

Inicio
    // Declaración de variables
    real r, A, C

    // Entrada de datos
    Ler r

    // Proceso de cálculo
    A = 3.1416*r*r
    C = 3.1416*r

    // Salida de resultados
    Imprimir A, C
Fin
```

### 1.1.8. Sentencias de asignación

La asignación consiste en almacenar un valor en una variable. La forma general de asignación es la siguiente:

**variable** = expresión

El símbolo igual (=) se denomina *operador de asignación simple*.

Por ejemplo:

```
area = b * h
```

Esto almacena en la variable area el resultado de multiplicar b por h.

### 1.1.9. Operadores aritméticos

En la tabla que sigue, se muestran los operadores aritméticos del lenguaje Java.

**Tabla 3** Operadores aritméticos

Operador	Significado	Ejemplo	Resultado
-	Resta	a - b	Resta de a y b
+	Suma	a + b	Suma de a y b
*	Multiplicación	a * b	Producto de a por b
/	División	a / b	Cociente de a entre b
%	Residuo	a % b	Residuo de a entre b

Los operadores aritméticos pueden utilizarse con tipos enteros y reales. Si ambos operandos son enteros, el resultado es un entero; si alguno de ellos es real, el resultado es real.

#### Ejemplo 11

2 + 5	produce el valor	7
2.0 + 5	produce el valor	7.0
2 + 5.0	produce el valor	7.0
2.0 + 5.0	produce el valor	7.0
10/4	produce el valor	2
10/4.0	produce el valor	2.5
10.0/4/4	produce el valor	2.5
15/2	produce el valor	7
15%2	produce el valor	1
4/10	produce el valor	0
4%10	produce el valor	4

# Problemas propuestos

## Actividades

1. Diseñe un algoritmo que determine el área (A) y el perímetro (P) de un rectángulo del que se conoce su base (b) y su altura (h). Considere las siguientes fórmulas:

$$A = b \times h$$

$$P = 2 \times (b + h)$$

2. Diseñe un algoritmo que determine el área (A) de un rombo del que se conoce su base mayor (B), su base menor (b) y su altura (h). Considere la siguiente fórmula:

$$A = \frac{(b + B) \times h}{2}$$

3. Diseñe un algoritmo que determine el área (A) de un rombo del que se conoce su diagonal mayor (D) y su diagonal menor (d). Considere la siguiente fórmula:

$$A = \frac{D \times d}{2}$$

## Autoevaluación

1. Diseñe un algoritmo que determine la frecuencia cardíaca de un varón conociendo su edad en años y su peso en kilogramos de acuerdo con la siguiente fórmula:

$$\text{frecuencia} = 210 - (0.5 \times \text{edad}) - (0.01 \times \text{peso} + 4)$$

2. Diseñe un algoritmo que determine el área ( $A$ ) y el volumen ( $V$ ) de un cubo del que se conoce su lado ( $l$ ). Considere las siguientes fórmulas:

$$A = 6l^2$$

$$V = l^3$$

## Para recordar

- Un algoritmo es un método para resolver un problema expresado mediante una serie de pasos.
- El código fuente de un programa se obtiene expresando las instrucciones de un algoritmo mediante un lenguaje de programación.
- El pseudocódigo es la expresión del algoritmo mediante un lenguaje informal de pseudoprogramación que es una imitación de uno a más lenguajes de programación.
- Una variable es el nombre de una localización o casillero de memoria RAM en el que se puede guardar un dato.
- Antes de usar una variable, primero hay que declararla.
- Declarar una variable consiste en especificar su tipo de dato y su nombre.
- Una sentencia de asignación es una instrucción que almacena un valor en la memoria RAM.
- Los operadores aritméticos pueden utilizarse con tipos enteros y reales. Si ambos operandos son enteros, el resultado es un entero; si alguno de ellos es real, el resultado es real.



# ESTRUCTURAS DE SECUENCIA

---

## LOGRO DE LA UNIDAD DE APRENDIZAJE

Al finalizar la unidad, el alumno diseña algoritmos que involucran procesos secuenciales siguiendo las etapas de desarrollo de un algoritmo.

## TEMARIO

### 2.1 Tema 2 : Estructuras de Secuencia

- 2.2.1 : Estructuras de Secuencia
- 2.2.2 : Algoritmos con fórmulas definidas
- 2.2.3 : Algoritmos de repartos
- 2.2.4 : Algoritmos de ventas
- 2.2.5 : Algoritmos de sueldos

## ACTIVIDADES PROPUESTAS

- Los alumnos desarrollan algoritmos que involucren estructuras de secuencia.

## 2.1. TEMA 2: ESTRUCTURAS DE SECUENCIA

### 2.1.1. Estructura de secuencia

Una *estructura de secuencia* es aquella en la que las instrucciones están una a continuación de la otra siguiendo una secuencia única.

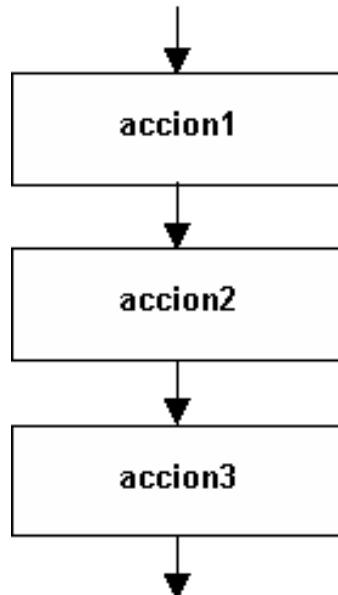


Figura 1 Estructura de Secuencia

### 2.1.2. Algoritmos con fórmulas definidas

#### Problema 1

Diseñe un programa que determine el área total ( $A$ ) y el volumen ( $V$ ) de un cilindro del que se conoce su radio ( $r$ ) y su altura ( $h$ ). Considere las siguientes fórmulas:

$$A = 2\pi r(r + h)$$

$$V = 2\pi r^2 h$$

#### Algoritmo

```

Inicio
    // Declaración de variables
    real r, h, are, vol

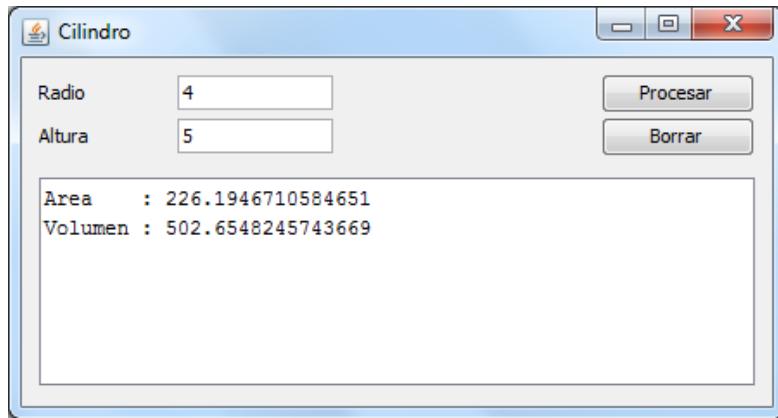
    // Entrada de datos
    Leer r, h

    // Proceso de cálculo
    are = 2*3.1416*r*(r+h)
    vol = 2*3.1416*r*r*h

    // Salida de resultados
    Imprimir are, vol
Fin

```

## Interfaz Gráfica



## Programa

```

package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Cilindro extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblRadio;
    private JLabel lblAltura;
    private JTextField txtRadio;
    private JTextField txtAltura;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Cilindro frame = new Cilindro();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}

```

```
// Crea la GUI
public Cilindro() {
    setTitle("Cilindro");
    setBounds(100, 100, 450, 239);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblRadio = new JLabel("Radio");
    lblRadio.setBounds(10, 13, 80, 14);
    getContentPane().add(lblRadio);

    lblAltura = new JLabel("Altura");
    lblAltura.setBounds(10, 38, 80, 14);
    getContentPane().add(lblAltura);

    txtRadio = new JTextField();
    txtRadio.setBounds(90, 10, 90, 20);
    getContentPane().add(txtRadio);
    txtRadio.setColumns(10);

    txtAltura = new JTextField();
    txtAltura.setBounds(90, 35, 90, 20);
    getContentPane().add(txtAltura);
    txtAltura.setColumns(10);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(335, 34, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 69, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declaración de variables
    double r, h, are, vol;

    // Entrada de datos
    r = Double.parseDouble(txtRadio.getText());
    h = Double.parseDouble(txtAltura.getText());

    // Proceso
    are = 2 * Math.PI * r * (r + h);
    vol = 2 * Math.PI * r * r * h;

    // Salida de resultados
    txtS.setText("Area: " + are + "\n");
    txtS.append ("Volumen : " + vol);
}
```

```
// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtRadio.setText("");
    txtAltura.setText("");
    txtS.setText("");
    txtRadio.requestFocus();
}
}
```

## Problema 2

Diseñe un programa que determine el área de la base ( $A$ ) y el volumen ( $V$ ) de una pirámide de base rectangular conociendo el largo ( $m$ ) y el ancho ( $n$ ) de la base y la altura ( $h$ ) de la pirámide. Considere las siguientes fórmulas:

$$A = mn$$

$$V = \frac{Ah}{3}$$

## Algoritmo

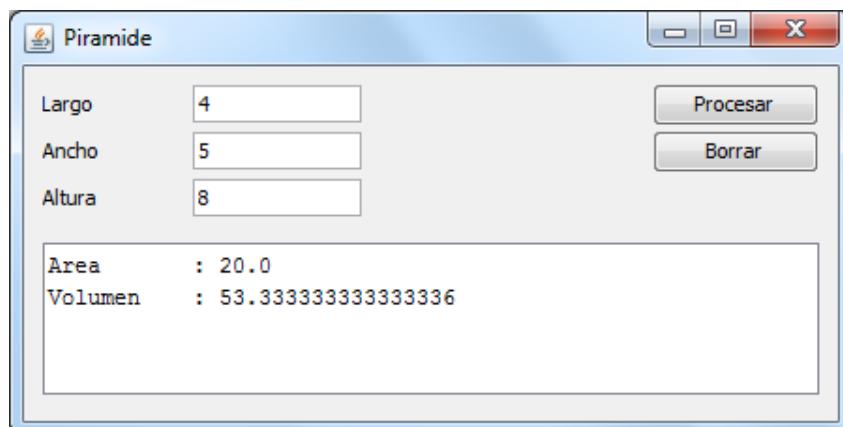
```
Inicio
    // Declaración de variables
    real m, n, h, are, vol

    // Entrada de datos
    Leer m, n, h

    // Proceso de cálculo
    are = m * n
    vol = are * h / 3

    // Salida de resultados
    Imprimir are, vol
Fin
```

## Interfaz Gráfica



## Programa

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Piramide extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblLargo;
    private JLabel lblAncho;
    private JLabel lblAltura;
    private JTextField txtLargo;
    private JTextField txtAncho;
    private JTextField txtAltura;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Piramide frame = new Piramide();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Piramide() {
        setTitle("Piramide");
        setBounds(100, 100, 450, 227);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        lblLargo = new JLabel("Largo");
        lblLargo.setBounds(10, 13, 80, 14);
        getContentPane().add(lblLargo);

        lblAncho = new JLabel("Ancho");
        lblAncho.setBounds(10, 38, 80, 14);
        getContentPane().add(lblAncho);

        lblAltura = new JLabel("Altura");
        lblAltura.setBounds(10, 63, 80, 14);
        getContentPane().add(lblAltura);
    }
}
```

```
txtLargo = new JTextField();
txtLargo.setBounds(90, 10, 90, 20);
getContentPane().add(txtLargo); txtLargo.setColumns(10);

txtAncho = new JTextField();
txtAncho.setBounds(90, 35, 90, 20);
getContentPane().add(txtAncho); txtAncho.setColumns(10);

txtAltura = new JTextField();
txtAltura.setBounds(90, 60, 90, 20);
getContentPane().add(txtAltura); txtAltura.setColumns(10);

btnProcesar = new JButton("Procesar");
btnProcesar.addActionListener(this);
btnProcesar.setBounds(335, 9, 89, 23);
getContentPane().add(btnProcesar);

btnBorrar = new JButton("Borrar");
btnBorrar.addActionListener(this);
btnBorrar.setBounds(335, 34, 89, 23);
getContentPane().add(btnBorrar);

scpScroll = new JScrollPane();
scpScroll.setBounds(10, 94, 414, 81);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);

    }
    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declaración de variables
    double m, n, h, are, vol;

    // Entrada de datos
    m = Double.parseDouble(txtLargo.getText());
    n = Double.parseDouble(txtAncho.getText());
    h = Double.parseDouble(txtAltura.getText());

    // Proceso de cálculo
    are = m * n;
    vol = are * h / 3;

    // Salida de resultados
    txtS.setText("Area : " + are + "\n");
    txtS.append("Volumen : " + vol + "\n");
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtLargo.setText("");
    txtAncho.setText("");
    txtAltura.setText("");
    txtS.setText("");
    txtLargo.requestFocus();
}
```

### Problema 3

Dada una cantidad de dinero y las edades de tres personas, diseñe un programa que reparta el dinero en forma proporcional a las edades.

El monto que le corresponde a cada persona se calcula con la siguiente fórmula:

$$\text{monto de la persona} = \frac{\text{edad de la persona} \times \text{dinero a repartir}}{\text{suma total de edades}}$$

### Algoritmo

```

Inicio
    // Declaración de variables
    real montoP1, montoP2, montoP3, montoRepartir
    entero edadP1, edadP2, edadP3, sumaEdades

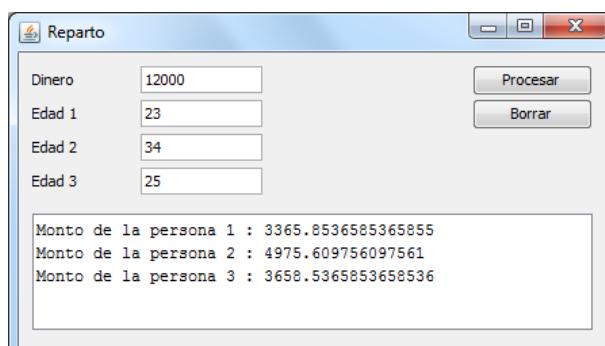
    // Entrada de datos
    Ler montoRepartir, edadP1, edadP2, edadP3

    // Calcula la suma total de edades
    sumaEdades = edadP1 + edadP2 + edadP3

    // Calcula la cantidad de dinero de cada persona
    montoP1 = (edadP1 * montoRepartir) / sumaEdades
    montoP2 = (edadP2 * montoRepartir) / sumaEdades
    montoP3 = (edadP3 * montoRepartir) / sumaEdades

    // Salida de resultados
    Imprimir montoP1, montoP2, montoP3
Fin
```

### Interfaz Gráfica



### Programa

```

package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
```

```
public class Reparto extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblDinero;
    private JLabel lblEdad1;
    private JLabel lblEdad2;
    private JLabel lblEdad3;
    private JTextField txtDinero;
    private JTextField txtEdad1;
    private JTextField txtEdad2;
    private JTextField txtEdad3;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Reparto frame = new Reparto();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Reparto() {
        setTitle("Reparto");
        setBounds(100, 100, 450, 255);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        lblDinero = new JLabel("Dinero");
        lblDinero.setBounds(10, 13, 80, 14);
        getContentPane().add(lblDinero);

        lblEdad1 = new JLabel("Edad 1");
        lblEdad1.setBounds(10, 38, 80, 14);
        getContentPane().add(lblEdad1);

        lblEdad2 = new JLabel("Edad 2");
        lblEdad2.setBounds(10, 63, 80, 14);
        getContentPane().add(lblEdad2);

        lblEdad3 = new JLabel("Edad 3");
        lblEdad3.setBounds(10, 88, 80, 14);
        getContentPane().add(lblEdad3);

        txtDinero = new JTextField();
        txtDinero.setBounds(90, 10, 90, 20);
        getContentPane().add(txtDinero);
        txtDinero.setColumns(10);

        txtEdad1 = new JTextField();
        txtEdad1.setBounds(90, 35, 90, 20);
        getContentPane().add(txtEdad1);
        txtEdad1.setColumns(10);
```

```
txtEdad2 = new JTextField();
txtEdad2.setBounds(90, 60, 90, 20);
getContentPane().add(txtEdad2);
txtEdad2.setColumns(10);

txtEdad3 = new JTextField();
txtEdad3.setColumns(10);
txtEdad3.setBounds(90, 85, 90, 20);
getContentPane().add(txtEdad3);

btnProcesar = new JButton("Procesar");
btnProcesar.addActionListener(this);
btnProcesar.setBounds(335, 9, 89, 23);
getContentPane().add(btnProcesar);

btnBorrar = new JButton("Borrar");
btnBorrar.addActionListener(this);
btnBorrar.setBounds(335, 34, 89, 23);
getContentPane().add(btnBorrar);

scpScroll = new JScrollPane();
scpScroll.setBounds(10, 119, 414, 86);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);

}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {

    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }

}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declaración de variables
    double monto1, monto2, monto3, dinero;
    int edad1, edad2, edad3, sumaEdades;

    // Entrada de datos
    dinero = Double.parseDouble(txtDinero.getText());
    edad1 = Integer.parseInt(txtEdad1.getText());
    edad2 = Integer.parseInt(txtEdad2.getText());
    edad3 = Integer.parseInt(txtEdad3.getText());

    // Calcula la suma total de edades
    sumaEdades = edad1 + edad2 + edad3;

    // Calcula la cantidad de dinero de cada persona
    monto1 = (edad1 * dinero) / sumaEdades;
    monto2 = (edad2 * dinero) / sumaEdades;
    monto3 = (edad3 * dinero) / sumaEdades;

    // Salida de resultados
    txtS.setText("Monto de la persona 1 : " + monto1 + "\n");
    txtS.append("Monto de la persona 2 : " + monto2 + "\n");
    txtS.append("Monto de la persona 3 : " + monto3);

}
```

```
// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtDinero.setText("");
    txtEdad1.setText("");
    txtEdad2.setText("");
    txtEdad3.setText("");
    txtS.setText("");
    txtDinero.requestFocus();
}
}
```

### 2.1.3. Algoritmos de reparto

#### Problema 4

Una empresa expondrá sus productos en una feria. La empresa considera que el monto total de dinero a invertir estará distribuido de la siguiente manera:

Rubro	Porcentaje
Alquiler de espacio en la feria	23%
Publicidad	7%
Transporte	26%
Servicios feriales	12%
Decoración	21%
Gastos varios	11%

Dado el monto total de dinero a invertir, diseñe un programa que determine cuánto gastará la empresa en cada rubro.

#### Algoritmo

```

Inicio
    // Declaración de variables
    real montoTotal, rubro1, rubro2, rubro3, rubro4, rubro5, rubro6

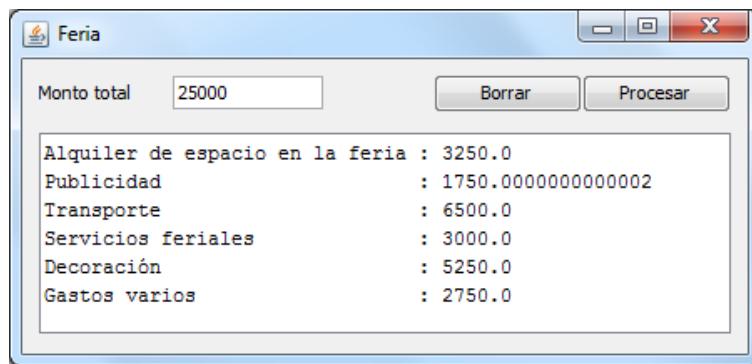
    // Entrada de datos
    Leer montoTotal

    // Proceso de cálculo
    rubro1 = 0.13*montoTotal
    rubro2 = 0.07*montoTotal
    rubro3 = 0.26*montoTotal
    rubro4 = 0.12*montoTotal
    rubro5 = 0.21*montoTotal
    rubro6 = 0.11*montoTotal

    // Salida de resultados
    Imprimir rubro1, rubro2, rubro3, rubro4, rubro5, rubro6
Fin

```

## Interfaz Gráfica



## Programa

```

package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Feria extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblMontoTotal;
    private JTextField txtMontoTotal;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Feria frame = new Feria();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Feria() {
        setTitle("Feria");
        setBounds(100, 100, 450, 214);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);
    }
}

```

```
lblMontoTotal = new JLabel("Monto total");
lblMontoTotal.setBounds(10, 13, 80, 14);
getContentPane().add(lblMontoTotal);

txtMontoTotal = new JTextField();
txtMontoTotal.setBounds(90, 10, 90, 20);
getContentPane().add(txtMontoTotal);
txtMontoTotal.setColumns(10);

btnProcesar = new JButton("Procesar");
btnProcesar.addActionListener(this);
btnProcesar.setBounds(335, 9, 89, 23);
getContentPane().add(btnProcesar);

btnBorrar = new JButton("Borrar");
btnBorrar.addActionListener(this);
btnBorrar.setBounds(246, 9, 89, 23);
getContentPane().add(btnBorrar);

scpScroll = new JScrollPane();
scpScroll.setBounds(10, 44, 414, 120);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declaración de variables
    double montoTotal, rubro1, rubro2, rubro3, rubro4, rubro5, rubro6;

    // Entrada de datos
    montoTotal = Double.parseDouble(txtMontoTotal.getText());

    // Proceso de cálculo
    rubro1 = 0.13 * montoTotal;
    rubro2 = 0.07 * montoTotal;
    rubro3 = 0.26 * montoTotal;
    rubro4 = 0.12 * montoTotal;
    rubro5 = 0.21 * montoTotal;
    rubro6 = 0.11 * montoTotal;

    // Salida de resultados
    txtS.setText("Alquiler de espacio en la feria : " + rubro1 + "\n");
    txtS.append("Publicidad : " + rubro2 + "\n");
    txtS.append("Transporte : " + rubro3 + "\n");
    txtS.append("Servicios feriales : " + rubro4 + "\n");
    txtS.append("Decoración : " + rubro5 + "\n");
    txtS.append("Gastos varios: " + rubro6);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtMontoTotal.setText("");
    txtS.setText("");
    txtMontoTotal.requestFocus();
}
```

## Problema 5

Un padre repartirá una cantidad de dinero entre sus cinco hijos. Cada uno recibirá una cantidad equivalente a:

- Tamar: 85% del monto recibido por Josué
- Josué: 27% de la cantidad a repartir
- Caleb: 23% del monto total recibido entre Josué y Daniel
- Daniel: 25% de la cantidad a repartir
- David: lo que queda del dinero a repartir

Dada la cantidad de dinero a repartir, diseñe un programa que determine cuánto de dinero recibirá cada hijo.

## Algoritmo

```

Inicio
    // Declaración de variables
    real dinero, dinTamar, dinJosue, dinCaleb, dinDaniel, dinDavid

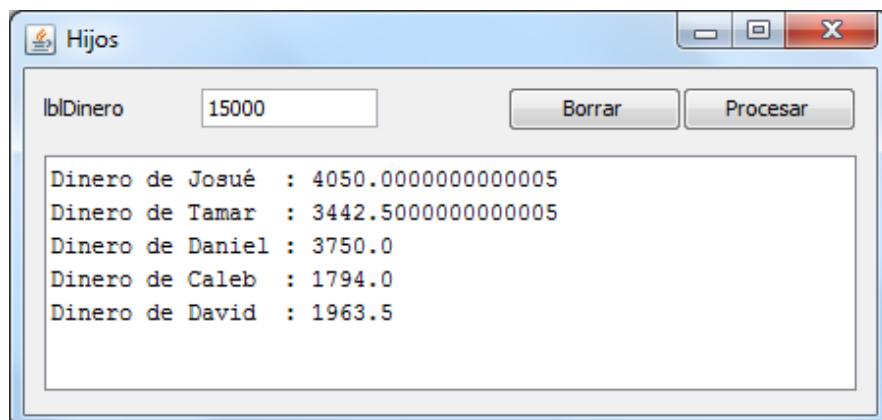
    // Entrada de datos
    Leer dinero

    // Proceso de cálculo
    dinJosue = 0.27*dinero
    dinTamar = 0.85*dinJosue
    dinDaniel = 0.25*dinero
    dinCaleb = 0.23*(dinJosue + dinDaniel)
    dinDavid = dinero - (dinTamar + dinJosue + dinCaleb + dinDaniel)

    // Salida de resultados
    Imprimir dinJosue, dinTamar, dinDaniel, dinCaleb, dinDavid
Fin

```

## Interfaz Gráfica



## Programa

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Hijos extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel Dinero;
    private JTextField txtDinero;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
            } catch (Throwable e) {
                e.printStackTrace();
            }
            EventQueue.invokeLater(new Runnable() {
                public void run() {
                    try {
                        Hijos frame = new Hijos();
                        frame.setVisible(true);
                    } catch (Exception e) {
                        e.printStackTrace();
                    }
                }
            });
        }
    }

    // Crea la GUI
    public Hijos() {
        setTitle("Hijos");
        setBounds(100, 100, 450, 214);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        Dinero = new JLabel("lblDinero");
        Dinero.setBounds(10, 13, 80, 14);
        getContentPane().add(Dinero);

        txtDinero = new JTextField();
        txtDinero.setBounds(90, 10, 90, 20);
        getContentPane().add(txtDinero);
        txtDinero.setColumns(10);

        btnProcesar = new JButton("Procesar");
        btnProcesar.addActionListener(this);
        btnProcesar.setBounds(335, 9, 89, 23);
        getContentPane().add(btnProcesar);

        btnBorrar = new JButton("Borrar");
        btnBorrar.addActionListener(this);
        btnBorrar.setBounds(246, 9, 89, 23);
        getContentPane().add(btnBorrar);
    }
}
```

```
scpScroll = new JScrollPane();
scpScroll.setBounds(10, 44, 414, 120);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {

    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declaración de variables
    double dinero, dinTamar, dinJosue, dinCaleb, dinDaniel, dinDavid;

    // Entrada de datos
    dinero = Double.parseDouble(txtDinero.getText());

    // Proceso de cálculo
    dinJosue = 0.27 * dinero;
    dinTamar = 0.85 * dinJosue;
    dinDaniel = 0.25 * dinero;
    dinCaleb = 0.23 * (dinJosue + dinDaniel);
    dinDavid = dinero - (dinTamar + dinJosue + dinCaleb + dinDaniel);

    // Salida de resultados
    txtS.setText("Dinero de Josué : " + dinJosue + "\n");
    txtS.append("Dinero de Tamar : " + dinTamar + "\n");
    txtS.append("Dinero de Daniel : " + dinDaniel + "\n");
    txtS.append("Dinero de Caleb : " + dinCaleb + "\n");
    txtS.append("Dinero de David : " + dinDavid);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtDinero.setText("");
    txtS.setText("");
    txtDinero.requestFocus();
}
}
```

## Problema 6

Dadas las cantidades de dinero aportadas por Débora, Raquel y Séfora para formar un capital, diseñe un programa que determine el monto del capital formado y el porcentaje de dicho capital que aporta cada una.

## Algoritmo

```

Inicio
    // Declaración de variables
    real dineDeb, dineRaq, dineSef, capital, porcDeb, porcRaq, porcSef

    // Entrada de datos
    Ler dineDeb, dineRaq, dineSef

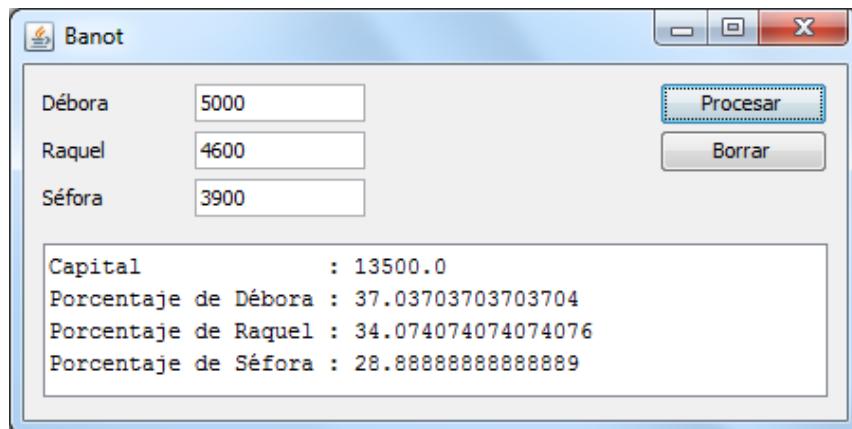
    // Determina el capital formado
    capital = dineDeb + dineRaq + dineSef

    // Determina los porcentajes
    porcDeb = dineDeb*100/capital
    porcRaq = dineRaq*100/capital
    porcSef = dineSef*100/capital

    // Salida de resultados
    Imprimir capital, porcDeb, porcRaq, porcSef
Fin

```

## Interfaz Gráfica



## Programa

```

package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

```

```
public class Banot extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblDebora;
    private JLabel lblRaquel;
    private JLabel lblSefora;
    private JTextField txtDebora;
    private JTextField txtRaquel;
    private JTextField txtSefora;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Banot frame = new Banot();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Banot() {
        setTitle("Banot");
        setBounds(100, 100, 450, 264);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        lblDebora = new JLabel("Débora");
        lblDebora.setBounds(10, 13, 80, 14);
        getContentPane().add(lblDebora);

        lblRaquel = new JLabel("Raquel");
        lblRaquel.setBounds(10, 38, 80, 14);
        getContentPane().add(lblRaquel);

        lblSefora = new JLabel("Séfora");
        lblSefora.setBounds(10, 63, 80, 14);
        getContentPane().add(lblSefora);

        txtDebora = new JTextField();
        txtDebora.setBounds(90, 10, 90, 20);
        getContentPane().add(txtDebora);
        txtDebora.setColumns(10);

        txtRaquel = new JTextField();
        txtRaquel.setBounds(90, 35, 90, 20);
        getContentPane().add(txtRaquel);
        txtRaquel.setColumns(10);

        txtSefora = new JTextField();
        txtSefora.setBounds(90, 60, 90, 20);
        getContentPane().add(txtSefora);
        txtSefora.setColumns(10);

        btnProcesar = new JButton("Procesar");
        btnProcesar.addActionListener(this);
        btnProcesar.setBounds(335, 9, 89, 23);
        getContentPane().add(btnProcesar);
    }
}
```

```
btnBorrar = new JButton("Borrar");
btnBorrar.addActionListener(this);
btnBorrar.setBounds(335, 34, 89, 23);
getContentPane().add(btnBorrar);

scpScroll = new JScrollPane();
scpScroll.setBounds(10, 94, 414, 120);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {

    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declaración de variables
    double dineDeb, dineRaq, dineSef, capital, porcDeb, porcRaq, porcSef;

    // Entrada de datos
    dineDeb = Double.parseDouble(txtDebora.getText());
    dineRaq = Double.parseDouble(txtRaquel.getText());
    dineSef = Double.parseDouble(txtSefora.getText());

    // Determina el capital formado
    capital = dineDeb + dineRaq + dineSef;

    // Determina los porcentajes
    porcDeb = dineDeb * 100 / capital;
    porcRaq = dineRaq * 100 / capital;
    porcSef = dineSef * 100 / capital;

    // Salida de resultados
    txtS.setText("Capital : " + capital + "\n");
    txtS.append("Porcentaje de Débora : " + porcDeb + "\n");
    txtS.append("Porcentaje de Raquel : " + porcRaq + "\n");
    txtS.append("Porcentaje de Séfora : " + porcSef);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtDebora.setText("");
    txtRaquel.setText("");
    txtSefora.setText("");
    txtS.setText("");
    txtDebora.requestFocus();
}
}
```

## 2.1.4. Algoritmos de ventas

### Problema 7

Una tienda ha puesto en oferta la venta de un producto ofreciendo un 11% de descuento sobre el importe de la compra.

El importe de la compra se calcula multiplicando el precio del producto por la cantidad de unidades adquiridas. El importe a pagar se calcula restando el importe de la compra menos el importe del descuento.

Como incentivo especial, la tienda obsequia 2 caramelos por cada unidad adquirida.

Dado el precio del producto y la cantidad de unidades adquiridas, diseñe un algoritmo que determine el importe de la compra, el importe del descuento y el importe a pagar.

### Algoritmo

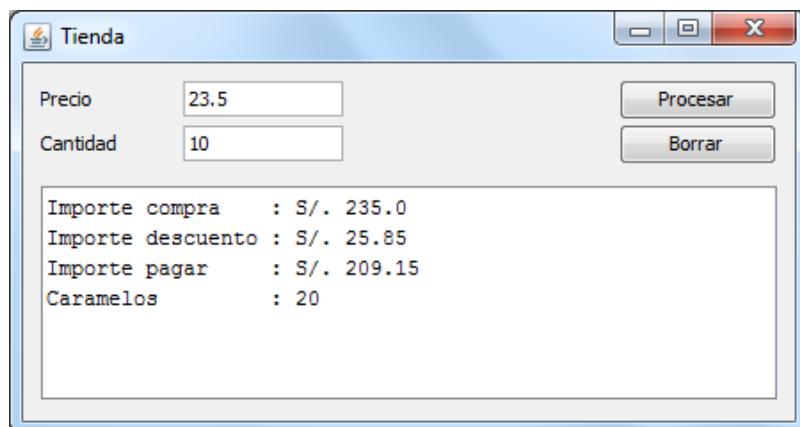
```
Inicio
    // Declaración de variables
    real pre, impcom, impdes, imppag
    entero can, car

    // Entrada de datos
    Leer pre, can

    // Cálculo de importes
    impcom = can*pre
    impdes = 0.11*impcom
    imppag = impcom - impdes
    car = 2*can

    // Salida de resultados
    Imprimir impcom, impdes, imppag, car
Fin
```

### Interfaz Gráfica



## Programa

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Tienda extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblPrecio;
    private JLabel lblCantidad;
    private JTextField txtPrecio;
    private JTextField txtCantidad;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Tienda frame = new Tienda();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Tienda() {
        setTitle("Tienda");
        setBounds(100, 100, 450, 239);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        lblPrecio = new JLabel("Precio");
        lblPrecio.setBounds(10, 13, 80, 14);
        getContentPane().add(lblPrecio);

        lblCantidad = new JLabel("Cantidad");
        lblCantidad.setBounds(10, 38, 80, 14);
        getContentPane().add(lblCantidad);

        txtPrecio = new JTextField();
        txtPrecio.setBounds(90, 10, 90, 20);
        getContentPane().add(txtPrecio);

        txtCantidad = new JTextField();
        txtCantidad.setBounds(90, 35, 90, 20);
        getContentPane().add(txtCantidad);
    }
}
```

```
btnProcesar = new JButton("Procesar");
btnProcesar.addActionListener(this);
btnProcesar.setBounds(335, 9, 89, 23);
getContentPane().add(btnProcesar);

btnBorrar = new JButton("Borrar");
btnBorrar.addActionListener(this);
btnBorrar.setBounds(335, 34, 89, 23);
getContentPane().add(btnBorrar);

scpScroll = new JScrollPane();
scpScroll.setBounds(10, 69, 414, 120);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declaración de variables
    int can, car;
    double pre, impcom, impdes, imppag;

    // Entrada de datos
    pre = Double.parseDouble(txtPrecio.getText());
    can = Integer.parseInt(txtCantidad.getText());

    // Proceso de cálculo
    impcom = pre * can;
    impdes = 0.11 * impcom;
    imppag = impcom - impdes;
    car = 2 * can;

    // Salida de resultados
    txtS.setText("Importe compra : S/. " + impcom + "\n");
    txtS.append("Importe descuento : S/. " + impdes + "\n");
    txtS.append("Importe pagar : S/. " + imppag + "\n");
    txtS.append("Caramelos : " + car);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtPrecio.setText("");
    txtCantidad.setText("");
    txtS.setText("");
    txtPrecio.requestFocus();
}
}
```

### Problema 8

Una tienda ha puesto en oferta la venta de camisas ofreciendo un descuento, por temporada de verano, denominado 7% + 7%. Los cálculos se efectúan de la siguiente manera:

- El importe de la compra es igual al producto del precio de la camisa por la cantidad de unidades adquiridas.
- El importe del primer descuento es igual al 7% del importe de la compra.
- El importe del segundo descuento es igual al 7% de lo que queda de restar el importe de la compra menos el importe del primer descuento.
- El importe del descuento total es igual a la suma de los dos descuentos anteriores.
- El importe a pagar es igual al importe de la compra menos el importe del descuento total.

Dado el precio del producto y la cantidad de unidades adquiridas, diseñe un programa que determine el importe de la compra, el importe del primer descuento, el importe del segundo descuento, el importe del descuento total y el importe a pagar

### Algoritmo

```

Inicio
    // Declaración de variables entero can
    real pre, impcom, impdes, imppag, des1, des2

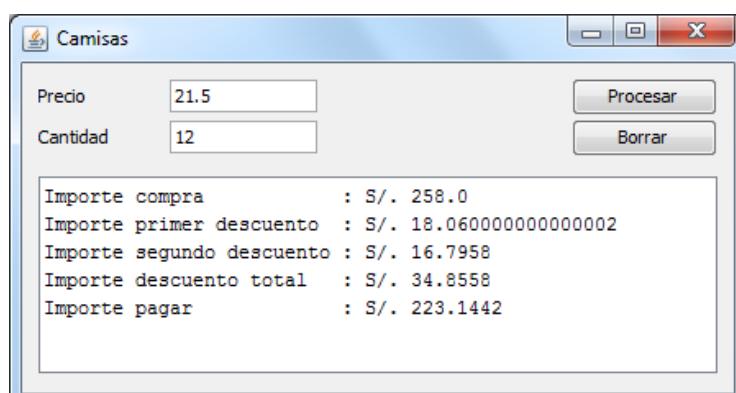
    // Entrada de datos
    Leer pre, can

    // Proceso de cálculo
    impcom = pre*can
    des1 = 0.07*impcom
    des2 = 0.07*(impcom - des1)
    impdes = des1 + des2
    imppag = impcom - impdes

    // Salida de resultados
    Imprimir impcom, des1, des2, impdes, imppag
Fin

```

### Interfaz Gráfica



## Programa

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Camisas extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblPrecio;
    private JLabel lblCantidad;
    private JTextField txtPrecio;
    private JTextField txtCantidad;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Camisas frame = new Camisas();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}

// Crea la GUI
public Camisas() {
    setTitle("Camisas");
    setBounds(100, 100, 450, 239);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblPrecio = new JLabel("Precio");
    lblPrecio.setBounds(10, 13, 80, 14);
    getContentPane().add(lblPrecio);

    lblCantidad = new JLabel("Cantidad");
    lblCantidad.setBounds(10, 38, 80, 14);
    getContentPane().add(lblCantidad);

    txtPrecio = new JTextField();
    txtPrecio.setBounds(90, 10, 90, 20);
    getContentPane().add(txtPrecio);
    txtPrecio.setColumns(10);
```

```
txtCantidad = new JTextField();
txtCantidad.setBounds(90, 35, 90, 20);
getContentPane().add(txtCantidad);
txtCantidad.setColumns(10);

btnProcesar = new JButton("Procesar");
btnProcesar.addActionListener(this);
btnProcesar.setBounds(335, 9, 89, 23);
getContentPane().add(btnProcesar);

btnBorrar = new JButton("Borrar");
btnBorrar.addActionListener(this);
btnBorrar.setBounds(335, 34, 89, 23);
getContentPane().add(btnBorrar);

scpScroll = new JScrollPane();
scpScroll.setBounds(10, 69, 414, 120);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declaración de variables
    int can;
    double pre, impcom, impdes, imppag, des1, des2;

    // Entrada de datos
    pre = Double.parseDouble(txtPrecio.getText());
    can = Integer.parseInt(txtCantidad.getText());

    // Proceso de cálculo
    impcom = pre * can;
    des1 = 0.07 * impcom;
    des2 = 0.07 * (impcom - des1);
    impdes = des1 + des2;
    imppag = impcom - impdes;

    // Salida de resultados
    txtS.setText("Importe compra : S/. " + impcom + "\n");
    txtS.append("Importe primer descuento : S/. " + des1 + "\n");
    txtS.append("Importe segundo descuento : S/. " + des2 + "\n");
    txtS.append("Importe descuento total : S/. " + impdes + "\n");
    txtS.append("Importe pagar: S/. " + imppag);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtPrecio.setText("");
    txtCantidad.setText("");
    txtS.setText("");
    txtPrecio.requestFocus();
}
```

## 2.1.5. Algoritmos de sueldos

### Problema 9

El cálculo del pago mensual de un empleado de una empresa se efectúa de la siguiente manera:

- Sueldo básico: horas trabajadas por la tarifa horaria
- Bonificación: 20% del sueldo básico
- Sueldo bruto: sueldo básico más bonificación
- Descuento: 10% del sueldo bruto
- Sueldo neto: sueldo bruto menos descuento

Dadas las horas trabajadas y la tarifa horaria de un empleado, diseñe un programa que determine el sueldo básico, la bonificación, el sueldo bruto, el descuento y el sueldo neto que le corresponden.

### Algoritmo

```

Inicio
    // Declaración de variables
    real horasTrab, tarifaHor
    real sueldoBas, montoBoni, sueldoBru, montoDesc, sueldoNet

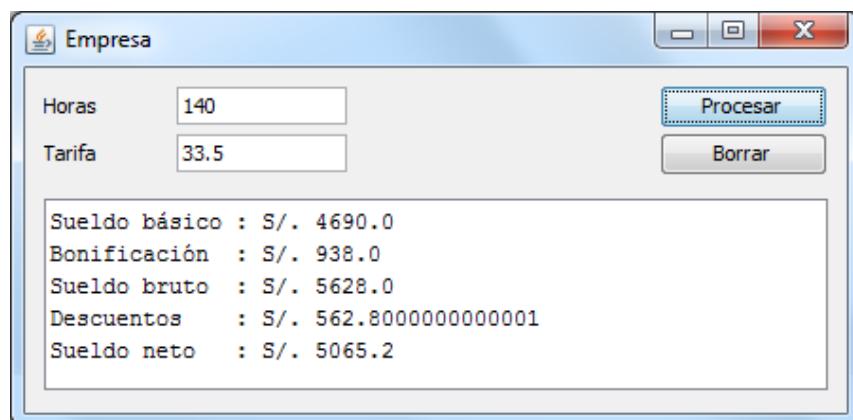
    // Entrada de datos
    Leer horasTrab, tarifaHor

    // Proceso de cálculo
    sueldoBas = horasTrab*tarifaHor
    montoBoni = 0.20*sueldoBas
    sueldoBru = sueldoBas+montoBoni
    montoDesc = 0.10*sueldoBru
    sueldoNet = sueldoBru-montoDesc

    //Salida de resultados
    Imprimir sueldoBas, montoBoni, sueldoBru, montoDesc, sueldoNet
Fin

```

### Interfaz Gráfica



## Programa

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Empresa extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblHoras;
    private JLabel lblTarifa;
    private JTextField txtHoras;
    private JTextField txtTarifa;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Empresa frame = new Empresa();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Empresa() {
        setTitle("Empresa");
        setBounds(100, 100, 450, 239);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        lblHoras = new JLabel("Horas");
        lblHoras.setBounds(10, 13, 70, 14);
        getContentPane().add(lblHoras);

        lblTarifa = new JLabel("Tarifa");
        lblTarifa.setBounds(10, 38, 70, 14);
        getContentPane().add(lblTarifa);

        txtHoras = new JTextField();
        txtHoras.setBounds(80, 10, 90, 20);
        getContentPane().add(txtHoras);
        txtHoras.setColumns(10);
    }
}
```

```
txtTarifa = new JTextField();
txtTarifa.setBounds(80, 35, 90, 20);
getContentPane().add(txtTarifa);
txtTarifa.setColumns(10);

btnProcesar = new JButton("Procesar");
btnProcesar.addActionListener(this);
btnProcesar.setBounds(335, 9, 89, 23);
getContentPane().add(btnProcesar);

btnBorrar = new JButton("Borrar");
btnBorrar.addActionListener(this);
btnBorrar.setBounds(335, 34, 89, 23);
getContentPane().add(btnBorrar);

scpScroll = new JScrollPane();
scpScroll.setBounds(10, 69, 414, 120);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declaración de variables
    double horasTrab, tarifaHor;
    double sueldoBas, montoBoni, sueldoBru, montoDesc, sueldoNet;

    // Entrada de datos
    horasTrab = Double.parseDouble(txtHoras.getText());
    tarifaHor = Double.parseDouble(txtTarifa.getText());

    // Cálculo de montos

    sueldoBas = horasTrab * tarifaHor;
    montoBoni = 0.20 * sueldoBas;
    sueldoBru = sueldoBas + montoBoni;
    montoDesc = 0.10 * sueldoBru;
    sueldoNet = sueldoBru - montoDesc;

    // Salida de resultados
    txtS.setText("Sueldo básico : S/. " + sueldoBas + "\n");
    txtS.append("Bonificación : S/. " + montoBoni + "\n");
    txtS.append("Sueldo bruto : S/. " + sueldoBru + "\n");
    txtS.append("Descuentos : S/. " + montoDesc + "\n");
    txtS.append("Sueldo neto : S/. " + sueldoNet);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtHoras.setText("");
    txtTarifa.setText("");
    txtS.setText("");
    txtHoras.requestFocus();
}
}
```

# Problemas propuestos

## Actividades

- Diseñe un algoritmo que determine el área lateral (AL), el área total (AT) y el área de la base (AB) de un cilindro del que se conoce su radio (r) y su altura (h). Considere las siguientes fórmulas:

$$AT = 2 \times AB + AL$$

$$AB = 3.1416 \times r^2$$

$$AL = 2 \times 3.1416 \times r \times h$$

- Diseñe un algoritmo que determine el área de la base (AB), el perímetro de la base (PB) y el área lateral (AL) de un cono del que se conoce el radio de su base (R) y su generatriz (G). Considere las siguientes fórmulas:

$$AB = 3.14 \times R^2$$

$$AL = \frac{PB \times G}{2}$$

$$PB = 6.28 \times R$$

- Diseñe un algoritmo que determine el área (A) y el volumen (V) de una esfera de la que se conoce su radio (r). Considere las siguientes fórmulas:

$$A = 12.57 \times r^2$$

$$V = \frac{12.57 \times r^3}{3}$$

- Una institución social ha recibido una donación en dinero que lo repartirá entre cinco áreas. Cada área recibirá una parte de la donación equivalente a:

- Centro de salud: 25% de la donación
- Comedor: 45% del monto recibido por la escuela
- Biblioteca: 17% del monto total recibido entre el comedor y la escuela
- Escuela: 35% de la donación
- Biblioteca: 40% del monto recibido por el centro de salud
- Asilo de ancianos: lo que queda de la donación

Dado el importe de la donación, diseñe un algoritmo que determine qué cantidad de dinero le corresponde a cada área.

- Una empresa ha recibido una donación en dinero que lo repartirá entre cinco áreas. Cada área recibirá una parte de la donación equivalente a:

- Área de producción: 25% del monto de la donación
- Área de contabilidad: 40% del monto total recibido entre las áreas de marketing y soporte
- Área de marketing: 15% del monto total recibido entre las áreas de producción y soporte
- Área de soporte: 20% del monto de la donación
- Área de recursos humanos: lo que queda del monto de la donación

- Dado el importe de la donación, diseñe un algoritmo que determine el monto de dinero que recibirá cada área.
6. Una tienda ha puesto en oferta la venta de un producto ofreciendo un descuento igual al 15% del importe de la compra. El importe de la compra se calcula multiplicando el precio del producto por la cantidad de unidades adquiridas. El importe a pagar se calcula restando el importe de la compra menos el importe del descuento. Dado el precio del producto y la cantidad de unidades adquiridas, diseñe un algoritmo que determine el importe de la compra, el importe del descuento y el importe a pagar.
  7. Una empresa de transportes aplica un descuento igual al 7% del importe de la compra. El importe de la compra se calcula multiplicando el precio del pasaje por la cantidad de pasajes adquiridos. El importe a pagar se calcula restando el importe de la compra menos el importe del descuento. Como incentivo adicional, la empresa obsequia 3 chocolates por cada pasaje adquirido. Dado el precio del pasaje y la cantidad de pasajes adquiridos, diseñe un algoritmo que determine el importe de la compra, el importe del descuento, el importe a pagar y la cantidad de chocolates de obsequio que le corresponden a un cliente.
  8. Una imprenta ha lanzado al mercado la venta al por mayor del cuaderno de 100 hojas mentor que es distribuido a nivel nacional. El importe compra se calcula multiplicando el precio de la docena por la cantidad de docenas adquiridas. Como oferta, la imprenta aplica un descuento del 12% del importe compra. El importe a pagar se calcula restando el importe de la compra menos el importe del descuento y sumando el costo del transporte. Dado el precio de la docena, la cantidad de docenas adquiridas y el costo del transporte, diseñe un algoritmo que determine el importe compra, el importe del descuento y el importe a pagar que le corresponden a un cliente.
  9. Una empresa paga a sus empleados por horas trabajadas. El sueldo bruto se calcula multiplicando las horas trabajadas por la tarifa horaria del empleado. Por ley, todo empleado está sujeto a un descuento igual al 15% del sueldo bruto. El sueldo neto se calcula restando el sueldo bruto menos el importe del descuento. Dado el número de horas trabajadas y la tarifa horaria de un empleado, diseñe un algoritmo que determine el sueldo bruto, el descuento y el sueldo neto del empleado.
  10. Una empresa paga a sus vendedores un sueldo bruto que se calcula sumando un sueldo básico más una comisión. El sueldo básico es S/. 350.75. La comisión es igual al 5% del importe total vendido en el mes. Por ley, todo empleado está sujeto a un descuento igual al 15% del sueldo bruto. El sueldo neto se calcula restando el sueldo bruto menos el importe del descuento. Dado el importe total vendido en el mes, diseñe un algoritmo que imprima la boleta de un vendedor indicando el sueldo básico, la comisión, el sueldo bruto, el descuento y el sueldo neto.

## Autoevaluación

1. Diseñe un algoritmo que lea un ángulo en grados sexagesimales (S) y lo convierta a sus equivalentes en grados centesimales (C) y radianes (R). Considere las siguientes fórmulas:

$$C = \frac{200 \times S}{180}$$

$$R = \frac{3.1416 \times S}{180}$$

2. Diseñe un algoritmo que lea una temperatura en grados Centígrados (C) y la convierta a sus equivalentes en grados Fahrenheit (F), grados Kelvin (K) y grados Rankine(R). Utilice las siguientes fórmulas:

$$K = R - 187$$

$$R = C + 460$$

$$F = \frac{9}{5} \times C + 32$$

3. Para estimar el peso de un niño en situaciones de emergencias pediátricas, se utiliza la siguiente fórmula:

$$\text{peso en kilogramos} = 3 \times \text{edad en años} + 7$$

Dada la edad de un niño en años, diseñe un algoritmo que determine el peso estimado del niño.

4. Un hospital ha recibido una donación especial que será repartida entre las áreas de Pediatría, Medicina General, Ginecología y Traumatología. Cada área recibirá una parte de la donación equivalente a:

- Pediatría: 20% del monto total recibido entre Medicina General y Ginecología
- Medicina General: 45% de la donación
- Ginecología: 80% del monto recibido por Medicina General
- Traumatología: lo que resta la donación

Dado el monto de la donación, diseñe un algoritmo que determine cuánto recibirá cada área

5. Un padre desea repartir una cantidad de dinero entre sus tres hijos. Cada uno recibirá una parte del dinero equivalente a:

- Juan: 45% del dinero a repartir
- Pedro: 60% del monto recibido por Juan
- Luis: Lo que queda del monto de dinero a repartir

Dado el monto de dinero a repartir, diseñe un algoritmo que determine qué cantidad de dinero le corresponde a cada hijo.

6. Una empresa confecciona polos de tamaño estándar aplicando un descuento del 11.5% del importe de la compra. El importe de la compra se calcula multiplicando el precio del polo por la cantidad de polos adquiridos. El importe a pagar se calcula restando el importe compra menos el importe del descuento. Adicionalmente, la empresa obsequia dos lapiceros por cada polo adquirido. Dado el precio del polo y la cantidad de polos adquiridos, diseñe un algoritmo que determine el importe de la compra, el importe del descuento, el importe a pagar y la cantidad de lapiceros de obsequio que le corresponden a un cliente.
7. Una tienda ha puesto en oferta la venta un producto ofreciendo un descuento denominado 10%+10% que consiste en aplicar dos descuentos del 10%. El importe compra se calcula multiplicando el precio del producto por la cantidad de unidades adquiridas. El primer descuento es igual al 10% del importe compra. El segundo descuento es igual al 10% del importe que queda de restar el importe compra menos el importe del primer descuento. El importe del descuento total se calcula sumando el primer y el segundo descuento. El importe a pagar se calcula restando el importe compra menos el importe del descuento total. Dado el precio del producto y la cantidad de unidades adquiridas, diseñe un algoritmo que determine el importe de la compra, el importe del descuento total y el importe a pagar.
8. Los cálculos salariales de los vendedores de una empresa se efectúan de la siguiente manera:
  - Sueldo básico mensual: S/.300
  - Comisión por ventas: 9% del importe total vendido en el mes
  - Sueldo bruto: sueldo básico más comisión
  - Descuento: 11% de sueldo bruto
  - Sueldo neto: sueldo bruto menos descuentoDado el importe total vendido en el mes, diseñe un algoritmo que imprima un reporte indicando el sueldo básico, la comisión, el sueldo bruto, el descuento y el sueldo neto.
9. Los cálculos salariales de los empleados de una empresa se efectúan de la siguiente manera:
  - Sueldo bruto: horas trabajadas por la tarifa horaria del empleado
  - Descuento por ESSALUD: 9% del sueldo bruto
  - Descuento por AFP: 10% del sueldo bruto
  - Descuento total: suma de los descuentos anteriores
  - Sueldo neto: sueldo bruto menos descuento totalDado el número de horas trabajadas y la tarifa horaria, diseñe un algoritmo que determine el sueldo bruto, el descuento por ESSALUD, el descuento por AFP, el descuento total y el sueldo neto.
10. Una empresa ha decidido otorgar una bonificación a sus empleados por única vez. La bonificación estará compuesta de la suma de una bonificación por hijos más una bonificación por tiempo de servicio. La bonificación por hijos será igual a S/. 25 por cada hijo. La bonificación por tiempo de servicio será igual a S/. 50 por cada año de tiempo de servicio. Dado el número de hijos y el número de años de tiempo de servicio, diseñe un algoritmo que determine el importe de la bonificación por hijos, el importe de la bonificación por tiempo de servicio y el importe de la bonificación total que le corresponden a un empleado.

## Para recordar

- Las instrucciones secuenciales se efectúan de arriba hacia abajo, por lo que si en un cierto punto del programa se requiere el valor de una variable, esta debe haber sido asignada previamente.



# ESTRUCTURAS DE SELECCIÓN

## LOGRO DE LA UNIDAD DE APRENDIZAJE

Al finalizar la unidad, el alumno diseña algoritmos que involucran procesos selectivos usando las estructuras de selección más apropiadas.

## TEMARIO

- 3.1 Tema 3 : Estructuras de Selección**
  - 3.3.1 : Operadores lógicos y relacionales
  - 3.3.2 : Estructura de selección if, if – else, if – else – if
  - 3.3.3 : Estructura de selección switch

## ACTIVIDADES PROPUESTAS

- Los alumnos desarrollan algoritmos que involucren estructuras de selección if
- Los alumnos desarrollan algoritmos que involucren la estructura de selección if – else.
- Los alumnos desarrollan algoritmos que involucren la estructura if-else-if.
- Los alumnos desarrollan algoritmos que involucren estructuras de selección múltiple.

## 3.1. TEMA 3: ESTRUCTURAS DE SELECCIÓN

### 3.1.1. Operadores lógicos y relacionales

Son operadores que se utilizan para crear condiciones lógicas. Una condición lógica es una expresión lógica que puede ser verdadera (true) o falsa (false) y puede incluir operadores aritméticos.

#### Operadores lógicos

Son operadores que permiten relacionar varias expresiones lógicas. El conjunto de operadores lógicos se muestra en la Tabla 1.

**Tabla 1** Operadores lógicos

Operador	Significado
	OR lógico (ó)
&&	AND lógico (y)
!	NOT lógico (no)

Las tablas de verdad de los operadores lógicos son las mismas de la lógica matemática, como se muestra en la Tabla 2.

**Tabla 2** Tabla de verdad de los operadores lógicos

p	q	p && q	p    q	!p
true	true	true	true	false
true	false	false	true	false
false	true	false	true	true
false	false	false	false	true

#### Operadores relacionales

Los operadores relacionales se utilizan para escribir condiciones que describan la relación entre dos valores. El conjunto de operadores relacionales se muestra en la Tabla 3.

**Tabla 3** Operadores relacionales

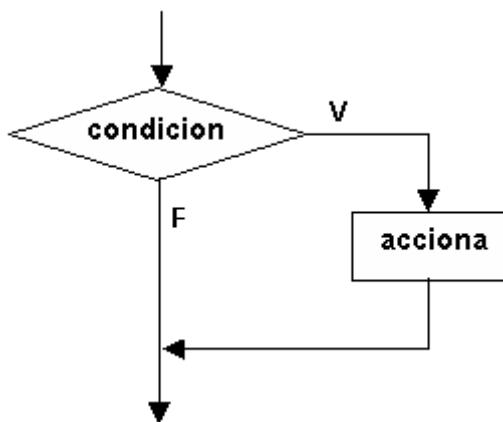
Operador	Significado
==	Igual a
!=	Diferente de
>	Mayor que
<	Menor que
>=	Mayor o igual que
<=	Menor o igual que

### 3.1.2. Estructura de selección if, if – else, if – else – if

#### Estructura de selección simple if

##### Definición

La estructura de selección **simple if** evalúa una condición lógica y, en caso resulte verdadera, efectúa la acción **acciona**. En caso de que la condición resulte falsa, continúa con la siguiente instrucción del programa. La acción **acciona** puede ser una **acción simple** (una sola acción) o una **acción compuesta** (bloque de acciones).



**Figura 1** Diagrama de flujo de la estructura de selección simple if

En la tabla que sigue, se muestra el código y el pseudocódigo de la estructura de selección simple if. Note que, en el caso de bloques de acciones (más de una acción), estas deben estar encerradas entre llaves de bloque {}.

Código Java	Pseudocódigo
<code>if( condicion )     accionA;</code>	<code>si( condicion )     accionA</code>
<code>if( condicion ){     accionA1;     accionA2;     .     .     .     accionAn; }</code>	<code>si( condicion ){     accionA1     accionA2     .     .     .     accionAn }</code>

## Problemas propuestos

### Problema 1

Una tienda vende un producto a precios unitarios que dependen de la cantidad de unidades adquiridas de acuerdo con la siguiente tabla:

Unidades adquiridas	Precio unitario
1 a 25	S/. 27.7
26 a 50	S/. 25.5
51 a 75	S/. 23.5
76 en adelante	S/. 21.5

Adicionalmente, si el cliente adquiere más de 50 unidades la tienda le descuenta el 15% del importe de la compra; en caso contrario, sólo le descuenta el 5% del importe compra. Diseñe un programa que determine el importe de la compra, el importe del descuento y el importe a pagar por la compra de cierta cantidad de unidades del producto.

### Algoritmo

```

Inicio
    // Declaración de variables
    entero unidades;
    real impcom = 0, impdes, imppag

    // Entrada de datos
    Ler unidades

    // Calcula el importe de la compra
    si(unidades >= 1 && unidades <= 25)
        impcom = unidades*27.5

    si(unidades >= 26 && unidades <= 50)
        impcom = unidades*25.5

    si(unidades >= 51 && unidades <= 75)
        impcom = unidades*27.5

    si(unidades >= 76)
        impcom = unidades*27.5

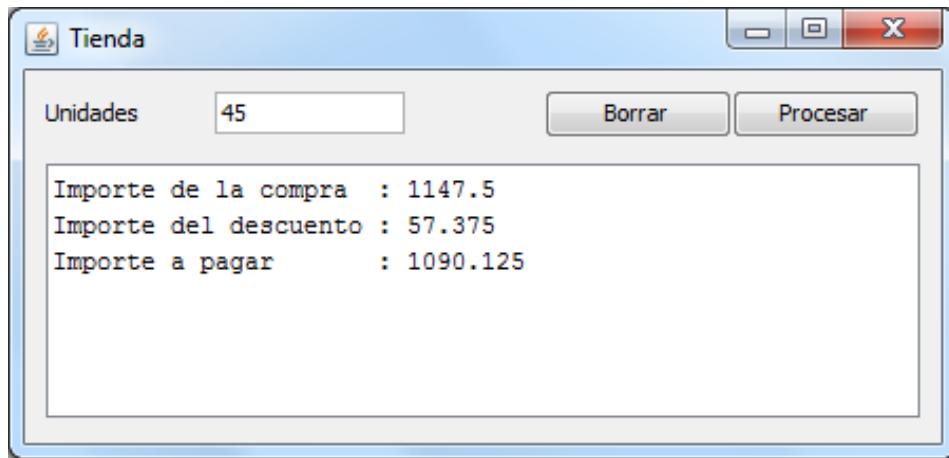
    // Calcula el importe del descuento
    si(unidades > 50)
        impdes = 0.15*impcom
    sino
        impdes = 0.05*impcom

    // Calcula el importe a pagar
    imppag = impcom - impdes;

    // Salida de resultados
    Imprimir impcom, impdes, imppag
Fin

```

## Interfaz Gráfica



## Programa

```
package cibertec;

import java.awt.EventQueue;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import java.awt.Font;

public class Tienda extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblUnidades;
    private JTextField txtUnidades;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {

        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Tienda frame = new Tienda();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}
```

```
// Crea la GUI
public Tienda() {
    setTitle("Tienda");
    setBounds(100, 100, 450, 214);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblUnidades = new JLabel("Unidades");
    lblUnidades.setBounds(10, 13, 80, 14);
    getContentPane().add(lblUnidades);

    txtUnidades = new JTextField();
    txtUnidades.setBounds(90, 10, 90, 20);
    getContentPane().add(txtUnidades);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(246, 9, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 44, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declaración de variables
    int unidades;
    double impcom = 0, impdes, imppag;

    // Entrada de datos
    unidades = Integer.parseInt(txtUnidades.getText());

    // Calcula el importe de la compra
    if (unidades >= 1 && unidades <= 25)
        impcom = unidades * 27.5;

    if (unidades >= 26 && unidades <= 50)
        impcom = unidades * 25.5;

    if (unidades >= 51 && unidades <= 75)
        impcom = unidades * 27.5;

    if (unidades >= 76)
        impcom = unidades * 27.5;

    // Calcula el importe del descuento
    if (unidades > 50)
        impdes = 0.15 * impcom;
    else
        impdes = 0.05 * impcom;
```

```

    // Calcula el importe a pagar
    imppag = impcom - impdes;

    // Salida de resultados
    txtS.setText("Importe de la compra : " + impcom + "\n");
    txtS.append("Importe del descuento : " + impdes + "\n");
    txtS.append("Importe a pagar      : " + imppag);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtUnidades.setText("");
    txtS.setText("");
    txtUnidades.requestFocus();
}
}

```

## Problema 2

Una tienda ha decidido incentivar a sus clientes con un obsequio. Para ello, el cliente debe extraer un bolo de una urna que contiene 100 bolos numerados del 1 al 100.

Luego, sobre la base del número del bolo se obtiene el obsequio de acuerdo con la siguiente tabla:

Número del bolo	Obsequio
1 a 20	Un lapicero
21 a 40	Un cuaderno de 100 hojas
41 a 60	Una caja de plumones
61 a 80	Un cuaderno espiral
81 a 99	Una agenda
100	Una mochila

Dado el número del bolo obtenido por un cliente, diseñe un programa que determine qué obsequio le corresponde.

En caso de que el número ingresado sea incorrecto, como obsequio mostrar "Ninguno".

## Algoritmo

```

Inicio
    // Declaración de variables
    entero numero
    cadena obsequio = "Ninguno"

    // Entrada de datos
    Leer numero

    // Determina el obsequio
    si(numero >= 1 && numero <= 20)
        obsequio = "Un lapicero"

    si(numero >= 21 && numero <= 40)
        obsequio = "Un cuaderno de 100 hojas"

    si(numero >= 41 && numero <= 60)
        obsequio = "Una caja de 12 plumones"

```

```

    si(numero >= 61 && numero <= 80)
        obsequio = "Un cuaderno espiral"

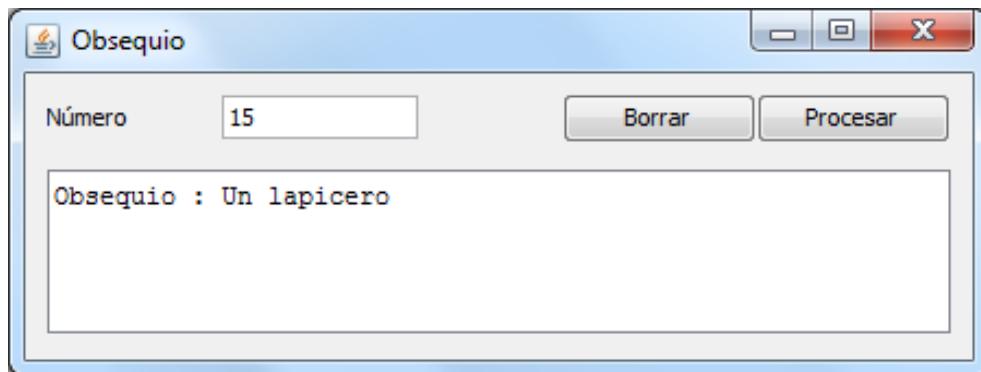
    si(numero >= 81 && numero <= 99)
        obsequio = "Una agenda"

    si(numero == 100)
        obsequio = "Una mochila"

    // Salida de resultados
    Imprimir obsequio
Fin

```

## Interfaz Gráfica



## Programa

```

package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Obsequio extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblNumero;
    private JTextField txtNumero;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {

        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
    }
}

```

```
EventQueue.invokeLater(new Runnable() {
    public void run() {
        try {
            Obsequio frame = new Obsequio();
            frame.setVisible(true);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
});

// Crea la GUI
public Obsequio() {
    setTitle("Obsequio");
    setBounds(100, 100, 450, 169);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblNumero = new JLabel("Número");
    lblNumero.setBounds(10, 13, 80, 14);
    getContentPane().add(lblNumero);

    txtNumero = new JTextField();
    txtNumero.setBounds(90, 10, 90, 20);
    getContentPane().add(txtNumero);
    txtNumero.setColumns(10);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(246, 9, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 44, 414, 75);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declaración de variables
    int numero;
    String obsequio = "Ninguno";

    // Entrada de datos
    numero = Integer.parseInt(txtNumero.getText());

    // Determina el obsequio
    if (numero >= 1 && numero <= 20)
        obsequio = "Un lapicero";
}
```

```

if (numero >= 21 && numero <= 40)
    obsequio = "Un cuaderno de 100 hojas";

if (numero >= 41 && numero <= 60)
    obsequio = "Una caja de 12 plumones";

if (numero >= 61 && numero <= 80)
    obsequio = "Un cuaderno espiral";

if (numero >= 81 && numero <= 99)
    obsequio = "Una agenda";

if (numero == 100)
    obsequio = "Una mochila";

// Salida de resultados
txtS.setText("Obsequio : " + obsequio);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtNumero.setText("");
    txtS.setText("");
    txtNumero.requestFocus();
}
}

```

### Problema 3

Una dulcería vende chocolates a los precios dados en la siguiente tabla:

Tipo de chocolate	Precio unitario
Primor	S/. 8.5
Dulzura	S/. 10.0
Tentación	S/. 7.0
Explosión	S/. 12.5

Como oferta, la tienda aplica un porcentaje de descuento sobre el importe de la compra, sobre la base de la cantidad de chocolates adquiridos, de acuerdo con la siguiente tabla:

Cantidad de chocolates	Descuento
< 5	4.0%
≥ 5 y < 10	6.5%
≥ 10 y < 15	9.0%
≥ 15	11.5%

Adicionalmente, si el importe a pagar es no menor de S/. 250, la tienda obsequia 3 caramelos por cada chocolate; en caso contrario, obsequia 2 caramelos por cada chocolate.

Dado el tipo de chocolate y la cantidad de unidades adquiridas, diseñe un programa que determine el importe de la compra, el importe del descuento, el importe a pagar y la cantidad de caramelos de obsequio.

## Algoritmo

```

Inicio
// Declaración de variables
    entero tipo, cantidad, caramelos
    real impcom = 0, impdes = 0, imppag

    // Entrada de datos
    Ler tipo, cantidad

    // Calcula el importe de la compra
    si(tipo == 0)
        impcom = 8.5*cantidad

    si(tipo == 1)
        impcom = 10.0*cantidad

    si(tipo == 2)
        impcom = 7.0*cantidad

    si(tipo == 3)
        impcom = 12.5*cantidad

    // Calcula el importe del descuento
    si(cantidad < 5)
        impdes = 0.04*impcom

    si(cantidad >= 5 && cantidad < 10)
        impdes = 0.065*impcom

    si(cantidad >= 10 && cantidad < 15)
        impdes = 0.09*impcom

    si(cantidad >= 15)
        impdes = 0.115*impcom

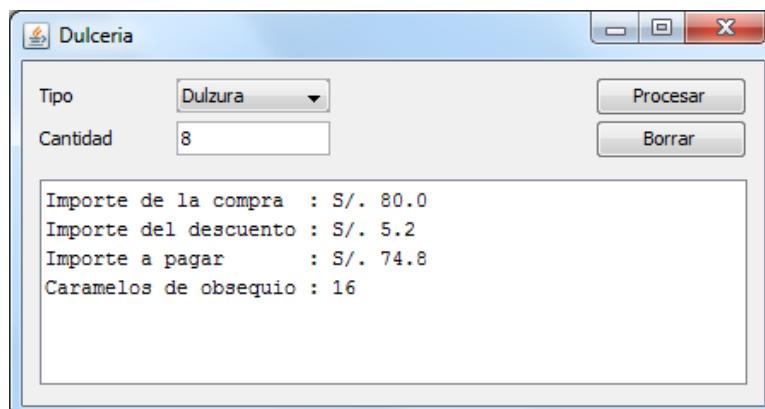
    // Calcula el importe a pagar
    imppag = impcom - impdes

    // Calcula la cantidad de caramelos de regalo
    si(imppag < 250)
        caramelos = 2*cantidad
    sino
        caramelos = 3*cantidad

    // Salida de resultados
    Imprimir impcom, impdes, imppag, caramelos
Fin

```

## Interfaz Gráfica



## Programa

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JComboBox;
import javax.swing.DefaultComboBoxModel;

public class Dulceria extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblTipo;
    private JLabel lblCantidad;
    private JComboBox<String> cboTipo;
    private JTextField txtCantidad;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Dulceria frame = new Dulceria();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Dulceria() {
        setTitle("Dulceria");
        setBounds(100, 100, 450, 239);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        lblTipo = new JLabel("Tipo");
        lblTipo.setBounds(10, 13, 80, 14);
        getContentPane().add(lblTipo);

        lblCantidad = new JLabel("Cantidad");
        lblCantidad.setBounds(10, 38, 80, 14);
        getContentPane().add(lblCantidad);

        cboTipo = new JComboBox<String>();
        cboTipo.setModel(new DefaultComboBoxModel<String>(new String[] {
        "Primor", "Dulzura", "Tentación", "Explosión" }));
        cboTipo.setBounds(90, 10, 90, 20);
        getContentPane().add(cboTipo);
    }
}
```

```
txtCantidad = new JTextField();
txtCantidad.setBounds(90, 35, 90, 20);
getContentPane().add(txtCantidad);
txtCantidad.setColumns(10);

btnProcesar = new JButton("Procesar");
btnProcesar.addActionListener(this);
btnProcesar.setBounds(335, 9, 89, 23);
getContentPane().add(btnProcesar);

btnBorrar = new JButton("Borrar");
btnBorrar.addActionListener(this);
btnBorrar.setBounds(335, 34, 89, 23);
getContentPane().add(btnBorrar);

scpScroll = new JScrollPane();
scpScroll.setBounds(10, 69, 414, 120);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declaración de variables
    int tipo, cantidad, caramelos;
    double impcom = 0, impdes = 0, imppag;

    // Entrada de datos
    tipo = cboTipo.getSelectedIndex();
    cantidad = Integer.parseInt(txtCantidad.getText());

    // Calcula el importe de la compra
    if (tipo == 0)
        impcom = 8.5 * cantidad;

    if (tipo == 1)
        impcom = 10.0 * cantidad;

    if (tipo == 2)
        impcom = 7.0 * cantidad;

    if (tipo == 3)
        impcom = 12.5 * cantidad;

    // Calcula el importe del descuento
    if (cantidad < 5)
        impdes = 0.04 * impcom;

    if (cantidad >= 5 && cantidad < 10)
        impdes = 0.065 * impcom;

    if (cantidad >= 10 && cantidad < 15)
        impdes = 0.09 * impcom;

    if (cantidad >= 15)
        impdes = 0.115 * impcom;
}
```

```

// Calcula el importe a pagar
imppag = impcom - impdes;

// Calcula la cantidad de caramelos de regalo
if (imppag < 250)
    caramelos = 2 * cantidad;
else
    caramelos = 3 * cantidad;

// Salida de resultados
txtS.setText("Importe de la compra : S/. " + impcom + "\n");
txtS.append("Importe del descuento : S/. " + impdes + "\n");
txtS.append("Importe a pagar : S/. " + imppag + "\n");
txtS.append("Caramelos de obsequio : " + caramelos);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    cboTipo.setSelectedIndex(0);
    txtCantidad.setText("");
    txtS.setText("");
    txtCantidad.requestFocus();
}
}

```

#### Problema 4

Los cálculos salariales de los empleados de una empresa se efectúan de la siguiente manera:

Sueldo bruto	: horas trabajadas x tarifa horaria
Descuento	: 15% del sueldo bruto
Sueldo neto	: sueldo bruto – descuento

La tarifa horaria depende de la categoría del trabajador de acuerdo con la siguiente tabla:

Categoría	Tarifa horaria (S.)
C1	45.0
C2	37.5
C3	35.0
C4	32.5

Dadas las horas trabajadas y la categoría de un empleado, diseñe un programa que determine la tarifa horaria, el sueldo bruto, el descuento y el sueldo neto del empleado.

#### Algoritmo

```

Inicio
    // Declaración de variables
    entero categoria
    real horas, suelbru, desc, suelnet, tarifa = 0

    // Entrada de datos
    Leer categoria, horas

    // Determina la tarifa horaria
    si(categoría == 0)
        tarifa = 45.0

    si(categoría == 1)
        tarifa = 37.5

```

```

    si(categoría == 2)
        tarifa = 35.0

    si(categoría == 3)
        tarifa = 32.5

    // Calcula el sueldo bruto
    suelbru = horas*tarifa

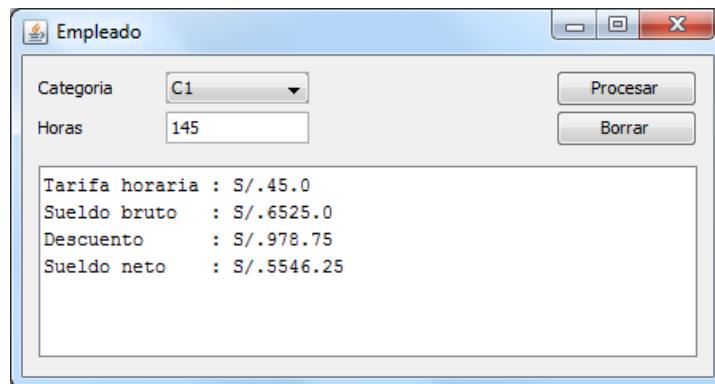
    // Calcula el descuento
    desc = 0.15*suelbru

    // Calcula el sueldo neto
    suelnet = suelbru - desc

    // Salida de resultados
    Imprimir tarifa, suelbru, desc, suelnet
Fin

```

## Interfaz Gráfica



## Programa

```

package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JComboBox;
import javax.swing.DefaultComboBoxModel;

public class Empleado extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblCategoría;
    private JLabel lblHoras;
    private JTextField txtHoras;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;
    private JComboBox<String> cboCategoria;
}

```

```
// Lanza la aplicación
public static void main(String[] args) {
    try {
        UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
    } catch (Throwable e) {
        e.printStackTrace();
    }

    EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                Empleado frame = new Empleado();
                frame.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

// Crea la GUI
public Empleado() {

    setTitle("Empleado");
    setBounds(100, 100, 450, 239);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblCategoria = new JLabel("Categoria");
    lblCategoria.setBounds(10, 13, 80, 14);
    getContentPane().add(lblCategoria);

    lblHoras = new JLabel("Horas");
    lblHoras.setBounds(10, 38, 80, 14);
    getContentPane().add(lblHoras);

    cboCategoria = new JComboBox<String>();
    cboCategoria.setModel(new DefaultComboBoxModel<String>(new String[] {
        "C1", "C2", "C3", "C4" }));
    cboCategoria.setBounds(90, 10, 90, 20);
    getContentPane().add(cboCategoria);

    txtHoras = new JTextField();
    txtHoras.setBounds(90, 35, 90, 20);
    getContentPane().add(txtHoras);
    txtHoras.setColumns(10);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(335, 34, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 69, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
}
```

```
    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }

    // Procesa la pulsación del botón Procesar
    protected void actionPerformedBtnProcesar(ActionEvent arg0) {

        // Declaración de variables
        int categoria;
        double horas, suelbru, desc, suelnet, tarifa = 0;

        // Entrada de datos
        categoria = cboCategoria.getSelectedIndex();
        horas = Double.parseDouble(txtHoras.getText());

        // Determina la tarifa horaria
        if (categoria == 0)
            tarifa = 45.0;

        if (categoria == 1)
            tarifa = 37.5;

        if (categoria == 2)
            tarifa = 35.0;

        if (categoria == 3)
            tarifa = 32.5;

        // Calcula el sueldo bruto
        suelbru = horas * tarifa;

        // Calcula el descuento
        desc = 0.15 * suelbru;

        // Calcula el sueldo neto
        suelnet = suelbru - desc;

        // Salida de resultados
        txtS.setText("Tarifa horaria : S/." + tarifa + "\n");
        txtS.append("Sueldo bruto : S/." + suelbru + "\n");
        txtS.append("Descuento : S/." + desc + "\n");
        txtS.append("Sueldo neto : S/." + suelnet);

    }

    // Procesa la pulsación del botón Borrar
    protected void actionPerformedBtnBorrar(ActionEvent arg0) {

        cboCategoria.setSelectedIndex(0);
        txtHoras.setText("");
        txtS.setText("");
        txtHoras.requestFocus();

    }
}
```

## Problema 5

Los ángulos se clasifican de la siguiente manera:

Magnitud	Clasificación
$\beta = 0^\circ$	Nulo
$0^\circ < \beta < 90^\circ$	Agudo
$\beta = 90^\circ$	Recto
$90^\circ < \beta < 180^\circ$	Obtuso
$\beta = 180^\circ$	Llano
$180^\circ < \beta < 360^\circ$	Cóncavo
$\beta = 360^\circ$	Completo

Diseñe un algoritmo que determine la clasificación de un ángulo dado en grados, minutos y segundos.

Asuma que el ángulo está en el intervalo de  $0^\circ$  a  $360^\circ$ .

### Algoritmo

```

Inicio
    // Declaración de variables
    entero grados, minutos, segundos
    real beta
    cadena tipo = ""

    // Entrada de datos
    Ler grados, minutos, segundos

    // Determina el ángulo en grados
    beta = grados + minutos/60.0 + segundos/3600.0

    // Determina el tipo de ángulo
    si( beta == 0 )
        tipo = "Nulo"

    si( beta > 0 && beta < 90 )
        tipo = "Agudo"

    si( beta == 90 )
        tipo = "Recto"

    si( beta > 90 && beta < 180 )
        tipo = "Obtuso"

    si( beta == 180 )
        tipo = "Llano"

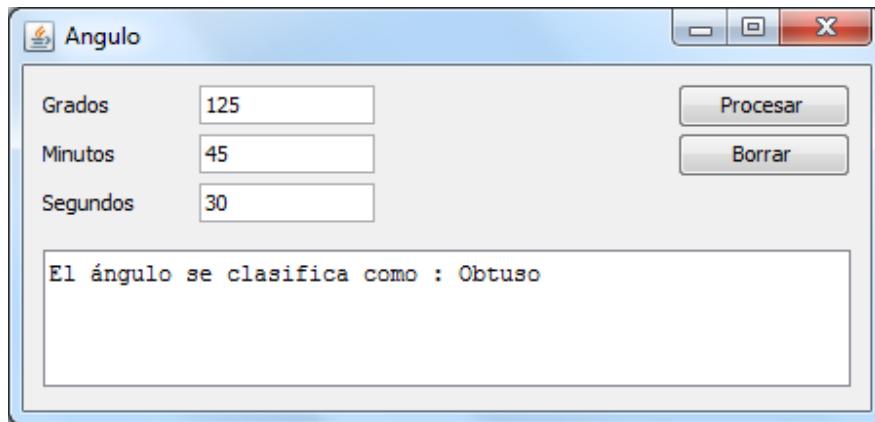
    si( beta > 180 && beta < 360 )
        tipo = "Cóncavo"

    si( beta == 360 )
        tipo = "Completo"

    // Salida de resultados
    Imprimir tipo
Fin

```

## Interfaz Gráfica



## Programa

```

package cibertec;

import java.awt.EventQueue;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import java.awt.Font;

public class Angulo extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblGrados;
    private JLabel lblMinutos;
    private JLabel lblSegundos;
    private JTextField txtGrados;
    private JTextField txtMinutos;
    private JTextField txtSegundos;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Angulo frame = new Angulo();

                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}

```

```
// Crea la GUI
public Angulo() {
    setTitle("Angulo");
    setBounds(100, 100, 450, 214);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblGrados = new JLabel("Grados");
    lblGrados.setBounds(10, 13, 80, 14);
    getContentPane().add(lblGrados);

    lblMinutos = new JLabel("Minutos");
    lblMinutos.setBounds(10, 38, 80, 14);
    getContentPane().add(lblMinutos);

    lblSegundos = new JLabel("Segundos");
    lblSegundos.setBounds(10, 63, 80, 14);
    getContentPane().add(lblSegundos);

    txtGrados = new JTextField();
    txtGrados.setBounds(90, 10, 90, 20);
    getContentPane().add(txtGrados);
    txtGrados.setColumns(10);

    txtMinutos = new JTextField();
    txtMinutos.setBounds(90, 35, 90, 20);
    getContentPane().add(txtMinutos);

    txtSegundos = new JTextField();
    txtSegundos.setBounds(90, 60, 90, 20);
    getContentPane().add(txtSegundos);
    txtSegundos.setColumns(10);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(335, 34, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 94, 414, 70);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declaración de variables
    int grados, minutos, segundos;
    double beta;
    String tipo = "";
}
```

```
// Entrada de datos
grados = Integer.parseInt(txtGrados.getText());
minutos = Integer.parseInt(txtMinutos.getText());
segundos = Integer.parseInt(txtSegundos.getText());

// Determina el ángulo en grados
beta = grados + minutos / 60.0 + segundos / 3600.0;

// Determina el tipo de ángulo
if (beta == 0)
    tipo = "Nulo";

if (beta > 0 && beta < 90)
    tipo = "Agudo";

if (beta == 90)
    tipo = "Recto";

if (beta > 90 && beta < 180)
    tipo = "Obtuso";

if (beta == 180)
    tipo = "Llano";

if (beta > 180 && beta < 360)
    tipo = "Cóncavo";

if (beta == 360)
    tipo = "Completo";

// Salida de resultados
txtS.setText("El ángulo se clasifica como : " + tipo);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtGrados.setText("");
    txtMinutos.setText("");
    txtSegundos.setText("");
    txtS.setText("");
    txtGrados.requestFocus();
}
```

## Problema 6

El promedio final de un curso se obtiene sobre la base del promedio simple de tres prácticas calificadas. Para ayudar a los alumnos, el profesor del curso ha decidido incrementar en dos puntos las notas de las prácticas no menores que 10. Dadas las tres notas de práctica de un estudiante, diseñe un programa que determine el promedio final que le corresponde. Considere que la nota máxima es 20.

### Algoritmo

```

Inicio
    // Declaración de variables
    real p1, p2, p3, promedio

    // Entrada de datos
    Leer p1, p2, p3

    // Si amerita, añade 2 puntos a p1
    si( p1 >= 10 ){
        p1 = p1 + 2

        si( p1 > 20 )
            p1 = 20
    }

    // Si amerita, añade 2 puntos a p2
    si( p2 >= 10 ){
        p2 = p2 + 2

        si( p2 > 20 )
            p2 = 20
    }

    // Si amerita, añade 2 puntos a p3
    si( p3 >= 10 ){
        p3 = p3 + 2

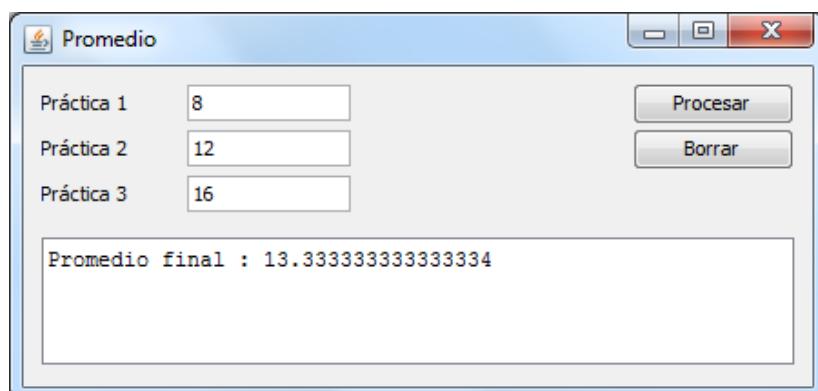
        si( p3 > 20 )
            p3 = 20
    }

    // Cálculo del promedio
    promedio = (p1+p2+p3)/3

    // Salida de resultados
    Imprimir promedio
Fin

```

### Interfaz Gráfica



## Programa

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Promedio extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblPractical;
    private JLabel lblPractica2;
    private JLabel lblPractica3;
    private JTextField txtPractical;
    private JTextField txtPractica2;
    private JTextField txtPractica3;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Promedio frame = new Promedio();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Promedio() {

        setTitle("Promedio");
        setBounds(100, 100, 450, 214);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        lblPractical = new JLabel("Práctica 1");
        lblPractical.setBounds(10, 13, 80, 14);
        getContentPane().add(lblPractical);

        lblPractica2 = new JLabel("Práctica 2");
        lblPractica2.setBounds(10, 38, 80, 14);
        getContentPane().add(lblPractica2);

        lblPractica3 = new JLabel("Práctica 3");
        lblPractica3.setBounds(10, 63, 80, 14);
        getContentPane().add(lblPractica3);
    }
}
```

```
txtPractical = new JTextField();
txtPractical.setBounds(90, 10, 90, 20);
getContentPane().add(txtPractical);
txtPractical.setColumns(10);

txtPractica2 = new JTextField();
txtPractica2.setBounds(90, 35, 90, 20);
getContentPane().add(txtPractica2);
txtPractica2.setColumns(10);

txtPractica3 = new JTextField();
txtPractica3.setBounds(90, 60, 90, 20);
getContentPane().add(txtPractica3);
txtPractica3.setColumns(10);

btnProcesar = new JButton("Procesar");
btnProcesar.addActionListener(this);
btnProcesar.setBounds(335, 9, 89, 23);
getContentPane().add(btnProcesar);

btnBorrar = new JButton("Borrar");
btnBorrar.addActionListener(this);
btnBorrar.setBounds(335, 34, 89, 23);
getContentPane().add(btnBorrar);

scpScroll = new JScrollPane();
scpScroll.setBounds(10, 94, 414, 70);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);

}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declaración de variables
    double promedio;
    int p1, p2, p3;

    // Entrada de datos
    p1 = Integer.parseInt(txtPractical.getText());
    p2 = Integer.parseInt(txtPractica2.getText());
    p3 = Integer.parseInt(txtPractica3.getText());

    // Si amerita, añade 2 puntos a p1
    if (p1 >= 10) {
        p1 = p1 + 2;

        if (p1 > 20)
            p1 = 20;
    }

    // Si amerita, añade 2 puntos a p2
    if (p2 >= 10) {
        p2 = p2 + 2;

        if (p2 > 20)
            p2 = 20;
    }
}
```

```

    // Si amerita, añade 2 puntos a p3
    if (p3 >= 10) {
        p3 = p3 + 2;

        if (p3 > 20)
            p3 = 20;
    }

    // Calcula el promedio
    promedio = (p1 + p2 + p3) / 3.0;

    // Salida de resultados
    txtS.setText("Promedio final : " + promedio);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtPractical.setText("");
    txtPractica2.setText("");
    txtPractica3.setText("");
    txtS.setText("");
    txtPractical.requestFocus();
}
}

```

### Problema 7

Dadas las edades de tres personas, diseñe un programa que determine la edad mayor.

### Algoritmo

```

Inicio
    // Declaración de variables
    entero edad1, edad2, edad3, edadMayor

    // Entrada de datos
    Ler edad1, edad2, edad3

    // Determina la edad mayor
    edadMayor = edad1

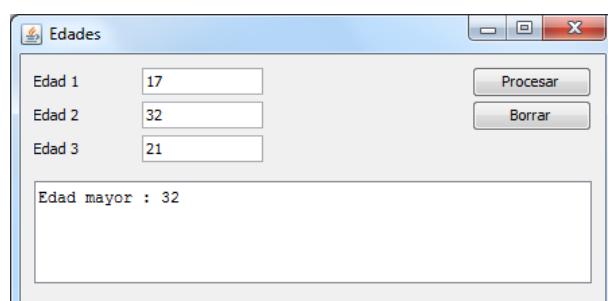
    si (edad2 > edadMayor)
        edadMayor = edad2

    si (edad3 > edadMayor)
        edadMayor = edad3

    // Salida de resultados
    Imprimir edadMayor
Fin

```

### Interfaz Gráfica



## Programa

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Edades extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblEdad1;
    private JLabel lblEdad2;
    private JLabel lblEdad3;
    private JTextField txtEdad1;
    private JTextField txtEdad2;
    private JTextField txtEdad3;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
            } catch (Throwable e) {
                e.printStackTrace();
            }

            EventQueue.invokeLater(new Runnable() {
                public void run() {
                    try {
                        Edades frame = new Edades();
                        frame.setVisible(true);
                    } catch (Exception e) {
                        e.printStackTrace();
                    }
                }
            });
        }
    }

    // Crea la GUI
    public Edades() {

        setTitle("Edades");
        setBounds(100, 100, 450, 222);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        lblEdad1 = new JLabel("Edad 1");
        lblEdad1.setBounds(10, 13, 80, 14);
        getContentPane().add(lblEdad1);

        lblEdad2 = new JLabel("Edad 2");
        lblEdad2.setBounds(10, 38, 80, 14);
        getContentPane().add(lblEdad2);

        lblEdad3 = new JLabel("Edad 3");
        lblEdad3.setBounds(10, 63, 80, 14);
        getContentPane().add(lblEdad3);
    }
}
```

```
txtEdad1 = new JTextField();
txtEdad1.setBounds(90, 10, 90, 20);
getContentPane().add(txtEdad1);
txtEdad1.setColumns(10);

txtEdad2 = new JTextField();
txtEdad2.setBounds(90, 35, 90, 20);
getContentPane().add(txtEdad2);
txtEdad2.setColumns(10);

txtEdad3 = new JTextField();
txtEdad3.setBounds(90, 60, 90, 20);
getContentPane().add(txtEdad3);
txtEdad3.setColumns(10);

btnProcesar = new JButton("Procesar");
btnProcesar.addActionListener(this);
btnProcesar.setBounds(335, 9, 89, 23);
getContentPane().add(btnProcesar);

btnBorrar = new JButton("Borrar");
btnBorrar.addActionListener(this);
btnBorrar.setBounds(335, 34, 89, 23);
getContentPane().add(btnBorrar);

scpScroll = new JScrollPane();
scpScroll.setBounds(10, 94, 414, 76);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {

    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declaración de variables
    int edad1, edad2, edad3, edadMayor;

    // Entrada de datos
    edad1 = Integer.parseInt(txtEdad1.getText());
    edad2 = Integer.parseInt(txtEdad2.getText());
    edad3 = Integer.parseInt(txtEdad3.getText());

    // Determina la edad mayor
    edadMayor = edad1;

    if (edad2 > edadMayor)
        edadMayor = edad2;

    if (edad3 > edadMayor)
        edadMayor = edad3;

    // Salida de resultados
    txtS.setText("Edad mayor : " + edadMayor);
}
}
```

```
// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtEdad1.setText("");
    txtEdad2.setText("");
    txtEdad3.setText("");
    txtS.setText("");
    txtEdad1.requestFocus();
}
```

# Problemas propuestos

## Actividades

1. De un partido de fútbol jugado entre dos equipos A y B se tienen los goles anotados por cada equipo. Diseñe un algoritmo que determine el resultado del partido entre *Ganó A*, *Ganó B* o *Empate*.
  
2. Dado un número decimal, diseñe un algoritmo que determine si el número es *Positivo*, *Negativo* o *Cero*.
  
3. Diseñe un algoritmo que determine la categoría de un estudiante sobre la base de su promedio ponderado, de acuerdo con la siguiente tabla:

Promedio	Categoría
17	A
14 pero < 17	B
12 pero < 14	C
< 12	D

4. Una tienda vende tres tipos de productos P1, P2 y P3 a los precios dados en la siguiente tabla:

Producto	Precio
P1	S/. 17.5
P2	S/. 25.0
P3	S/. 15.5

Como oferta la tienda ofrece un porcentaje de descuento sobre el importe de la compra de acuerdo con la siguiente tabla:

Unidades adquiridas	Descuento
1 a 10	5.0%
11 a 20	7.5%
Más de 21	10.0%

Diseñe un algoritmo que determine el importe de la compra, el importe del descuento y el importe a pagar por la compra de cierta cantidad de unidades de un mismo tipo de producto.

5. Un supermercado vende yogurt en botellas de 1 litro a los precios dados en la siguiente tabla:

Marca	Precio por litro
Buena vida	S/. 3.90
Pura salud	S/. 3.80
Todo sabor	S/. 4.20
Cielo	S/. 3.60

Como oferta, la tienda aplica un porcentaje de descuento sobre el importe de la compra, basándose en la cantidad de botellas adquiridas, de acuerdo con la siguiente tabla:

Cantidad de botellas	Descuento
Menos de 15	4.0%
16 a 30	6.5%
31 a 45	9.0%
Más de 45	11.5%

Diseñe un algoritmo que determine el importe de la compra, el importe del descuento y el importe a pagar por la compra de cierta cantidad de botellas de una misma marca de yogurt.

## Autoevaluación

- Un padre da a su hijo una propina de S/. 10 por cada examen aprobado. Dadas las notas de los exámenes de los cursos de Matemática, Física y Química, diseñe un programa que determine el monto total de la propina. Use únicamente la estructura de selección simple en la solución.
- En una autopista, se multa a los conductores de vehículos que exceden el límite de velocidad permitido de acuerdo con la siguiente tabla.

Velocidad (km/h)	Multa
Hasta 70	Sin sanción
71 a 90	100 euros
91 a 100	140 euros
Más de 100	200 euros

Dada la velocidad de un vehículo, diseñe un programa que determine cuánto de multa deberá pagar el conductor.

- Una tienda vende un producto a un precio unitario que depende del número de unidades adquiridas de acuerdo con la siguiente tabla:

Unidades adquiridas	Precio unitario
1 a 50	S/. 25.5
51 a 100	S/. 22.5
101 a 150	S/. 20.0
151 en adelante	S/. 18.0

Como oferta, la tienda ofrece un descuento igual al 15% del importe de la compra si es que el número de unidades adquiridas es mayor que 50; en caso contrario, sólo descuenta el 5% del importe de la compra.

Dada la cantidad de unidades adquiridas del producto, diseñe un programa que determine el importe de la compra, el importe del descuento y el importe a pagar.

- Un curso se evalúa basándose en cuatro notas de práctica de las cuales se elimina la nota menor y se promedian las tres notas más altas.

Diseñe un programa que determine la nota eliminada y el promedio final de un alumno.

- Diseñe un programa que lea un número entero del intervalo 1 a 7, correspondiente a un día de la semana, y determine el nombre del día.

Considere: 1 para lunes, 2 para martes, 3 para miércoles, 4 para jueves, 5 para viernes, 6 para sábado y 7 para domingo. Si el número no es del intervalo 1 a 7, imprima el mensaje "Día desconocido".

6. ¿Qué imprime el siguiente fragmento de programa?

```
int z;
z = 5;

if(z > 2);
    z = 3;

txtS.append("El valor de z es " + z);
```

7. ¿Qué imprime el siguiente fragmento de programa?

```
int a, b;
a = 8;

if(a < 20)
    b = 1;

if(a < 15)
    b = 2;

if(a < 10)
    b = 3;

if(a < 5)
    b = 4;

txtS.append("El valor de b es " + b);
```

8. ¿Qué imprimen los siguientes fragmentos de programa?

#### Fragmento 1

```
int a, b, c;
b = 5;
c = 1;
a = 10;

if(a > 2)
    b = 3;

c = 2;
a = b+c;

txtS.append("El valor de a es " + a);
```

#### Fragmento 2

```
int a, b, c;
b = 5;
c = 1;
a = 10;

if(a > 2){
    b = 3;
    c = 2;
}

a = b+c;

txtS.append("El valor de a es " + a);
```

# Para recordar

- Colocar un ; al final de la condición de un **if** hace que la acción del **if** sea nula.
- Si el cuerpo de un **if** incluye varias acciones simples, estas deben ir encerradas entre llaves de bloque {}.

## Estructura de selección doble if – else

### Definición

La estructura de selección doble **if...else (si...sino)** evalúa una condición lógica y en caso de que resulte verdadera efectúa la acción **acción a**; de lo contrario, efectúa la acción **acción b**. Tanto **acción a** como **acción b** pueden ser **acciones simples** (una sola acción) o **acciones compuestas** (un bloque de acciones).

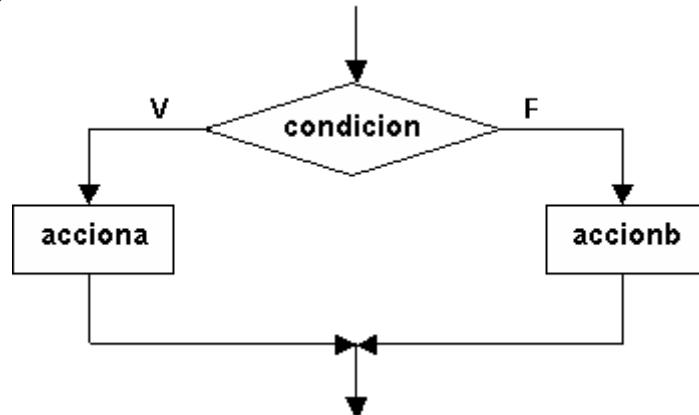


Figura 1 Diagrama de flujo de la estructura de selección if-else

En la tabla que sigue, se muestra el código y el pseudocódigo de la estructura de selección simple if. Note que, en el caso de bloques de acciones (más de una acción), estas deben estar encerradas entre llaves de bloque {}.

Código Java	Pseudocódigo
<pre>if( condicion )     accionA; else     accionB;</pre>	<pre>si( condicion )     accionA sino     accionB</pre>
<pre>if( condicion ){     acciónA1;     acciónA2;     .     .     .     acciónAn; } else {     acciónB1;     acciónB2;     .     .     .     acciónBn; }</pre>	<pre>si( condicion ){     acciónA1     acciónA2     .     .     .     acciónAn } sino {     acciónB1     acciónB2     .     .     .     acciónBn }</pre>

## Problemas propuestos

### Problema 1

Una tienda ha puesto en oferta la venta de un producto ofreciendo un porcentaje de descuento sobre el importe de la compra de acuerdo con la siguiente tabla:

Docenas adquiridas	Descuento
$\geq 10$	15%
$< 10$	11%

Adicionalmente, la tienda obsequia lapiceros de acuerdo con la siguiente tabla:

Importe a pagar	Lapiceros
$\geq 200$	2 por cada docena
$< 200$	0

Dado el precio de la docena y la cantidad de docenas adquiridas, diseñe un programa que determine el importe de la compra, el importe del descuento, el importe a pagar y la cantidad de lapiceros de obsequio.

### Algoritmo

```

Inicio
    // Declaración de variables
    entero cantidad, lapiceros
    real impcom, impdes, imppag, precio

    // Entrada de datos
    Leer cantidad, precio

    // Calcula el importe de la compra
    impcom = cantidad*precio

    // Calcula el importe del descuento
    si(cantidad >= 10)
        impdes = 0.15*impcom
    sino
        impdes = 0.11*impcom

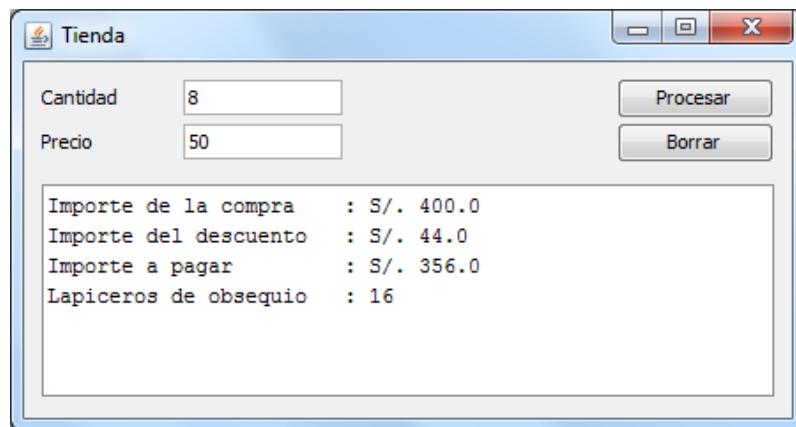
    // Calcula el importe a pagar
    imppag = impcom - impdes

    // Calcula los lapiceros de obsequio
    si(imppag >= 200)
        lapiceros = 2*cantidad
    sino
        lapiceros = 0

    // Salida de resultados
    Imprimir impcom, impdes, imppag, lapiceros
Fin

```

## Interfaz Gráfica



## Programa

```

package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Tienda extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblCantidad;
    private JLabel lblPrecio;
    private JTextField txtCantidad;
    private JTextField txtPrecio;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Tienda frame = new Tienda();
                    frame.setVisible(true);

                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}

```

```
// Crea la GUI
public Tienda() {
    setTitle("Tienda");
    setBounds(100, 100, 450, 239);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblCantidad = new JLabel("Cantidad");
    lblCantidad.setBounds(10, 13, 80, 14);
    getContentPane().add(lblCantidad);

    lblPrecio = new JLabel("Precio");
    lblPrecio.setBounds(10, 38, 80, 14);
    getContentPane().add(lblPrecio);

    txtCantidad = new JTextField();
    txtCantidad.setBounds(90, 10, 90, 20);
    getContentPane().add(txtCantidad);
    txtCantidad.setColumns(10);

    txtPrecio = new JTextField();
    txtPrecio.setBounds(90, 35, 90, 20);
    getContentPane().add(txtPrecio);
    txtPrecio.setColumns(10);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(335, 34, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 69, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {

    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declaración de variables
    int cantidad, lapiceros;
    double impcom, impdes, imppag, precio;

    // Entrada de datos
    cantidad = Integer.parseInt(txtCantidad.getText());
    precio = Double.parseDouble(txtPrecio.getText());

    // Calcula el importe de la compra
    impcom = cantidad * precio;
}
```

```

// Calcula el importe del descuento
if (cantidad >= 10)
    impdes = 0.15 * impcom;
else
    impdes = 0.11 * impcom;

// Calcula el importe a pagar
imppag = impcom - impdes;

// Calcula los lapiceros de obsequio
if (imppag >= 200)
    lapiceros = 2 * cantidad;
else
    lapiceros = 0;

// Salida de resultados
txtS.setText("Importe de la compra      : S/. " + impcom + "\n");
txtS.append("Importe del descuento     : S/. " + impdes + "\n");
txtS.append("Importe a pagar       : S/. " + imppag + "\n");
txtS.append("Lapiceros de obsequio   : " + lapiceros);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtCantidad.setText("");
    txtPrecio.setText("");
    txtS.setText("");
    txtCantidad.requestFocus();
}
}

```

## Problema 2

Una empresa de transportes cubre la ruta Lima-Huánuco en dos turnos: mañana y noche. Los precios de los pasajes se dan en la siguiente tabla:

Turno	Precio del pasaje
Mañana	S/. 37.5
Noche	S/. 45.0

Como oferta especial, la empresa aplica un porcentaje de descuento sobre el importe de la compra de acuerdo con a la siguiente tabla:

Cantidad de pasajes	Descuento
$\geq 15$	8%
$< 15$	5%

Adicionalmente, la empresa obsequia caramelos de acuerdo con la siguiente tabla:

Importe a pagar	Caramelos
$> 200$	2 por cada boleto
$\leq 200$	0

Dado el turno y la cantidad de pasajes adquiridos por un cliente, diseñe un programa que determine el importe de la compra, el importe del descuento, el importe a pagar y la cantidad de caramelos de obsequio.

## Algoritmo

```

Inicio
    // Declaración de variables
    entero turno, cantidad, caramelos
    real impcom, impdes, imppag

    // Entrada de datos
    Leer turno, cantidad

    // Calcula el importe compra
    si (turno == 0)
        impcom = 37.5 * cantidad
    sino
        impcom = 45.0 * cantidad

    // Calcula el importe del descuento
    si (cantidad >= 15)
        impdes = 0.08 * impcom
    sino
        impdes = 0.05 * impcom

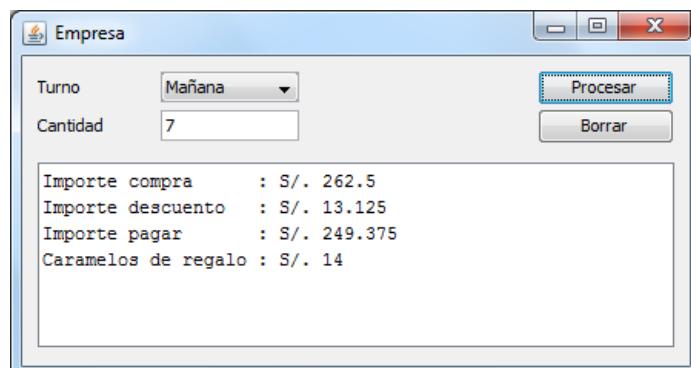
    // Calcula el importe a pagar
    imppag = impcom - impdes

    // Calcula los caramelos de obsequio
    si (imppag > 200)
        caramelos = 2 * cantidad
    sino
        caramelos = 0

    // Salida de resultados
    Imprimir impdes, impdes, imppag, caramelos
Fin

```

## Interfaz Gráfica



## Programa

```

package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JComboBox;
import javax.swing.DefaultComboBoxModel;

```

```
public class Empresa extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblTurno;
    private JLabel lblCantidad;
    private JComboBox<String> cboTurno;
    private JTextField txtCantidad;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Empresa frame = new Empresa();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Empresa() {

        setTitle("Empresa");
        setBounds(100, 100, 450, 239);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        lblTurno = new JLabel("Turno");
        lblTurno.setBounds(10, 13, 80, 14);
        getContentPane().add(lblTurno);

        lblCantidad = new JLabel("Cantidad");
        lblCantidad.setBounds(10, 38, 80, 14);
        getContentPane().add(lblCantidad);

        cboTurno = new JComboBox<String>();
        cboTurno.setModel(new DefaultComboBoxModel<String>(new String[] {"Mañana", "Noche"}));
        cboTurno.setBounds(90, 10, 90, 20);
        getContentPane().add(cboTurno);

        txtCantidad = new JTextField();
        txtCantidad.setBounds(90, 35, 90, 20);
        getContentPane().add(txtCantidad);
        txtCantidad.setColumns(10);

        btnProcesar = new JButton("Procesar");
        btnProcesar.addActionListener(this);
        btnProcesar.setBounds(335, 9, 89, 23);
        getContentPane().add(btnProcesar);

        btnBorrar = new JButton("Borrar");
        btnBorrar.addActionListener(this);
        btnBorrar.setBounds(335, 34, 89, 23);
        getContentPane().add(btnBorrar);
    }
}
```

```
scpScroll = new JScrollPane();
scpScroll.setBounds(10, 69, 414, 120);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declaración de variables
    int turno, cantidad, caramelos;
    double impcom, impdes, imppag;

    // Entrada de datos
    turno = cboTurno.getSelectedIndex();
    cantidad = Integer.parseInt(txtCantidad.getText());

    // Calcula el importe de la compra
    if (turno == 0)
        impcom = 37.5 * cantidad;
    else
        impcom = 45.0 * cantidad;

    // Calcula el importe del descuento
    if (cantidad >= 15)
        impdes = 0.08 * impcom;
    else
        impdes = 0.05 * impcom;

    // Calcula el importe a pagar
    imppag = impcom - impdes;

    // Calcula los caramelos de obsequio
    if (imppag > 200)
        caramelos = 2 * cantidad;
    else
        caramelos = 0;

    // Salida de resultados
    txtS.setText("Importe compra : S/. " + impcom + "\n");
    txtS.append("Importe descuento : S/. " + impdes + "\n");
    txtS.append("Importe pagar: S/. " + imppag + "\n");
    txtS.append("Caramelos de regalo : S/. " + caramelos);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtCantidad.setText("");
    txtS.setText("");
    txtCantidad.requestFocus();
}
```

### Problema 3

El sueldo bruto de los empleados de una empresa se calcula sumando el sueldo básico más la bonificación por hijos.

El sueldo básico se calcula multiplicando las horas trabajadas por la tarifa horaria. La tarifa horaria depende de la categoría del empleado de acuerdo con la siguiente tabla:

Categoría	Tarifa horaria (S/.)
A	45.0
B	37.5

La bonificación por hijos se calcula de acuerdo con la siguiente tabla:

Número de hijos	Bonificación
Hasta 3	S/. 40.5 por cada hijo
Más de 3	S/. 35.0 por cada hijo

Por ley, todo empleado está sujeto a un porcentaje de descuento sobre el sueldo bruto de acuerdo con la siguiente tabla:

Sueldo bruto (S/.)	Descuento
$\geq 3500$	13.5%
$< 3500$	10.0%

Dadas la categoría y la cantidad de horas trabajadas de un empleado, diseñe un programa que determine el sueldo básico, el sueldo bruto, el descuento y el sueldo neto que le corresponden.

### Algoritmo

```

Inicio

    // Declaración de variables
    entero hij, cat
    real bonif, suelbas, suelbru, desc, suelnet, hor

    // Entrada de datos
    Ler hor, cat, hij

    // Calcula el sueldo bruto
    si (cat == 0 )
        suelbas = hor * 45.0
    sino
        suelbas = hor * 37.5

    // Calcula la bonificación por hijos
    si (hij <= 3)
        bonif = 40.5 * hij
    sino
        bonif = 35.0 * hij

    // Calcula el sueldo bruto
    suelbru = suelbas + bonif

```

```

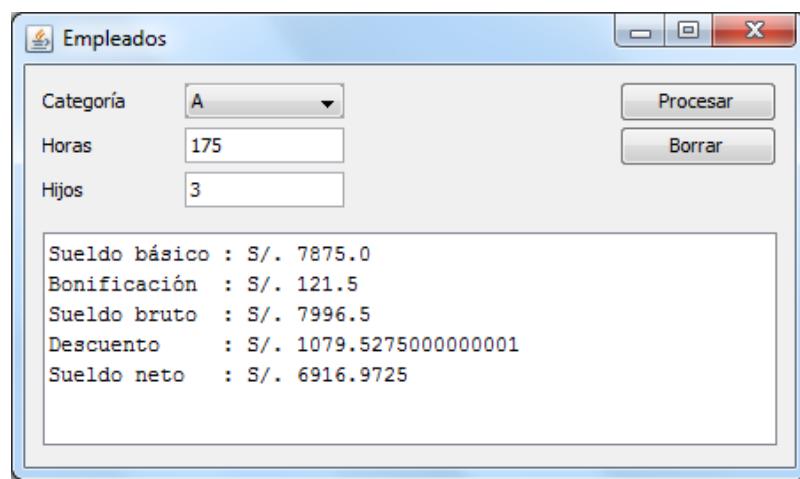
// Calcula el descuento
si (suelbru >= 3500)
    desc = 0.135 * suelbru
sino
    desc = 0.10 * suelbru

// Calcula el sueldo neto
suelnet = suelbru - desc

// Salida de resultados
Imprimir suelbas, bonif, suelbru, desc, suelnet
Fin

```

## Interfaz Gráfica



## Programa

```

package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JComboBox;
import javax.swing.DefaultComboBoxModel;

public class Empleados extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblCategoria;
    private JLabel lblHoras;
    private JLabel lblHijos;
    private JComboBox<String> cboCategoria;
    private JTextField txtHoras;
    private JTextField txtHijos;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

```

```
// Lanza la aplicación
public static void main(String[] args) {
    try {
        UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
    } catch (Throwable e) {
        e.printStackTrace();
    }

    EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                Empleados frame = new Empleados();
                frame.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

// Crea la GUI
public Empleados() {

    setTitle("Empleados");
    setBounds(100, 100, 450, 264);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblCategoria = new JLabel("Categoría");
    lblCategoria.setBounds(10, 13, 80, 14);
    getContentPane().add(lblCategoria);

    lblHoras = new JLabel("Horas");
    lblHoras.setBounds(10, 38, 80, 14);
    getContentPane().add(lblHoras);

    lblHijos = new JLabel("Hijos");
    lblHijos.setBounds(10, 63, 80, 14);
    getContentPane().add(lblHijos);

    cboCategoria = new JComboBox<String>();
    cboCategoria.setModel(
        new DefaultComboBoxModel<String>(new String[] { "A", "B" }));
    cboCategoria.setBounds(90, 10, 90, 20);
    getContentPane().add(cboCategoria);

    txtHoras = new JTextField();
    txtHoras.setBounds(90, 35, 90, 20);
    getContentPane().add(txtHoras);
    txtHoras.setColumns(10);

    txtHijos = new JTextField();
    txtHijos.setBounds(90, 60, 90, 20);
    getContentPane().add(txtHijos);
    txtHijos.setColumns(10);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(335, 34, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 94, 414, 120);
    getContentPane().add(scpScroll);
}
```

```
txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declaración de variables
    int hij, cat;
    double bonif, suelbas, suelbru, desc, suelnet, hor;

    // Entrada de datos
    hor = Double.parseDouble(txtHoras.getText());
    cat = cboCategoria.getSelectedIndex();
    hij = Integer.parseInt(txtHijos.getText());

    // Calcula el sueldo básico
    if (cat == 0)
        suelbas = hor * 45.0;
    else
        suelbas = hor * 37.5;

    // Calcula la bonificación por hijos
    if (hij <= 3)
        bonif = 40.5 * hij;
    else
        bonif = 35.0 * hij;

    // Calcula el sueldo bruto
    suelbru = suelbas + bonif;

    // Calcula el descuento
    if (suelbru >= 3500)
        desc = 0.135 * suelbru;
    else
        desc = 0.10 * suelbru;

    // Calcula el sueldo neto
    suelnet = suelbru - desc;

    // Salida de resultados
    txtS.setText("Sueldo básico : S/. " + suelbas + "\n");
    txtS.append("Bonificación : S/. " + bonif + "\n");
    txtS.append("Sueldo bruto : S/. " + suelbru + "\n");
    txtS.append("Descuento : S/. " + desc + "\n");
    txtS.append("Sueldo neto : S/. " + suelnet);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    cboCategoria.setSelectedIndex(0);
    txtHoras.setText("");
    txtHijos.setText("");
    txtS.setText("");
    txtHoras.requestFocus();
}
}
```

## Problema 4

Los cálculos salariales de los vendedores de una empresa se calculan de la siguiente manera:

- Sueldo básico: S/.600
- Comisión: 7% del importe total vendido si es que el importe total vendido es mayor a S/.15000; en caso contrario, 5% del importe total vendido
- Bonificación: S/.25 por cada hijo si es que el número de hijos es menor a 5; en caso contrario, S/.22 por cada hijo
- Sueldo bruto: La suma del sueldo básico, más la comisión y más la bonificación
- Descuento: 15% del sueldo bruto si es que el sueldo bruto es mayor que S/.3500; en caso contrario, 11% del sueldo bruto
- Sueldo neto: La resta del sueldo bruto menos el descuento

Dado el importe total vendido y el número de hijos de un vendedor, diseñe un programa que determine el sueldo básico, la comisión, la bonificación, el sueldo bruto, el descuento y el sueldo neto.

## Algoritmo

```

Inicio
    // Declaración de variables
    real impven, suelbas, comi, bonif, suelbru, desc, suelnet
    entero hijos

    // Entrada de datos
    Leer impven, hijos

    // Asigna el sueldo básico
    suelbas = 600

    // Calcula la comisión
    si (impven > 15000)
        comi = 0.07 * impven
    sino
        comi = 0.05 * impven

    // Calcula la bonificación
    si (hijos < 5)
        bonif= 25 * hijos
    sino
        bonif= 22 * hijos

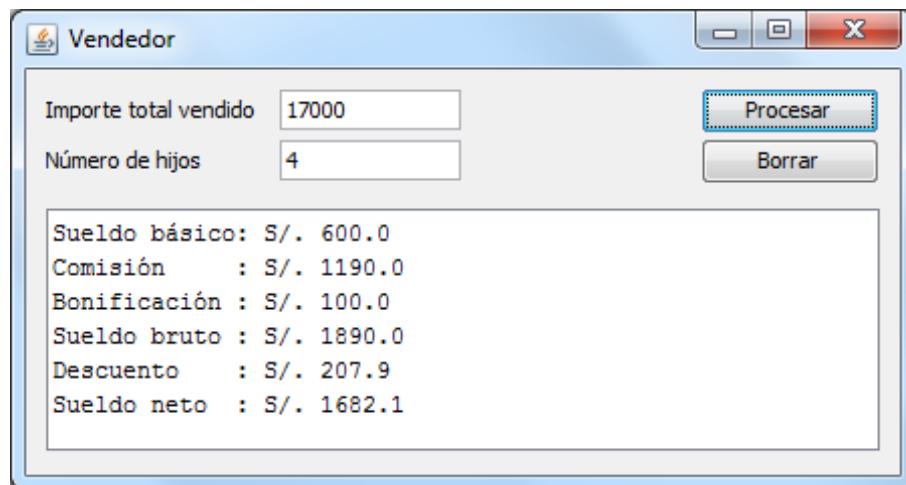
    // Calcula el sueldo bruto
    suelbru = suelbas + comi + bonif

    // Calcula el descuento
    si (suelbru > 3500)
        desc = 0.15 * suelbru
    sino
        desc = 0.11 * suelbru

    // Calcula el sueldo neto
    suelnet = suelbru - desc

    // Salida de resultados
    Imprimir suelbas, comi, bonif, suelbru, desc, suelnet
Fin
```

## Interfaz Gráfica



## Programa

```

package cibertec;

import java.awt.Font;
import java.awt.EventQueue;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Vendedor extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblImporte;
    private JLabel lblHijos;
    private JTextField txtImporte;
    private JTextField txtHijos;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Vendedor frame = new Vendedor();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}

```

```
// Crea la GUI
public Vendedor() {
    setTitle("Vendedor");
    setBounds(100, 100, 450, 239);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);

    lblImporte = new JLabel("Importe total vendido");
    lblImporte.setBounds(10, 13, 115, 14);
    getContentPane().add(lblImporte);

    txtImporte = new JTextField();
    txtImporte.setBounds(126, 10, 90, 20);
    getContentPane().add(txtImporte);

    txtImporte.setColumns(10);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);

    lblHijos = new JLabel("Número de hijos");
    lblHijos.setBounds(10, 38, 115, 14);
    getContentPane().add(lblHijos);

    txtHijos = new JTextField();
    txtHijos.setBounds(126, 35, 90, 20);
    getContentPane().add(txtHijos);

    txtHijos.setColumns(10);
    btnBorrar.setBounds(335, 34, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 69, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declaración de variables
    double impven, suelbas, comi, bonif, suelbru, desc, suelnet;
    int hijos;

    // Entrada de datos
    impven = Double.parseDouble(txtImporte.getText());
    hijos = Integer.parseInt(txtHijos.getText());

    // Asignación del sueldo básico
    suelbas = 600;
```

```
// Calcula la comisión
if (impven > 15000)
    comi = 0.07 * impven;
else
    comi = 0.05 * impven;

// Calcula la bonificación
if (hijos < 5)
    bonif = 25 * hijos;
else
    bonif = 22 * hijos;

// Calcula el sueldo bruto
suelbru = suelbas + comi + bonif;

// Calcula el descuento
if (suelbru > 3500)
    desc = 0.15 * suelbru;
else
    desc = 0.11 * suelbru;

// Calculo el sueldo neto
suelnet = suelbru - desc;

// Salida de resultados
txtS.setText("Sueldo básico: S/. " + suelbas + "\n");
txtS.append("Comisión : S/. " + comi + "\n");
txtS.append("Bonificación : S/. " + bonif + "\n");
txtS.append("Sueldo bruto : S/. " + suelbru + "\n");
txtS.append("Descuento : S/. " + desc + "\n");
txtS.append("Sueldo neto : S/. " + suelnet);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtImporte.setText("");
    txtHijos.setText("");
    txtS.setText("");
    txtImporte.requestFocus();
}
}
```

### Problema 5

Una institución benéfica recibe anualmente una donación y lo reparte entre un centro de salud, un comedor de niños y una parte lo invierte en la bolsa de acuerdo con lo siguiente:

- Si el monto de la donación es de \$10000 o más: 30% se destina al centro de salud, 50% al comedor de niños y el resto se invierte en la bolsa.
- Si el monto de la donación es menor de \$10000: 25% se destina al centro de salud, 60% al comedor de niños y el resto se invierte en la bolsa.

Dado el monto de la donación, diseñe un programa que determine el monto de dinero que recibirá cada rubro.

### Algoritmo

```

Inicio
    // Declaración de variables
    real donacion, comedor, salud, bolsa

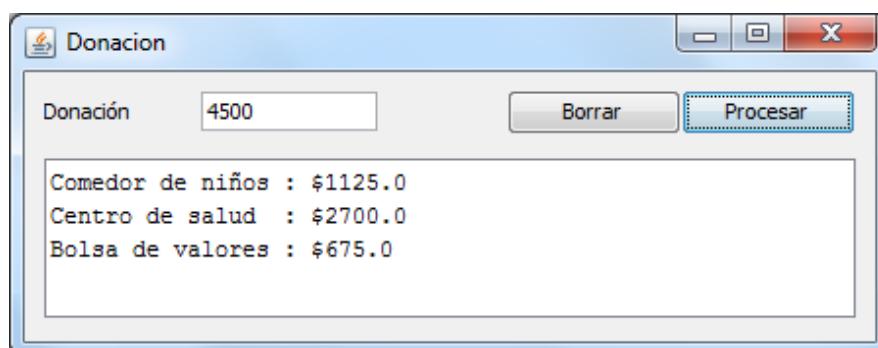
    // Entrada de datos
    Ler donacion

    // Reparte la donación
    si( donacion >= 10000 ){
        comedor = 0.30*donacion
        salud = 0.50*donacion
        bolsa = 0.20*donacion
    } sino {
        comedor = 0.25*donacion
        salud = 0.60*donacion
        bolsa = 0.15*donacion
    }

    // Salida de resultados
    Imprimir comedor, salud, bolsa
Fin

```

### Interfaz Gráfica



## Programa

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;

public class Donacion extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblDonacion;
    private JTextField txtDonacion;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Donacion frame = new Donacion();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Donacion() {
        setTitle("Donacion");
        setBounds(100, 100, 450, 214);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        lblDonacion = new JLabel("Donación");
        lblDonacion.setBounds(10, 13, 80, 14);
        getContentPane().add(lblDonacion);

        txtDonacion = new JTextField();
        txtDonacion.setBounds(90, 10, 90, 20);
        getContentPane().add(txtDonacion);

        btnProcesar = new JButton("Procesar");
        btnProcesar.addActionListener(this);
        btnProcesar.setBounds(335, 9, 89, 23);
        getContentPane().add(btnProcesar);

        btnBorrar = new JButton("Borrar");
        btnBorrar.addActionListener(this);
        btnBorrar.setBounds(246, 9, 89, 23);
        getContentPane().add(btnBorrar);
    }
}
```

```
scpScroll = new JScrollPane();
scpScroll.setBounds(10, 44, 414, 120);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declaración de variables
    double donacion, comedor, salud, bolsa;

    // Entrada de datos
    donacion = Double.parseDouble(txtDonacion.getText());

    // Reparte la donación
    if (donacion >= 10000) {
        comedor = 0.30 * donacion;
        salud = 0.50 * donacion;
        bolsa = 0.20 * donacion;
    } else {
        comedor = 0.25 * donacion;
        salud = 0.60 * donacion;
        bolsa = 0.15 * donacion;
    }

    // Salida de resultados
    txtS.setText("Comedor de niños : $" + comedor + "\n");
    txtS.append("Centro de salud      : $" + salud + "\n");
    txtS.append("Bolsa de valores : $" + bolsa);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtDonacion.setText("");
    txtS.setText("");
    txtDonacion.requestFocus();
}
}
```

### Comentario

Note el uso de las llaves {} en el bloque **if** y en el bloque **else**, dado que cada bloque tiene más de una instrucción. En caso de que el **if** o el **else** tuviera una sola instrucción, el uso de llaves es opcional.

## Problema 6

En una oficina de empleos categorizan a los postulantes en función del sexo y de la edad de acuerdo con lo siguiente:

- Si la persona es de sexo femenino: categoría FA si tiene menos de 23 años y FB, en caso contrario.
- Si la persona es de sexo masculino: categoría MA si tiene menos de 25 años y MB, en caso contrario.

Dado el sexo y la edad de un postulante, diseñe un programa que determine su categoría.

## Algoritmo

```

Inicio
    // Declaración de variables
    entero sexo, edad
    cadena categoria

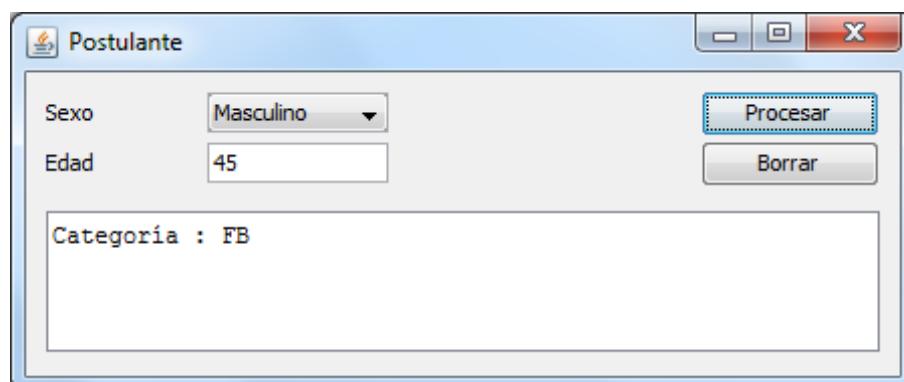
    // Entrada de datos
    Leer sexo, edad

    // Determina la categoría
    si( sexo == 0 ){
        si( edad < 23 )
            categoria = "FA"
        sino
            categoria = "FB"
    } sino {
        si( edad < 25 )
            categoria = "MA"
        sino
            categoria = "MB"
    }

    // Salida de resultados
    Imprimir categoria
Fin

```

## Interfaz Gráfica



## Programa

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JComboBox;
import javax.swing.DefaultComboBoxModel;

public class Postulante extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblSexo;
    private JLabel lblEdad;
    private JComboBox<String> cboSexo;
    private JTextField txtEdad;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Postulante frame = new Postulante();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Postulante() {
        setTitle("Postulante");
        setBounds(100, 100, 450, 239);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        lblSexo = new JLabel("Sexo");
        lblSexo.setBounds(10, 13, 80, 14);
        getContentPane().add(lblSexo);

        lblEdad = new JLabel("Edad");
        lblEdad.setBounds(10, 38, 80, 14);
        getContentPane().add(lblEdad);

        cboSexo = new JComboBox<String>();
        cboSexo.setModel(
            new DefaultComboBoxModel<String>(new String[] { "Masculino", "Femenino" }));
        cboSexo.setBounds(90, 10, 90, 20);
        getContentPane().add(cboSexo);
    }
}
```

```
txtEdad = new JTextField();
txtEdad.setBounds(90, 35, 90, 20);
getContentPane().add(txtEdad);
txtEdad.setColumns(10);

btnProcesar = new JButton("Procesar");
btnProcesar.addActionListener(this);
btnProcesar.setBounds(335, 9, 89, 23);
getContentPane().add(btnProcesar);

btnBorrar = new JButton("Borrar");
btnBorrar.addActionListener(this);
btnBorrar.setBounds(335, 34, 89, 23);
getContentPane().add(btnBorrar);

scpScroll = new JScrollPane();
scpScroll.setBounds(10, 69, 414, 120);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {

    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declaración de variables
    int sexo, edad;
    String categoria;

    // Entrada de datos
    sexo = cboSexo.getSelectedIndex();
    edad = Integer.parseInt(txtEdad.getText());

    // Determina la categoría
    if (sexo == 0) {
        if (edad < 23)
            categoria = "FA";
        else
            categoria = "FB";
    } else {
        if (edad < 25)
            categoria = "MA";
        else
            categoria = "MB";
    }

    // Salida de resultados
    txtS.setText("Categoría : " + categoria);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtEdad.setText("");
    txtS.setText("");
    txtEdad.requestFocus();
}
}
```

### Problema 7

Una papelera ha puesto en oferta la venta de papel bond en paquetes de medio millar de acuerdo con los siguientes criterios:

- Para compras menores o iguales a 20 paquetes, se paga el precio normal.
- Para compras mayores de 20 paquetes, por los primeros 20 paquetes se paga el precio normal; pero, por los paquetes que exceden de 20, sólo se paga el 85% del precio normal.

Adicionalmente, para compras de más de 50 paquetes, el cliente recibe dos paquetes adicionales.

Dado el precio normal del paquete y la cantidad de paquetes adquiridos, diseñe un programa que determine el importe a pagar y la cantidad total de paquetes que recibirá el cliente.

### Algoritmo

```

Inicio
    // Declaración de variables
    real imppag, precio
    entero cantidad, cantot

    // Entrada de datos
    Leer precio, cantidad

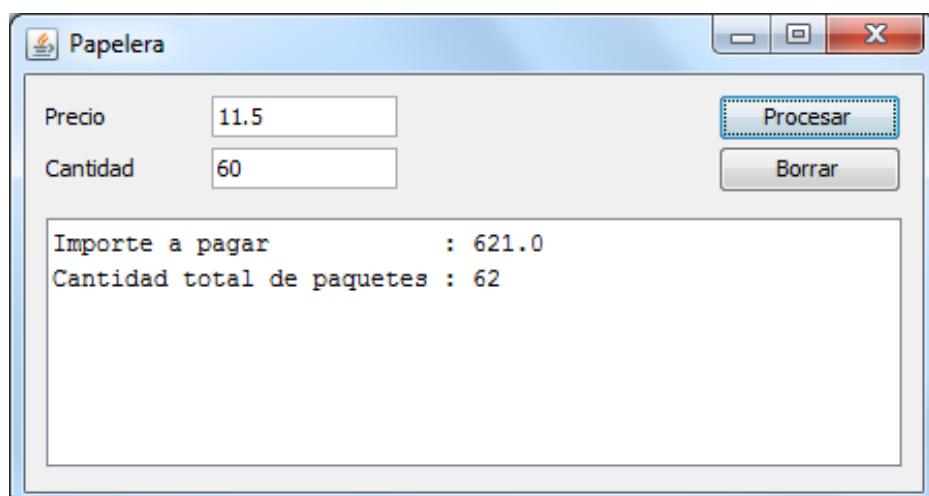
    // Determina el importe a pagar
    si(cantidad <= 20)
        imppag = precio*cantidad
    sino
        imppag = precio*20 + 0.85*precio*(cantidad-20)

    // Determina la cantidad total de paquetes
    si(cantidad > 50)
        cantot = cantidad + 2
    sino
        cantot = cantidad

    // Salida de resultados
    Imprimir imppag, cantot
Fin

```

### Interfaz Gráfica



## Programa

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Papelera extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblPrecio;
    private JLabel lblCantidad;
    private JTextField txtPrecio;
    private JTextField txtCantidad;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Papelera frame = new Papelera();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Papelera() {

        setTitle("Papelera");
        setBounds(100, 100, 450, 239);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        lblPrecio = new JLabel("Precio");
        lblPrecio.setBounds(10, 13, 80, 14);
        getContentPane().add(lblPrecio);

        lblCantidad = new JLabel("Cantidad");
        lblCantidad.setBounds(10, 38, 80, 14);
        getContentPane().add(lblCantidad);

        txtPrecio = new JTextField();
        txtPrecio.setBounds(90, 10, 90, 20);
        getContentPane().add(txtPrecio);
        txtPrecio.setColumns(10);
```

```
txtCantidad = new JTextField();
txtCantidad.setBounds(90, 35, 90, 20);
getContentPane().add(txtCantidad);
txtCantidad.setColumns(10);

btnProcesar = new JButton("Procesar");
btnProcesar.addActionListener(this);
btnProcesar.setBounds(335, 9, 89, 23);
getContentPane().add(btnProcesar);

btnBorrar = new JButton("Borrar");
btnBorrar.addActionListener(this);
btnBorrar.setBounds(335, 34, 89, 23);
getContentPane().add(btnBorrar);

scpScroll = new JScrollPane();
scpScroll.setBounds(10, 69, 414, 120);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declaración de variables
    double imppag, precio;
    int cantidad, canttot;

    // Entrada de datos
    precio = Double.parseDouble(txtPrecio.getText());
    cantidad = Integer.parseInt(txtCantidad.getText());

    // Determina el importe a pagar
    if (cantidad <= 20)
        imppag = precio * cantidad;
    else
        imppag = precio * 20 + 0.85 * precio * (cantidad - 20);

    // Determina la cantidad total de paquetes
    if (cantidad > 50)
        canttot = cantidad + 2;
    else
        canttot = cantidad;

    // Salida de resultados
    txtS.setText("Importe a pagar : " + imppag + "\n");
    txtS.append("Cantidad total de paquetes : " + canttot);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtPrecio.setText("");
    txtCantidad.setText("");
    txtS.setText("");
    txtPrecio.requestFocus();
}
```

# Problemas propuestos

## Actividad

1. Una tienda ha puesto en oferta la venta de un producto. Si el cliente adquiere más de 10 unidades del producto, la tienda aplica un descuento igual al 15% del importe de la compra; en caso contrario, solo descuenta el 5% del importe compra. Como incentivo adicional, la tienda ha decidido obsequiar una *agenda* a los clientes cuyo importe a pagar es mayor de S/. 200; en caso contrario, obsequiará un cuaderno. Dado el precio del producto y la cantidad de unidades adquiridas, diseñe un algoritmo que determine el importe de la compra, el importe del descuento, el importe a pagar y el obsequio.
  
2. Una empresa paga a sus empleados por horas trabajadas. El sueldo bruto se calcula multiplicando las horas trabajadas por la tarifa horaria del empleado. Por ley, todo empleado está sujeto a un descuento del 15% del sueldo bruto si es que el sueldo bruto es mayor de S/. 3500; en caso contrario, solo se descuenta el 11% del sueldo bruto. El sueldo neto se calcula restando el sueldo bruto menos el importe del descuento. Dado el número de horas trabajadas y la tarifa horaria de un empleado, diseñe un algoritmo que determine el sueldo bruto, el descuento y el sueldo neto del empleado.
  
3. Un curso se evalúa sobre la base de tres prácticas calificadas P<sub>1</sub>, P<sub>2</sub> y P<sub>3</sub>. El promedio final del curso *Pf* se obtiene con la siguiente fórmula:

$$PF = 0.20 \times P1 + 0.35 \times P2 + 0.45 \times P3$$

Para aprobar el curso se requiere obtener un promedio final mínimo de 13.0. Dadas las tres notas de práctica de un estudiante, diseñe un algoritmo que determine el promedio final y la condición de aprobado o desaprobado.

## Autoevaluación

1. En un supermercado hay una promoción según la cual el cliente raspa una tarjeta que contiene un número oculto y en base al número de la tarjeta se obtiene un descuento. Si el número de la tarjeta es mayor de 100, el cliente obtiene un descuento del 15% sobre el importe de la compra; en caso contrario, obtiene un descuento del 5% del importe compra.

Dado el número oculto de la tarjeta y el importe de la compra, diseñe un algoritmo que determine el importe del descuento y el importe a pagar.

2. Una empresa de bienes raíces ofrece casas de interés social bajo las siguientes condiciones: si el ingreso mensual del comprador es menos de \$1250, la cuota inicial será igual al 15% del costo de la casa y el resto se distribuirá en 120 cuotas mensuales; pero, si el ingreso mensual del comprador es mayor o igual a \$1250, la cuota inicial será igual al 30% del costo de la casa y el resto se distribuirá en 75 cuotas mensuales.

Dado el ingreso mensual y el costo de la casa, diseñe un algoritmo que determine la cuota inicial y la cuota mensual.

3. Una empresa paga a sus vendedores un sueldo bruto igual a la suma de un sueldo básico más una comisión por ventas. El sueldo básico es S/. 300. La comisión por ventas es 15% del monto total vendido. Por otro lado, si el sueldo bruto del vendedor es mayor de S/. 1800, recibe un descuento del 15% del sueldo bruto; en caso contrario, recibe un descuento del 11% del sueldo bruto. Además, como incentivo, la empresa obsequia 5 polos si es que el monto vendido es mayor de S/. 1500; en caso contrario, obsequia 2 polos.

Dado el monto total vendido, diseñe un programa que imprima un reporte mostrando el sueldo básico, la comisión, el sueldo bruto, el descuento, el sueldo neto y el número de polos de obsequio.

4. Una empresa calcula el sueldo bruto de sus trabajadores en base a las horas trabajadas. Hasta 48 horas, se paga una tarifa horaria normal. Para las horas en exceso sobre 48, se paga un recargo del 15% respecto a la tarifa horaria normal. Por otro lado, si el sueldo bruto es superior a S/. 3500, se aplica un descuento del 15% del sueldo bruto; en caso contrario, se aplica un descuento del 11% del sueldo bruto.

Dadas las horas trabajadas y la tarifa horaria normal, diseñe un programa que determine el sueldo bruto, el descuento y el sueldo neto de un trabajador.

5. Una empresa ha decidido otorgar una bonificación por fiestas patrias a sus empleados. Si el empleado tiene más de dos hijos, recibirá una bonificación igual al 12.5% de su sueldo bruto más S/. 30 por cada hijo; en caso contrario, recibirá una bonificación igual al 12.5% de su sueldo bruto más S/. 40 por cada hijo.

Dado el sueldo bruto y el número de hijos de un empleado, diseñe un programa que determine la bonificación por fiestas patrias que le corresponde.

6. Un padre ha decidido dar una propina a su hijo sobre la base de sus notas en los cursos de Matemáticas, Física e Historia del Perú.

- Si la nota de Matemática es no menor de 17, le dará S/. 3.0 de propina por cada punto de la nota; en caso contrario, sólo le dará S/. 1.0 por cada punto
- Si la nota de Física es no menor de 15, le dará S/. 2.0 de propina por cada punto de la nota; en caso contrario, sólo le dará S/.0.5 por cada punto
- Si la nota de Historia del Perú es no menor de 15, le dará S/. 1.5 por cada punto de la nota; en caso contrario, sólo le dará S/. 0.3 por cada punto

Dadas las notas de los cursos de Matemática, Física e Historia del Perú, diseñe un algoritmo que determine el monto total de la propina.

7. Una empresa ha decidido adquirir varias piezas a una fábrica de refacciones. La empresa, dependiendo del monto total de la compra, decidirá qué hacer para pagar al fabricante.

Si el monto total de la compra excede de \$50000, la empresa pedirá prestado al banco el 30% e invertirá el resto de su propio dinero; en caso contrario, pedirá prestado al banco el 20% e invertirá el resto de su propio dinero.

Dado el monto total de la compra, diseñe un programa que determine cuánto tendrá que pagar la empresa de su propio dinero y cuánto deberá pedir prestado al banco.

8. Corrija los errores de los siguientes fragmentos de programa:

a. `if(a > 20);  
b = 10;  
else  
b = 8;`

b. `if(a+b = 5)  
c = a;  
d = c;  
else  
c = 0;`

c. `if( a > 30 ){  
if(a < 70){  
b = 5;  
c = 2;  
else  
b = 1;  
} else {  
b = 0;  
}`

d. `if( a > 10 && a <= 20)  
b = 5  
else;  
b = 2;`

9. Determine qué imprimen los siguientes fragmentos de programa cuando **a** tiene **9** y **b** tiene **11**, y cuando **a** tiene **11** y **b** tiene **9**.

a. if(a < 10)  
if(b > 10)  
txtS.append("\*\*\*\*\*\n");  
else txtS.append("#####\n");  
txtS.append("@@@@\n");

b. if(a < 10) {  
if(b > 10)  
txtS.append("\*\*\*\*\*\n");  
} else {  
txtS.append("#####\n");  
txtS.append("@@@@\n");  
}

10. Modifique el código que sigue para producir la salida mostrada. No puede hacer ningún cambio con excepción de la inserción de llaves. Use sangrías (márgenes) para darle claridad al código, pero recuerde que las sangrías son opcionales.

```
if( a == 8 )  
if( b == 5 )  
txtS.append("@@@@\n");  
else  
txtS.append("####\n");  
txtS.append("$$$$\n");  
txtS.append("&&&\n");
```

- a. Para **a** igual a **5** y **b** igual a **8**, debe mostrarse la siguiente salida

@@@  
\$\$\$  
&&&

- b. Para **a** igual a **5** y **b** igual a **8**, debe mostrarse la siguiente salida  
@@@

- c. Para **a** igual a **5** y **b** igual a **8**, debe mostrarse la siguiente salida  
@@  
@@  
&&&  
&

- d. Para **a** igual a **5** y **b** igual a **7**, debe mostrarse la siguiente salida

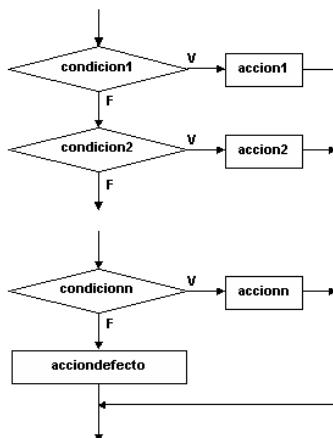
###  
\$\$\$  
&&&

# Para recordar

- Si el cuerpo del **if** o el cuerpo del **else** incluyen más de una acción, estas deben ir encerradas entre llaves de bloque **{ }**.

### Estructura de selección doble encadenada if-else-if

La estructura de selección doble encadenada **if...else...if** evalúa un conjunto de condiciones en orden descendente, pasando de una condición a otra siempre que la condición anterior sea falsa, y en el momento que encuentre una condición verdadera, efectúa la acción correspondiente a dicha condición y abandona el resto de la estructura. La estructura tiene una acción por defecto que se efectúa en el caso que todas las condiciones sean falsas.



**Figura 1** Estructura de selección doble encadenada

En las tablas que siguen, se muestran el código Java y el pseudocódigo correspondiente a los dos formatos de escritura de la estructura if-else-if.

## Estructura de Selección if – else –if (Formato 1)

Código Java	Pseudocódigo
<pre> if( condicion1 )     accion1; else     if( condicion2 )         accion2;     else         if( condicion3 )             accion3;         .         .         else             if( condicionn )                 accionN;             else                 accionDefecto; </pre>	<pre> si( condicion1 )     accion1; sino     si( condicion2 )         accion2;     sino         si( condicion3 )             accion3;         .         .         sino             si( condicionn )                 accionN;             sino                 accionDefecto; </pre>

## Estructura de Selección if – else –if (Formato 2)

Código Java	Pseudocódigo
<pre> <b>if</b>( condicion1 )     accion1; <b>else if</b>( condicion2 )     accion2; <b>else if</b>( condicion3 )     accion3; . . <b>else if</b>( condicionn )     accionN; <b>else</b>     accionDefecto; </pre>	<pre> <b>si</b>( condicion1 )     accion1; <b>sino si</b>( condicion2 )     accion2; <b>sino si</b>( condicion3 )     accion3; . . <b>sino si</b>( condicionn )     accionN; <b>sino</b>     accionDefecto; </pre>

En el caso de acciones compuestas, estas deben estar encerradas entre llaves de bloque {}.

## Problemas resueltos

### Ejemplo 1

Los ángulos se clasifican de la siguiente manera:

Magnitud	Clasificación
$\beta = 0^\circ$	Nulo
$0^\circ < \beta < 90^\circ$	Agudo
$\beta = 90^\circ$	Recto
$90^\circ < \beta < 180^\circ$	Obtuso
$\beta = 180^\circ$	Llano
$180^\circ < \beta < 360^\circ$	Cóncavo
$\beta = 360^\circ$	Completo

Dada la medida de un ángulo en grados, minutos y segundos, diseñe un algoritmo que determine la clasificación del ángulo. Asuma que el ángulo está en el intervalo de  $0^\circ$  a  $360^\circ$ .

### Algoritmo

```

Inicio
    // Declaración de variables
    entero grados, minutos, segundos
    real beta
    cadena tipo

    // Entrada de datos
    Leer grados, minutos, segundos

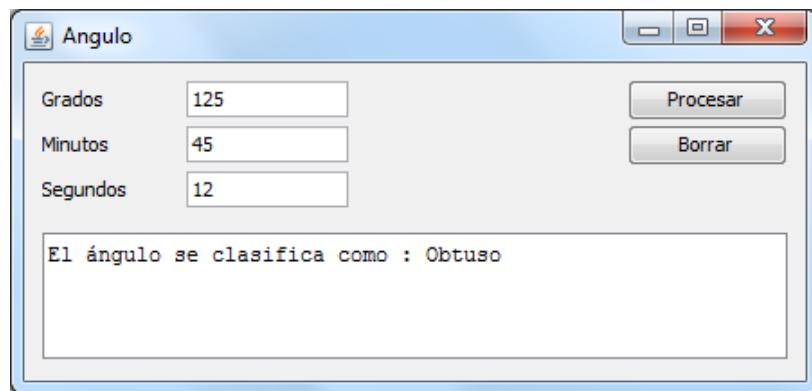
    // Determina el ángulo en grados
    beta = grados + minutos/60.0 + segundos/3600.0

    // Determina el tipo de ángulo
    si( beta == 0 )
        tipo = "Nulo"
    sino si(beta < 90 )
        tipo = "Agudo"
    sino si( beta == 90 )
        tipo = "Recto"
    sino si( beta < 180 )
        tipo = "Obtuso"
    sino si( beta == 180 )
        tipo = "Llano"
    sino si( beta < 360 )
        tipo = "Cóncavo"
    sino
        tipo = "Completo"

    // Salida de resultados
    Imprimir tipo
Fin

```

## Interfaz Gráfica



## Programa

```

package cibertec;

import java.awt.EventQueue;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import java.awt.Font;

public class Angulo extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblGrados;
    private JLabel lblMinutos;
    private JLabel lblSegundos;
    private JTextField txtGrados;
    private JTextField txtMinutos;
    private JTextField txtSegundos;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Angulo frame = new Angulo();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}

```

```
// Crea la GUI
public Angulo() {
    setTitle("Angulo");
    setBounds(100, 100, 450, 214);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblGrados = new JLabel("Grados");
    lblGrados.setBounds(10, 13, 80, 14);
    getContentPane().add(lblGrados);

    lblMinutos = new JLabel("Minutos");
    lblMinutos.setBounds(10, 38, 80, 14);
    getContentPane().add(lblMinutos);

    lblSegundos = new JLabel("Segundos");
    lblSegundos.setBounds(10, 63, 80, 14);
    getContentPane().add(lblSegundos);

    txtGrados = new JTextField();
    txtGrados.setBounds(90, 10, 90, 20);
    getContentPane().add(txtGrados);
    txtGrados.setColumns(10);

    txtMinutos = new JTextField();
    txtMinutos.setBounds(90, 35, 90, 20);
    getContentPane().add(txtMinutos);
    txtMinutos.setColumns(10);

    txtSegundos = new JTextField();
    txtSegundos.setBounds(90, 60, 90, 20);
    getContentPane().add(txtSegundos);
    txtSegundos.setColumns(10);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(335, 34, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 94, 414, 70);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declaración de variables
    int grados, minutos, segundos;
    double beta;
    String tipo;
```

```
// Entrada de datos
grados = Integer.parseInt(txtGrados.getText());
minutos = Integer.parseInt(txtMinutos.getText());
segundos = Integer.parseInt(txtSegundos.getText());

// Determina el ángulo en grados
beta = grados + minutos / 60.0 + segundos / 3600.0;

// Determina el tipo de ángulo
if (beta == 0)
    tipo = "Nulo";
else if (beta < 90)
    tipo = "Agudo";
else if (beta == 90)
    tipo = "Recto";
else if (beta < 180)
    tipo = "Obtuso";
else if (beta == 180)
    tipo = "Llano";
else if (beta < 360)
    tipo = "Cóncavo";
else
    tipo = "Completo";

// Salida de resultados
txtS.setText("El ángulo se clasifica como : " + tipo);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtGrados.setText("");
    txtMinutos.setText("");
    txtSegundos.setText("");
    txtS.setText("");
    txtGrados.requestFocus();
}
```

**Problema 2**

Resuelva el problema 1 verificando que los datos ingresados sean correctos. Para el efecto, se requiere que la cantidad de grados sea de 0 a 360 y que la cantidad de minutos y de segundos sean de 0 a 59.

**Solución 1: Validación usando la estructura if – else – if**

```
// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declaración de variables
    int grados, minutos, segundos;
    double beta;
    String tipo = "";

    // Entrada de datos
    grados = Integer.parseInt(txtGrados.getText());
    minutos = Integer.parseInt(txtMinutos.getText());
    segundos = Integer.parseInt(txtSegundos.getText());

    // Verifica los grados
    if (grados < 0 || grados > 360) {
        JOptionPane.showMessageDialog(this, "Grados debe ser de 0 a 360");
        txtGrados.requestFocus();
        txtGrados.selectAll();
    }

    // Verifica los minutos
    else if (minutos < 0 || minutos > 59) {
        JOptionPane.showMessageDialog(this, "Minutos debe ser de 0 a 59");
        txtMinutos.requestFocus();
        txtMinutos.selectAll();
    }

    // Verifica los segundos
    else if (segundos < 0 || segundos > 59) {
        JOptionPane.showMessageDialog(this, "Segundos debe ser de 0 a 59");
        txtSegundos.requestFocus();
        txtSegundos.selectAll();
    }

    // Continua con datos válidos
    else {
        // Determina el ángulo en grados
        beta = grados + minutos / 60.0 + segundos / 3600.0;

        // Determina el tipo de ángulo
        if (beta == 0)
            tipo = "Nulo";
        else if (beta < 90)
            tipo = "Agudo";
        else if (beta == 90)
            tipo = "Recto";
        else if (beta < 180)
            tipo = "Obtuso";
        else if (beta == 180)
            tipo = "Llano";
        else if (beta < 360)
            tipo = "Cóncavo";
        else
            tipo = "Completo";

        // Salida de resultados
        txtS.setText("El ángulo se clasifica como : " + tipo);
    }
}
```

**Solución 2: Validación usando la estructura if y return**

```
// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declaración de variables
    int grados, minutos, segundos;
    double beta;
    String tipo;

    // Entrada de datos
    grados = Integer.parseInt(txtGrados.getText());
    minutos = Integer.parseInt(txtMinutos.getText());
    segundos = Integer.parseInt(txtSegundos.getText());

    // Verifica los grados
    if (grados < 0 || grados > 360) {
        JOptionPane.showMessageDialog(this, "Grados debe ser de 0 a 360");
        txtGrados.requestFocus();
        txtGrados.selectAll();
        return; // Salir si grados es incorrecto
    }

    // Verifica los minutos
    if (minutos < 0 || minutos > 59) {
        JOptionPane.showMessageDialog(this, "Minutos debe ser de 0 a 59");
        txtMinutos.requestFocus();
        txtMinutos.selectAll();
        return; // Salir si minutos es incorrecto
    }

    // Verifica los segundos
    if (segundos < 0 || segundos > 59) {
        JOptionPane.showMessageDialog(this, "Segundos debe ser de 0 a 59");
        txtSegundos.requestFocus();
        txtSegundos.selectAll();
        return; // Salir si segundos es incorrecto
    }

    // Determina el ángulo en grados
    beta = grados + minutos / 60.0 + segundos / 3600.0;

    // Determina el tipo de ángulo
    if (beta == 0)
        tipo = "Nulo";
    else if (beta < 90)
        tipo = "Agudo";
    else if (beta == 90)
        tipo = "Recto";
    else if (beta < 180)
        tipo = "Obtuso";
    else if (beta == 180)
        tipo = "Llano";
    else if (beta < 360)
        tipo = "Cóncavo";
    else
        tipo = "Completo";

    // Salida de resultados
    txtS.setText("El ángulo se clasifica como : " + tipo);
}
```

### Problema 3

En una universidad, los alumnos están clasificados en cuatro categorías. A cada categoría le corresponde una pensión mensual distinta dada en la siguiente tabla:

Categoría	Pensión
A	S/. 550
B	S/. 500
C	S/. 460
D	S/. 400

Semestralmente, la universidad efectúa rebajas en las pensiones de sus estudiantes a partir del segundo ciclo basándose en el promedio ponderado del ciclo anterior en porcentajes dados en la tabla siguiente:

Promedio	Descuento
0 a 13.99	No hay descuento
14.00 a 15.99	10 %
16.00 a 17.99	12 %
18.00 a 20.00	15 %

Dado el promedio ponderado y la categoría de un estudiante, diseñe un programa que determine cuánto de rebaja recibirá sobre su pensión actual y a cuánto asciende su nueva pensión.

### Algoritmo

```

Inicio
    // Declaración de variables
    entero categoria
    real actualpen, nuevapen, descuento, promedio

    // Entrada de datos
    Leer categoria, promedio

    // Calcula la pensión actual
    si( categoria == 0 )
        actualpen = 550
    sino si ( categoria == 1 )
        actualpen = 500
    sino si ( categoria == 2 )
        actualpen = 460
    sino
        actualpen = 400

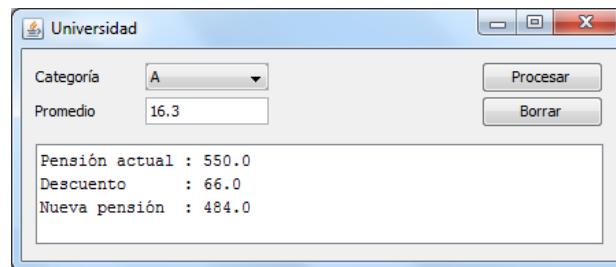
    // Calcula el descuento
    si( promedio <= 13.99 )
        descuento = 0
    sino si( promedio <= 15.99 )
        descuento = 0.10*actualpen
    sino si( promedio <= 17.99 )
        descuento = 0.12*actualpen
    sino
        descuento = 0.15*actualpen

    // Calcula la nueva pensión
    nuevapen = actualpen - descuento

```

```
// Salida de resultados
Imprimir actualpen, nuevapen
Fin
```

## Interfaz Gráfica



## Programa

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JComboBox;
import javax.swing.DefaultComboBoxModel;

public class Universidad extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblCategoria;
    private JLabel lblPromedio;
    private JComboBox<String> cboCategoria;
    private JTextField txtPromedio;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Universidad frame = new Universidad();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}
```

```
// Crea la GUI
public Universidad() {

    setTitle("Universidad");
    setBounds(100, 100, 450, 239);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblCategoria = new JLabel("Categoría");
    lblCategoria.setBounds(10, 13, 80, 14);
    getContentPane().add(lblCategoria);

    lblPromedio = new JLabel("Promedio");
    lblPromedio.setBounds(10, 38, 80, 14);
    getContentPane().add(lblPromedio);

    cboCategoria = new JComboBox<String>();
    cboCategoria.setModel(
        new DefaultComboBoxModel<String>(new String[] { "A", "B", "C", "D" }));
    cboCategoria.setBounds(90, 10, 90, 20);
    getContentPane().add(cboCategoria);

    txtPromedio = new JTextField();
    txtPromedio.setBounds(90, 35, 90, 20);
    getContentPane().add(txtPromedio);
    txtPromedio.setColumns(10);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(335, 34, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 69, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);

}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {

    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }

}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declaración de variables
    int categoria;
    double actualpen, nuevapen, descuento, promedio;

    // Entrada de datos
    categoria = cboCategoria.getSelectedIndex();
    promedio = Double.parseDouble(txtPromedio.getText());
}
```

```
// Calcula la pensión actual
if (categoria == 0)
    actualpen = 550;
else if (categoria == 1)
    actualpen = 500;

else if (categoria == 2)
    actualpen = 460;
else
    actualpen = 400;

// Calcula el descuento
if (promedio <= 13.99)
    descuento = 0;
else if (promedio <= 15.99)
    descuento = 0.10 * actualpen;
else if (promedio <= 17.99)
    descuento = 0.12 * actualpen;
else
    descuento = 0.15 * actualpen;

// Calcula la nueva pensión
nuevapen = actualpen - descuento;

// Salida de resultados
txtS.setText("Pensión actual : " + actualpen + "\n");
txtS.append("Descuento : " + descuento + "\n");
txtS.append("Nueva pensión : " + nuevapen);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtPromedio.setText("");
    txtS.setText("");
    txtPromedio.requestFocus();
}
```

### Problema 4

Una empresa de préstamos tiene el siguiente esquema de cobros:

Monto del préstamo (S.)	Número de cuotas
Hasta 5000	2
Más de 5000 hasta 10000	4
Más de 10000 hasta 15000	6
Más de 15000	10

Si el monto del préstamo es mayor de S/. 10000, la empresa cobra 3% de interés mensual; en caso contrario, cobra 5% de interés mensual. Dado el monto del préstamo de un cliente, diseñe un programa que determine el número de cuotas, el monto de la cuota mensual y el monto del interés total entre todas las cuotas.

### Algoritmo

```

Inicio
    // Declaración de variables
    real montoprestamo, montointeres, tasainterres, montocuota
    entero cuotas

    // Entrada de datos
    Leer montoprestamo

    // Obtiene el número de cuotas
    si( montoprestamo <= 5000 )
        cuotas = 2
    sino si( montoprestamo <= 10000 )
        cuotas = 4
    sino si( montoprestamo <= 15000 )
        cuotas = 6
    sino
        cuotas = 10

    // Obtiene la tasa de interés
    si( montoprestamo > 10000 )
        tasainterres = 0.03
    sino
        tasainterres = 0.05

    // Calcula el monto del interés total
    montointeres = tasainterres*montoprestamo*cuotas

    // Calcula el monto de la cuota
    montocuota = (montoprestamo+ montointeres) /cuotas

    // Salida de resultados
    Imprimir cuotas, montocuota, montointeres
Fin

```

## Interfaz Gráfica



## Programa

```

package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Prestamo extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblMontoPrestamo;
    private JTextField txtMontoPrestamo;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Prestamo frame = new Prestamo();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Prestamo() {
        setTitle("Prestamo");
        setBounds(100, 100, 450, 214);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);
    }
}

```

```
btnProcesar = new JButton("Procesar");
btnProcesar.addActionListener(this);
btnProcesar.setBounds(335, 9, 89, 23);
getContentPane().add(btnProcesar);

lblMontoPrestamo = new JLabel("Monto del préstamo");
lblMontoPrestamo.setBounds(10, 13, 115, 14);
getContentPane().add(lblMontoPrestamo);

txtMontoPrestamo = new JTextField();
txtMontoPrestamo.setBounds(125, 10, 90, 20);
getContentPane().add(txtMontoPrestamo);
txtMontoPrestamo.setColumns(10);

btnBorrar = new JButton("Borrar");
btnBorrar.addActionListener(this);
btnBorrar.setBounds(246, 9, 89, 23);
getContentPane().add(btnBorrar);

scpScroll = new JScrollPane();
scpScroll.setBounds(10, 44, 414, 120);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declaración de variables
    double montoprestamo, montointeres, tasainterres, montocuota;
    int cuotas;

    // Entrada de datos
    montoprestamo = Double.parseDouble(txtMontoPrestamo.getText());

    // Obtiene el número de cuotas
    if (montoprestamo <= 5000)
        cuotas = 2;
    else if (montoprestamo <= 10000)
        cuotas = 4;
    else if (montoprestamo <= 15000)
        cuotas = 6;
    else
        cuotas = 10;

    // Obtiene la tasa de interés
    if (montoprestamo > 10000)
        tasainterres = 0.03;
    else
        tasainterres = 0.05;

    // Calcula el monto del interés total
    montointeres = tasainterres * montoprestamo * cuotas;

    // Calcula el monto de la cuota
    montocuota = (montoprestamo + montointeres) / cuotas;
}
```

```

    // Salida de resultados
    txtS.setText("Número de cuotas : " + cuotas + "\n");
    txtS.append("Cuota mensual: S/. " + montocuota + "\n");
    txtS.append("Interés total: S/. " + montointeres);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtMontoPrestamo.setText("");
    txtS.setText("");
    txtMontoPrestamo.requestFocus();
}
}

```

### Problema 5

Una empresa evalúa a sus empleados bajo dos criterios: puntualidad y rendimiento. En cada caso, el empleado recibe un puntaje que va de 1 a 10, de acuerdo con los siguientes criterios:

**Puntaje por puntualidad:**- está en función de los minutos de tardanza de acuerdo con la siguiente tabla:

Minutos de tardanza	Puntaje
0	10
1 a 2	8
3 a 5	6
6 a 9	4
Más de 9	0

**Puntaje por rendimiento:**- está en función de la cantidad de observaciones efectuadas al empleado por no cumplir sus obligaciones de acuerdo con la siguiente tabla:

Observaciones efectuadas	Puntaje
0	10
1	8
2	5
3	1
Más de 3	0

El puntaje total del empleado es la suma del puntaje por puntualidad más el puntaje por rendimiento. Basándose en el puntaje total, el empleado recibe una bonificación anual de acuerdo con la siguiente tabla:

Puntaje total	Bonificación
Menos de 11	S/. 2.5 por punto
11 a 13	S/. 5.0 por punto
14 a 16	S/. 7.5 por punto
17 a 19	S/. 10.0 por punto
20	S/. 12.5 por punto

Dados los minutos de tardanza y el número de observaciones de un empleado, diseñe un programa que determine el puntaje por puntualidad, el puntaje por rendimiento, el puntaje total y la bonificación que le corresponden.

## Algoritmo

```

Inicio
    // Declaración de variables
    entero minutosTar, numeroObs, puntajePun, puntajeRen, puntajeTot
    real bonificacion

    // Entrada de datos
    Ler minutosTar, numeroObs

    // Determina el puntaje por puntualidad
    si(minutosTar == 0)
        puntajePun = 10
    sino si(minutosTar <= 2)
        puntajePun = 8
    sino si(minutosTar <= 5)
        puntajePun = 6
    sino si(minutosTar <= 9)
        puntajePun = 4
    sino
        puntajePun = 0

    // Determina el puntaje por rendimiento
    si(numeroObs == 0)
        puntajeRen = 10
    sino si(numeroObs == 1)
        puntajeRen = 8
    sino si(numeroObs == 2)
        puntajeRen = 5
    sino si(numeroObs == 3)
        puntajeRen = 1
    sino
        puntajeRen = 0

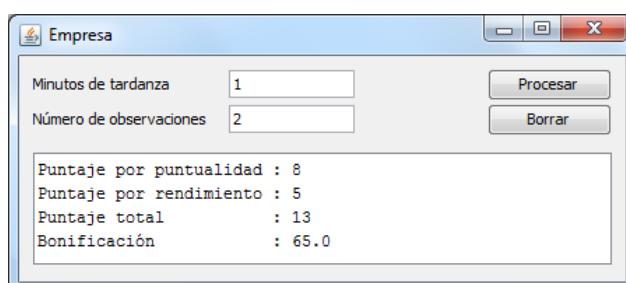
    // Determina el puntaje total
    puntajeTot = puntajePun + puntajeRen

    // Determina la bonificación
    si(puntajeTot < 11)
        bonificacion = 2.5*puntajeTot
    sino si(puntajeTot <= 13)
        bonificacion = 5.0*puntajeTot
    sino si(puntajeTot <= 16)
        bonificacion = 7.5*puntajeTot
    sino si(puntajeTot <= 19)
        bonificacion = 10.0*puntajeTot
    sino
        bonificacion = 12.5*puntajeTot

    // Salida de resultados
    Imprimir puntajePun, puntajeRen, puntajeTot, bonificacion
Fin

```

## Interfaz Gráfica



## Programa

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Empresa extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblMinutosTardanza;
    private JLabel lblNumeroObservaciones;
    private JTextField txtMinutosTardanza;
    private JTextField txtNumeroObservaciones;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Empresa frame = new Empresa();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Empresa() {

        setTitle("Empresa");
        setBounds(100, 100, 450, 239);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        lblNumeroObservaciones = new JLabel("Número de observaciones");
        lblNumeroObservaciones.setBounds(10, 38, 140, 14);
        getContentPane().add(lblNumeroObservaciones);

        lblMinutosTardanza = new JLabel("Minutos de tardanza");
        lblMinutosTardanza.setBounds(10, 13, 140, 14);
        getContentPane().add(lblMinutosTardanza);

        txtMinutosTardanza = new JTextField();
        txtMinutosTardanza.setBounds(150, 10, 90, 20);
        getContentPane().add(txtMinutosTardanza);
        txtMinutosTardanza.setColumns(10);
    }
}
```

```
txtNumeroObservaciones = new JTextField();
txtNumeroObservaciones.setBounds(150, 35, 90, 20);
getContentPane().add(txtNumeroObservaciones);
txtNumeroObservaciones.setColumns(10);

btnProcesar = new JButton("Procesar");
btnProcesar.addActionListener(this);
btnProcesar.setBounds(335, 9, 89, 23);
getContentPane().add(btnProcesar);

btnBorrar = new JButton("Borrar");
btnBorrar.addActionListener(this);
btnBorrar.setBounds(335, 34, 89, 23);
getContentPane().add(btnBorrar);

scpScroll = new JScrollPane();
scpScroll.setBounds(10, 69, 414, 120);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declaración de variables
    int minutosTar, numeroObs, puntajePun, puntajeRen, puntajeTot;
    double bonificacion;

    // Entrada de datos
    minutosTar = Integer.parseInt(txtMinutosTardanza.getText());
    numeroObs = Integer.parseInt(txtNumeroObservaciones.getText());

    // Determina el puntaje por puntuabilidad
    if (minutosTar == 0)
        puntajePun = 10;
    else if (minutosTar <= 2)
        puntajePun = 8;
    else if (minutosTar <= 5)
        puntajePun = 6;
    else if (minutosTar <= 9)
        puntajePun = 4;
    else
        puntajePun = 0;

    // Determina el puntaje por rendimiento
    if (numeroObs == 0)
        puntajeRen = 10;
    else if (numeroObs == 1)
        puntajeRen = 8;
    else if (numeroObs == 2)
        puntajeRen = 5;
    else if (numeroObs == 3)
        puntajeRen = 1;
    else
        puntajeRen = 0;

    // Determina el puntaje total
    puntajeTot = puntajePun + puntajeRen;
```

```

// Determina la bonificación
if (puntajeTot < 11)
    bonificacion = 2.5 * puntajeTot;
else if (puntajeTot <= 13)
    bonificacion = 5.0 * puntajeTot;
else if (puntajeTot <= 16)
    bonificacion = 7.5 * puntajeTot;
else if (puntajeTot <= 19)
    bonificacion = 10.0 * puntajeTot;
else
    bonificacion = 12.5 * puntajeTot;

// Salida de resultados
txtS.setText("Puntaje por puntuabilidad : " + puntajePun + "\n");
txtS.append("Puntaje por rendimiento : " + puntajeRen + "\n");
txtS.append("Puntaje total: " + puntajeTot + "\n");
txtS.append("Bonificación : " + bonificacion);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtMinutosTardanza.setText("");
    txtNumeroObservaciones.setText("");
    txtS.setText("");
    txtMinutosTardanza.requestFocus();
}
}

```

### Problema 6

Una dulcería vende chocolates a los precios dados en la siguiente tabla:

Tipo de chocolate	Precio unitario
Primor	S/. 8.5
Dulzura	S/. 10.0
Tentación	S/. 7.0
Explosión	S/. 12.5

Como oferta, la tienda aplica un porcentaje de descuento sobre el importe de la compra, basándose en la cantidad de chocolates adquiridos, de acuerdo con la siguiente tabla:

Cantidad de chocolates	Descuento
< 5	4.0%
≥ 5 y < 10	6.5%
≥ 10 y < 15	9.0%
≥ 15	11.5%

Adicionalmente, si el importe a pagar es no menor de S/. 250, la tienda obsequia 3 caramelos por cada chocolate; en caso contrario, obsequia 2 caramelos por cada chocolate.

Dado el tipo de chocolate y la cantidad de unidades adquiridas, diseñe un programa que determine el importe de la compra, el importe del descuento, el importe a pagar y la cantidad de caramelos de obsequio.

## Algoritmo

```

Inicio
    // Declaración de variables
    entero tipo, cantidad, caramelos
    real impcom, impdes, imppag

    // Entrada de datos
    Leer tipo, cantidad

    // Calcula el importe de la compra
    si(tipo == 0)
        impcom = 8.5*cantidad
    sino si(tipo == 1)
        impcom = 10.0*cantidad
    sino si(tipo == 2)
        impcom = 7.0*cantidad
    sino
        impcom = 12.5*cantidad

    // Calcula el importe del descuento
    si(cantidad < 5)
        impdes = 0.04*impcom
    sino si(cantidad < 10)
        impdes = 0.065*impcom
    sino si(cantidad < 15)
        impdes = 0.09*impcom
    sino
        impdes = 0.115*impcom

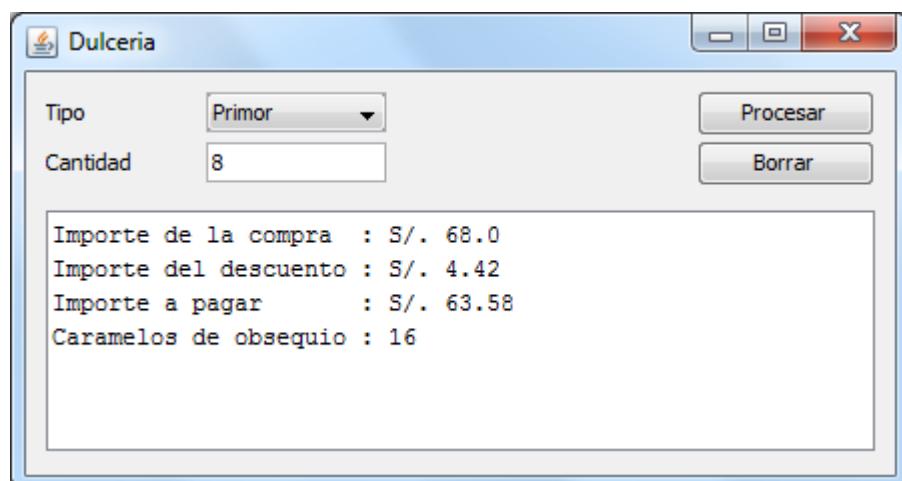
    // Calcula el importe a pagar
    imppag = impcom - impdes

    // Calcula la cantidad de caramelos de regalo
    si(imppag < 250)
        caramelos = 2*cantidad
    sino
        caramelos = 3*cantidad

    // Salida de resultados
    Imprimir impcom, impdes, imppag, caramelos
Fin

```

## Interfaz Gráfica



## Programa

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JComboBox;
import javax.swing.DefaultComboBoxModel;

public class Dulceria extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblTipo;
    private JLabel lblCantidad;
    private JComboBox<String> cboTipo;
    private JTextField txtCantidad;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Dulceria frame = new Dulceria();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Dulceria() {
        setTitle("Dulceria");
        setBounds(100, 100, 450, 239);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        lblTipo = new JLabel("Tipo");
        lblTipo.setBounds(10, 13, 80, 14);
        getContentPane().add(lblTipo);

        lblCantidad = new JLabel("Cantidad");
        lblCantidad.setBounds(10, 38, 80, 14);
        getContentPane().add(lblCantidad);

        cboTipo = new JComboBox<String>();
        cboTipo.setModel(new DefaultComboBoxModel<String>(new String[] {
        "Primor", "Dulzura", "Tentación", "Explosión" }));
        cboTipo.setBounds(90, 10, 90, 20);
        getContentPane().add(cboTipo);
    }
}
```

```
txtCantidad = new JTextField();
txtCantidad.setBounds(90, 35, 90, 20);
getContentPane().add(txtCantidad);
txtCantidad.setColumns(10);

btnProcesar = new JButton("Procesar");
btnProcesar.addActionListener(this);
btnProcesar.setBounds(335, 9, 89, 23);
getContentPane().add(btnProcesar);

btnBorrar = new JButton("Borrar");
btnBorrar.addActionListener(this);

btnBorrar.setBounds(335, 34, 89, 23);
getContentPane().add(btnBorrar);

scpScroll = new JScrollPane();
scpScroll.setBounds(10, 69, 414, 120);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {

    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declaración de variables
    int tipo, cantidad, caramelos;
    double impcom, impdes, imppag;

    // Entrada de datos
    tipo = cboTipo.getSelectedIndex();
    cantidad = Integer.parseInt(txtCantidad.getText());

    // Calcula el importe de la compra
    if (tipo == 0)
        impcom = 8.5 * cantidad;
    else if (tipo == 1)
        impcom = 10.0 * cantidad;
    else if (tipo == 2)
        impcom = 7.0 * cantidad;
    else
        impcom = 12.5 * cantidad;

    // Calcula el importe del descuento
    if (cantidad < 5)
        impdes = 0.04 * impcom;
    else if (cantidad < 10)
        impdes = 0.065 * impcom;
    else if (cantidad < 15)
        impdes = 0.09 * impcom;
    else
        impdes = 0.115 * impcom;

    // Calcula el importe a pagar
    imppag = impcom - impdes;
}
```

```

// Calcula la cantidad de caramelos de regalo
if (imppag < 250)
    caramelos = 2 * cantidad;
else
    caramelos = 3 * cantidad;

// Salida de resultados
txtS.setText("Importe de la compra : S/. " + impcom + "\n");
txtS.append("Importe del descuento : S/. " + impdes + "\n");
txtS.append("Importe a pagar : S/. " + imppag + "\n");
txtS.append("Caramelos de obsequio : " + caramelos);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    cboTipo.setSelectedIndex(0);
    txtCantidad.setText("");
    txtS.setText("");
    txtCantidad.requestFocus();
}
}

```

### Problema 7

Una web clasifica a sus usuarios basándose en la cantidad de mensajes publicados en los foros de acuerdo con la siguiente tabla:

Número de mensajes	Clase de usuario
0 a 49	Desconocido
50 a 99	Humano
100 a 249	Diclonius Inicial
250 a 499	Diclonius Novato
500 a 999	Diclonius Experimentado
1000 a 1999	Diclonius Elite
2000 a 4999	Diclonius Definitivo
5000 a más	Diclonius Legendario

Dado el número de mensajes publicados por un usuario, diseñe un programa que determine la clasificación que le corresponde.

### Algoritmo

```

Inicio
    // Declaración de variables
    entero num
    cadena clase

    // Entrada de datos
    Leer num

    // Determina la clase del usuario
    si (num < 50)
        clase = "Desconocido"
    sino si (num < 100)
        clase = "Humano"
    sino si (num < 250)
        clase = "Diclonius Inicial"

```

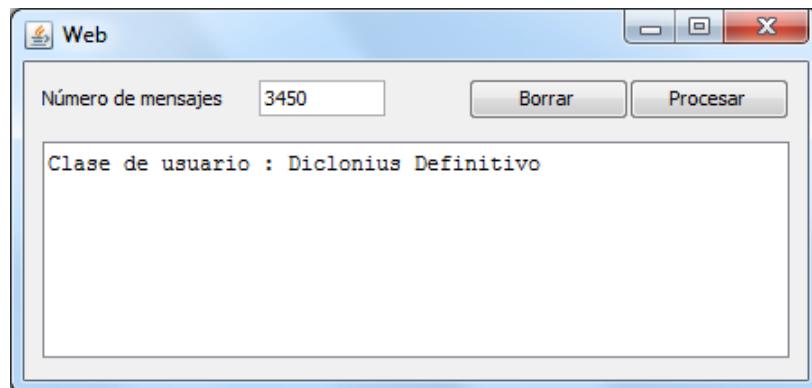
```

    sino si (num < 500)
        clase = "Diclonius Novato"
    sino si (num < 1000)
        clase = "Diclonius Experimentado"
    sino si (num < 2000)
        clase = "Diclonius Elite"
    sino si (num < 5000)
        clase = "Diclonius Definitivo"
    sino
        clase = "Diclonius Legendario"

    // Salida de resultados
    Imprimir clase
Fin

```

## Interfaz Gráfica



## Programa

```

package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Web extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblNumero;
    private JTextField txtNumero;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
    }
}

```

```
EventQueue.invokeLater(new Runnable() {
    public void run() {
        try {
            Web frame = new Web();
            frame.setVisible(true);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
});

// Crea la GUI
public Web() {
    setTitle("Web");
    setBounds(100, 100, 450, 214);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(246, 9, 89, 23);
    getContentPane().add(btnBorrar);

    lblNumero = new JLabel("Número de mensajes");
    lblNumero.setBounds(10, 13, 120, 14);
    getContentPane().add(lblNumero);

    txtNumero = new JTextField();
    txtNumero.setBounds(130, 10, 70, 20);
    getContentPane().add(txtNumero);
    txtNumero.setColumns(10);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 44, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {

    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declaración de variables
    int num;
    String clase;

    // Entrada de datos
    num = Integer.parseInt(txtNumero.getText());
```

```
// Determina la clase del usuario
if (num < 50)
    clase = "Desconocido";
else if (num < 100)
    clase = "Humano";
else if (num < 250)
    clase = "Diclonius Inicial";
else if (num < 500)
    clase = "Diclonius Novato";
else if (num < 1000)
    clase = "Diclonius Experimentado";
else if (num < 2000)
    clase = "Diclonius Elite";
else if (num < 5000)
    clase = "Diclonius Definitivo";
else
    clase = "Diclonius Legendario";

// Salida de resultados
txtS.setText("Clase de usuario : " + clase);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtNumero.setText("");
    txtS.setText("");
    txtNumero.requestFocus();
}
```

# Problemas propuestos

## Actividad

1. Diseñe un programa que determine la categoría de un estudiante basándose en su promedio ponderado de acuerdo con la siguiente tabla:

Promedio	Categoría
17	A
14 y < 17	B
12 y < 14	C
< 12	D

2. Un supermercado vende aceite en botellas de 1 litro a los precios por litro dados en la siguiente tabla:

Aceite	Precio por litro
Primor	S/. 5.99
Girasol	S/. 5.50
Cil	S/. 4.50
Cocinero	S/. 4.70

Como oferta, el supermercado ofrece un porcentaje de descuento sobre el importe de la compra de acuerdo con la siguiente tabla:

Cantidad de litros	Descuento
< 4	5.0%
4 y < 7	7.5%
7 y < 10	10.0%
10	12.5%

Diseñe un algoritmo que determine el importe de la compra, el importe del descuento y el importe a pagar por la compra de cierta cantidad de litros de una misma marca de aceite.

3. Un supermercado vende yogurt en botellas de 1 litro a los precios dados en la siguiente tabla:

Yogurt	Precio por litro
Buena vida	S/. 3.90
Pura salud	S/. 3.80
Todo sabor	S/. 4.20
Cielo	S/. 3.60

Como oferta, la tienda aplica un porcentaje de descuento sobre el importe de la compra, basándose en la cantidad de botellas adquiridas, de acuerdo con la siguiente tabla:

Cantidad de botellas	Descuento
$\geq 45$	11.5%
$\geq 30 \text{ y } < 45$	9.0%
$\geq 15 \text{ y } < 30$	6.5%
$< 15$	4.0%

Diseñe un algoritmo que determine el importe de la compra, el importe del descuento y el importe a pagar por la compra de cierta cantidad de botellas de una misma marca de yogur.

4. El índice de masa corporal (IMC) permite medir el grado de sobrepeso u obesidad de una persona. El IMC de una persona se calcula con la fórmula:

$$IMC = \frac{\text{peso}}{\text{estatura}^2}$$

estando el peso en kilogramos y la estatura en metros. Basándose en el valor del IMC, se obtiene el grado de obesidad de la persona de acuerdo a la tabla adjunta. Diseñe un algoritmo que determine el grado de obesidad de una persona conociendo su peso y su estatura.

IMC	Grado de obesidad
$< 20$	Delgado
$\geq 20 \text{ pero } < 25$	Normal
$\geq 25 \text{ pero } < 27$	Sobrepeso
$\geq 27$	Obesidad

## Autoevaluación

1. Diseñar un programa que lea la temperatura promedio de un día e imprima el tipo de clima correspondiente de acuerdo con la siguiente tabla:

Temperatura	Clima
$\leq 10$	Frío
$>10$ pero $\leq 20$	Nublado
$>20$ pero $\leq 30$	Caluroso
$> 30$	Trópico

2. Se denomina velocidad de escape a la velocidad mínima inicial que necesita un objeto para escapar de la gravedad de un cuerpo astronómico y continuar desplazándose sin tener que hacer otro esfuerzo propulsor. En la tabla adjunta, se dan velocidades de escape para los planetas del sistema solar, en km/seg. Dado un planeta del sistema solar, diseñe un programa que determine la velocidad de escape correspondiente.

Planeta	Velocidad de escape (km/seg)
Mercurio	4.2
Venus	10.3
Tierra	11.2
Marte	5.0
Júpiter	61.0
Saturno	36.0
Urano	22.0
Neptuno	24.0
Plutón	5.3
Luna	2.4

3. Dadas las estaturas de Juan, Pedro y Miguel, diseñe un programa que determine quiénes de ellos tienen una estatura menor que la estatura promedio e imprima un mensaje como: “Juan y Pedro miden menos que el promedio” o “Pedro y Miguel miden menos que el promedio” o “Juan mide menos que el promedio”, etc.
4. Dado el porcentaje de fósforo de un mineral de fierro, diseñe un programa que determine la clasificación del mineral de acuerdo con la siguiente tabla:

Porcentaje de fósforo	Clasificación
$< 0.05$	Bessemer
$0.05$ pero $0.18$	No Bessemer
$> 0.18$	Fosforoso

5. Por la altura de sus tallos aéreos, las plantas se clasifican en la forma indicada en la tabla adjunta. Dada la altura de un tallo, diseñe un programa que determine la clasificación correspondiente.

Altura del tallo	Clasificación
Hasta 1.00 m.	Mata
Más de 1.00 m pero no más que 4.00 m.	Arbusto
Más de 4.00 m pero no más que 8.00 m.	Arbolillo
Más de 8.00 m.	Árbol

6. Una compañía cobra las cuotas mensuales de sus clientes de acuerdo con lo siguiente:

- Si el cliente paga dentro de los primeros diez días del mes, obtiene un descuento igual al mayor valor entre \$5 y el 2% de la cuota.
- Si el cliente paga en los siguientes diez días, no tiene derecho a ningún descuento; deberá pagar exactamente la suma adeudada.
- Si el cliente paga dentro de los restantes días del mes, tendrá un recargo igual al mayor valor entre \$10 y el 3% de la cuota.

Dado el número del día de pago de un mes y la cuota mensual, diseñe un programa que determine el importe total a pagar.

7. Una librería estima los precios de sus libros de la siguiente forma: el precio básico de un libro es de \$5.00 más \$0.15 por página. Sin embargo, si el número de páginas excede de 300, el precio sufrirá un recargo adicional de \$10. Además, si el número de páginas excede de 550, el precio se incrementará en otros \$7.50. Diseñe un programa que determine el precio de un libro.
8. Una empresa calcula el sueldo bruto de sus empleados multiplicando las horas trabajadas por una tarifa horaria que depende de la categoría del trabajador de acuerdo con la siguiente tabla:

Categoría	Tarifa
A	S/. 21.0
B	S/. 19.5
C	S/. 17.0
D	S/. 15.5

Por ley, todo empleado se somete a un porcentaje de descuento del sueldo bruto: 20% si el sueldo bruto es mayor de S/. 2500 y 15%, en caso contrario.

Dada la categoría y las horas trabajadas, diseñe un programa que determine el sueldo bruto, el descuento y el sueldo neto correspondientes.

9. Una empresa química paga a sus vendedores un sueldo bruto que es igual a la suma de un sueldo básico quincenal de S/.250 más una comisión igual a un porcentaje del total de las ventas efectuadas de acuerdo con la siguiente tabla:

Monto vendido	Comisión
20000	16%
15000 pero < 20000	14%
10000 pero < 15000	12%
< 10000	10%

Por otro lado, si el sueldo bruto del vendedor supera los S/.1800, este se somete a un descuento del 11%. Dado el monto total vendido en una quincena por un vendedor, diseñe un programa que determine el sueldo bruto, el descuento y el sueldo neto correspondientes.

10. Un curso se evalúa de la siguiente forma: se toman cinco prácticas calificadas, se determina el promedio de las cuatro notas más altas y se le da al estudiante una categoría que puede ser A, B, C o D según la tabla siguiente:

Promedio	Categoría
$\geq 17$	A
$\geq 14$ pero $< 17$	B
$\geq 10$ pero $< 14$	C
$< 10$	D

Dadas las notas de práctica de un estudiante, diseñe un programa que determine el promedio y la categoría que le corresponden.

11. En una empresa, cada empleado tiene un código entero de tres cifras. Diseñe un programa que lea el código de un empleado y determine de qué tipo de empleado se trata de acuerdo con los siguientes criterios:

- Si el código es divisible por 2, por 3 y por 5, el tipo de empleado es “Administrativo”.
- Si el código es divisible por 3 y por 5 pero no por 2, el tipo de empleado es “Directivo”.
- Si el código es divisible por 2, pero no por 3 ni por 5, el tipo de empleado es “Vendedor”.
- Si el código no es divisible por 2, ni por 3 ni por 5, el tipo de empleado es “Seguridad”.

12. En una librería, obsequian lapiceros Lucas, Cross y/o Novo por la compra de cuadernos, de acuerdo con lo siguiente:

- Si el número de cuadernos adquiridos es menos de 12, no se obsequia ningún lapicero.
- Si el número de cuadernos adquiridos es no menor de 12, pero menos de 24, se obsequia 1 lapicero Lucas por cada 4 cuadernos adquiridos.
- Si el número de cuadernos adquiridos es no menor de 24, pero menos de 36, se obsequia 2 lapiceros Cross por cada 4 cuadernos adquiridos.
- Si el número de cuadernos adquiridos es no menor de 36, se obsequia 2 lapiceros Novo por cada 4 cuadernos adquiridos, más 1 lapicero Lucas y más 1 lapicero Cross.

Dada la cantidad de cuadernos adquiridos por un cliente, diseñe un programa que determine cuántos lapiceros Lucas, Cross y Novo le corresponden.

13. Diseñe un programa que lea un número entero y determine si tiene 1, 2, 3, 4 o más de 4 dígitos.
14. Se desea un programa para obtener el grado de eficiencia de un operario de torno de una fábrica productora de tornillos de acuerdo con las siguientes condiciones que se le impone para un período de 15 días.

Condiciones impuestas al operario:

- No más de 1.5 horas de ausencia al trabajo
- Menos de 300 tornillos defectuosos producidos
- Más de 10000 tornillos no defectuosos producidos

15. Los grados de eficiencia para cada trabajador son asignados de la siguiente manera:

- Si no cumple ninguna condición, grado 5
- Si sólo cumple la primera condición, grado 7
- Si sólo cumple la segunda condición, grado 8
- Si sólo cumple la tercera condición, grado 9
- Si cumple la primera y segunda condición, grado 12
- Si cumple la primera y tercera condición, grado 13
- Si cumple la segunda y tercera condición, grado 15
- Si cumple las tres condiciones, grado 20

16. Una compañía alquila automóviles a los costos por día dados en la siguiente tabla:

Días de alquiler	Costo por día
1	\$ 50
2	\$ 45
3 a 5	\$ 40
6 a 10	\$ 35
Más de 10	\$ 30

Adicionalmente, si el cliente alquila un automóvil por más de 10 días, la compañía le obsequia una agenda; en caso contrario, le obsequia un cuaderno. Dado el número de días de alquiler, diseñe un programa que determine el importe a pagar y el obsequio correspondientes.

17. Planos S.A se dedica al fotocopiado de planos topográficos a los precios indicados en la siguiente tabla:

Plano	Costo por copia
Menos de 500 hectáreas	S/. 25
De 501 a 1000 hectáreas	S/. 30
De 1001 a 2000 hectáreas	S/. 40
De 2001 a 5000 hectáreas	S/. 50
De 5001 a 10000 hectáreas	S/. 75
Más de 10000 hectáreas	S/. 100

Como oferta, Planos S.A ofrece un descuento del 5% para la segunda copia y del 10% para cada una de las demás copias. Dada la cantidad de copias de un plano, diseñe un programa que determine el importe bruto, el importe de descuento (o si no hay descuento) y el importe a pagar.

18. Un vivero municipal vende plantones a los precios indicados en la siguiente tabla:

Plantones	Costo por unidad
Forestal	S/. 0.35
Forestal ornamental	S/. 0.50
Frutal	S/. 2.00
Frutal injertado	S/. 3.00
Bonsái	S/. 5.00

Por otro lado, si el cliente adquiere más de 10 docenas de plantones de cualquier tipo, el vivero le obsequia 3 plantones por cada docena adquirida; en caso contrario, sólo le obsequia 1 plantón por cada docena.

Diseñe un programa que determine el importe a pagar y el número de plantones de obsequio por la compra de cierta cantidad de plantones de un mismo tipo.

19. En una elección democrática a la presidencia de un club femenino, participan Débora, Raquel y Séfora. Para ganar la elección se requiere obtener la mitad de los votos emitidos más uno.

En caso de no haber un ganador, pasan a una segunda vuelta los candidatos que alcanzaron los dos primeros puestos o se anula la elección, si hay empate entre los tres o, si hay empate por el segundo puesto.

Dados los votos obtenidos por cada candidato, se le pide diseñar un programa que determine el nombre del candidato ganador o los nombres de los candidatos que pasan a la segunda vuelta o un mensaje indicando la anulación de la elección.

20. Escriba una versión más simple para cada uno de los siguientes fragmentos de código:

```

a. if(a > 2 || b > 2 || c > 2)
    x = 1;
else if(a%2 == 1 && b <= 2)
    x = 2;
else if(b%2 == 1 && c <= 2)
    x = 3;
else if(c%2 == 1 && a <= 2)
    x = 4;
else
    x = 5;

b. if(x%3 == 0) {
    a = 0;
    b = 2;
    c = 3;
} else if(x%2 == 1) {
    a = 1;
    b = 2;
    c = 3;
} else {
    a = 2;
    b = 2;
    c = 3;
}

```

21. Explique y corrija los errores de sintaxis de los siguientes fragmentos de programa.

a. if(a > 20)  
    x = 1;  
    y = 2;  
else if(a > 15)  
    x = 2;  
    y = 3;  
else {  
    x = 3;  
    y = 4;  
}

b. if(x%3 == 0) {  
    a = 0;  
else if(x%2 == 1)  
    a = 1;  
else  
    a = 2;  
}

22. Determine qué imprime el siguiente fragmento de programa para los siguientes casos:

a). x igual a 1, b). x igual a 2, c). x igual a 7 y d). x igual a 5

```
if(x == 1){  
    txtS.append("aaaa\n");  
    x = x + 1;  
}  
if(x == 2){  
    txtS.append("bbbb\n");  
    x = x + 2;  
}  
if(x >= 8)  
    txtS.append("cccc\n");  
else if(x >= 6)  
    txtS.append("dddd\n");  
else  
    txtS.append("eeee\n");  
txtS.append("ffff");
```

# Para recordar

- Colocar ; al final de cualquier condición de la estructura **if...else...if** causa un error de sintaxis.
- En caso qué las acciones de un **if**, de un **else if** o de un **else** sean compuestas (varias acciones simples), estas deben ir entre llaves de bloque { }.

### 3.1.3. Estructura de selección múltiple switch

#### Definición

La estructura de selección múltiple **switch** permite seleccionar una ruta de entre varias rutas posibles basándose en el valor de una variable **selector** que se compara con una lista de constantes enteras o de carácter **c1, c2, c3, ..., cn**. Cuando se encuentra una correspondencia entre el valor de la variable **selector** y una constante, se ejecuta la acción o el grupo de acciones asociadas a dicha constante. Si el selector no coincide con ninguna constante, se efectúa la acción por defecto, si es que existe.

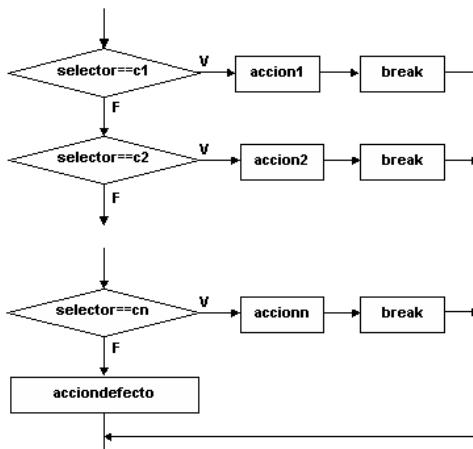


Figura 1 Estructura de Selección Múltiple **switch**

En la tabla que sigue, se muestra el código y el pseudocódigo de la estructura **switch**.

Código Java	Pseudocódigo
<pre> switch( selector ){     case c1:         accion1;         break;     case c2:         accion2;         break;         .         .     case cn:         accionN;         break;     default :         accionDefecto; } </pre>	<pre> según( selector ){     caso c1:         accion1         salir     caso c2:         accion2         salir         .         .     caso cn:         accionN         salir     defecto :         accionDefecto } </pre>

#### Consideraciones:

- Las sentencias **break** y el caso por defecto **default** son opcionales.
- El caso por defecto **default** no tiene que ser el último de todos sino que puede ser el primero u ocupar una posición intermedia.
- Luego de efectuarse la acción o las acciones de un **case** o del **default**, se proseguirá con la ejecución de la acción o las acciones de los **case** que siguen hasta encontrar un **break** o hasta llegar al final de la estructura **switch**; lo que ocurra primero.
- Es un error de sintaxis tener casos duplicados.
- Las acciones pueden ser acciones simples o acciones compuestas. En el caso de acciones compuestas, no es necesario colocarlas entre llaves de bloque.

## Problemas resueltos

### Problema 1

Un centro comercial ha decidido hacer un obsequio a los clientes cuyo importe total pagado es mayor de S/. 500. Para obtener el obsequio, el cliente debe extraer un bolo de una urna que contiene 50 bolos numerados del 1 al 50. Con el número del bolo, el obsequio se obtiene de la siguiente tabla:

Número del bolo	Obsequio
10	Una agenda
20	Un reloj
30	Una memoria USB
40	Un perfume
50	Una radio
Otro	Una pelota

Dado el importe total pagado y el número del bolo, diseñe un programa que determine el obsequio correspondiente.

### Algoritmo

```

Inicio
    // Declaración de variables
    entero numbolo
    real imptotpag
    cadena obsequio

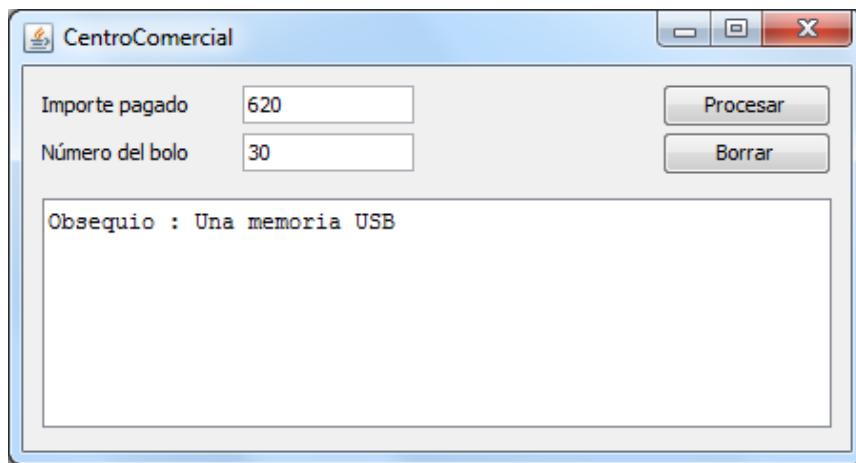
    // Entrada de datos
    Leer imptotpag, numbolo

    // Determina el obsequio
    si (imptotpag > 500) {
        segun (numbolo) {
            caso 10:
                obsequio = "Una agenda"
                salir
            caso 20:
                obsequio = "Un reloj"
                salir
            caso 30:
                obsequio = "Una memoria USB"
                salir
            caso 40:
                obsequio = "Un perfume"
                salir
            defecto:
                obsequio = "Una pelota"
        }
    }

    // Salida de resultados
    Imprimir obsequio
}
Sino
    Imprimir "No le corresponde ningún obsequio"
Fin

```

## Interfaz Gráfica



## Programa

```

package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class CentroComercial extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblImportePagado;
    private JLabel lblNumeroBolo;
    private JTextField txtImportePagado;
    private JTextField txtNumeroBolo;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    CentroComercial frame = new CentroComercial();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}

```

```
// Crea la GUI
public CentroComercial() {

    setTitle("CentroComercial");
    setBounds(100, 100, 450, 239);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblImportePagado = new JLabel("Importe pagado");
    lblImportePagado.setBounds(10, 13, 105, 14);
    getContentPane().add(lblImportePagado);

    lblNumeroBolo = new JLabel("Número del bolo");
    lblNumeroBolo.setBounds(10, 38, 105, 14);
    getContentPane().add(lblNumeroBolo);

    txtImportePagado = new JTextField();
    txtImportePagado.setBounds(115, 10, 90, 20);
    getContentPane().add(txtImportePagado);
    txtImportePagado.setColumns(10);

    txtNumeroBolo = new JTextField();
    txtNumeroBolo.setBounds(115, 35, 90, 20);
    getContentPane().add(txtNumeroBolo);
    txtNumeroBolo.setColumns(10);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(335, 34, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 69, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);

}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {

    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }

}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declaración de variables
    int numbolo;
    double imptopag;
    String obsequio;

    // Entrada de datos
    imptopag = Double.parseDouble(txtImportePagado.getText());
    numbolo = Integer.parseInt(txtNumeroBolo.getText());
}
```

```

// Determina el obsequio
if (imptotpag > 500) {
    switch (numbolo) {
        case 10:
            obsequio = "Una agenda";
            break;
        case 20:
            obsequio = "Un reloj";
            break;
        case 30:
            obsequio = "Una memoria USB";
            break;
        case 40:
            obsequio = "Un perfume";
            break;
        default:
            obsequio = "Una pelota";
    }

    //Salida de resultados
    txtS.setText("Obsequio : " + obsequio);

} else {
    txtS.setText("No le corresponde ningún obsequio");
}
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtImportePagado.setText("");
    txtNumeroBolo.setText("");
    txtS.setText("");
    txtImportePagado.requestFocus();
}
}

```

## Problema 2

Una dulcería vende chocolates a los precios dados en la siguiente tabla:

Tipo de chocolate	Precio unitario
Primor	S/. 8.5
Dulzura	S/. 10.0
Tentación	S/. 7.0
Explosión	S/. 12.5

Como oferta, la tienda aplica un porcentaje de descuento sobre el importe de la compra, basándose en la cantidad de chocolates adquiridos, de acuerdo con la siguiente tabla:

Cantidad de chocolates	Descuento
< 5	4.0%
≥ 5 y < 10	6.5%
≥ 10 y < 15	9.0%
≥ 15	11.5%

Adicionalmente, si el importe a pagar es no menor de S/. 250, la tienda obsequia 3 caramelos por cada chocolate; en caso contrario, obsequia 2 caramelos por cada chocolate.

Dado el tipo de chocolate y la cantidad de unidades adquiridas, diseñe un programa que determine el importe de la compra, el importe del descuento, el importe a pagar y la cantidad de caramelos de obsequio.

## Algoritmo

```

Inicio
    // Declaración de variables
    entero tipo, cantidad, caramelos
    real impcom, impdes, imppag

    // Entrada de datos
    Leer tipo, cantidad

    // Calcula el importe de la compra
    segun (tipo) {
        caso 0:
            impcom = 8.5*cantidad
            salir
        caso 1:
            impcom = 10.0*cantidad
            salir
        caso 2:
            impcom = 7.0*cantidad
            salir
        defecto:
            impcom = 12.5*cantidad
    }

    // Calcula el importe del descuento
    segun (cantidad) {
        caso 1:
        caso 2:
        caso 3:
        caso 4:
            impdes = 0.04*impcom
            salir
        caso 5:
        caso 6:
        caso 7:
        caso 8:
        caso 9:
            impdes = 0.065*impcom
            salir
        caso 10:
        caso 11:
        caso 12:
        caso 13:
        caso 14:
            impdes = 0.09*impcom
            salir
        defecto:
            impdes = 0.115*impcom
    }

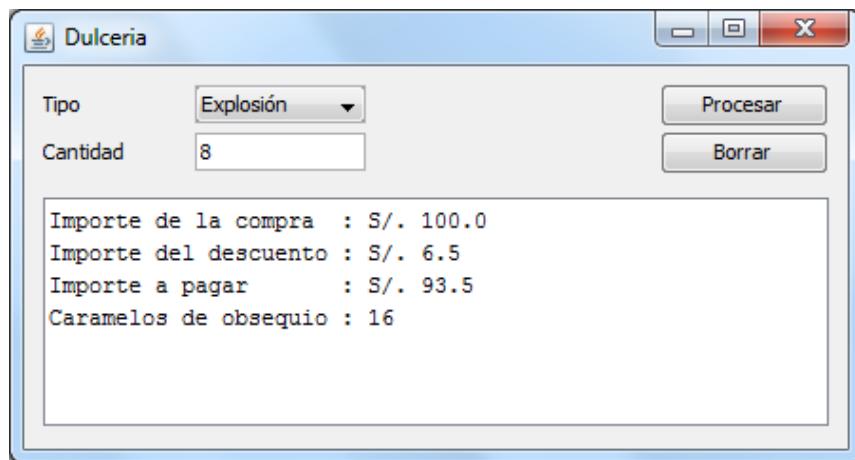
    // Calcula el importe a pagar
    imppag = impcom - impdes

    // Calcula la cantidad de caramelos de regalo
    si(imppag < 250)
        caramelos = 2*cantidad
    sino
        caramelos = 3*cantidad

    // Salida de resultados
    Imprimir impcom, impdes, imppag, caramelos
Fin

```

## Interfaz Gráfica



## Programa

```

package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JComboBox;
import javax.swing.DefaultComboBoxModel;

public class Dulceria extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblTipo;
    private JLabel lblCantidad;
    private JComboBox<String> cboTipo;
    private JTextField txtCantidad;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Dulceria frame = new Dulceria();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}

```

```
// Crea la GUI
public Dulceria() {
    setTitle("Dulceria");
    setBounds(100, 100, 450, 239);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblTipo = new JLabel("Tipo");
    lblTipo.setBounds(10, 13, 80, 14);
    getContentPane().add(lblTipo);

    lblCantidad = new JLabel("Cantidad");
    lblCantidad.setBounds(10, 38, 80, 14);
    getContentPane().add(lblCantidad);

    cboTipo = new JComboBox<String>();
    cboTipo.setModel(
        new DefaultComboBoxModel<String>(new String[] { "Primor",
"Dulzura", "Tentación", "Explosión" }));
    cboTipo.setBounds(90, 10, 90, 20);
    getContentPane().add(cboTipo);

    txtCantidad = new JTextField();
    txtCantidad.setBounds(90, 35, 90, 20);
    getContentPane().add(txtCantidad);
    txtCantidad.setColumns(10);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(335, 34, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 69, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);

}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {

    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }

}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declaración de variables
    int tipo, cantidad, caramelos;
    double impcom, impdes, imppag;

    // Entrada de datos
    tipo = cboTipo.getSelectedIndex();
    cantidad = Integer.parseInt(txtCantidad.getText());
}
```

```
// Calcula el importe de la compra
switch (tipo) {
    case 0:
        impcom = 8.5 * cantidad;
        break;
    case 1:
        impcom = 10.0 * cantidad;
        break;
    case 2:
        impcom = 7.0 * cantidad;
        break;
    default:
        impcom = 12.5 * cantidad;
}

// Calcula el importe del descuento
switch (cantidad) {
    case 1:
    case 2:
    case 3:
    case 4:
        impdes = 0.04 * impcom;
        break;
    case 5:
    case 6:
    case 7:
    case 8:
    case 9:
        impdes = 0.065 * impcom;
        break;
    case 10:
    case 11:
    case 12:
    case 13:
    case 14:
        impdes = 0.09 * impcom;
        break;
    default:
        impdes = 0.115 * impcom;
}

// Calcula el importe a pagar
imppag = impcom - impdes;

// Calcula la cantidad de caramelos de regalo
if (imppag < 250)
    caramelos = 2 * cantidad;
else
    caramelos = 3 * cantidad;

// Salida de resultados
txtS.setText("Importe de la compra : S/. " + impcom + "\n");
txtS.append("Importe del descuento : S/. " + impdes + "\n");
txtS.append("Importe a pagar : S/. " + imppag + "\n");
txtS.append("Caramelos de obsequio : " + caramelos);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    cboTipo.setSelectedIndex(0);
    txtCantidad.setText("");
    txtS.setText("");
    txtCantidad.requestFocus();
}
```

### Problema 3

Dados los siguientes tipos de papel y sus respectivas dimensiones:

Tamaño de papel	Dimensiones
A4	297 x 210 mm
B5	182 x 257 mm
A5	148 x 210 mm
Carta	8½ x 11 pulg
Legal	8½ x 14 pulg
Ejecutivo	7¼ x 10½ pulg
Media carta	5½ x 8½ pulg

Se conoce como área imprimible al área que queda libre luego de descontar los márgenes superior, inferior, izquierdo y derecho. Dado el tamaño del papel y los márgenes superior, inferior, izquierdo y derecho en centímetros, diseñe un programa que determine el área imprimible en cm<sup>2</sup>.

### Algoritmo

```

Inicio
    // Declaración de variables
    entero tamaño
    real mrgsup, mrginf, mrgder, mrgizq, ancho, alto, area

    // Entrada de datos
    Ler tamaño, mrgsup, mrginf, mrgder, mrgizq

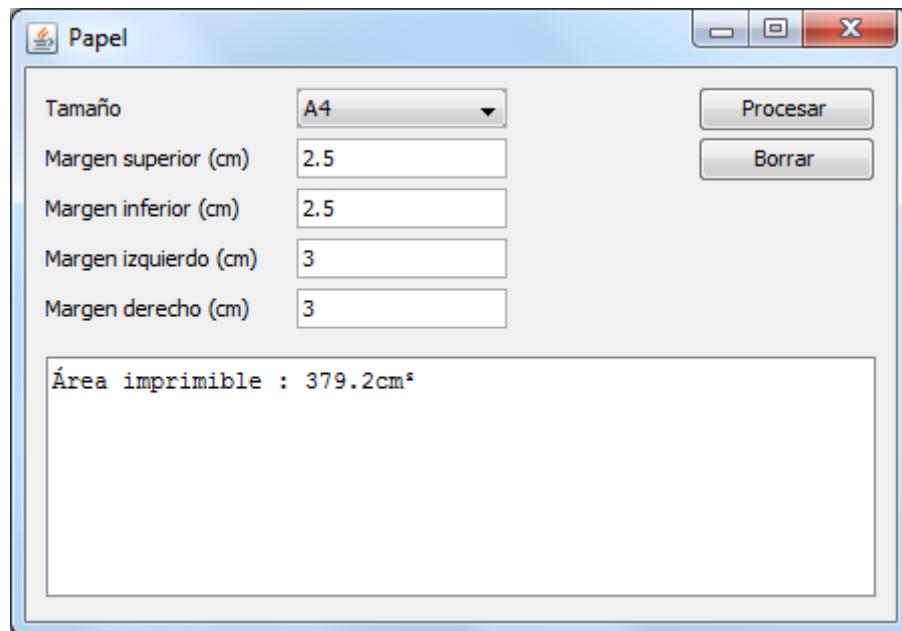
    // Determina el ancho y el alto del papel en cm
    segun (tamaño) {
        caso 0:
            ancho = 29.7
            alto = 21.0
            salir
        caso 1:
            ancho = 18.2
            alto = 25.7
            salir
        caso 2:
            ancho = 14.8
            alto = 21.0
            salir
        caso 3:
            ancho = 8.5*2.54
            alto = 11*2.54
            salir
        caso 4:
            ancho = 8.5*2.54
            alto = 14*2.54
            salir
        caso 5:
            ancho = 7.25*2.54
            alto = 10.5*2.54
            salir
        defecto:
            ancho = 5.5*2.54
            alto = 8.5*2.54
    }

    // Calcula el área imprimible
    area = (ancho-mrgizq-mrgder) * (alto-mrgsup-mrginf)

```

```
// Muestra al área imprimible
Imprimir area
Fin
```

## Interfaz Gráfica



## Programa

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JComboBox;
import javax.swing.DefaultComboBoxModel;

public class Papel extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblTamaño;
    private JLabel lblMargenSuperior;
    private JLabel lblMargenInferior;
    private JLabel lblMargenIzquierdo;
    private JLabel lblMargenDerecho;
    private JTextField txtMargenSuperior;
    private JTextField txtMargenInferior;
    private JTextField txtMargenIzquierdo;
    private JTextField txtMargenDerecho;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;
    private JComboBox<String> cboTamaño;
```

```
// Lanza la aplicación
public static void main(String[] args) {
    try {
        UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
    } catch (Throwable e) {
        e.printStackTrace();
    }

    EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                Papel frame = new Papel();
                frame.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

// Crea la GUI
public Papel() {
    setTitle("Papel");
    setBounds(100, 100, 450, 314);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblTamaño = new JLabel("Tama\u00f1o");
    lblTamaño.setBounds(10, 13, 125, 14);
    getContentPane().add(lblTamaño);

    lblMargenSuperior = new JLabel("Margen superior (cm)");
    lblMargenSuperior.setBounds(10, 38, 125, 14);
    getContentPane().add(lblMargenSuperior);

    lblMargenInferior = new JLabel("Margen inferior (cm)");
    lblMargenInferior.setBounds(10, 63, 125, 14);
    getContentPane().add(lblMargenInferior);

    lblMargenIzquierdo = new JLabel("Margen izquierdo (cm)");
    lblMargenIzquierdo.setBounds(10, 88, 125, 14);
    getContentPane().add(lblMargenIzquierdo);

    lblMargenDerecho = new JLabel("Margen derecho (cm)");
    lblMargenDerecho.setBounds(10, 113, 125, 14);
    getContentPane().add(lblMargenDerecho);

    cboTamaño = new JComboBox<String>();
    cboTamaño.setModel(new DefaultComboBoxModel<String>(
        new String[] { "A4", "B5", "A5", "Carta", "Legal", "Ejecutivo", "Media carta" }));
    cboTamaño.setBounds(135, 10, 105, 20);
    getContentPane().add(cboTamaño);

    txtMargenSuperior = new JTextField();
    txtMargenSuperior.setBounds(135, 35, 105, 20);
    getContentPane().add(txtMargenSuperior);
    txtMargenSuperior.setColumns(10);

    txtMargenInferior = new JTextField();
    txtMargenInferior.setBounds(135, 60, 105, 20);
    getContentPane().add(txtMargenInferior);
    txtMargenInferior.setColumns(10);

    txtMargenIzquierdo = new JTextField();
    txtMargenIzquierdo.setColumns(10);
    txtMargenIzquierdo.setBounds(135, 85, 105, 20);
    getContentPane().add(txtMargenIzquierdo);

    txtMargenDerecho = new JTextField();
    txtMargenDerecho.setColumns(10);
    txtMargenDerecho.setBounds(135, 110, 105, 20);
    getContentPane().add(txtMargenDerecho);
```

```
btnProcesar = new JButton("Procesar");
btnProcesar.addActionListener(this);
btnProcesar.setBounds(335, 9, 89, 23);
getContentPane().add(btnProcesar);

btnBorrar = new JButton("Borrar");
btnBorrar.addActionListener(this);
btnBorrar.setBounds(335, 34, 89, 23);
getContentPane().add(btnBorrar);

scpScroll = new JScrollPane();
scpScroll.setBounds(10, 144, 414, 120);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declaración de variables
    int tamaño;
    double mrgsup, mrginf, mrgder, mrgizq, ancho, alto, area;

    // Entrada de datos
    tamaño = cboTamaño.getSelectedIndex();
    mrgsup = Double.parseDouble(txtMargenSuperior.getText());
    mrginf = Double.parseDouble(txtMargenInferior.getText());
    mrgder = Double.parseDouble(txtMargenDerecho.getText());
    mrgizq = Double.parseDouble(txtMargenIzquierdo.getText());

    // Determina el ancho y el alto del papel en cm
    switch (tamaño) {
        case 0:
            ancho = 29.7;
            alto = 21.0;
            break;
        case 1:
            ancho = 18.2;
            alto = 25.7;
            break;
        case 2:
            ancho = 14.8;
            alto = 21.0;
            break;
        case 3:
            ancho = 8.5 * 2.54;
            alto = 11 * 2.54;
            break;
        case 4:
            ancho = 8.5 * 2.54;
            alto = 14 * 2.54;
            break;
        case 5:
            ancho = 7.25 * 2.54;
            alto = 10.5 * 2.54;
            break;
        default:
            ancho = 5.5 * 2.54;
            alto = 8.5 * 2.54;
    }
}
```

```

    // Calcula el área imprimible
    area = (ancho - mrgizq - mrgder) * (alto - mrgsup - mrginf);

    // Muestra al área imprimible
    txtS.setText("Área imprimible : " + area + "cm2");
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    cboTamaño.setSelectedIndex(0);
    txtMargenSuperior.setText("");
    txtMargenInferior.setText("");
    txtMargenIzquierdo.setText("");
    txtMargenDerecho.setText("");
    txtS.setText("");
    txtMargenSuperior.requestFocus();
}
}

```

#### Problema 4

Una empresa de transportes brinda servicios en dos rutas (Lima-Huánuco y Lima-Huancayo) en tres calidades de servicio a los precios por boleto dados en la siguiente tabla:

Calidad	Precio del boleto	
	Lima-Huánuco	Lima-Huancayo
A	S/. 45	S/. 38
B	S/. 35	S/. 33
C	S/. 30	S/. 28

Como oferta, la empresa efectúa 5% de descuento sobre el importe de la compra únicamente para compras de boletos de calidad A, independientemente de la ruta elegida, siempre y cuando la cantidad de boletos adquiridos sea más de 4.

Dada la ruta elegida, la calidad del servicio y la cantidad de boletos adquiridos, diseñe un programa que determine el importe de la compra, el importe del descuento y el importe a pagar.

#### Algoritmo

```

Inicio

    // Declaración de variables
    entero cal, rut, can
    real impcom, impdes, imppag

    // Entrada de datos
    Leer rut, cal, can

    // Determina el importe de la compra
    segun (cal) {
        caso 0:
            si (rut == 0)
                impcom = 45 * can
            sino
                impcom = 38 * can

        salir
    }

```

```

caso 1:
    si (rut == 0)
        impcom = 35 * can
    sino
        impcom = 33 * can

    salir
defecto:
    si (rut == 0)
        impcom = 30 * can
    sino
        impcom = 28 * can
}

// Determina el importe del descuento
si (cal == 0 && can > 4)
    impdes = 0.05 * impcom
sino
    impdes = 0

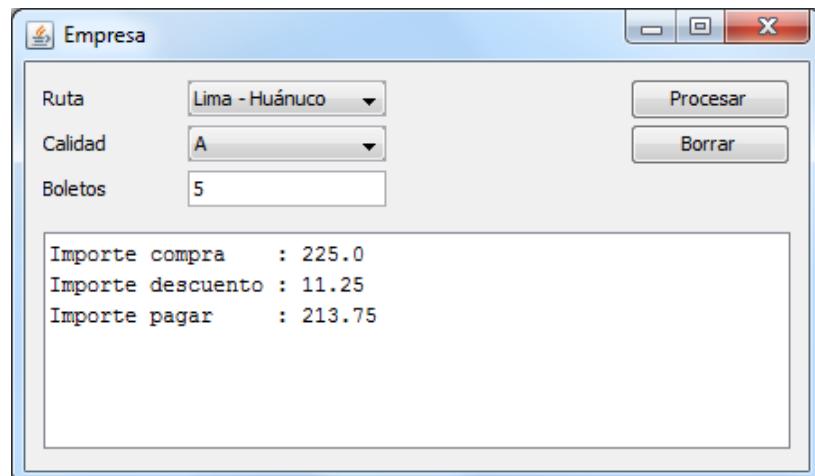
// Determina el importe a pagar
imppag = impcom - impdes

// Salida de resultados
Imprimir impcom, impdes, imppag

Fin

```

## Interfaz Gráfica



## Programa

```

package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JComboBox;
import javax.swing.DefaultComboBoxModel;

```

```
public class Empresa extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblRuta;
    private JLabel lblCalidad;
    private JLabel lblBoletos;
    private JComboBox<String> cboRuta;
    private JComboBox<String> cboCalidad;
    private JTextField txtBoletos;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Empresa frame = new Empresa();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Empresa() {
        setTitle("Empresa");
        setBounds(100, 100, 450, 264);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        lblRuta = new JLabel("Ruta");
        lblRuta.setBounds(10, 13, 80, 14);
        getContentPane().add(lblRuta);

        lblCalidad = new JLabel("Calidad");
        lblCalidad.setBounds(10, 38, 80, 14);
        getContentPane().add(lblCalidad);

        lblBoletos = new JLabel("Boletos");
        lblBoletos.setBounds(10, 63, 80, 14);
        getContentPane().add(lblBoletos);

        cboRuta = new JComboBox<String>();
        cboRuta.setModel(new DefaultComboBoxModel<String>(new String[] { "Lima - Huánuco", "Lima - Huancayo" }));
        cboRuta.setBounds(90, 10, 110, 20);
        getContentPane().add(cboRuta);

        txtBoletos = new JTextField();
        txtBoletos.setBounds(90, 60, 110, 20);
        getContentPane().add(txtBoletos);
        txtBoletos.setColumns(10);

        cboCalidad = new JComboBox<String>();
        cboCalidad.setModel(new DefaultComboBoxModel<String>(new String[] { "A", "B", "C" }));
        cboCalidad.setBounds(90, 35, 110, 20);
        getContentPane().add(cboCalidad);
    }
}
```

```
btnProcesar = new JButton("Procesar");
btnProcesar.addActionListener(this);
btnProcesar.setBounds(335, 9, 89, 23);
getContentPane().add(btnProcesar);

btnBorrar = new JButton("Borrar");
btnBorrar.addActionListener(this);
btnBorrar.setBounds(335, 34, 89, 23);
getContentPane().add(btnBorrar);

scpScroll = new JScrollPane();
scpScroll.setBounds(10, 94, 414, 120);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declaración de variables
    int cal, rut, can;
    double impcom, impdes, imppag;

    // Entrada de datos
    rut = cboRuta.getSelectedIndex();
    cal = cboCalidad.getSelectedIndex();
    can = Integer.parseInt(txtBoletos.getText());

    // Determina el importe de la compra
    switch (cal) {
        case 0:
            if (rut == 0)
                impcom = 45 * can;
            else
                impcom = 38 * can;
            break;
        case 1:
            if (rut == 0)
                impcom = 35 * can;
            else
                impcom = 33 * can;
            break;
        default:
            if (rut == 0)
                impcom = 30 * can;
            else
                impcom = 28 * can;
    }

    // Determina el importe del descuento
    if (cal == 0 && can > 4)
        impdes = 0.05 * impcom;
    else
        impdes = 0;

    // Determina el importe a pagar
    imppag = impcom - impdes;
}
```

```

        // Salida de resultados
        txtS.setText("Importe compra : " + impcom + "\n");
        txtS.append("Importe descuento : " + impdes + "\n");
        txtS.append("Importe pagar: " + imppag);
    }

    // Procesa la pulsación del botón Borrar
    protected void actionPerformedBtnBorrar(ActionEvent arg0) {
        cboRuta.setSelectedIndex(0);
        cboCalidad.setSelectedIndex(0);
        txtBoletos.setText("");
        txtS.setText("");
        txtBoletos.requestFocus();
    }
}

```

**Problema 5**

Reemplace la siguiente estructura **if – else – if** por la estructura **switch**. Considere que **producto** es de tipo **int**.

```

if (producto == 0)
    precio = 25;
else if (producto == 1)
    precio = 15;
else if (producto == 2)
    precio = 10;
else
    precio = 12;

```

**Solución**

El equivalente **switch** es el siguiente:

```

switch (producto) {
case 0:
    precio = 25;
    break;
case1:
    precio = 15;
    break;
case2:
    precio = 10;
    break;
default:
    precio = 12;
}

```

**Problema 6**

Reemplace la estructura **if – else – if** por la estructura **switch**. Considere que **z** es de tipo **int**.

```

if (z == 0)
    a = 10;
else if (z == 1 || z == 3 || z == 5)
    a = 2;
else if (z == 2 || z == 4)
    a = 7;
else
    a = 3;

```

## Solución

```
switch (z) {  
    case 0:  
        a = 10;  
        break; case 1:  
    case 3:  
    case 5:  
        a = 2;  
        break;  
    case 2:  
    case 4:  
        a = 7;  
        break;  
    default:  
        a = 3;  
}
```

## Problema 7

Reemplace la siguiente estructura **if – else – if** por la estructura **switch**. Considere que **can** es de tipo **int**.

```
if (can < 4)  
    impcom = 17.5*can;  
else if (can < 8)  
    impcom = 16.5*can;  
else  
    impcom = 15.5*can;
```

## Solución

```
switch (can) {  
    case 1:  
    case 2:  
    case 3:  
        impcom = 17.5 * can;  
        break;  
    case 4:  
    case 5:  
    case 6:  
    case 7:  
        impcom = 16.5 * can;  
        break;  
    default:  
        impcom = 15.5 * can;  
}
```

# Problemas propuestos

## Actividad

- Una heladería vende helados a los precios unitarios dados en la siguiente tabla:

Helado	Precio unitario
Sol	S/. 1.5
Fresa	S/. 2.0
Mar	S/. 1.7
Rico	S/. 2.5

Como oferta, si el importe compra es mayor de S/. 250, la heladería aplica un descuento igual al 5% del importe compra; en caso contrario, no hay descuento.

Dado el tipo de helado y la cantidad de unidades adquiridas, diseñe un algoritmo que determine el importe de la compra, el importe del descuento y el importe a pagar.

- Dado el puntaje obtenido en el lanzamiento de un dado, diseñe un algoritmo que determine la calificación que corresponde de acuerdo con la siguiente tabla:

Puntaje	Calificación
1 ó 2	Pésimo
3 ó 4	Regular
5	Muy bien
6	Excelente

- Diseñe un algoritmo que lea un número entero y determine el estado civil correspondiente de acuerdo con la siguiente tabla:

Número	Estado civil
1	Soltero
2	Casado
3	Viudo
4	Divorciado

- Una librería ha puesto a la venta los libros indicados en la siguiente tabla:

Libro	Precio
Java 2 a Fondo	\$ 30
HTML 5 Fácil	\$ 27
Aprenda C++	\$ 20
Compendio de JavaScript	\$ 35

Como oferta, la librería aplica un porcentaje de descuento sobre el importe compra de acuerdo con la siguiente tabla:

Importe compra	Descuento
< 100	5%
$\geq 100$ y $< 200$	7%
$\geq 200$ y $< 300$	9%
$\geq 300$	11%

Dado el libro y la cantidad de ejemplares adquiridos, diseñe un algoritmo que determine el importe compra, el importe del descuento y el importe a pagar.

## Autoevaluación

1. Diseñe un programa que lea un número entero del intervalo 1 a 7 y determine el nombre del día de la semana de acuerdo con la siguiente tabla:

Número	Día
1	Lunes
2	Martes
3	Miércoles
4	Jueves
5	Viernes
6	Sábado
7	Domingo

2. En un cine, los precios de las entradas se dan en la siguiente tabla:

Día	General	Niños
Lunes	S/. 9	S/.7
Martes	S/. 7	S/.7
Miércoles a Viernes	S/. 10	S/.8
Sábado y Domingo	S/. 12	S/.9

Dados los datos de una compra (día, cantidad de entradas generales y cantidad de entradas de niños), diseñe un algoritmo que determine el importe total a pagar.

3. Determine qué imprime el siguiente fragmento de programa para cada uno de los siguientes casos:
- v igual a 1
  - v igual a 2
  - v igual a 3
  - v igual a 4.

```
switch(v) {
    case 1:
        txtS.append("Uno\n");
    case 2:
        txtS.append("Dos\n");
    case 3:
        txtS.append("Tres\n");
    default:
        txtS.append("Defecto\n");
}
```

4. Determine que imprime el siguiente fragmento de programa para v igual a 5.

```
switch(v) {
    case 1:
        txtS.append("Uno\n");
    case 2:
        txtS.append("Dos\n");
    default:
        txtS.append("Defecto\n");
    case 3:
        txtS.append("Tres\n");
}
```

5. Reescriba los siguientes fragmentos de programa usando la estructura if-else-if.

a. switch(estacion) {  
 case 0:  
 txtS.append("Primavera");  
 break;  
 case 1:  
 txtS.append("Verano");  
 break;  
 case 2:  
 txtS.append("Otoño");  
 break;  
 default:  
 txtS.append("Invierno");  
}

b. switch(numero) {  
 case 0: case 1: case 2:  
 x = 10;  
 break;  
 case 3: case 4: case 5:  
 x = 15;  
 break;  
 default:  
 x = 20;  
}

## Para recordar

- Si un **case** no tiene **break**, sucederá que al ejecutar las acciones de dicho **case** se ejecutarán, también, las acciones de los **case** que siguen hasta encontrar un **break** o hasta llegar al final del **switch**.
- Se puede usar la estructura **switch** en una toma de decisiones únicamente si las condiciones consisten en comparaciones de una misma variable con una lista de constantes enteras o de carácter.



# MÉTODOS

## LOGRO DE LA UNIDAD DE APRENDIZAJE

Al finalizar la unidad, el alumno diseña programas de manera modular mediante el uso de métodos con valor de retorno y sin valor de retorno.

## TEMARIO

### 4.1 Tema 4 : Métodos

- 4.1.1 : Programación modular
- 4.1.2 : Variables locales y globales
- 4.1.3 : Métodos tipo void
- 4.1.4 : Métodos con valor de retorno

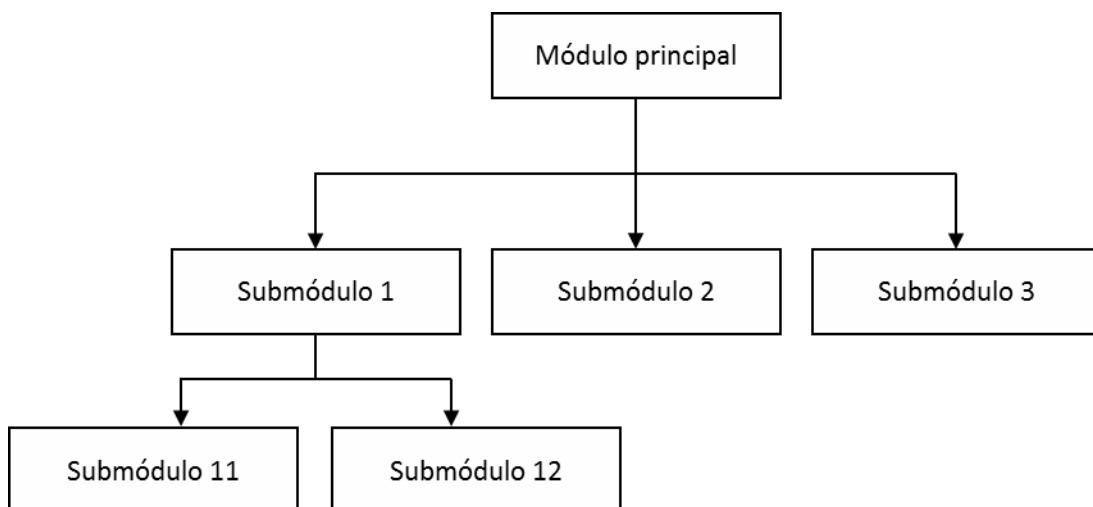
## ACTIVIDADES PROPUESTAS

- Los alumnos desarrollan programas mediante descomposición modular.
- Los alumnos desarrollan programas usando métodos con valor de retorno y tipo void.

## 4.1. TEMA 4: MÉTODOS

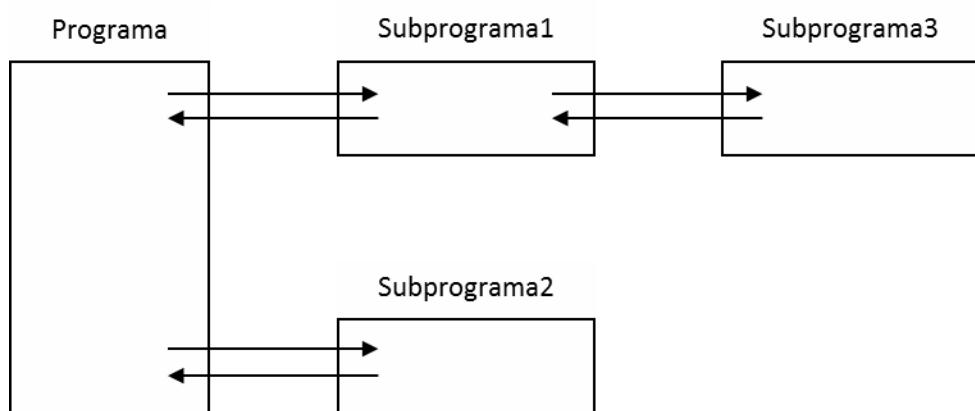
### 4.1.1. Programación modular

La programación modular es una metodología de programación que permite construir un programa grande descomponiéndolo en pequeños subprogramas o módulos. Para ello, se parte de un módulo principal que se descompone en varios submódulos que son controlados por el módulo principal. Si la tarea asignada a un módulo es demasiado compleja, este deberá descomponerse en otros módulos más pequeños hasta lograr módulos que hagan tareas relativamente sencillas. A este proceso de refinamiento sucesivo se conoce también como la técnica de “divide y vencerás”.



**Figura 1** Descomposición modular de un programa

Las tareas asignadas a los subprogramas pueden ser de diversa índole: entrada, salida, cálculos, control de otros módulos, etc. Para que un subprograma pueda efectuar su tarea, tiene que ser llamado o invocado por el programa principal o por algún otro módulo que considere necesario el servicio del subprograma. Una vez que el subprograma termina su tarea, devuelve el control al punto donde se hizo la llamada. Un subprograma puede llamar, a su vez, a otros subprogramas.



**Figura 2** Un programa con diferentes niveles de subprograma.

En el lenguaje Java, a los módulos o subprogramas se denominan **métodos**, mientras que, en el lenguaje algorítmico, se denominan **subalgoritmos**.

### 4.1.2. Variables locales y globales

Los métodos pueden utilizar sus propias variables denominadas **variables locales** o variables de uso compartido, comunes a todos los métodos, denominadas **variables globales**.

#### Variables locales

Una *variable local* es una variable que se declara en el interior de un método por lo que su ámbito es el interior del método, es decir, sólo puede ser utilizada dentro del método donde fue declarada. Este tipo de variable se crea vacía al iniciar la ejecución del método y se destruye al finalizar la ejecución del método.

#### Variables globales

Una *variable global* es una variable que se declara dentro del programa, pero en el exterior de todos los métodos; por lo que, su ámbito es el interior de todo el programa, es decir, puede ser utilizada en cualquier parte del programa. Este tipo de variable se crea al iniciar la ejecución del programa y se destruye al finalizar. Por otro lado, una variable global se inicializa automáticamente: **o** si es de tipo **int**, **o.0** si es de tipo **double**, **false** si es de tipo **boolean**, **'\0'** si es de tipo **char** y **null** si es de tipo **String**.

### 4.1.3. Métodos tipo void

Un **método tipo void** es un módulo de programa que puede recibir datos de entrada a través de variables locales denominadas **parámetros**; pero que no retorna ningún resultado al punto donde es invocado, razón por el que se le conoce también como **método sin valor de retorno**.

Los métodos tipo void pueden dividirse a su vez en dos tipos:

- Métodos tipo void sin parámetros
- Métodos tipo void con parámetros

#### Métodos tipo void sin parámetros

Estos métodos no pueden recibir datos de entrada ni retornar ningún resultado al punto de su invocación.

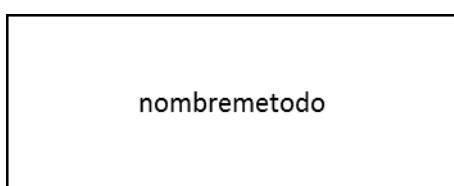


Figura 3 Método tipo void sin parámetros

Cuando se programa usando métodos, se siguen dos etapas. Primero, el método debe definirse. Esto consiste en crear el método ubicándolo en alguna parte del programa. Segundo, el método creado debe ser invocado en el lugar donde se requiera. Esto consiste en poner el método en ejecución.

## Definición

Este tipo de método se define de la siguiente manera:

```
void nombreMetodo () {
    Declaración de variables locales
    Cuerpo del método
}
```

*Donde:*

nombreMetodo : Es el nombre del método.

## Llamada

Este tipo de método se invoca de la siguiente manera:

```
nombreMetodo ();
```

*Donde:*

nombremetodo : Es el nombre del método invocado.

## Métodos tipo void con parámetros

Estos métodos reciben datos de entrada a través de variables locales al método denominadas parámetros; pero, igual que en el caso anterior, no pueden retornar ningún resultado al punto de su invocación.

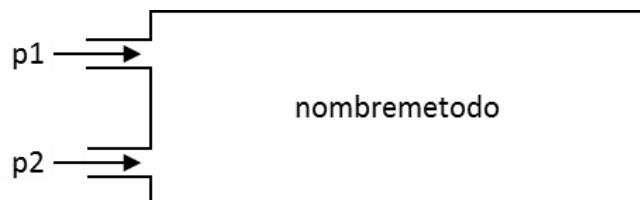


Figura 4 Método tipo void con parámetros

Donde **p1**, **p2**, etc. son los parámetros del método. El número de parámetros es variable y depende de las necesidades del método.

## Definición

Este tipo de método se define de la siguiente manera:

```
void nombreMetodo (tipo1 p1, tipo2 p2, ...) {
    Declaración de variables locales
    Cuerpo del método
}
```

*Donde:*

nombreMetodo : Es el nombre del método.

p1, p2,... : Son los nombres de los parámetros.

## Llamada

Este tipo de método se invoca de la siguiente manera:

```
nombreMetodo (v1, v2, ...);
```

Donde:

- |              |   |
|--------------|---|
| nombreMetodo | : Es el nombre del método invocado.         |
| v1, v2,...   | : Son los valores pasados a los parámetros. |

## Problemas resueltos

### Problema 1

En una universidad, los alumnos están clasificados en cuatro categorías. A cada categoría le corresponde una pensión mensual distinta dada en la siguiente tabla:

Categoría	Pensión
A	S/. 550
B	S/. 500
C	S/. 460
D	S/. 400

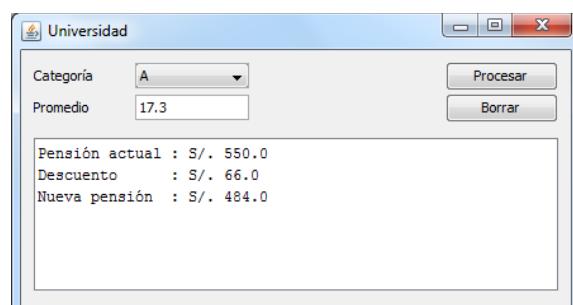
Semestralmente, la universidad efectúa rebajas en las pensiones de sus estudiantes a partir del segundo ciclo basándose en el promedio ponderado del ciclo anterior en porcentajes dados en la tabla siguiente:

Promedio	Descuento
0 a 13.99	No hay descuento
14.00 a 15.99	10 %
16.00 a 17.99	12 %
18.00 a 20.00	15 %

Dado el promedio ponderado y la categoría de un estudiante, diseñe un programa que determine cuánto de rebaja recibirá sobre su pensión actual y a cuánto asciende su nueva pensión.

Declare todas las variables como globales y use métodos tipo void.

## Interfaz Gráfica



## Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JComboBox;
import javax.swing.DefaultComboBoxModel;

public class Universidad extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblCategoria;
    private JLabel lblPromedio;
    private JComboBox<String> cboCategoria;
    private JTextField txtPromedio;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // -----
    // Declaración de variables globales
    int categoria;
    double actualpen, nuevapen, descuento, promedio;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Universidad frame = new Universidad();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Universidad() {
        setTitle("Universidad");
        setBounds(100, 100, 450, 239);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        lblCategoria = new JLabel("Categoría");
        lblCategoria.setBounds(10, 13, 80, 14);
        getContentPane().add(lblCategoria);

        lblPromedio = new JLabel("Promedio");
        lblPromedio.setBounds(10, 38, 80, 14);
        getContentPane().add(lblPromedio);
```

```
cboCategoria = new JComboBox<String>();
cboCategoria.setModel(new DefaultComboBoxModel<String>(new String[] {
"A", "B", "C", "D")));
cboCategoria.setBounds(90, 10, 90, 20);
getContentPane().add(cboCategoria);

txtPromedio = new JTextField();
txtPromedio.setBounds(90, 35, 90, 20);
getContentPane().add(txtPromedio);

btnProcesar = new JButton("Procesar");
btnProcesar.addActionListener(this);
btnProcesar.setBounds(335, 9, 89, 23);
getContentPane().add(btnProcesar);

btnBorrar = new JButton("Borrar");
btnBorrar.addActionListener(this);
btnBorrar.setBounds(335, 34, 89, 23);
getContentPane().add(btnBorrar);

scpScroll = new JScrollPane();
scpScroll.setBounds(10, 69, 414, 120);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    ingresarDatos();
    calcularPensionActual();
    calcularDescuento();
    calcularNuevaPension();
    mostrarResultados();
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtPromedio.setText("");
    txtS.setText("");
    txtPromedio.requestFocus();
}

// Ingrera datos
void ingresarDatos() {
    categoria = cboCategoria.getSelectedIndex();
    promedio = Double.parseDouble(txtPromedio.getText());
}

// Calcula la pensión actual
void calcularPensionActual() {
    if (categoria == 0)
        actualpen = 550;
    else if (categoria == 1)
        actualpen = 500;
    else if (categoria == 2)
        actualpen = 460;
    else
        actualpen = 400;
}
```

```

// Calcula el descuento
void calcularDescuento() {
    if (promedio <= 13.99)
        descuento = 0;
    else if (promedio <= 15.99)
        descuento = 0.10 * actualpen;
    else if (promedio <= 17.99)
        descuento = 0.12 * actualpen;
    else
        descuento = 0.15 * actualpen;
}

// Calcula la nueva pensión
void calcularNuevaPension() {
    nuevapen = actualpen - descuento;
}

// Muestra resultados
void mostrarResultados() {
    txtS.setText("");
    imprimir("Pensión actual : S/. " + actualpen);
    imprimir("Descuento : S/. " + descuento);
    imprimir("Nueva pensión : S/. " + nuevapen);
}

// Imprime una línea de texto incluyendo un salto de línea
void imprimir(String cad) {
    txtS.append(cad + "\n");
}
}

```

## Problema 2

Una empresa evalúa a sus empleados bajo dos criterios: puntualidad y rendimiento. En cada caso, el empleado recibe un puntaje que va de 1 a 10, de acuerdo con los siguientes criterios:

**Puntaje por puntualidad:**- está en función de los minutos de tardanza de acuerdo con la siguiente tabla:

Minutos de tardanza	Puntaje
0	10
1 a 2	8
3 a 5	6
6 a 9	4
Más de 9	0

**Puntaje por rendimiento:**- está en función de la cantidad de observaciones efectuadas al empleado por no cumplir sus obligaciones de acuerdo con la siguiente tabla:

Observaciones efectuadas	Puntaje
0	10
1	8
2	5
3	1
Más de 3	0

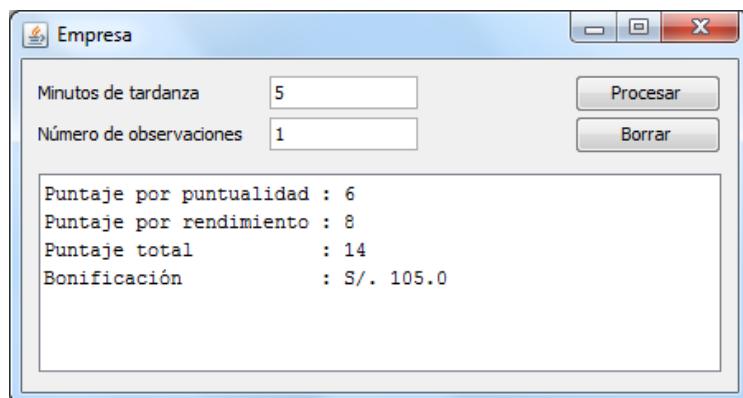
El puntaje total del empleado es la suma del puntaje por puntuabilidad más el puntaje por rendimiento. Basándose en el puntaje total, el empleado recibe una bonificación anual de acuerdo con la siguiente tabla:

Puntaje total	Bonificación
Menos de 11	S/. 2.5 por punto
11 a 13	S/. 5.0 por punto
14 a 16	S/. 7.5 por punto
17 a 19	S/. 10.0 por punto
20	S/. 12.5 por punto

Dados los minutos de tardanza y el número de observaciones de un empleado, diseñe un programa que determine el puntaje por puntuabilidad, el puntaje por rendimiento, el puntaje total y la bonificación que le corresponden.

Declare todas las variables como globales y use métodos tipo void.

### Interfaz Gráfica



### Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
```

```
public class Empresa extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblMinutosTardanza;
    private JLabel lblNumeroObservaciones;
    private JTextField txtMinutosTardanza;
    private JTextField txtNumeroObservaciones;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;
```

```
// Declaración de variables globales
int minutosTar, numeroObs, puntajePun, puntajeRen, puntajeTot;
double bonificacion;
// ----

// Lanza la aplicación
public static void main(String[] args) {
    try {

        UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Empresa frame = new Empresa();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Empresa() {

        setTitle("Empresa");
        setBounds(100, 100, 450, 239);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        lblNumeroObservaciones = new JLabel("Número de observaciones");
        lblNumeroObservaciones.setBounds(10, 38, 140, 14);
        getContentPane().add(lblNumeroObservaciones);

        lblMinutosTardanza = new JLabel("Minutos de tardanza");
        lblMinutosTardanza.setBounds(10, 13, 140, 14);
        getContentPane().add(lblMinutosTardanza);

        txtMinutosTardanza = new JTextField();
        txtMinutosTardanza.setBounds(150, 10, 90, 20);
        getContentPane().add(txtMinutosTardanza);
        txtMinutosTardanza.setColumns(10);

        txtNumeroObservaciones = new JTextField();
        txtNumeroObservaciones.setBounds(150, 35, 90, 20);
        getContentPane().add(txtNumeroObservaciones);
        txtNumeroObservaciones.setColumns(10);

        btnProcesar = new JButton("Procesar");
        btnProcesar.addActionListener(this);
        btnProcesar.setBounds(335, 9, 89, 23);
        getContentPane().add(btnProcesar);

        btnBorrar = new JButton("Borrar");
        btnBorrar.addActionListener(this);
        btnBorrar.setBounds(335, 34, 89, 23);
        getContentPane().add(btnBorrar);

        scpScroll = new JScrollPane();
        scpScroll.setBounds(10, 69, 414, 120);
        getContentPane().add(scpScroll);

        txtS = new JTextArea();
        txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
        scpScroll.setViewportView(txtS);
    }
}
```

```
// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {

    if (arg0.getSource() == btnProcesar) {

        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }

}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    ingresarDatos();
    determinarPuntajePuntualidad();
    determinarPuntajeRendimiento();
    determinarPuntajeTotal();
    determinarBonificacion();
    mostrarResultados();
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtMinutosTardanza.setText("");
    txtNumeroObservaciones.setText("");
    txtS.setText("");
    txtMinutosTardanza.requestFocus();
}

// Efectúa la entrada de datos
void ingresarDatos() {
    minutosTar = Integer.parseInt(txtMinutosTardanza.getText());
    numeroObs = Integer.parseInt(txtNumeroObservaciones.getText());
}

// Determina el puntaje por puntualidad
void determinarPuntajePuntualidad() {
    if (minutosTar == 0)
        puntajePun = 10;
    else if (minutosTar <= 2)
        puntajePun = 8;
    else if (minutosTar <= 5)
        puntajePun = 6;
    else if (minutosTar <= 9)
        puntajePun = 4;
    else
        puntajePun = 0;
}

// Determina el puntaje por rendimiento
void determinarPuntajeRendimiento() {
    if (numeroObs == 0)
        puntajeRen = 10;
    else if (numeroObs == 1)
        puntajeRen = 8;
    else if (numeroObs == 2)
        puntajeRen = 5;
    else if (numeroObs == 3)
        puntajeRen = 1;
    else
        puntajeRen = 0;
}

// Determina el puntaje total
void determinarPuntajeTotal() {
    puntajeTot = puntajePun + puntajeRen;
}
```

```

// Determina la bonificación
void determinarBonificacion() {
    if (puntajeTot < 11)
        bonificacion = 2.5 * puntajeTot;
    else if (puntajeTot <= 13)
        bonificacion = 5.0 * puntajeTot;
    else if (puntajeTot <= 16)
        bonificacion = 7.5 * puntajeTot;
    else if (puntajeTot <= 19)
        bonificacion = 10.0 * puntajeTot;
    else
        bonificacion = 12.5 * puntajeTot;
}

// Muestra los resultados
void mostrarResultados() {
    txtS.setText("");
    imprimir("Puntaje por puntualidad : " + puntajePun);
    imprimir("Puntaje por rendimiento : " + puntajeRen);
    imprimir("Puntaje total : " + puntajeTot);
    imprimir("Bonificación : S/. " + bonificacion);
}

// Imprime una línea de texto incluyendo un salto de línea al final
void imprimir(String cad) {
    txtS.append(cad + "\n");
}
}

```

### Problema 3

Una dulcería vende chocolates a los precios dados en la siguiente tabla:

Tipo de chocolate	Precio unitario
Primor	S/. 8.5
Dulzura	S/. 10.0
Tentación	S/. 7.0
Explosión	S/. 12.5

Como oferta, la tienda aplica un porcentaje de descuento sobre el importe de la compra, basándose en la cantidad de chocolates adquiridos, de acuerdo con la siguiente tabla:

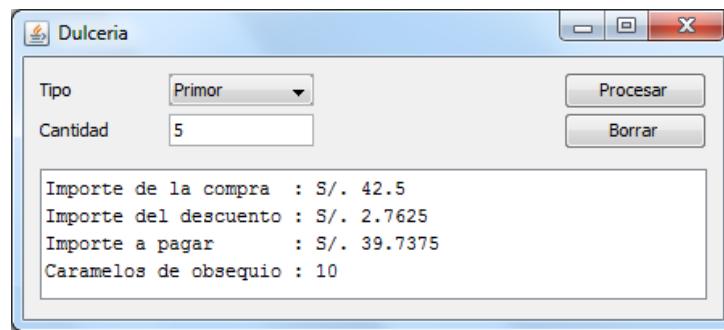
Cantidad de chocolates	Descuento
< 5	4.0%
≥ 5 y < 10	6.5%
≥ 10 y < 15	9.0%
≥ 15	11.5%

Adicionalmente, si el importe a pagar es no menor de S/. 250, la tienda obsequia 3 caramelos por cada chocolate; en caso contrario, obsequia 2 caramelos por cada chocolate.

Dado el tipo de chocolate y la cantidad de unidades adquiridas, diseñe un programa que determine el importe de la compra, el importe del descuento, el importe a pagar y la cantidad de caramelos de obsequio.

Declare todas las variables como globales y use métodos tipo void.

## Interfaz Gráfica



## Solución

```

package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JComboBox;
import javax.swing.DefaultComboBoxModel;

public class Dulceria extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblTipo;
    private JLabel lblCantidad;
    private JComboBox<String> cboTipo;
    private JTextField txtCantidad;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Declaración de variables globales
    int tipo, cantidad, caramelos;
    double impcom, impdes, imppag;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Dulceria frame = new Dulceria();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}

```

```
// Crea la GUI
public Dulceria() {
    setTitle("Dulceria");
    setBounds(100, 100, 450, 239);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblTipo = new JLabel("Tipo");
    lblTipo.setBounds(10, 13, 80, 14);
    getContentPane().add(lblTipo);

    lblCantidad = new JLabel("Cantidad");
    lblCantidad.setBounds(10, 38, 80, 14);
    getContentPane().add(lblCantidad);

    cboTipo = new JComboBox<String>();
    cboTipo.setModel(
        new DefaultComboBoxModel<String>(new String[] { "Primor",
"Dulzura", "Tentación", "Explosión" }));
    cboTipo.setBounds(90, 10, 90, 20);
    getContentPane().add(cboTipo);

    txtCantidad = new JTextField();
    txtCantidad.setBounds(90, 35, 90, 20);
    getContentPane().add(txtCantidad);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(335, 34, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 69, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    ingresarDatos();
    calcularImporteCompra();
    calcularImporteDescuento();
    calcularImportePagar();
    calcularCaramelos();
    mostrarResultados();
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    cboTipo.setSelectedIndex(0);
    txtCantidad.setText("");
    txtS.setText("");
    txtCantidad.requestFocus();
}
```

```
// Efectúa la entrada de datos
void ingresarDatos() {
    tipo = cboTipo.getSelectedIndex();
    cantidad = Integer.parseInt(txtCantidad.getText());
}

// Calcula el importe de la compra
void calcularImporteCompra() {
    switch (tipo) {
        case 0:
            impcom = 8.5 * cantidad;
            break;
        case 1:
            impcom = 10.0 * cantidad;
            break;
        case 2:
            impcom = 7.0 * cantidad;
            break;
        default:
            impcom = 12.5 * cantidad;
    }
}

// Calcula el importe del descuento
void calcularImporteDescuento() {
    if (cantidad < 5)
        impdes = 0.04 * impcom;
    else if (cantidad < 10)
        impdes = 0.065 * impcom;
    else if (cantidad < 15)
        impdes = 0.09 * impcom;
    else
        impdes = 0.115 * impcom;
}

// Calcula el importe a pagar
void calcularImportePagar() {
    imppag = impcom - impdes;
}

// Calcula los caramelos de obsequio
void calcularCaramelos() {
    if (imppag < 250)
        caramelos = 2 * cantidad;
    else
        caramelos = 3 * cantidad;
}

// Muestra los resultados obtenidos
void mostrarResultados() {
    txtS.setText("");
    imprimir("Importe de la compra : S/. " + impcom);
    imprimir("Importe del descuento : S/. " + impdes);
    imprimir("Importe a pagar : S/. " + imppag);
    imprimir("Caramelos de obsequio : " + caramelos);
}

// Imprime una línea de texto incluyendo un salto de línea al final
void imprimir(String cad) {
    txtS.append(cad + "\n");
}
```

### Problema 4

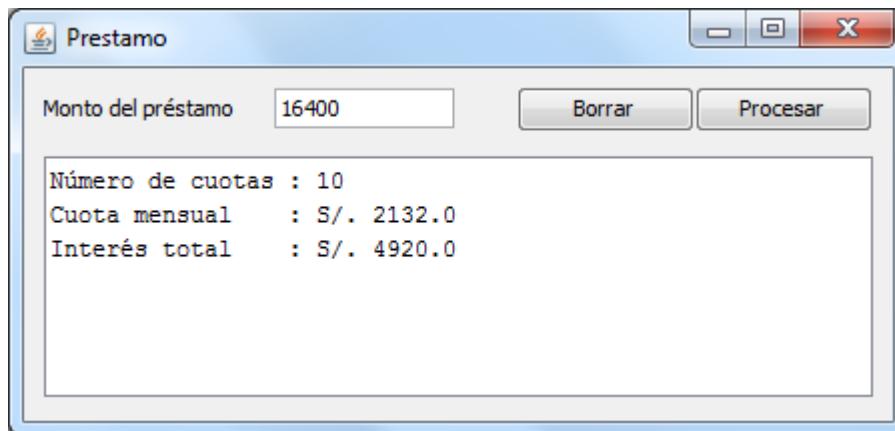
Una empresa de préstamos tiene el siguiente esquema de cobros:

Monto del préstamo (S/.)	Número de cuotas
Hasta 5000	2
Más de 5000 hasta 10000	4
Más de 10000 hasta 15000	6
Más de 15000	10

Si el monto del préstamo es mayor de S/. 10000, la empresa cobra 3% de interés mensual; en caso contrario, cobra 5% de interés mensual.

Dado el monto del préstamo de un cliente, diseñe un programa que determine el número de cuotas, el monto de la cuota mensual y el monto del interés total entre todas las cuotas.

### Interfaz Gráfica



### Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
```

```
public class Prestamo extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblMontoPrestamo;
    private JTextField txtMontoPrestamo;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;
```

```
// Declaración de variables globales
double montoprestamo, montointeres, tasainterest, montocuota;
int cuotas;

// Lanza la aplicación
public static void main(String[] args) {
    try {
        UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Prestamo frame = new Prestamo();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

// Crea la GUI
public Prestamo() {
    setTitle("Prestamo");
    setBounds(100, 100, 450, 214);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    lblMontoPrestamo = new JLabel("Monto del préstamo");
    lblMontoPrestamo.setBounds(10, 13, 115, 14);
    getContentPane().add(lblMontoPrestamo);

    txtMontoPrestamo = new JTextField();
    txtMontoPrestamo.setBounds(125, 10, 90, 20);
    getContentPane().add(txtMontoPrestamo);
    txtMontoPrestamo.setColumns(10);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(246, 9, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 44, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {

    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}
```

```
// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    ingresarDatos();
    obtenerNumeroCuotas();
    obtenerTasaInteres();
    calcularMontoInteresTotal();
    calcularMontoCuota();
    mostrarResultados();
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtMontoPrestamo.setText("");
    txtS.setText("");
    txtMontoPrestamo.requestFocus();
}

// Efectúa la entrada de datos
void ingresarDatos() {
    montoprestamo = Double.parseDouble(txtMontoPrestamo.getText());
}

// Obtiene el número de cuotas
void obtenerNumeroCuotas() {
    if (montoprestamo <= 5000)
        cuotas = 2;
    else if (montoprestamo <= 10000)
        cuotas = 4;
    else if (montoprestamo <= 15000)
        cuotas = 6;
    else
        cuotas = 10;
}

// Obtiene ka tasa de interés
void obtenerTasaInteres() {
    if (montoprestamo > 10000)
        tasainterres = 0.03;
    else
        tasainterres = 0.05;
}

// Calcula el monto del interés total
void calcularMontoInteresTotal() {
    montointeres = tasainterres * montoprestamo * cuotas;
}

// Calcula el monto de la cuota}
void calcularMontoCuota() {
    montocuota = (montoprestamo + montointeres) / cuotas;
}

// Muestra los resultados obtenidos
void mostrarResultados() {
    txtS.setText("");
    imprimir("Número de cuotas : " + cuotas);
    imprimir("Cuota mensual : S/. " + montocuota);
    imprimir("Interés total : S/. " + montointeres);
}

// Imprime una línea de texto incluyendo un salto de línea al final
void imprimir(String cad) {
    txtS.append(cad + "\n");
}
```

# Problemas propuestos

## Actividad

- Una tienda vende tres tipos de productos a los precios dados en la siguiente tabla:

Producto	Precio
P1	S/. 15.0
P2	S/. 17.5
P3	S/. 20.0

Como oferta la tienda ofrece un regalo de acuerdo con la siguiente tabla:

Unidades adquiridas	Regalo
Menos de 26	Un lapicero
26 a 50	Un cuaderno
Más de 50	Una agenda

Dado el tipo de producto y la cantidad de unidades adquiridas, diseñe un programa que determine el importe a pagar y el regalo.

Declare todas las variables como globales y use métodos tipo void.

- En un parque de diversiones los precios de los juegos se dan en la siguiente tabla:

Juego	Precio
El gusanito	S/. 3.5
El trencito	S/. 5.0
El pulpo	S/. 2.5

Dado el juego elegido y la cantidad de boletos adquiridos, diseñe un programa que determine el importe a pagar.

Declare todas las variables como globales y use métodos tipo void.

## Autoevaluación

- Una empresa de transportes, que cubre la ruta Lima-Huánuco, brinda servicio en tres turnos a los precios por pasaje dados en la siguiente tabla:

Turno	Precio del pasaje (S.)
Mañana	30.0
Tarde	35.0
Noche	42.5

Como oferta, la empresa aplica un porcentaje de descuento sobre el importe compra de acuerdo con la siguiente tabla:

Cantidad de pasajes	Descuento
1 a 5	12%
6 a 10	14%
Más de 10	16%

Dado el turno y la cantidad de pasajes adquiridos, diseñe un programa que determine el importe de la compra, el importe del descuento y el importe a pagar correspondientes.

Declare todas las variables como globales y use los siguientes métodos tipo void:

- actionPerformedBtnProcesar:** método del botón Procesar
- ingresarDatos:** efectúa la entrada de datos
- calcularImporteCompra:** calcula el importe de la compra
- calcularImporteDescuento:** calcula el importe del descuento
- calcularImportePagar:** calcula el importe a pagar
- mostrarResultados:** muestra los resultados solicitados.

- Una empresa paga a sus vendedores un sueldo bruto que es igual a la suma de un sueldo básico más una comisión.

El sueldo básico se obtiene de la siguiente tabla, basándose en la categoría del vendedor.

Categoría	Sueldo básico (S.)
E1	2500
E2	2250
E3	2000

La comisión se obtiene de la siguiente tabla, basándose en el importe vendido en el mes:

Importe vendido	Comisión
$\geq 9000$	11%
$\geq 6000 \text{ y } < 9000$	9%
$\geq 3000 \text{ y } < 6000$	7%
$< 3000$	5%

Por ley, todo vendedor está sujeto a un descuento igual al 15% del sueldo bruto.

Dado el importe vendido en el mes y la categoría, diseñe un programa que determine el sueldo básico, la comisión, el sueldo bruto, el descuento y el sueldo neto correspondientes.

Declare todas las variables como globales y use los siguientes métodos tipo void:

- **actionPerformedBtnProcesar:** método del botón Procesar
- **ingresarDatos:** efectúa la entrada de datos
- **calcularSueldoBasico:** calcula el sueldo básico
- **calcularComision:** calcula la comisión
- **calcularSueldoBruto:** calcula el sueldo bruto
- **calcularDescuento:** calcula el descuento
- **calcularSueldoNeto:** calcula el sueldo neto
- **mostrarResultados:** muestra los resultados solicitados.

3. Una empresa calcula el sueldo bruto de sus empleados multiplicando las horas trabajadas por una tarifa horaria que depende de la categoría del trabajador de acuerdo con la siguiente tabla:

Categoría	Tarifa
A	S/. 21.0
B	S/. 19.5
C	S/. 17.0
D	S/. 15.5

Por ley, todo trabajador se somete a un porcentaje de descuento sobre el sueldo bruto de acuerdo con la siguiente tabla:

Sueldo bruto	Descuento
> 2500	20% del sueldo bruto
≤ 2500	15% del sueldo bruto

Dada la categoría y el número de horas trabajadas, diseñe un programa que determine el sueldo bruto, el descuento y el sueldo neto correspondientes.

Declare todas las variables como globales y use los siguientes métodos tipo void.

- **actionPerformedBtnProcesar:** método del botón Procesar
- **ingresarDatos:** efectúa la entrada de datos
- **calcularSueldoBruto:** calcula el sueldo bruto
- **calcularDescuento:** calcula el descuento
- **calcularSueldoNeto:** calcula el sueldo neto
- **mostrarResultados:** muestra los resultados solicitados

## Para recordar

- Una variable local sólo puede ser utilizada dentro del método donde fue declarada.
- Una variable global puede ser utilizada en todo el programa.
- Las variables locales se crean vacías.
- Las variables globales se crean y se inicializan automáticamente.

#### 4.1.4. Métodos con valor de retorno

Un método con valor de retorno es un módulo de programa que puede recibir datos de entrada a través de variables locales denominadas parámetros y que retorna un resultado al punto donde es invocado. Este tipo de método se utiliza para efectuar cualquier tipo de proceso que produzca un resultado

Estos métodos pueden dividirse a su vez en dos tipos:

- Métodos con valor de retorno sin parámetros
- Métodos con valor de retorno con parámetros

##### Métodos con valor de retorno sin parámetros

Este tipo de método no recibe datos de entrada a través de parámetros; pero retorna un valor al punto donde es invocado.

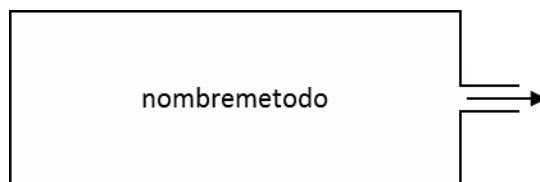


Figura 1 Método con valor de retorno sin parámetros

##### Definición

Este tipo de método se define de la siguiente manera:

```
tipoRetorno nombreMetodo() {
    Declaración de variables locales
    Cuerpo del método
    return valor;
}
```

Donde:

nombreMetodo	: Es el nombre del método.
tipoRetorno	: Es el tipo del valor de retorno.
valor	: Es el valor de retorno.

##### Llamada

Este tipo de método se invoca de la siguiente manera:

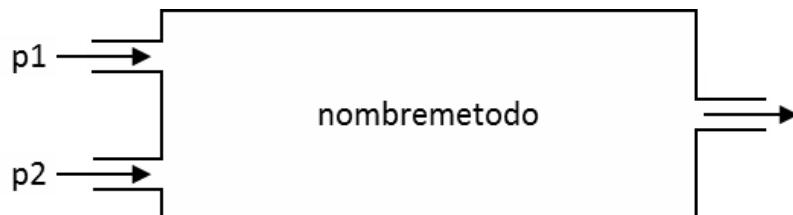
```
variable = nombreMetodo();
```

Donde:

nombremetodo	: Es el nombre del método invocado.
variable	: Es la variable que recibe el valor de retorno.

## Métodos con valor de retorno con parámetros

Este tipo de método recibe datos de entrada a través de parámetros y retorna un resultado al punto donde es invocado. En la Figura 2, p1 y p2 son parámetros.



**Figura 2** Método con valor de retorno y con parámetros

### Definición

Este tipo de método se define de la siguiente manera:

```

tipoRetorno nombreMetodo(tipo1 p1, tipo2 p2, tipo2 p3, ...) {
    Declaración de variables locales
    Cuerpo del método
    return valor;
}
  
```

Donde:

- |                          |                                      |
|--------------------------|--------------------------------------|
| nombreMetodo             | : Es el nombre del método.           |
| tipoRetorno              | : Es el tipo del valor de retorno.   |
| p1, p2, p3, ...          | : Son los nombres de los parámetros. |
| tipo1, tipo2, tipo3, ... | : Son los tipos de los parámetros.   |
| valor                    | : Es el valor de retorno.            |

### Llamada

Este tipo de método se invoca de la siguiente manera:

```

variable = nombreMetodo(v1, v2, v3, ...);
  
```

Donde:

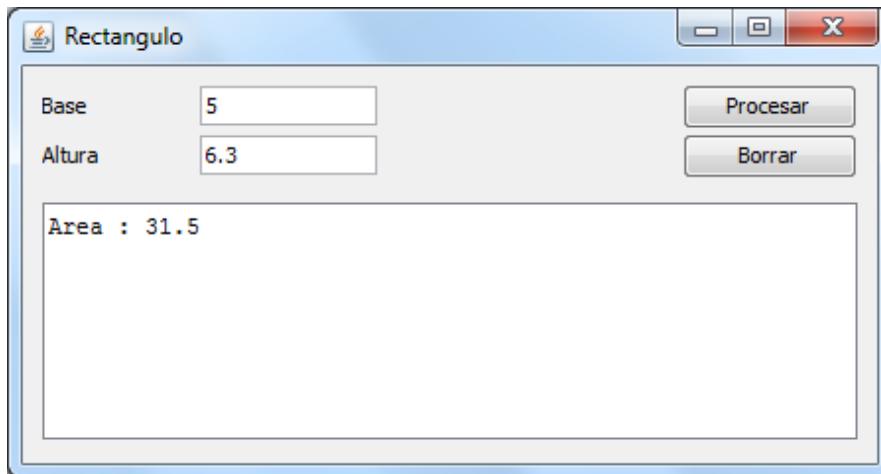
- |                 |  |
|-----------------|--|
| nombreMetodo    | : Es el nombre del método invocado.              |
| variable        | : Es la variable que recibe el valor de retorno. |
| v1, v2, v3, ... | : Son los valores dados a los parámetros.        |

### Problemas resueltos

#### Problema 1

Diseñe un programa que determine el área de un rectángulo. Declare todas las variables como locales y use métodos para la entrada de datos, para el cálculo del área y para la salida de resultados.

## Interfaz Gráfica



### Solución 1

```

package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Rectangulo extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblBase;
    private JLabel lblAltura;
    private JTextField txtBase;
    private JTextField txtAltura;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Rectangulo frame = new Rectangulo();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}

```

```
// Crea la GUI
public Rectangulo() {

    setTitle("Rectangulo");
    setBounds(100, 100, 450, 239);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblBase = new JLabel("Base");
    lblBase.setBounds(10, 13, 80, 14);
    getContentPane().add(lblBase);

    lblAltura = new JLabel("Altura");
    lblAltura.setBounds(10, 38, 80, 14);
    getContentPane().add(lblAltura);

    txtBase = new JTextField();
    txtBase.setBounds(90, 10, 90, 20);
    getContentPane().add(txtBase);
    txtBase.setColumns(10);

    txtAltura = new JTextField();
    txtAltura.setBounds(90, 35, 90, 20);
    getContentPane().add(txtAltura);
    txtAltura.setColumns(10);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(335, 34, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 69, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declaración de variables locales
    double b, h, arearec;

    // Llama a getBase
    // getBase envía como retorno la base del rectángulo
    // El valor retornado es almacenado en b
    b = getBase();

    // Llama a getAltura
    // getAltura envía como retorno la altura del rectángulo
    // El valor retornado es almacenado en h
    h = getAltura();
```

```
// Llama a calcularArea y le envía los valores de b y h
// El valor de b es recibido por el parámetro bas de calcularArea
// El valor de h es recibido por el parámetro alt de calcularArea
// calcularArea retorna el área del rectángulo
// El valor returned es almacenado en arearec
arearec = calcularArea(b, h);

// Llama a mostrarArea y le envía el valor de arearec
// El valor de arearec es recibido por el parámetro are
// mostrarArea retorna sin ningún resultado
// Un método que no retorna un resultado es de tipo void
mostrarArea(arearec);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtBase.setText("");
    txtAltura.setText("");
    txtS.setText("");
    txtBase.requestFocus();
}

// Lee y retorna la base
// Dado que la base es de tipo double, el tipo de retorno es double
double getBase() {
    return Double.parseDouble(txtBase.getText());
}

// Lee y retorna la altura
// Dado que la altura es de tipo double, el tipo de retorno es double
double getAltura() {
    return Double.parseDouble(txtAltura.getText());
}

// Calcula y retorna el área
// El parámetro bas recibirá el valor de la base
// El parámetro alt recibirá el valor de la altura
// Dado que el área es de tipo double, el tipo de retorno es double
// Los valores son enviados desde actionPerformedBtnProcesar
double calcularArea(double bas, double alt) {
    return bas * alt;
}

// Muestra el área
// El parámetro are recibirá el valor del área
// El valor del área es enviado desde actionPerformedBtnProcesar
// El método no retorna ningún resultado; por lo que, es de tipo void
void mostrarArea(double are) {
    txtS.setText("Area : " + are);
}
}
```

## Solución 2

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Rectangulo extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblBase;
    private JLabel lblAltura;
    private JTextField txtBase;
    private JTextField txtAltura;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Rectangulo frame = new Rectangulo();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Rectangulo() {

        setTitle("Rectangulo");
        setBounds(100, 100, 450, 239);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        lblBase = new JLabel("Base");
        lblBase.setBounds(10, 13, 80, 14);
        getContentPane().add(lblBase);

        lblAltura = new JLabel("Altura");
        lblAltura.setBounds(10, 38, 80, 14);
        getContentPane().add(lblAltura);

        txtBase = new JTextField();
        txtBase.setBounds(90, 10, 90, 20);
        getContentPane().add(txtBase);
        txtBase.setColumns(10);
```

```
txtAltura = new JTextField();
txtAltura.setBounds(90, 35, 90, 20);
getContentPane().add(txtAltura);
txtAltura.setColumns(10);

btnProcesar = new JButton("Procesar");
btnProcesar.addActionListener(this);
btnProcesar.setBounds(335, 9, 89, 23);
getContentPane().add(btnProcesar);

btnBorrar = new JButton("Borrar");
btnBorrar.addActionListener(this);
btnBorrar.setBounds(335, 34, 89, 23);
getContentPane().add(btnBorrar);

scpScroll = new JScrollPane();
scpScroll.setBounds(10, 69, 414, 120);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declaración de variables locales
    double b, h, arearec;

    // Llama a getBase
    // getBase envía como retorno la base del rectángulo
    // El valor returned es almacenado en b
    b = getBase();

    // Llama a getAltura
    // getAltura envía como retorno la altura del rectángulo
    // El valor returned es almacenado en h
    h = getAltura();

    // Llama a calcularArea y le envía los valores de b y h
    // El valor de b es recibido por el parámetro bas de calcularArea
    // El valor de h es recibido por el parámetro alt de calcularArea
    // calcularArea retorna el área del rectángulo
    // Dado que el área es de tipo double, el tipo de retorno es double
    // El valor returned es almacenado en arearec
    arearec = calcularArea(b, h);

    // Llama a mostrarArea y le envía el valor de arearec
    // El valor de arearec es recibido por el parámetro are
    // mostrarArea retorna sin ningún resultado
    // Un método que no retorna un resultado es de tipo void
    mostrarArea(arearec);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtBase.setText("");
    txtAltura.setText("");
    txtS.setText("");
    txtBase.requestFocus();
}
```

```

// Lee y retorna la base
// Dado que la base es de tipo double, el tipo de retorno es double
// Declara una variable local xb para guardar la base temporalmente
// Almacena la base en xb
// Retorna la base almacenada en xb
double getBase() {
    double xb;
    xb = Double.parseDouble(txtBase.getText());

    return xb;
}

// Lee y retorna la altura
// Dado que la altura es de tipo double, el tipo de retorno es double
// Declara una variable local xh para guardar la altura temporalmente
// Almacena la altura en xh
// Retorna la altura almacenada en xh
double getAltura() {
    double xh;

    xh = Double.parseDouble(txtAltura.getText());
    return xh;
}

// Calcula y retorna el área
// El parámetro bas recibirá el valor de la base
// El parámetro alt recibirá el valor de la altura
// Los valores son enviados desde actionPerformedBtnProcesar
// Declara una variable local xa para guardar el área temporalmente
// Almacena el área en xa
// Retorna el área almacenada en xa
double calcularArea(double bas, double alt) {
    double xa;
    xa = bas * alt;

    return xa;
}

// Muestra el área
// El parámetro are recibirá el valor del área
// El valor del área es enviado desde actionPerformedBtnProcesar
// El método no retorna ningún resultado; por lo que, es de tipo void
void mostrarArea(double are) {
    txtS.setText("Área : " + are);
}
}

```

## Problema 2

Una dulcería vende chocolates a los precios dados en la siguiente tabla:

<b>Tipo de chocolate</b>	<b>Precio unitario</b>
Primor	S/. 8.5
Dulzura	S/. 10.0
Tentación	S/. 7.0
Explosión	S/. 12.5

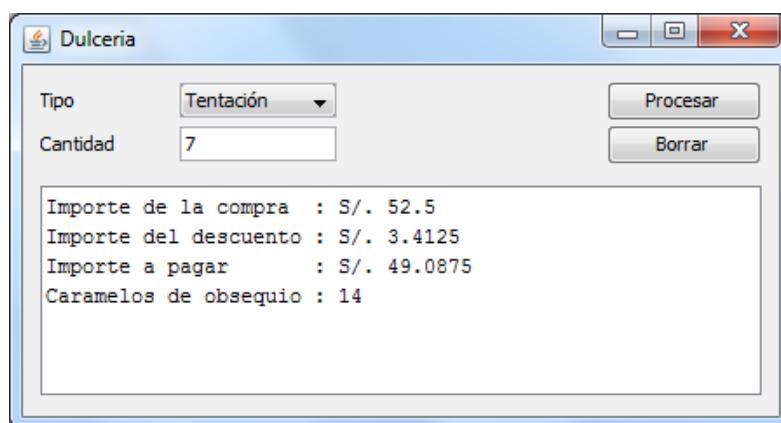
Como oferta, la tienda aplica un porcentaje de descuento sobre el importe de la compra, basándose en la cantidad de chocolates adquiridos, de acuerdo con la siguiente tabla:

Cantidad de chocolates	Descuento
< 5	4.0%
≥ 5 y < 10	6.5%
≥ 10 y < 15	9.0%
≥ 15	11.5%

Adicionalmente, si el importe a pagar es no menor de S/. 250, la tienda obsequia 3 caramelos por cada chocolate; en caso contrario, obsequia 2 caramelos por cada chocolate.

Dado el tipo de chocolate y la cantidad de unidades adquiridas, diseñe un programa que determine el importe de la compra, el importe del descuento, el importe a pagar y la cantidad de caramelos de obsequio.

### Interfaz Gráfica



### Solución 1

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JComboBox;
import javax.swing.DefaultComboBoxModel;

public class Dulceria extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblTipo;
    private JLabel lblCantidad;
    private JComboBox<String> cboTipo;
    private JTextField txtCantidad;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;
```

```
// Lanza la aplicación
public static void main(String[] args) {
    try {
        UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
    } catch (Throwable e) {
        e.printStackTrace();
    }
    EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                Dulceria frame = new Dulceria();
                frame.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

// Crea la GUI
public Dulceria() {
    setTitle("Dulceria");
    setBounds(100, 100, 450, 239);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblTipo = new JLabel("Tipo");
    lblTipo.setBounds(10, 13, 80, 14);
    getContentPane().add(lblTipo);

    lblCantidad = new JLabel("Cantidad");
    lblCantidad.setBounds(10, 38, 80, 14);
    getContentPane().add(lblCantidad);

    cboTipo = new JComboBox<String>();
    cboTipo.setModel(
        new DefaultComboBoxModel<String>(new String[] { "Primor",
"Dulzura", "Tentación", "Explosión" }));
    cboTipo.setBounds(90, 10, 90, 20);
    getContentPane().add(cboTipo);

    txtCantidad = new JTextField();
    txtCantidad.setBounds(90, 35, 90, 20);
    getContentPane().add(txtCantidad);
    txtCantidad.setColumns(10);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(335, 34, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 69, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
}
```

```
        if (arg0.getSource() == btnBorrar) {
            actionPerformedBtnBorrar(arg0);
        }

    // Procesa la pulsación del botón Procesar
    protected void actionPerformedBtnProcesar(ActionEvent arg0) {

        // Declaración de variables locales
        int tipo, cantidad, caramelos;
        double impcom, impdes, imppag;

        // Entrada de datos
        tipo = getTipo();
        cantidad = getCantidad();

        // Proceso de cálculo
        impcom = calcularImporteCompra(tipo, cantidad);
        impdes = calcularImporteDescuento(cantidad, impcom);
        imppag = calcularImportePagar(impcom, impdes);
        caramelos = calcularCaramelos(cantidad, imppag);

        // Salida de resultados
        mostrarResultados(impcom, impdes, imppag, caramelos);
    }

    // Procesa la pulsación del botón Borrar
    protected void actionPerformedBtnBorrar(ActionEvent arg0) {
        cboTipo.setSelectedIndex(0);
        txtCantidad.setText("");
        txtS.setText("");
        txtCantidad.requestFocus();
    }

    // Lee y retorna el tipo de chocolate
    int getTipo() {
        return cboTipo.getSelectedIndex();
    }

    // Lee y retorna la cantidad de chocolates
    int getCantidad() {
        return Integer.parseInt(txtCantidad.getText());
    }

    // Calcula y retorna el importe de la compra
    double calcularImporteCompra(int tip, int can) {
        switch (tip) {
        case 0:
            return 8.5 * can;
        case 1:
            return 10.0 * can;
        case 2:
            return 7.5 * can;
        default:
            return 12.5 * can;
        }
    }

    // Calcula y retorna el importe del descuento
    double calcularImporteDescuento(int can, double ic) {
        if (can < 5)
            return 0.04 * ic;
        else if (can < 10)
            return 0.065 * ic;
        else if (can < 15)
            return 0.09 * ic;
        else
            return 0.115 * ic;
    }
}
```

```

// Calcula y retorna el importe a pagar
double calcularImportePagar(double ic, double id) {
    return ic - id;
}

// Calcula y retorna los caramelos de obsequio
int calcularCaramelos(int can, double ip) {

    if (ip < 250)
        return 2 * can;
    else
        return 3 * can;

}

// Muestra los resultados obtenidos
void mostrarResultados(double ic, double id, double ip, int car) {
    txtS.setText("");
    imprimir("Importe de la compra : S/. " + ic);
    imprimir("Importe del descuento : S/. " + id);
    imprimir("Importe a pagar : S/. " + ip);
    imprimir("Caramelos de obsequio : " + car);
}

// Imprime una línea de texto incluyendo un salto de línea al final
void imprimir(String cad) {
    txtS.append(cad + "\n");
}
}

```

## Solución 2

```

package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JComboBox;
import javax.swing.DefaultComboBoxModel;

public class Dulceria extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblTipo;
    private JLabel lblCantidad;
    private JComboBox<String> cboTipo;
    private JTextField txtCantidad;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
    }
}

```

```
EventQueue.invokeLater(new Runnable() {
    public void run() {
        try {
            Dulceria frame = new Dulceria();
            frame.setVisible(true);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
});

// Crea la GUI
public Dulceria() {

    setTitle("Dulceria");
    setBounds(100, 100, 450, 239);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblTipo = new JLabel("Tipo");
    lblTipo.setBounds(10, 13, 80, 14);
    getContentPane().add(lblTipo);

    lblCantidad = new JLabel("Cantidad");
    lblCantidad.setBounds(10, 38, 80, 14);
    getContentPane().add(lblCantidad);

    cboTipo = new JComboBox<String>();
    cboTipo.setModel(
        new DefaultComboBoxModel<String>(new String[] { "Primor",
"Dulzura", "Tentación", "Explosión" }));
    cboTipo.setBounds(90, 10, 90, 20);
    getContentPane().add(cboTipo);

    txtCantidad = new JTextField();
    txtCantidad.setBounds(90, 35, 90, 20);
    getContentPane().add(txtCantidad);
    txtCantidad.setColumns(10);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(335, 34, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 69, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);

}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}
```

```
// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declaración de variables locales
    int tipo, cantidad, caramelos;
    double impcom, impdes, imppag;

    // Entrada de datos
    tipo = getTipo();
    cantidad = getCantidad();

    // Proceso de cálculo
    impcom = calcularImporteCompra(tipo, cantidad);
    impdes = calcularImporteDescuento(cantidad, impcom);
    imppag = calcularImportePagar(impcom, impdes);
    caramelos = calcularCaramelos(cantidad, imppag);

    // Salida de resultados
    mostrarResultados(impcom, impdes, imppag, caramelos);

}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {

    cboTipo.setSelectedIndex(0);
    txtCantidad.setText("");
    txtS.setText("");
    txtCantidad.requestFocus();

}

// Lee y retorna el tipo de chocolate
int getTipo() {

    int xt;
    xt = cboTipo.getSelectedIndex();

    return xt;

}

// Lee y retorna la cantidad de chocolates
int getCantidad() {

    int xc;
    xc = Integer.parseInt(txtCantidad.getText());

    return xc;

}

// Calcula y retorna el importe de la compra
double calcularImporteCompra(int tip, int can) {
    double xic;

    switch (tip) {
    case 0:
        xic = 8.5 * can;
        break;
    case 1:
        xic = 10.0 * can;
        break;
    case 2:
        xic = 7.5 * can;
        break;
    default:
        xic = 12.5 * can;
    }

    return xic;
}
```

```

// Calcula y retorna el importe del descuento
double calcularImporteDescuento(int can, double ic) {
    double xid;

    if (can < 5)
        xid = 0.04 * ic;
    else if (can < 10)
        xid = 0.065 * ic;
    else if (can < 15)
        xid = 0.09 * ic;
    else
        xid = 0.115 * ic;

    return xid;
}

// Calcula y retorna el importe a pagar
double calcularImportePagar(double ic, double id) {
    double xip;
    xip = ic - id;

    return xip;
}

// Calcula y retorna los caramelos de obsequio
int calcularCaramelos(int can, double ip) {
    int xcar;
    if (ip < 250)
        xcar = 2 * can;
    else
        xcar = 3 * can;

    return xcar;
}

// Muestra los resultados obtenidos
void mostrarResultados(double ic, double id, double ip, int car) {
    txtS.setText("");
    imprimir("Importe de la compra : S/. " + ic);
    imprimir("Importe del descuento : S/. " + id);
    imprimir("Importe a pagar : S/. " + ip);
    imprimir("Caramelos de obsequio : " + car);
}

// Imprime una línea de texto incluyendo un salto de línea al final
void imprimir(String cad) {
    txtS.append(cad + "\n");
}
}

```

### Problema 3

En una universidad, los alumnos están clasificados en cuatro categorías. A cada categoría le corresponde una pensión mensual distinta dada en la siguiente tabla:

Categoría	Pensión
A	S/. 550
B	S/. 500
C	S/. 460
D	S/. 400

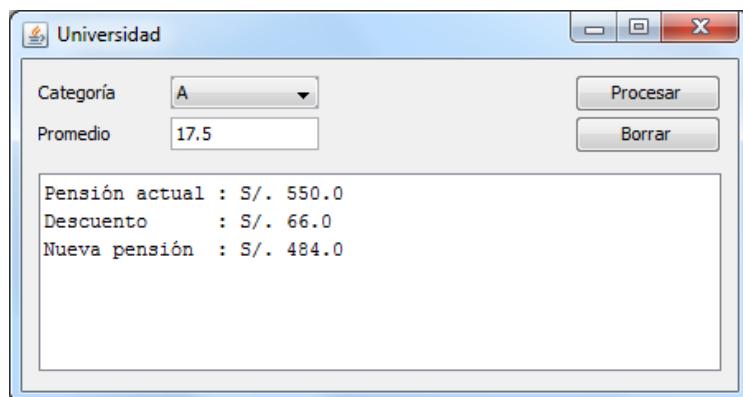
Semestralmente, la universidad efectúa rebajas en las pensiones de sus estudiantes a partir del segundo ciclo basándose en el promedio ponderado del ciclo anterior en porcentajes dados en la tabla siguiente:

Promedio	Descuento
0 a 13.99	No hay descuento
14.00 a 15.99	10 %
16.00 a 17.99	12 %
18.00 a 20.00	15 %

Dado el promedio ponderado y la categoría de un estudiante, diseñe un programa que determine cuánto de rebaja recibirá sobre su pensión actual y a cuánto asciende su nueva pensión.

Declare todas las variables como globales y use métodos tipo void.

### Interfaz Gráfica



### Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JComboBox;
import javax.swing.DefaultComboBoxModel;

public class Universidad extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblCategoria;
    private JLabel lblPromedio;
    private JComboBox<String> cboCategoria;
    private JTextField txtPromedio;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;
```

```
// Lanza la aplicación
public static void main(String[] args) {
    try {
        UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
    } catch (Throwable e) {
        e.printStackTrace();
    }
    EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                Universidad frame = new Universidad();
                frame.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

// Crea la GUI
public Universidad() {

    setTitle("Universidad");
    setBounds(100, 100, 450, 239);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblCategoria = new JLabel("Categoría");
    lblCategoria.setBounds(10, 13, 80, 14);
    getContentPane().add(lblCategoria);

    lblPromedio = new JLabel("Promedio");
    lblPromedio.setBounds(10, 38, 80, 14);
    getContentPane().add(lblPromedio);

    cboCategoria = new JComboBox<String>();
    cboCategoria.setModel(new DefaultComboBoxModel<String>(new String[] {
"A", "B", "C", "D }));
    cboCategoria.setBounds(90, 10, 90, 20);
    getContentPane().add(cboCategoria);

    txtPromedio = new JTextField();
    txtPromedio.setBounds(90, 35, 90, 20);
    getContentPane().add(txtPromedio);
    txtPromedio.setColumns(10);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(335, 34, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 69, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
}
```

```
if (arg0.getSource() == btnBorrar) {
    actionPerformedBtnBorrar(arg0);
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declaración de variables globales
    int categoria;
    double actualpen, nuevapen, descuento, promedio;

    // Entrada de datos
    categoria = getCategoría();
    promedio = getPromedio();

    // Proceso de cálculo
    actualpen = calcularPensionActual(categoría);
    descuento = calcularDescuento(promedio, actualpen);
    nuevapen = calcularNuevaPension(actualpen, descuento);

    // Salida de resultados
    mostrarResultados(actualpen, descuento, nuevapen);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtPromedio.setText("");
    txtS.setText("");
    txtPromedio.requestFocus();
}

// Lee y retorna la categoría
int getCategoría() {
    return cboCategoría.getSelectedIndex();
}

// Lee y retorna el promedio ponderado
double getPromedio() {
    return Double.parseDouble(txtPromedio.getText());
}

// Calcula y retorna la pensión actual
double calcularPensionActual(int cat) {
    switch (cat) {
        case 0:
            return 550.0;
        case 1:
            return 500.0;
        case 2:
            return 460.0;
        default:
            return 400.0;
    }
}

// Calcula y retorna el descuento
double calcularDescuento(double pro, double apen) {
    if (pro <= 13.99)
        return 0;
    else if (pro <= 15.99)
        return 0.10 * apen;
    else if (pro <= 17.99)
        return 0.12 * apen;
    else
        return 0.15 * apen;
}

// Calcula y retorna la nueva pensión
double calcularNuevaPension(double apen, double des) {
    return apen - des;
}
```

```

// Muestra los resultados obtenidos
void mostrarResultados(double apen, double des, double npen) {
    txtS.setText("");
    imprimir("Pensión actual : S/. " + apen);
    imprimir("Descuento : S/. " + des);
    imprimir("Nueva pensión : S/. " + npen);
}

// Imprime una línea de texto incluyendo un salto de línea
void imprimir(String cad) {
    txtS.append(cad + "\n");
}
}

```

#### Problema 4

Una empresa evalúa a sus empleados bajo dos criterios: puntualidad y rendimiento. En cada caso el empleado recibe un puntaje que va de 1 a 10, de acuerdo con los siguientes criterios:

**Puntaje por puntualidad:**- está en función de los minutos de tardanza de acuerdo con la siguiente tabla:

Minutos de tardanza	Puntaje
0	10
1 a 2	8
3 a 5	6
6 a 9	4
Más de 9	0

**Puntaje por rendimiento:**- está en función de la cantidad de observaciones efectuadas al empleado por no cumplir sus obligaciones de acuerdo con la siguiente tabla:

Observaciones efectuadas	Puntaje
0	10
1	8
2	5
3	1
Más de 3	0

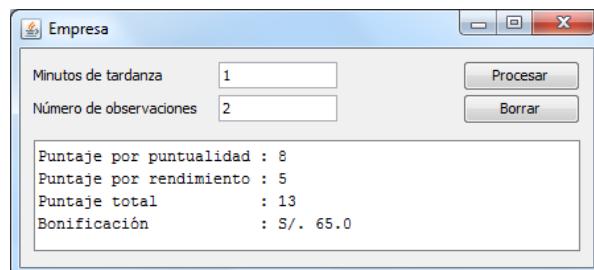
El puntaje total del empleado es la suma del puntaje por puntualidad más el puntaje por rendimiento. Basándose en el puntaje total, el empleado recibe una bonificación anual de acuerdo con la siguiente tabla:

Puntaje total	Bonificación
Menos de 11	S/. 2.5 por punto
11 a 13	S/. 5.0 por punto
14 a 16	S/. 7.5 por punto
17 a 19	S/. 10.0 por punto
20	S/. 12.5 por punto

Dados los minutos de tardanza y el número de observaciones de un empleado, diseñe un programa que determine el puntaje por puntualidad, el puntaje por rendimiento, el puntaje total y la bonificación que le corresponden.

Declare todas las variables como globales y use métodos tipo void.

### Interfaz Gráfica



### Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Empresa extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblMinutosTardanza;
    private JLabel lblNumeroObservaciones;
    private JTextField txtMinutosTardanza;
    private JTextField txtNumeroObservaciones;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Empresa frame = new Empresa();
                    frame.setVisible(true);

                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}
```

```
// Crea la GUI
public Empresa() {

    setTitle("Empresa");
    setBounds(100, 100, 450, 239);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblNumeroObservaciones = new JLabel("Nº de observaciones");
    lblNumeroObservaciones.setBounds(10, 38, 140, 14);
    getContentPane().add(lblNumeroObservaciones);

    lblMinutosTardanza = new JLabel("Minutos de tardanza");
    lblMinutosTardanza.setBounds(10, 13, 140, 14);
    getContentPane().add(lblMinutosTardanza);

    txtMinutosTardanza = new JTextField();
    txtMinutosTardanza.setBounds(150, 10, 90, 20);
    getContentPane().add(txtMinutosTardanza);
    txtMinutosTardanza.setColumns(10);

    txtNumeroObservaciones = new JTextField();
    txtNumeroObservaciones.setBounds(150, 35, 90, 20);
    getContentPane().add(txtNumeroObservaciones);
    txtNumeroObservaciones.setColumns(10);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(335, 34, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 69, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declaración de variables globales
    int minutosTar, numeroObs, puntajePun, puntajeRen, puntajeTot;
    double bonificacion;

    // Entrada de datos
    minutosTar = getMinutosTardanza();
    numeroObs = getNumeroObservaciones();

    // Proceso de cálculo
    puntajePun = determinarPuntajePuntualidad(minutosTar);
    puntajeRen = determinarPuntajeRendimiento(numeroObs);
    puntajeTot = determinarPuntajeTotal(puntajePun, puntajeRen);
    bonificacion = determinarBonificacion(puntajeTot);
```

```
// Salida de resultados
mostrarResultados(puntajePun, puntajeRen, puntajeTot, bonificacion);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {

    txtMinutosTardanza.setText("");
    txtNumeroObservaciones.setText("");
    txtS.setText("");
    txtMinutosTardanza.requestFocus();

}

// Lee y retorna los minutos de tardanza
int getMinutosTardanza() {
    return Integer.parseInt(txtMinutosTardanza.getText());
}

// Lee y retorna el número de observaciones
int getNumeroObservaciones() {
    return Integer.parseInt(txtNumeroObservaciones.getText());
}

// Determina y retorna el puntaje por puntualidad
int determinarPuntajePuntualidad(int min) {
    if (min == 0)
        return 10;
    else if (min <= 2)
        return 8;
    else if (min <= 5)
        return 6;
    else if (min <= 9)
        return 4;
    else
        return 0;
}

// Determina y retorna el puntaje por rendimiento
int determinarPuntajeRendimiento(int nob) {
    if (nob == 0)
        return 10;
    else if (nob == 1)
        return 8;
    else if (nob == 2)
        return 5;
    else if (nob == 3)
        return 1;
    else
        return 0;
}

// Determina y retorna el puntaje total
int determinarPuntajeTotal(int pp, int pr) {
    return pp + pr;
}

// Determina y retorna la bonificación
double determinarBonificacion(int pt) {
    if (pt < 11)
        return 2.5 * pt;
    else if (pt <= 13)
        return 5.0 * pt;
    else if (pt <= 16)
        return 7.5 * pt;
    else if (pt <= 19)
        return 10.0 * pt;
    else
        return 12.5 * pt;
}
```

```

// Muestra los resultados
void mostrarResultados(int pp, int pr, int pt, double bo) {
    txtS.setText("");
    imprimir("Puntaje por puntualidad : " + pp);
    imprimir("Puntaje por rendimiento : " + pr);
    imprimir("Puntaje total : " + pt);
    imprimir("Bonificación : S/. " + bo);
}

// Imprime una línea de texto incluyendo un salto de línea al final
void imprimir(String cad) {
    txtS.append(cad + "\n");
}
}

```

### Problema 5

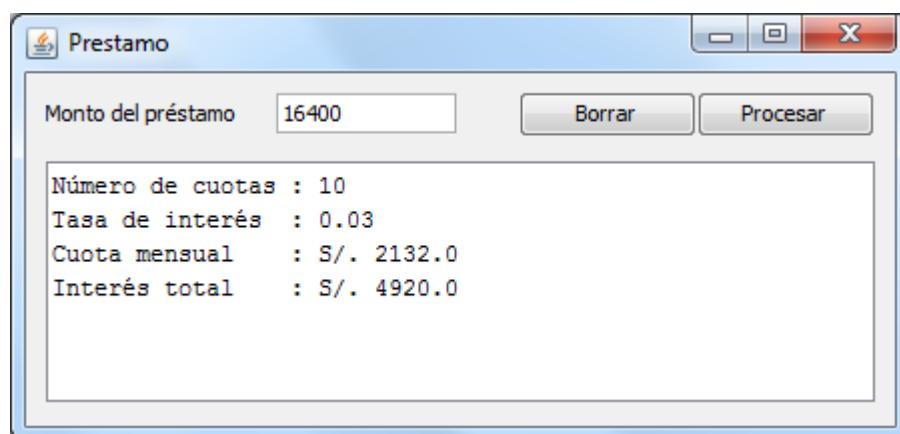
Una empresa de préstamos tiene el siguiente esquema de cobros:

Monto del préstamo (S.)	Número de cuotas
Hasta 5000	2
Más de 5000 hasta 10000	4
Más de 10000 hasta 15000	6
Más de 15000	10

Si el monto del préstamo es mayor de S/. 10000, la empresa cobra 3% de interés mensual; en caso contrario, cobra 5% de interés mensual.

Dado el monto del préstamo de un cliente, diseñe un programa que determine el número de cuotas, el monto de la cuota mensual y el monto del interés total entre todas las cuotas.

### Interfaz Gráfica



## Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Prestamo extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblMontoPrestamo;
    private JTextField txtMontoPrestamo;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Prestamo frame = new Prestamo();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Prestamo() {
        setTitle("Prestamo");
        setBounds(100, 100, 450, 214);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        btnProcesar = new JButton("Procesar");
        btnProcesar.addActionListener(this);
        btnProcesar.setBounds(335, 9, 89, 23);
        getContentPane().add(btnProcesar);

        lblMontoPrestamo = new JLabel("Monto del préstamo");
        lblMontoPrestamo.setBounds(10, 13, 115, 14);
        getContentPane().add(lblMontoPrestamo);

        txtMontoPrestamo = new JTextField();
        txtMontoPrestamo.setBounds(125, 10, 90, 20);
        getContentPane().add(txtMontoPrestamo);
        txtMontoPrestamo.setColumns(10);

        btnBorrar = new JButton("Borrar");
        btnBorrar.addActionListener(this);
        btnBorrar.setBounds(246, 9, 89, 23);
        getContentPane().add(btnBorrar);
    }
}
```

```
scpScroll = new JScrollPane();
scpScroll.setBounds(10, 44, 414, 120);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);

}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declaración de variables globales
    double montopres, montointe, tasainte, montocuo;
    int cuotas;

    // Entrada de datos
    montopres = getMontoPrestamo();

    // Proceso de cálculo
    cuotas = obtenerNumeroCuotas(montopres);
    tasainte = obtenerTasaInteres(montopres);
    montointe = calcularMontoInteresTotal(tasainte, montopres, cuotas);
    montocuo = calcularMontoCuota(montopres, montointe, cuotas);

    // Salida de resultados
    mostrarResultados(cuotas, tasainte, montointe, montocuo);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtMontoPrestamo.setText("");
    txtS.setText("");
    txtMontoPrestamo.requestFocus();
}

// Lee y retorna el monto del préstamo
double getMontoPrestamo() {
    return Double.parseDouble(txtMontoPrestamo.getText());
}

// Obtiene y retorna el número de cuotas
int obtenerNumeroCuotas(double mpre) {
    if (mpre <= 5000)
        return 2;
    else if (mpre <= 10000)
        return 4;
    else if (mpre <= 15000)
        return 6;
    else
        return 10;
}

// Obtiene y retorna la tasa de interés
double obtenerTasaInteres(double mpre) {
    if (mpre > 10000)
        return 0.03;
    else
        return 0.05;
}
```

```
// Calcula y retorna el monto del interés total
double calcularMontoInteresTotal(double tint, double mpre, int ncuo) {
    return tint * mpre * ncuo;
}

// Calcula y retorna el monto de la cuota
double calcularMontoCuota(double mpre, double mint, int ncuo) {
    return (mpre + mint) / ncuo;
}

// Muestra los resultados obtenidos
void mostrarResultados(int ncuo, double tint, double mint, double mcuo) {
    txtS.setText("");
    imprimir("Número de cuotas : " + ncuo);
    imprimir("Tasa de interés : " + tint);
    imprimir("Cuota mensual : S/. " + mcuo);
    imprimir("Interés total : S/. " + mint);
}

// Imprime una línea de texto incluyendo un salto de linea al final
void imprimir(String cad) {
    txtS.append(cad + "\n");
}
}
```

# Problemas propuestos

## Actividad

- Una tienda vende tres tipos de productos a los precios dados en la siguiente tabla:

Producto	Precio
P1	S/. 15.0
P2	S/. 17.5
P3	S/. 20.0

Como oferta, la tienda ofrece un regalo de acuerdo con la siguiente tabla:

Unidades adquiridas	Regalo
Menos de 26	Un lapicero
26 a 50	Un cuaderno
Más de 50	Una agenda

Dado el tipo de producto y la cantidad de unidades adquiridas, diseñe un programa que determine el importe a pagar y el regalo.

Declare todas las variables como locales y use métodos con valor de retorno

- En un parque de diversiones los precios de los juegos se dan en la siguiente tabla:

Juego	Precio
El gusanito	S/. 3.5
El trencito	S/. 5.0
El pulpo	S/. 2.5

Dado el juego elegido y la cantidad de boletos adquiridos, diseñe un programa que determine el importe a pagar.

Declare todas las variables como locales y use métodos con valor de retorno.

## Autoevaluación

1. Una empresa de transportes, que cubre la ruta Lima-Huánuco, brinda servicio en tres turnos a los precios por pasaje dados en la siguiente tabla:

Turno	Precio del pasaje (S.)
Mañana	30.0
Tarde	35.0
Noche	42.5

Como oferta, la empresa obsequia caramelos de acuerdo con la siguiente tabla:

Pasajes	Caramelos
Más de 10	4 caramelos por cada pasaje
6 a 10	3 caramelos por cada pasaje
1 a 5	2 caramelos por cada pasaje

Dado el turno y la cantidad de pasajes adquiridos, diseñe un programa que determine el importe a pagar y la cantidad de caramelos de regalo.

Declare todas las variables como locales y use los siguientes métodos:

- **actionPerformedBtnProcesar:** método del botón Procesar
- **getTurno:** lee y retorna el turno elegido
- **getCantidad:** lee y retorna la cantidad de pasajes adquiridos
- **calcularImportePagar:** calcula y retorna el importe a pagar
- **calcularCaramelos:** calcula y retorna la cantidad de caramelos
- **mostrarResultados:** muestra los resultados solicitados

2. Una empresa calcula el sueldo bruto de sus empleados multiplicando las horas trabajadas por una tarifa horaria que depende de la categoría del trabajador de acuerdo con la siguiente tabla:

Categoría	Tarifa
A	S/. 21.0
B	S/. 19.5
C	S/. 17.0
D	S/. 15.5

Como incentivo, la empresa obsequia gorros veraniegos basándose en la cantidad de horas trabajadas de acuerdo con la siguiente tabla:

Horas trabajadas	Gorros de obsequio
< 120	2
≥ 120 y < 140	4
≥ 140 y < 160	7
≥ 160	11

Dada la categoría y el número de horas trabajadas, diseñe un programa que determine el sueldo bruto y la cantidad de gorros de obsequio.

Declare todas las variables como locales y use los siguientes métodos.

- **actionPerformedBtnProcesar:** método del botón Procesar
- **getCategoria:** lee y retorna la categoría
- **getHoras:** lee y retorna las horas trabajadas
- **calcularSueldoBruto:** calcula y retorna el sueldo bruto
- **obtenerGorros:** obtiene y retorna la cantidad de gorros
- **mostrarResultados:** muestra los resultados solicitados

3. Los ángulos se clasifican de la siguiente manera:

Magnitud	Clasificación
$\beta = 0^\circ$	Nulo
$0^\circ < \beta < 90^\circ$	Agudo
$\beta = 90^\circ$	Recto
$90^\circ < \beta < 180^\circ$	Obtuso
$\beta = 180^\circ$	Llano
$180^\circ < \beta < 360^\circ$	Cóncavo
$\beta = 360^\circ$	Completo

Dada la medida de un ángulo en grados, diseñe un programa que determine la clasificación del ángulo.

Declare todas las variables como globales y use los siguientes métodos:

- **actionPerformedBtnProcesar:** método del botón Procesar
- **getAngulo:** lee y retorna la medida del ángulo
- **determinarClasificacion:** determina y retorna la clasificación del ángulo
- **mostrarClasificacion:** muestra la clasificación del ángulo

## Para recordar

- Un método con valor de retorno es un método que puede recibir datos de entrada a través de variables locales al método conocidas como parámetros y que retorna un valor al punto donde es invocado.



# CONTADORES Y ACUMULADORES

## LOGRO DE LA UNIDAD DE APRENDIZAJE

Al finalizar la unidad, el alumno diseña programas que involucran procesos de conteo y acumulación mediante el uso de variables locales y globales.

## TEMARIO

- 5.1 Tema 5 : Contadores y Acumuladores**
  - 5.1.1 : Operadores de incremento y decremento
  - 5.1.2 : Operadores de asignación compleja
  - 5.1.3 : Contadores y acumuladores

## ACTIVIDADES PROPUESTAS

- Los alumnos desarrollan programas usando contadores y acumuladores.

## 5.1. TEMA 5: CONTADORES Y ACUMULADORES

### 5.1.1. Operadores de incremento y decremento

Son operadores que permiten incrementar o decrementar en una unidad el valor de una variable numérica.

Operador	Uso	Equivalencia
<code>++</code>	<code>a++;</code>	<code>a = a + 1;</code>
<code>--</code>	<code>a--;</code>	<code>a = a - 1;</code>

#### Ejemplo 1

```
// Incrementa en uno el valor de x (Forma 1)
x = x + 1;

// Incrementa en uno el valor de x (Forma 2)
x++;

// Decrementa en 1 el valor de la variable z (Forma 1)
z = z - 1;

// Decrementa en 1 el valor de la variable z (Forma 2)
z--;
```

### 5.1.2. Operadores de asignación compleja

Son operadores que permiten asignar a una variable el valor de la variable más, menos, por o entre el valor de otra variable.

Operador	Ejemplo	Equivalencia
<code>+=</code>	<code>a += b;</code>	<code>a = a + b;</code>
<code>-=</code>	<code>a -= b;</code>	<code>a = a - b;</code>
<code>*=</code>	<code>a *= b;</code>	<code>a = a * b;</code>
<code>/=</code>	<code>a /= b;</code>	<code>a = a / b;</code>

#### Ejemplo 2

```
// Incrementa en 2 el valor de la variable z (Forma 1)
z = z + 2;

// Incrementa en 2 el valor de la variable z (Forma 2)
z += 2;

// Decrementa en 5 el valor de la variable m (Forma 1)
m = m - 5;

// Decrementa en 5 el valor de la variable m (Forma 2)
m -= 5;
```

### 5.1.3. Contadores y acumuladores

#### Contadores

Un **contador** es una variable que se utiliza para contar el número de ocurrencias de un suceso o el número de veces que se cumple una determinada condición. El proceso de conteo se efectúa, generalmente, de uno en uno y consiste en incrementar en 1 a la variable conteo cada vez que ocurre el evento o suceso que se pretende contar. Antes de iniciar el proceso de conteo, el contador debe ser inicializado en 0.

Por ejemplo, se necesita un **contador** para determinar:

- La cantidad de veces que se hizo clic en un botón
- La cantidad de notas ingresadas
- La cantidad de notas desaprobatorias
- La cantidad de ventas efectuadas
- La cantidad de ventas efectuadas

Una instrucción de conteo tiene la siguiente forma:

```
contador = contador + 1;
```

Que puede escribirse también como:

```
contador++;
```

#### Ejemplo 3

```
// Incrementa la cantidad de alumnos aprobados de una sección  
cantidadAprobados++;  
  
// Incrementa la cantidad de ventas efectuadas en un día  
cantidadVentasDia++;
```

#### Acumuladores

Un **acumulador** es una variable que se utiliza para acumular o totalizar cantidades de una misma especie: sueldos, edades, pesos, etc. El proceso de acumulación consiste en incrementar la variable que sirve de acumulador en la cantidad que se pretende acumular. Antes de iniciar el proceso de acumulación, el acumulador debe ser inicializado en 0.

Por ejemplo, se necesita un **acumulador** para determinar:

- El sueldo total de los empleados de una empresa
- La suma total de las notas de un alumno
- La suma total de los pesos de un grupo de personas
- La suma total de las edades de un grupo de personas
- La cantidad total de unidades vendidas de un producto

Una instrucción de acumulación tiene la siguiente forma:

```
acumulador = acumulador + cantidad;
```

Qué puede escribirse también como:

```
acumulador += cantidad;
```

#### Ejemplo 4

```
// Incrementa el monto total vendido  
montoTotalVendido += montoVenta;  
  
// Incrementa el sueldo total de los empleados de una empresa  
sueldoTotalEmpresa += sueldoEmpleado;  
  
// Incrementa la cantidad total de unidades vendidas de un producto  
cantidadUnidadesVendidas += cantidad;  
  
// Incrementa la suma total de notas de un alumno  
sumaNotas += nota;
```

#### Problemas resueltos

##### Problema 1

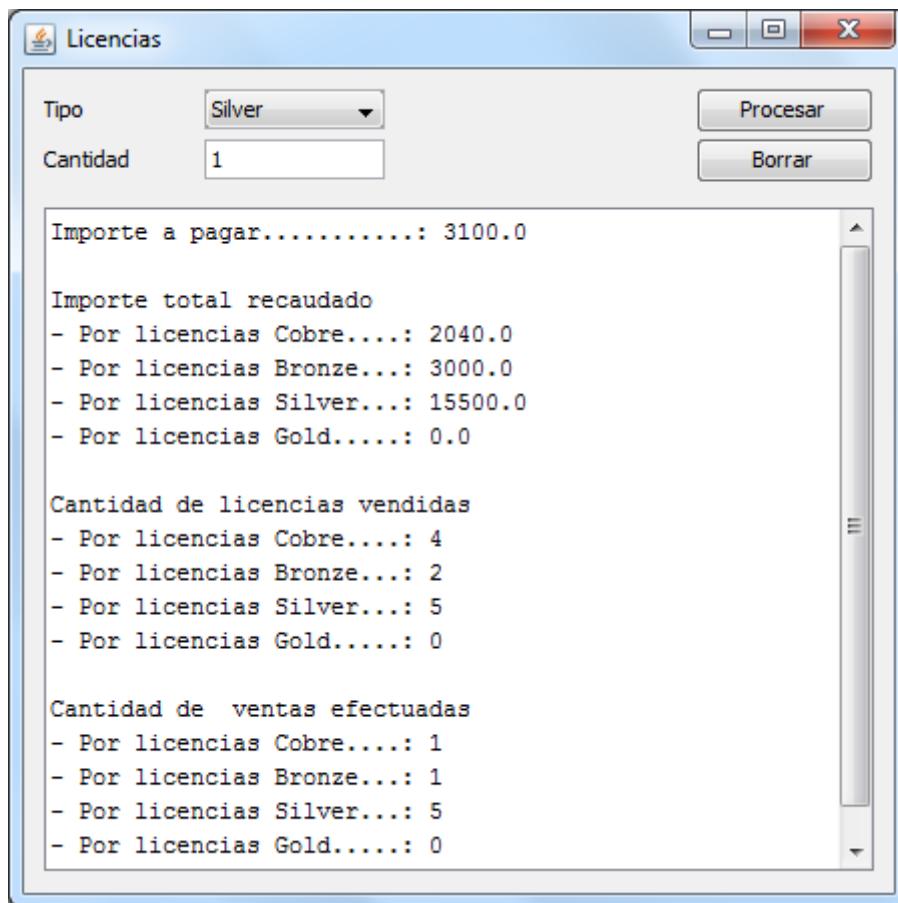
Una empresa desarrolladora de software ha puesto a la venta licencias de su programa de edición de video: Video Edit 2.0 a los siguientes costos unitarios:

Licencia	Costo
Cobre	\$ 510
Bronze	\$ 1500
Silver	\$ 3100
Gold	\$ 4500

Diseñe un programa que permita ingresar, por cada venta, el tipo de licencia y la cantidad de licencias, y muestre luego de cada venta:

- El importe a pagar para la venta efectuada.
- El importe total recaudado de cada tipo de licencia.
- La cantidad de licencias vendidas de cada tipo de licencia.
- La cantidad de ventas efectuadas de cada tipo de licencia.

## Interfaz Gráfica



## Solución

```

package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JComboBox;
import javax.swing.DefaultComboBoxModel;

public class Licencias extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblTipo;
    private JLabel lblCantidad;
    private JComboBox<String> cboTipo;
    private JTextField txtCantidad;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;
}

```

```
// Declaración de variables globales para el algoritmo
double imptot0, imptot1, imptot2, imptot3;
int canlic0, canlic1, canlic2, canlic3;
int canven0, canven1, canven2, canven3;

// Lanza la aplicación
public static void main(String[] args) {
    try {
        UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
    } catch (Throwable e) {
        e.printStackTrace();
    }

    EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                Licencias frame = new Licencias();
                frame.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

// Crea la GUI
public Licencias() {

    setTitle("Licencias");
    setBounds(100, 100, 450, 449);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblTipo = new JLabel("Tipo");
    lblTipo.setBounds(10, 13, 80, 14);
    getContentPane().add(lblTipo);

    lblCantidad = new JLabel("Cantidad");
    lblCantidad.setBounds(10, 38, 80, 14);
    getContentPane().add(lblCantidad);

    cboTipo = new JComboBox<String>();
    cboTipo.setModel(new DefaultComboBoxModel<String>(new String[] {
"Cobre", "Bronze", "Silver", "Gold" }));
    cboTipo.setBounds(90, 10, 90, 20);
    getContentPane().add(cboTipo);

    txtCantidad = new JTextField();
    txtCantidad.setBounds(90, 35, 90, 20);
    getContentPane().add(txtCantidad);
    txtCantidad.setColumns(10);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(335, 34, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 69, 414, 331);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);

}
```

```
// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {

    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    int tipo, cantidad;
    double imppag;

    tipo = getTipo();
    cantidad = getCantidad();
    imppag = calcularImportePagar(tipo, cantidad);

    efectuarIncrementos(tipo, cantidad, imppag);
    mostrarResultados(imppag);

}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtCantidad.setText("");
    txtS.setText("");
    txtCantidad.requestFocus();
}

// Lee y retorna el tipo de licencia
int getTipo() {
    return cboTipo.getSelectedIndex();
}

// Lee y retorna la cantidad de licencias
int getCantidad() {
    return Integer.parseInt(txtCantidad.getText());
}

// Calcula y retorna el importe a pagar
double calcularImportePagar(int tip, int can) {
    switch (tip) {
    case 0:
        return 510 * can;
    case 1:
        return 1500 * can;
    case 2:
        return 3100 * can;
    default:
        return 4500 * can;
    }
}

// Efectúa los incrementos necesarios
void efectuarIncrementos(int tip, int can, double ip) {
    switch (tip) {
    case 0:
        imptot0 += ip;
        canlico0 += can;
        canveno0++;
        break;
    case 1:
        imptot1 += ip;
        canlic1 += can;
        canven1++;
        break;
    }
}
```

```

        case 2:
            imptot2 += ip;
            canlic2 += can;
            canven2++;
            break;
        default:
            imptot3 += ip;
            canlic3 += can;
            canven3++;
    }
}

// Muestra el reporte solicitado
void mostrarResultados(double ip) {
    txtS.setText("");
    imprimir("Importe a pagar.....: " + ip);
    imprimir("");
    imprimir("Importe total recaudado");
    imprimir("- Por licencias Cobre....: " + imptot0);
    imprimir("- Por licencias Bronze....: " + imptot1);
    imprimir("- Por licencias Silver....: " + imptot2);
    imprimir("- Por licencias Gold.....: " + imptot3);
    imprimir("");
    imprimir("Cantidad de licencias vendidas");
    imprimir("- Por licencias Cobre....: " + canlic0);
    imprimir("- Por licencias Bronze....: " + canlic1);
    imprimir("- Por licencias Silver....: " + canlic2);
    imprimir("- Por licencias Gold.....: " + canlic3);
    imprimir("");
    imprimir("Cantidad de ventas efectuadas");
    imprimir("- Por licencias Cobre....: " + canven0);
    imprimir("- Por licencias Bronze....: " + canven1);
    imprimir("- Por licencias Silver....: " + canven2);
    imprimir("- Por licencias Gold.....: " + canven3);
}

// Imprime una cadena con un salto de línea al final
void imprimir(String cad) {
    txtS.append(cad + "\n");
}
}

```

### Problema 2

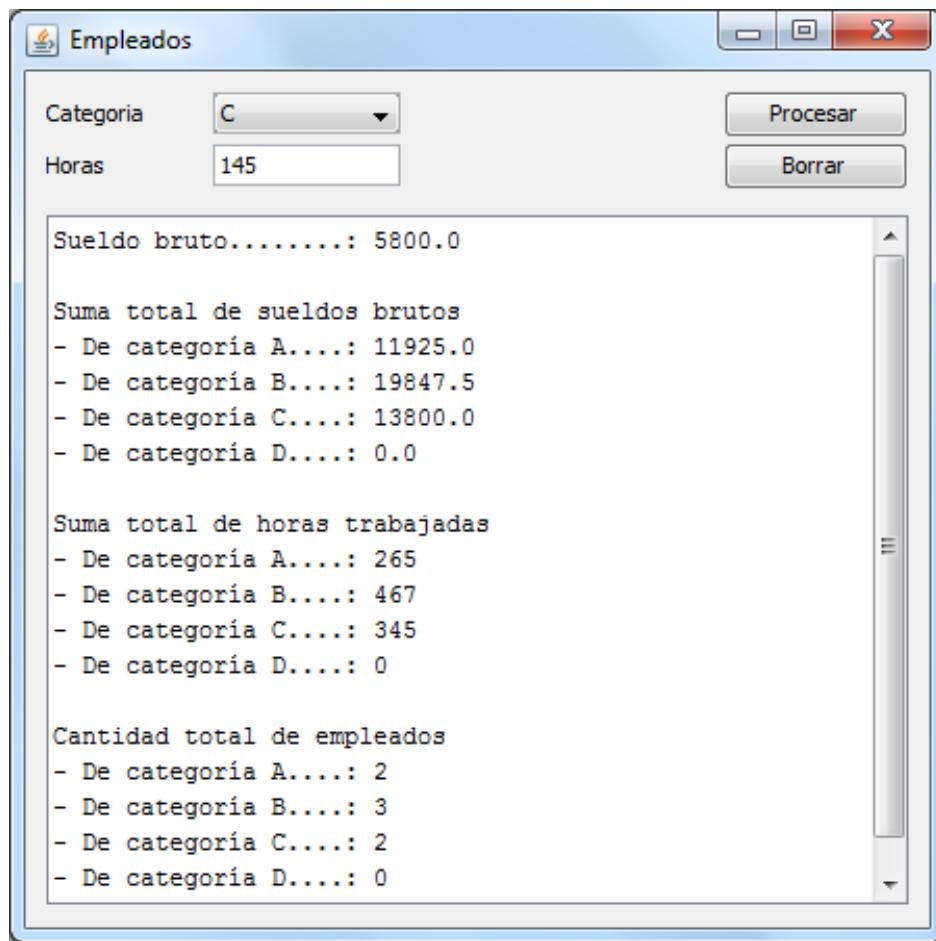
El sueldo bruto de los empleados de una empresa se calcula multiplicando las horas trabajadas por una tarifa horaria que depende de la categoría del empleado de acuerdo con la siguiente tabla:

Categoría	Tarifa
A	45.0
B	42.5
C	40.0
D	37.5

Diseñe un programa que permita ingresar, por cada empleado, la categoría y la cantidad de horas trabajadas, y muestre, luego de cada ingreso:

- El sueldo bruto del empleado
- La suma total de sueldos brutos de cada categoría
- La suma total de horas trabajadas de cada categoría
- La cantidad total de empleados de cada categoría

## Interfaz Gráfica



## Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JComboBox;
import javax.swing.DefaultComboBoxModel;

public class Empleados extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblCategoria;
    private JLabel lblHoras;
    private JComboBox<String> cboCategoria;
    private JTextField txtHoras;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;
```

```
// Declaración de variables globales para el algoritmo
double sbrutot0, sbrutot1, sbrutot2, sbrutot3;
int tothor0, tothor1, tothor2, tothor3;
int canemp0, canempl1, canemp2, canemp3;

// Lanza la aplicación
public static void main(String[] args) {
    try {
        UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
    } catch (Throwable e) {
        e.printStackTrace();
    }

    EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                Empleados frame = new Empleados();
                frame.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

// Crea la GUI
public Empleados() {

    setTitle("Empleados");
    setBounds(100, 100, 450, 449);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblCategoria = new JLabel("Categoria");
    lblCategoria.setBounds(10, 13, 80, 14);
    getContentPane().add(lblCategoria);

    lblHoras = new JLabel("Horas");
    lblHoras.setBounds(10, 38, 80, 14);
    getContentPane().add(lblHoras);

    cboCategoria = new JComboBox<String>();
    cboCategoria.setModel(new DefaultComboBoxModel<String>(new String[] {"A", "B", "C", "D"}));
    cboCategoria.setBounds(90, 10, 90, 20);
    getContentPane().add(cboCategoria);

    txtHoras = new JTextField();
    txtHoras.setBounds(90, 35, 90, 20);
    getContentPane().add(txtHoras);
    txtHoras.setColumns(10);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(335, 34, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 69, 414, 331);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}
}
```

```
// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    int categoria, horas;
    double suelbru;

    categoria = getCategoría();
    horas = getHoras();
    suelbru = calcularSueldoBruto(categoría, horas);

    efectuarIncrementos(categoría, horas, suelbru);
    mostrarResultados(suelbru);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtHoras.setText("");
    txtS.setText("");
    txtHoras.requestFocus();
}

// Lee y retorna la categoría
int getCategoría() {
    return cboCategoria.getSelectedIndex();
}

// Lee y retorna las cantidad de horas trabajadas
int getHoras() {
    return Integer.parseInt(txtHoras.getText());
}

// Calcula y retorna el sueldo bruto
double calcularSueldoBruto(int cat, int hor) {
    switch (cat) {
    case 0:
        return 45.0 * hor;
    case 1:
        return 42.5 * hor;
    case 2:
        return 40.0 * hor;
    default:
        return 37.5 * hor;
    }
}

// Efectúa los incrementos necesarios
void efectuarIncrementos(int cat, int hor, double sb) {
    switch (cat) {
    case 0:
        sbrutot0 += sb;
        tothor0 += hor;
        canemp0++;
        break;
    case 1:
        sbrutot1 += sb;
        tothor1 += hor;
        canemp1++;
        break;
    case 2:
        sbrutot2 += sb;
        tothor2 += hor;
        canemp2++;
        break;
    }
}
```

```

default:
    sbrutot3 += sb;
    tothor3 += hor;
    canemp3++;
}
}

// Muestra el reporte solicitado
void mostrarResultados(double sb) {
    txtS.setText("");
    imprimir("Sueldo bruto.....: " + sb);
    imprimir("");
    imprimir("Suma total de sueldos brutos");
    imprimir("- De categoría A....: " + sbrutot0);
    imprimir("- De categoría B....: " + sbrutot1);
    imprimir("- De categoría C....: " + sbrutot2);
    imprimir("- De categoría D....: " + sbrutot3);
    imprimir("");
    imprimir("Suma total de horas trabajadas");
    imprimir("- De categoría A....: " + tothor0);
    imprimir("- De categoría B....: " + tothor1);
    imprimir("- De categoría C....: " + tothor2);
    imprimir("- De categoría D....: " + tothor3);
    imprimir("");
    imprimir("Cantidad total de empleados");
    imprimir("- De categoría A....: " + canemp0);
    imprimir("- De categoría B....: " + canemp1);
    imprimir("- De categoría C....: " + canemp2);
    imprimir("- De categoría D....: " + canemp3);
}

// Imprime una cadena con un salto de línea al final
void imprimir(String cad) {
    txtS.append(cad + "\n");
}
}
}

```

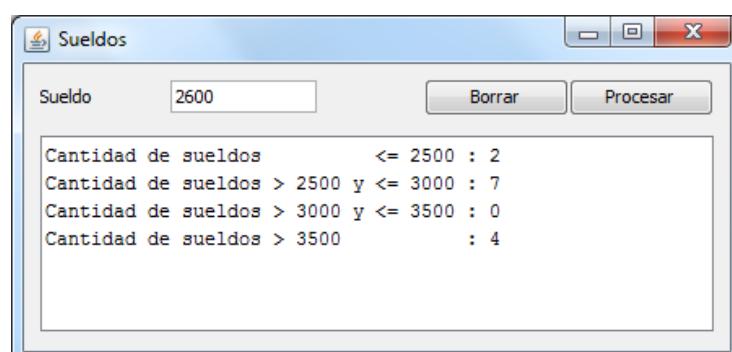
### Problema 3

Diseñe un programa que permita ingresar los sueldos de un conjunto de empleados.

Luego de cada ingreso, muestre un reporte actualizado indicando:

- La cantidad de sueldos menores o iguales que 2500
- La cantidad de sueldos mayores de 2500 pero menores o iguales a 3000
- La cantidad de sueldos mayores de 3000 pero menores o iguales a 3500
- La cantidad de sueldos mayores de 3500

### Interfaz Gráfica



## Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Sueldos extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblSueldo;
    private JTextField txtSueldo;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Declaración de variables globales para el algoritmo
    int conta1, conta2, conta3, conta4;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Sueldos frame = new Sueldos();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Sueldos() {
        setTitle("Sueldos");
        setBounds(100, 100, 450, 214);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        lblSueldo = new JLabel("Dato1");
        lblSueldo.setBounds(10, 13, 80, 14);
        getContentPane().add(lblSueldo);

        txtSueldo = new JTextField();
        txtSueldo.setBounds(90, 10, 90, 20);
        getContentPane().add(txtSueldo);
        txtSueldo.setColumns(10);

        btnProcesar = new JButton("Procesar");
        btnProcesar.addActionListener(this);
        btnProcesar.setBounds(335, 9, 89, 23);
        getContentPane().add(btnProcesar);
    }
}
```

```
btnBorrar = new JButton("Borrar");
btnBorrar.addActionListener(this);
btnBorrar.setBounds(246, 9, 89, 23);
getContentPane().add(btnBorrar);

scpScroll = new JScrollPane();
scpScroll.setBounds(10, 44, 414, 120);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    double sueldo;
    sueldo = getSueldo();

    efectuarIncrementos(sueldo);
    mostrarResultados();
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtSueldo.setText("");
    txtS.setText("");
    txtSueldo.requestFocus();
}

// Lee y retorna el sueldo
double getSueldo() {
    return Double.parseDouble(txtSueldo.getText());
}

// Efectúa los incrementos necesarios
void efectuarIncrementos(double suel) {
    if (suel <= 2500)
        conta1++;
    else if (suel <= 3000)
        conta2++;
    else if (suel <= 3500)
        conta3++;
    else
        conta4++;
}

// Muestra los resultados solicitados
void mostrarResultados() {
    txtS.setText("");
    imprimir("Cantidad de sueldos    <= 2500 : " + conta1);
    imprimir("Cantidad de sueldos > 2500 y <= 3000 : " + conta2);
    imprimir("Cantidad de sueldos > 3000 y <= 3500 : " + conta3);
    imprimir("Cantidad de sueldos > 3500      : " + conta4);
}

// Imprime una cadena con un salto de línea al final
void imprimir(String cad) {
    txtS.append(cad + "\n");
}
}
```

### Problema 4

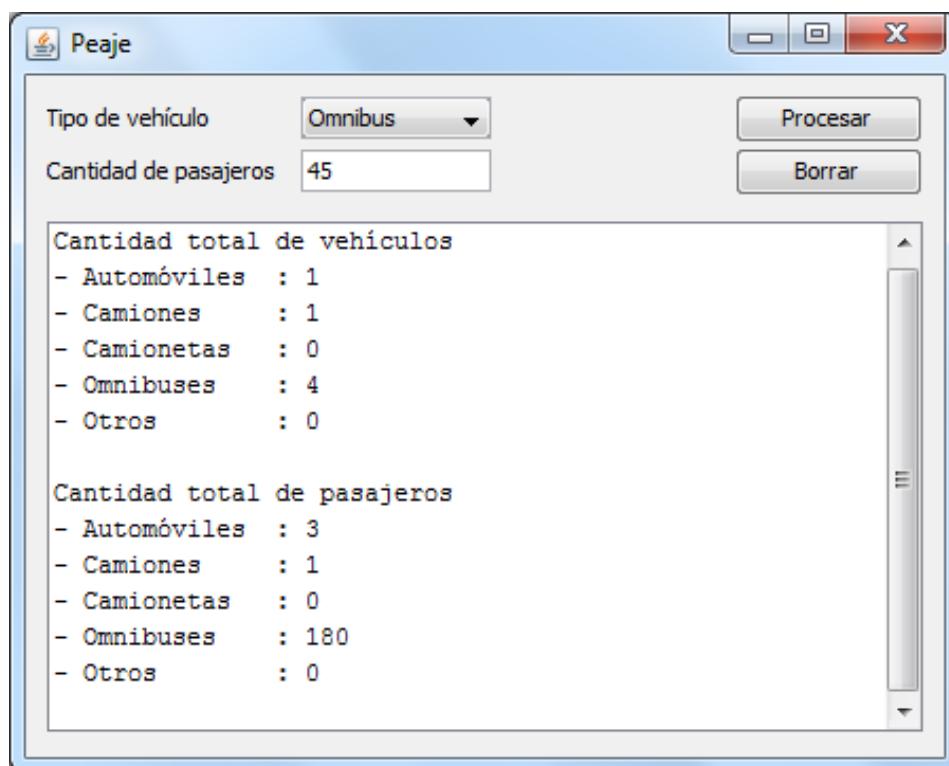
Diseñe un programa que lea, por cada vehículo, que pasa por un peaje, el tipo y la cantidad de pasajeros que transporta.

Muestre, a continuación, un reporte indicando:

- La cantidad total de vehículos de cada tipo que pasaron por el peaje
- La cantidad total de pasajeros por cada tipo de vehículo

Los tipos de vehículos por considerar son: automóvil, camión, camioneta, ómnibus y otros.

### Interfaz Gráfica



### Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JComboBox;
import javax.swing.DefaultComboBoxModel;
```

```
public class Peaje extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblTipoVehiculo;
    private JLabel lblCantidadPasajeros;
    private JComboBox<String> cboTipoVehiculo;
    private JTextField txtCantidadPasajeros;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // -----
    // Declaración de variables globales para el algoritmo
    int canveh0, canveh1, canveh2, canveh3, canveh4;
    int totpas0, totpas1, totpas2, totpas3, totpas4;
    // -----

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Peaje frame = new Peaje();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Peaje() {
        setTitle("Peaje");
        setBounds(100, 100, 450, 360);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        lblTipoVehiculo = new JLabel("Tipo de vehículo");
        lblTipoVehiculo.setBounds(10, 13, 120, 14);
        getContentPane().add(lblTipoVehiculo);

        lblCantidadPasajeros = new JLabel("Cantidad de pasajeros");
        lblCantidadPasajeros.setBounds(10, 38, 120, 14);
        getContentPane().add(lblCantidadPasajeros);

        cboTipoVehiculo = new JComboBox<String>();
        cboTipoVehiculo.setModel(new DefaultComboBoxModel<String>(
            new String[] { "Automóvil", "Camión", "Camioneta", "Omnibus", "Otro" }));
        cboTipoVehiculo.setBounds(130, 10, 90, 20);
        getContentPane().add(cboTipoVehiculo);

        txtCantidadPasajeros = new JTextField();
        txtCantidadPasajeros.setBounds(130, 35, 90, 20);
        getContentPane().add(txtCantidadPasajeros);
        txtCantidadPasajeros.setColumns(10);

        btnProcesar = new JButton("Procesar");
        btnProcesar.addActionListener(this);
        btnProcesar.setBounds(335, 9, 89, 23);
        getContentPane().add(btnProcesar);
    }
}
```

```
btnBorrar = new JButton("Borrar");
btnBorrar.addActionListener(this);
btnBorrar.setBounds(335, 34, 89, 23);
getContentPane().add(btnBorrar);

scpScroll = new JScrollPane();
scpScroll.setBounds(10, 69, 414, 241);
getContentPane().add(scpScroll);

txtS = new JTextArea();
txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    int tipoveh, cantpas;

    tipoveh = getVehiculo();
    cantpas = getPasajeros();

    efectuarIncrementos(tipoveh, cantpas);
    mostrarReporte();
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtCantidadPasajeros.setText("");
    txtS.setText("");
    txtCantidadPasajeros.requestFocus();
}

// Lee y retorna el tipo de vehículo
int getVehiculo() {
    return cboTipoVehiculo.getSelectedIndex();
}

// Lee y retorna la cantidad de pasajeros
int getPasajeros() {
    return Integer.parseInt(txtCantidadPasajeros.getText());
}

// Efectúa los incrementos necesarios
void efectuarIncrementos(int tipv, int canp) {
    switch (tipv) {
        case 0:
            canveh0++;
            totpass0 += canp;
            break;
        case 1:
            canveh1++;
            totpass1 += canp;
            break;
        case 2:
            canveh2++;
            totpass2 += canp;
            break;
        case 3:
            canveh3++;
            totpass3 += canp;
            break;
    }
}
```

```
default:  
    canveh4++;  
    totpas4 += canp;  
}  
  
// Muestra el reporte solicitado  
void mostrarReporte() {  
    txtS.setText("");  
    imprimir("Cantidad total de vehículos");  
    imprimir("- Automóviles : " + canveh0);  
    imprimir("- Camiones : " + canveh1);  
    imprimir("- Camionetas : " + canveh2);  
    imprimir("- Omnibuses : " + canveh3);  
    imprimir("- Otros : " + canveh4);  
    imprimir("");  
    imprimir("Cantidad total de pasajeros");  
    imprimir("- Automóviles : " + totpas0);  
    imprimir("- Camiones : " + totpas1);  
    imprimir("- Camionetas : " + totpas2);  
    imprimir("- Omnibuses : " + totpas3);  
    imprimir("- Otros : " + totpas4);  
}  
  
// Imprime una cadena con un salto de línea al final  
void imprimir(String cad) {  
    txtS.append(cad + "\n");  
}  
}
```

# Problemas propuestos

## Actividad

- Una tienda vende tres tipos de productos a los precios dados en la siguiente tabla:

Producto	Precio
A	S/. 15.0
B	S/. 17.5
C	S/. 20.0

Diseñe un programa que permita efectuar ventas (ingresando por cada una de ellas: producto elegido y cantidad de unidades adquiridas) y muestre, luego de cada venta:

- El importe a pagar
  - La cantidad de ventas efectuadas de cada tipo de producto
  - El importe total recaudado por cada tipo de producto Declare como globales a las variables absolutamente necesarias.
- En un parque de diversiones, los precios de los tickets para los juegos se dan en la siguiente tabla:

Juego	Precio
El gusanito	S/. 3.5
El trencito	S/. 5.0
El pulpo	S/. 2.5

Diseñe un programa que permita efectuar ventas (ingresando por cada una de ellas: juego elegido y cantidad de tickets adquiridos) y muestre, luego de cada venta:

- El importe a pagar
- La cantidad de ventas efectuadas de cada tipo de juego
- La cantidad total de tickets vendidos de cada tipo de juego Declare como globales a las variables absolutamente necesarias.

## Autoevaluación

1. Una empresa de transportes, que cubre la ruta Lima-Huánuco, brinda servicio en tres turnos a los precios por pasaje dados en la siguiente tabla:

Turno	Precio del pasaje (S/.)
Mañana	30.0
Tarde	35.0
Noche	42.5

Diseñe un programa que permita efectuar ventas (ingresando, por cada una de ellas, turno elegido y cantidad de pasajes adquiridos) y muestre, luego de cada venta:

- El importe a pagar
- La cantidad de ventas efectuadas de cada turno
- El importe total recaudado por cada turno
- La cantidad total de pasajes vendidos de cada turno Declare como globales a las variables absolutamente necesarias.

2. Una empresa calcula el sueldo bruto de sus empleado multiplicando las horas trabajadas por una tarifa horaria que depende de la categoría del trabajador de acuerdo con la siguiente tabla:

Categoría	Tarifa
A	S/. 21.0
B	S/. 19.5
C	S/. 17.0

Diseñe un programa que ingrese por cada empleado: categoría y horas trabajadas, y muestre, luego de cada ingreso:

- El sueldo bruto del empleado ingresado
- El sueldo bruto total de cada categoría
- La cantidad total de horas trabajadas de cada categoría Declare como globales a las variables absolutamente necesarias.

3. Diseñe un programa que permita ingresar las edades, de una en una, de un conjunto de personas y muestre, luego de cada ingreso:

- La cantidad de personas ingresadas
- La cantidad de personas mayores de edad
- La cantidad de personas menores de edad

Declare como globales a las variables absolutamente necesarias.

4. Diseñe un programa que permita ingresar donaciones, de una en una, para tres centros de ayuda social: un comedor de niños, una posta médica y una escuela infantil. El programa mostrará, luego de cada ingreso:

- El número de donantes por cada centro de ayuda social
- El monto total donado por cada centro de ayuda social

Declare como globales a las variables absolutamente necesarias.

5. Diseñe un programa que permita ingresar los sexos (Masculino o Femenino) y las preferencias en bebidas gaseosas (Pepsi Cola, Coca Cola o Fanta) de un conjunto de personas y muestre, luego de cada ingreso:

- Cuántas personas prefieren cada tipo de gaseosa.
- Cuántos varones prefieren cada tipo de gaseosa.
- Cuántas mujeres prefieren cada tipo de gaseosa.

Declare como globales a las variables absolutamente necesarias.

## Para recordar

- Las variables locales tienen alcance de método; en cambio, las variables globales tienen alcance de programa.
- Las variables locales no se inicializan automáticamente, a diferencia de las variables globales que si se inicializan automáticamente.



# ESTRUCTURAS DE REPETICIÓN

## LOGRO DE LA UNIDAD DE APRENDIZAJE

Al finalizar la unidad, el alumno diseña programas que involucran procesos repetitivos mediante el uso de las estructuras de repetición más apropiadas.

## TEMARIO

### 6.1 Tema 6 : Estructuras de Repetición

- 6.1.1 : Estructura while
- 6.1.2 : Estructura for
- 6.1.3 : Estructuras do – while

## ACTIVIDADES PROPUESTAS

- Los alumnos desarrollan algoritmos que involucren el uso de estructuras de repetición.

## 6.1. TEMA 6: ESTRUCTURAS DE REPETICIÓN

### Introducción

Se denominan **estructuras repetitivas** o **estructuras de repetición** a aquellas estructuras que permiten repetir instrucciones. A las estructuras repetitivas se conocen también como **estructuras iterativas** o **bucles**, a las instrucciones a repetir se conocen como el **cuerpo del bucle** y al hecho de repetir la secuencia de instrucciones se denomina **iteración**. En el caso del lenguaje Java, tenemos tres tipos de estructuras repetitivas: las estructuras **while**, **do...while** y **for**.

#### 6.1.1. Estructura de repetición while

La estructura **while** repite una acción o un conjunto de acciones mientras sea verdadera una determinada condición, para lo cual, primero, verifica la condición y, luego, ejecuta la acción. La acción puede ser una **acción simple** o una **acción compuesta** (bloque de acciones encerradas entre llaves). En las Figura 1, se muestra el diagrama de flujo de la estructura while.

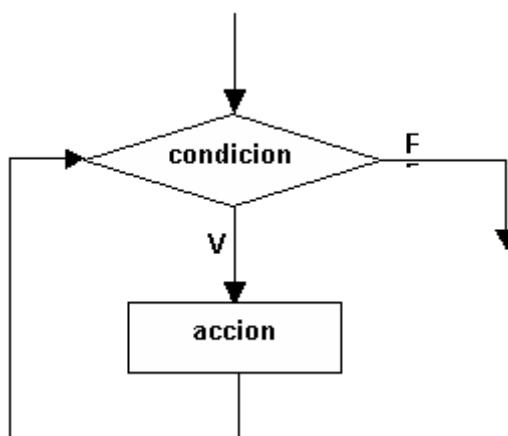


Figura 1 Diagrama de flujo de la estructura while

La sintaxis de la estructura de repetición **while** es la siguiente:

**Para una sola acción por repetir:**

```

while (condicion)
    accion;
  
```

**Para más de una acción por repetir:**

```

while (condicion) {
    accion1;
    accion2;
    ...
    accionN;
}
  
```

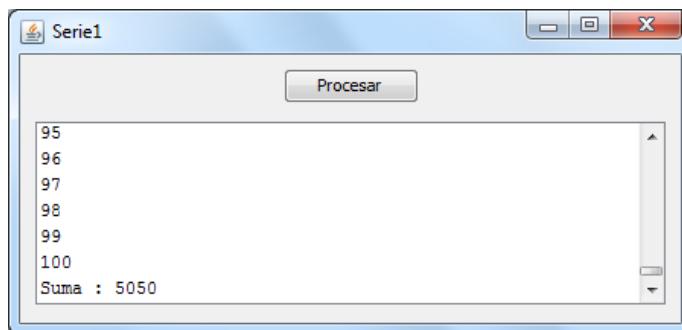
## Problemas resueltos

### Problema 1

Diseñe un programa que imprima y sume la siguiente serie:

1, 2, 3, ... , 100

#### Interfaz Gráfica



#### Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Serie1 extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Serie1 frame = new Serie1();
                    frame.setVisible(true);
                }
                catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}
```

```
// Crea la GUI
public Serie1() {
    setTitle("Serie1");
    setBounds(100, 100, 450, 214);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(173, 9, 89, 23);
    getContentPane().add(btnProcesar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 44, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declara e inicializa variables
    int ter = 1, sum = 0;

    // Genera los términos de la serie
    while (ter <= 100) {

        // Imprime el término actual
        txtS.append(ter + "\n");

        // Suma el término actual
        sum += ter;

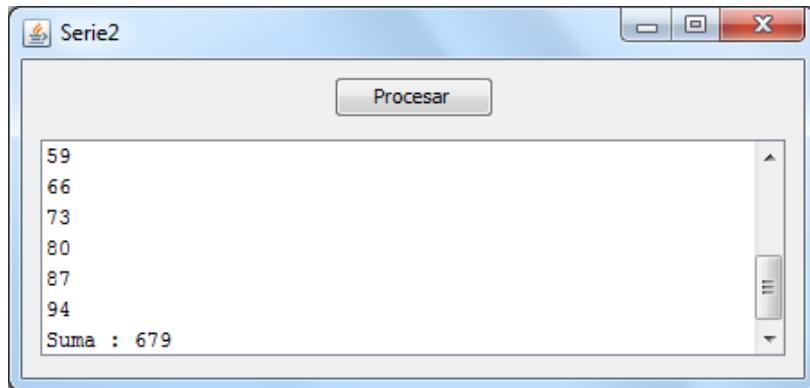
        // Pasa al siguiente término
        ter++;
    }

    // Imprime la suma de la serie
    txtS.append("Suma : " + sum);
}
}
```

**Problema 2**

Diseñe un programa que imprima y sume la siguiente serie:

3, 10, 17, 24, 31, ... , 94

**Interfaz Gráfica****Solución**

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Serie2 extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;
    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Serie2 frame = new Serie2();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}
```

```
// Crea la GUI
public Serie2() {
    setTitle("Serie2");
    setBounds(100, 100, 450, 214);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(173, 9, 89, 23);
    getContentPane().add(btnProcesar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 44, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declara e inicializa variables
    int ter = 3, sum = 0;

    // Genera los términos de la serie
    while (ter <= 94) {

        // Imprime el término actual
        txtS.append(ter + "\n");

        // Suma el término actual
        sum += ter;

        // Pasa al siguiente término
        ter += 7;
    }

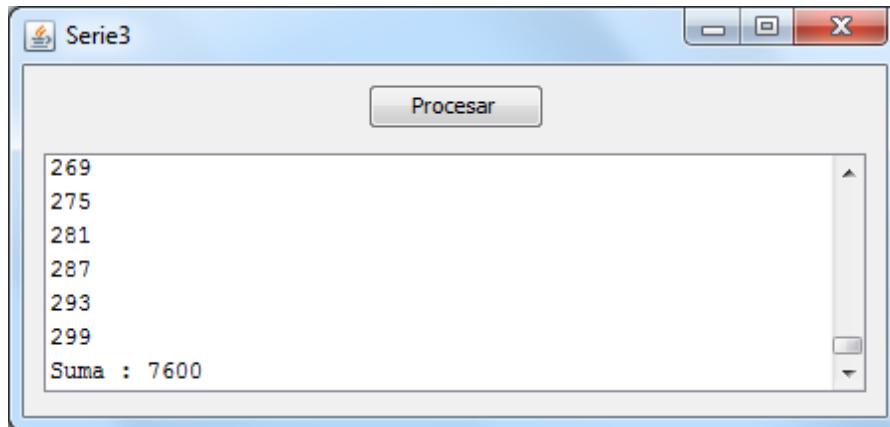
    // Imprime la suma de la serie
    txtS.append("Suma : " + sum);
}
}
```

### Problema 3

Diseñe un programa que imprima y sume 50 términos de la siguiente serie:

5, 11, 17, 23, 29, 35, ...

### Interfaz Gráfico



### Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Serie3 extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;
    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Serie3 frame = new Serie3();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}
```

```
// Crea la GUI
public Serie3() {
    setTitle("Serie3");
    setBounds(100, 100, 450, 214);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(173, 9, 89, 23);
    getContentPane().add(btnProcesar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 44, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declara e inicializa variables
    int ter = 5, sum = 0, c = 0;

    // Genera los términos de la serie
    while (c < 50) {

        // Imprime el término actual
        txtS.append(ter + "\n");

        // Suma el término actual
        sum += ter;

        // Pasa al siguiente término
        ter += 6;

        // Cuenta la cantidad de términos
        c++;
    }

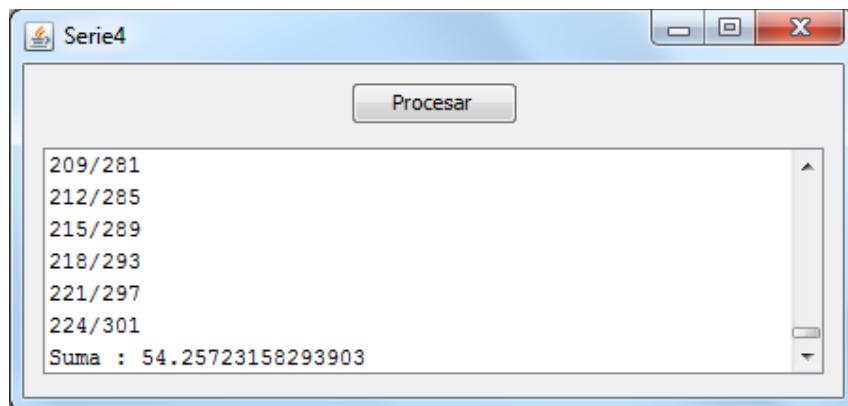
    // Imprime la suma de la serie
    txtS.append("Suma : " + sum);
}
}
```

## Problema 4

Diseñe un programa que imprima y sume 75 términos de la siguiente serie:

$$\frac{2}{5}, \frac{5}{9}, \frac{8}{13}, \frac{11}{17}, \dots$$

### Interfaz Gráfica



### Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Serie4 extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;
    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Serie4 frame = new Serie4();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}
```

```
// Crea la GUI
public Serie4() {
    setTitle("Serie4");
    setBounds(100, 100, 450, 214);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(173, 9, 89, 23);
    getContentPane().add(btnProcesar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 44, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declara e inicializa variables
    int num = 2, den = 5, c = 0;
    double sum = 0;

    // Genera los términos de la serie
    while (c < 75) {

        // Imprime el término actual
        txtS.append(num + "/" + den + "\n");

        // Suma el término actual
        sum += num * 1.0 / den;

        // Pasa al siguiente término
        num += 3;
        den += 4;

        // Cuenta la cantidad de términos
        c++;
    }

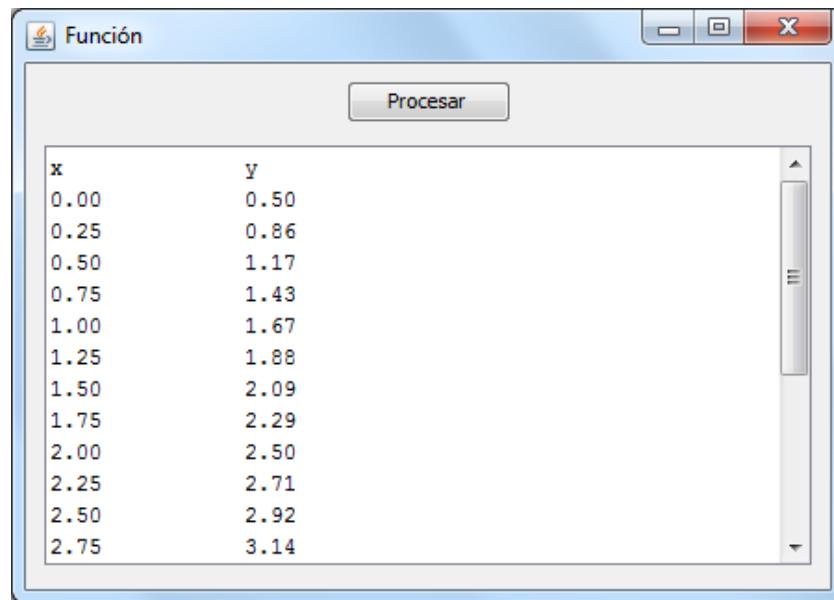
    // Imprime la suma de la serie
    txtS.append("Suma : " + sum);
}
}
```

## Problema 5

Diseñe un programa que imprima una tabla de valores de x e y, para valores de x en el intervalo de 0 a 2.75 cada 0.25, siendo:

$$y = \frac{x^3 + 3x + 1}{x^2 + 2}$$

## Interfaz Gráfica



## Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.util.Locale;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Funcion extends JFrame implements ActionListener {

    private static final long serialVersionUID = 1L;

    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
    }
}
```

```
EventQueue.invokeLater(new Runnable() {
    public void run() {
        try {
            Funcion frame = new Funcion();
            frame.setVisible(true);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
});

// Crea la GUI
public Funcion() {
    setTitle("Función");
    setBounds(100, 100, 450, 321);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(173, 9, 89, 23);
    getContentPane().add(btnProcesar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 44, 414, 226);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declara e inicializa variables
    double x = 0, y;

    // Imprime la cabecera
    txtS.append(String.format(Locale.US, "%-15.2s%-15.2s\n", "x", "y"));

    // Imprime la tabla
    while (x <= 2.75) {

        // Calcula el valor de y para el valor actual de x
        y = (x * x * x + 3 * x + 1) / (x * x + 2);

        // Imprime la pareja de valores x e y
        txtS.append(String.format(Locale.US, "%-15.2f%-15.2f\n", x, y));

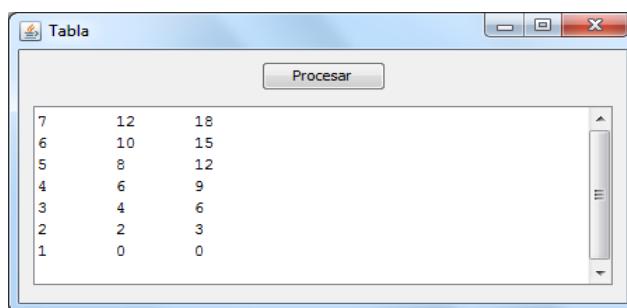
        // Incrementa x en 0.25
        x += 0.25;
    }
}
```

## Problema 6

Diseñe un programa que imprima la siguiente tabla de números:

7	12	18
6	10	15
5	8	12
4	6	9
3	4	6
2	2	3
1	0	0

## Interfaz Gráfica



## Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Tabla extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;
    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Tabla frame = new Tabla();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}
```

```
// Crea la GUI
public Tabla() {
    setTitle("Tabla");
    setBounds(100, 100, 450, 237);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(173, 9, 89, 23);
    getContentPane().add(btnProcesar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 44, 414, 141);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
}

//Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    //Declara e inicializa variables
    int a = 7, b = 12, c = 18;

    //Imprime la tabla
    while (a >= 1) {

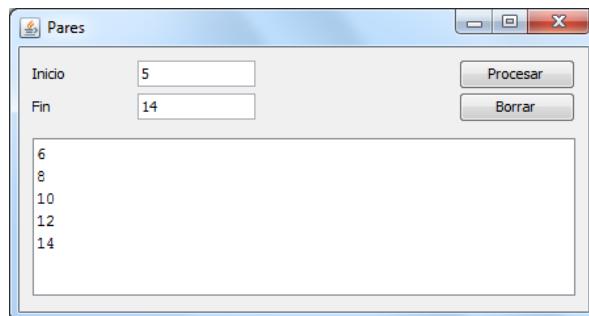
        //Imprime un elemento de cada columna
        txtS.append(a + "\t" + b + "\t" + c + "\n");

        //Pasa al siguiente elemento de cada columna
        a--;
        b -= 2;
        c -= 3;
    }
}
```

## Problema 7

Diseñe un programa que lea los extremos de un intervalo de números enteros e imprima todos los números pares del intervalo.

### Interfaz Gráfica



### Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Pares extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblInicio;
    private JLabel lblFin;
    private JTextField txtInicio;
    private JTextField txtFin;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Pares frame = new Pares();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}
```

```
// Crea la GUI
public Pares() {
    setTitle("Pares");
    setBounds(100, 100, 450, 239);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblInicio = new JLabel("Inicio");
    lblInicio.setBounds(10, 13, 80, 14);
    getContentPane().add(lblInicio);

    lblFin = new JLabel("Fin");
    lblFin.setBounds(10, 38, 80, 14);
    getContentPane().add(lblFin);

    txtInicio = new JTextField();
    txtInicio.setBounds(90, 10, 90, 20);
    getContentPane().add(txtInicio);
    txtInicio.setColumns(10);

    txtFin = new JTextField();
    txtFin.setBounds(90, 35, 90, 20);
    getContentPane().add(txtFin);
    txtFin.setColumns(10);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(335, 34, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 69, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declara variables
    int inicio, fin, numero;

    // Ingresa los extremos del intervalo
    inicio = Integer.parseInt(txtInicio.getText());
    fin = Integer.parseInt(txtFin.getText());

    // Imprime la lista de pares
    numero = inicio;
    while (numero <= fin) {
        if (numero % 2 == 0)
            txtS.append(numero + "\n");
        numero++;
    }
}
```

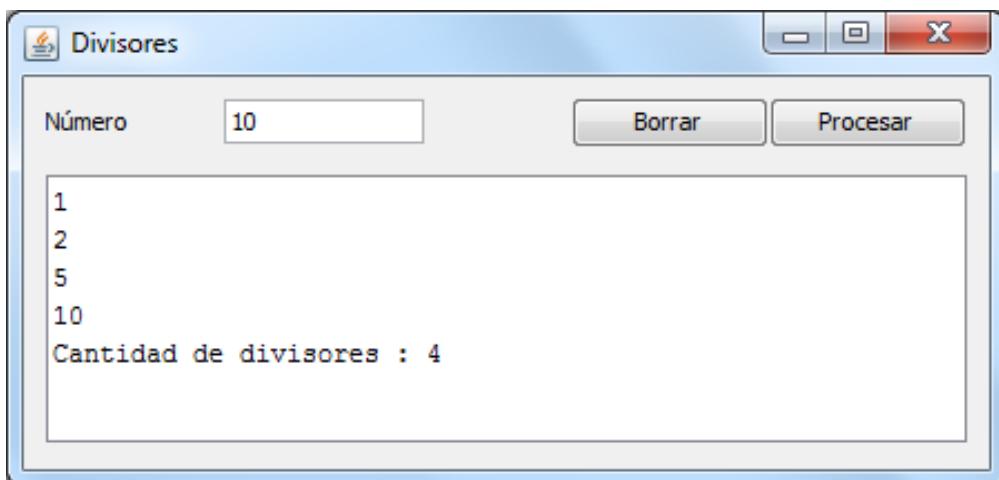
```
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtInicio.setText("");
    txtFin.setText("");
    txtS.setText("");
    txtInicio.requestFocus();
}
}
```

### Problema 8

Diseñe un programa que imprima los divisores de un número natural y la cantidad de divisores encontrados.

### Interfaz Gráfica



### Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
```

```
public class Divisores extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblNumero;
    private JTextField txtNumero;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;
```

```
// Lanza la aplicación
public static void main(String[] args) {
    try {
        UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
    } catch (Throwable e) {
        e.printStackTrace();
    }

    EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                Divisores frame = new Divisores();
                frame.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

// Crea la GUI
public Divisores() {

    setTitle("Divisores");
    setBounds(100, 100, 450, 214);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblNumero = new JLabel("Número");
    lblNumero.setBounds(10, 13, 80, 14);
    getContentPane().add(lblNumero);

    txtNumero = new JTextField();
    txtNumero.setBounds(90, 10, 90, 20);
    getContentPane().add(txtNumero);
    txtNumero.setColumns(10);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(246, 9, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 44, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {

    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}
```

```
// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declara variables
    int contadiv = 0, divisor = 1, numero;

    // Ingresa el número
    numero = Integer.parseInt(txtNumero.getText());

    // Imprime los divisores del número
    // Determina la cantidad de divisores del número
    while (divisor <= numero) {

        if (numero % divisor == 0) {
            txtS.append(divisor + "\n");
            contadiv++;
        }

        divisor++;
    }

    // Imprime la cantidad de divisores
    txtS.append("Cantidad de divisores : " + contadiv);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtNumero.setText("");
    txtS.setText("");
    txtNumero.requestFocus();
}
```

# Problemas propuestos

## Actividad

- Diseñe un programa que imprima y sume la siguiente serie. Los términos serán mostrados en una columna a razón de un término por fila.

2, 4, 6, 8, 10, 12, ..., 98, 100

- Diseñe un programa que imprima y sume la siguiente serie. Los términos serán mostrados en una columna a razón de un término por fila.

3, 7, 11, 15, 19, 23, ..., 199, 203

- Diseñe un programa que imprima y sume 55 términos de la siguiente serie. Los términos serán mostrados en una columna a razón de un término por fila.

5, 12, 19, 26, 33, 40, 47, ...

- Diseñe un programa que imprima y sume 100 términos de la siguiente serie. Los términos serán mostrados en una columna a razón de un término por fila.

$$\frac{3}{2}, \frac{7}{5}, \frac{11}{8}, \frac{15}{11}, \dots$$

- Diseñe un programa que imprima y sume n términos de la siguiente serie. Los términos serán mostrados en una columna a razón de un término por fila.

- Diseñe un programa que imprima y sume n términos de la siguiente serie. Los términos serán mostrados en una columna a razón de un término por fila.

$$\frac{2}{3}, \frac{5}{5}, \frac{8}{7}, \frac{11}{9}, \dots$$

## Autoevaluación

- Diseñe un programa que imprima la siguiente serie en una columna a razón de un término por fila.

3, 7, 11, 15, 19, 23, ..., 199

- Diseñe un programa que imprima y sume 50 términos de la siguiente serie. Los términos serán mostrados en una columna a razón de un término por fila.

5, 8, 11, 14, 17, 20, 23, ...

- Diseñe un programa que imprima y sume 100 términos de la siguiente serie. Los términos serán mostrados en una columna a razón de un término por fila.

$\frac{3}{4}, \frac{5}{7}, \frac{7}{10}, \frac{9}{13}, \frac{11}{16}, \dots$

- Diseñe un programa que imprima y sume n términos de la siguiente serie. Los términos serán mostrados en una columna a razón de un término por fila

$\frac{1}{2}, \frac{4}{4}, \frac{7}{6}, \frac{10}{8}, \dots$

- Diseñe un programa que halle la suma de n términos de la siguiente serie:

$\frac{1}{1}, \frac{6}{5}, \frac{11}{9}, \frac{16}{13}, \dots$

## Para recordar

- La estructura **while** es una estructura de propósito general que puede ser usada para resolver cualquier tipo de problema que involucre procesos repetitivos.
- Si la condición del **while** resulta **falsa** la primera vez que se evalúa su condición de control, el **while** no efectuará ninguna iteración.
- Si la condición del **while** no se hace **falsa** nunca, se genera un *bucle infinito*.

### 6.1.2. Estructura de repetición for

La estructura de repetición while es una estructura de propósito general: puede usarse para resolver cualquier problema que involucre procesos repetitivos. Pero, para los casos en que se conoce el número de iteraciones de un proceso repetitivo, la estructura ideal es la estructura for. Fue construida para ese tipo de procesos y es por ello por lo que puede manejar uno o más contadores propios. En la Figura 1, se muestra el diagrama de flujo de la estructura for.

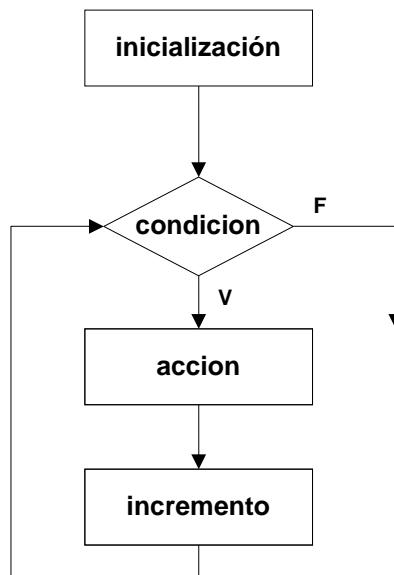


Figura 1 Diagrama de flujo de la estructura for

La sintaxis de la estructura de repetición **for** es la siguiente:

**Para una sola acción por repetir:**

```

for (inicio; condición; incremento)
    accion;
  
```

**Para más de una acción por repetir:**

```

for (inicio; condición; incremento){
    accion1;
    accion2;
    .
    .
    .
    accionN;
}
  
```

## Funcionamiento del **for**

- **Paso 1:** Se ejecuta la sentencia de **inicio**. En la sentencia de **inicio**, se declara y se inicializa el contador del **for**.
- **Paso 2:** Se evalúa la **condición**. En caso de que la **condición** sea verdadera, va al paso 3; en caso contrario, finaliza el **for**.
- **Paso 3:** Se ejecuta la **acción** o el conjunto de **acciones**.
- **Paso 4:** Se ejecuta la sentencia de **incremento** y vuelve al paso 2. En la sentencia de **incremento** se incrementa el contador del **for**.

### Ejemplo 1

- **for** que imprime los números: 0, 1, 2, 3, ..., 48, 49, 50

```
for (int i = 0; i <= 50; i++)
    txtS.append(i + "\n");
```

- **for** que imprime los números: 100, 99, 98, ..., 13, 12, 11, 10

```
for (int i = 100; i >= 10; i--)
    txtS.append( i + "\n" );
```

- **for** que imprime los números: 10, 12, 14, 16,..., 98, 99, 100

```
for (int i = 10; i <= 100; i += 2)
    txtS.append(i + "\n");
```

- **for** que imprime los números: 100, 97, 94, 91, ..., 18, 15, 12, 9

```
for (int i = 100; i >= 9; i -= 3)
    txtS.append(i + "\n");
```

- **for** que imprime la siguiente tabla:

10	30
11	29
12	28
13	27
14	26
15	25
16	24
17	23
18	22
19	21
20	20

```
for (int i = 10, j = 30; i <= 20; i++, j--)  
    txtS.append(i + "\t" + j + "\n");
```

**Nota.-** El contador del **for** puede ser declarado dentro del mismo **for**, en cuyo caso se considera como variable local al **for**, no siendo accesible fuera de él. De esta manera, en un mismo método, dos o más **for**, no anidados, pueden declarar contadores con el mismo nombre. Cada contador existe dentro del **for** en el que fue declarado.

## Números aleatorios

Para generar un número aleatorio entero en el intervalo de **min** a **máx.**, se usa la siguiente expresión:

```
n = (Entero) ((max - min + 1) * Math.random() + min))
```

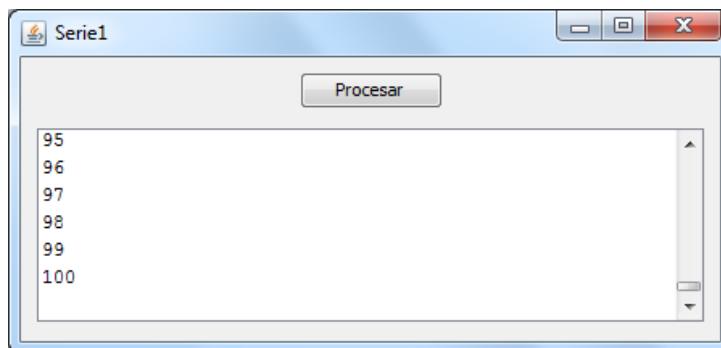
## Problemas resueltos

### Problema 1

Diseñe un programa que imprima y sume la siguiente serie:

1, 2, 3, ... , 100

## Interfaz Gráfica



## Solución

```
package cibertec;  
  
import java.awt.EventQueue;  
import java.awt.Font;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import javax.swing.JButton;  
import javax.swing.JFrame;  
import javax.swing.JScrollPane;  
import javax.swing.JTextArea;  
import javax.swing.UIManager;
```

```
public class Serie1 extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;
    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Serie1 frame = new Serie1();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Serie1() {
        setTitle("Serie1");
        setBounds(100, 100, 450, 214);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        btnProcesar = new JButton("Procesar");
        btnProcesar.addActionListener(this);
        btnProcesar.setBounds(173, 9, 89, 23);
        getContentPane().add(btnProcesar);

        scpScroll = new JScrollPane();
        scpScroll.setBounds(10, 44, 414, 120);
        getContentPane().add(scpScroll);

        txtS = new JTextArea();
        txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
        scpScroll.setViewportView(txtS);
    }

    // Direcciona eventos de tipo ActionEvent
    public void actionPerformed(ActionEvent arg0) {
        if (arg0.getSource() == btnProcesar) {
            actionPerformedBtnProcesar(arg0);
        }
    }

    // Procesa la pulsación del botón Procesar
    protected void actionPerformedBtnProcesar(ActionEvent arg0) {
        // Declara e inicializa la suma
        int sum = 0;

        // Imprime y suma la serie
        for (int ter = 1; ter <= 100; ter++) {

            // Imprime el término actual
            txtS.append(ter + "\n");

            // Suma el término actual
            sum += ter;
        }

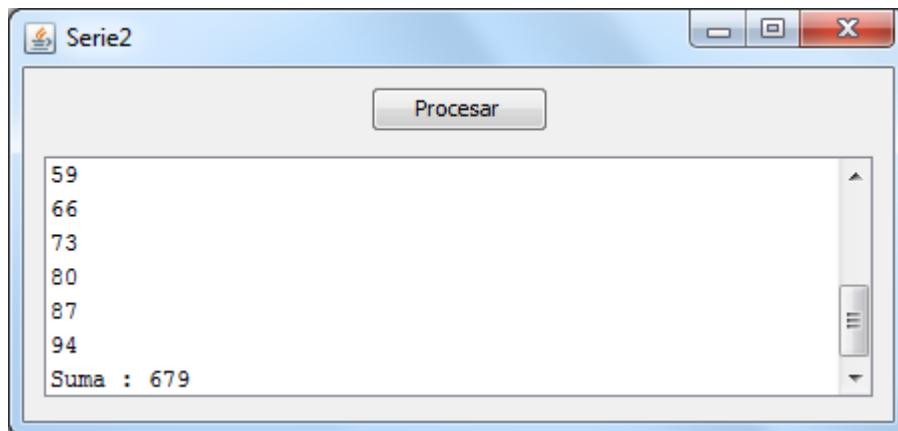
        // Imprime la suma
        txtS.append("Suma : " + sum);
    }
}
```

## Problema 2

Diseñe un programa que imprima y sume la siguiente serie:

3, 10, 17, 24, 31, ... , 94

### Interfaz Gráfica



### Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Serie2 extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;
    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Serie2 frame = new Serie2();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}
```

```

// Crea la GUI
public Serie2() {
    setTitle("Serie2");
    setBounds(100, 100, 450, 214);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(173, 9, 89, 23);
    getContentPane().add(btnProcesar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 44, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declara e inicializa la suma
    int sum = 0;

    // Imprime y suma la serie
    for (int ter = 3; ter <= 94; ter += 7) {
        // Imprime el término actual
        txtS.append(ter + "\n");

        // Suma el término actual
        sum += ter;
    }

    // Imprime la suma
    txtS.append("Suma : " + sum);
}
}

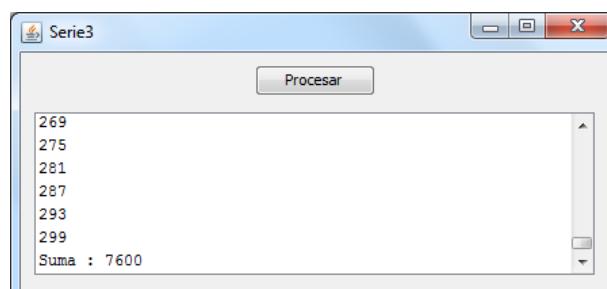
```

### Problema 3

Diseñe un programa que imprima y sume 50 términos de la siguiente serie:

5, 11, 17, 23, 29, 35, ...

#### Interfaz Gráfica



## Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Serie3 extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;
    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Serie3 frame = new Serie3();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Serie3() {

        setTitle("Serie3");
        setBounds(100, 100, 450, 214);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        btnProcesar = new JButton("Procesar");
        btnProcesar.addActionListener(this);
        btnProcesar.setBounds(173, 9, 89, 23);
        getContentPane().add(btnProcesar);

        scpScroll = new JScrollPane();
        scpScroll.setBounds(10, 44, 414, 120);
        getContentPane().add(scpScroll);

        txtS = new JTextArea();
        txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
        scpScroll.setViewportView(txtS);

    }

    // Direcciona eventos de tipo ActionEvent
    public void actionPerformed(ActionEvent arg0) {

        if (arg0.getSource() == btnProcesar) {
            actionPerformedBtnProcesar(arg0);
        }

    }
}
```

```

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declara e inicializa el término y la suma
    int ter = 5, sum = 0;

    // Imprime y suma la serie
    for (int c = 0; c < 50; c++) {
        // Imprime el término actual
        txtS.append(ter + "\n");

        // Suma el término actual
        sum += ter;

        // Pasa al siguiente término
        ter += 6;
    }

    // Imprime la suma
    txtS.append("Suma : " + sum);
}
}

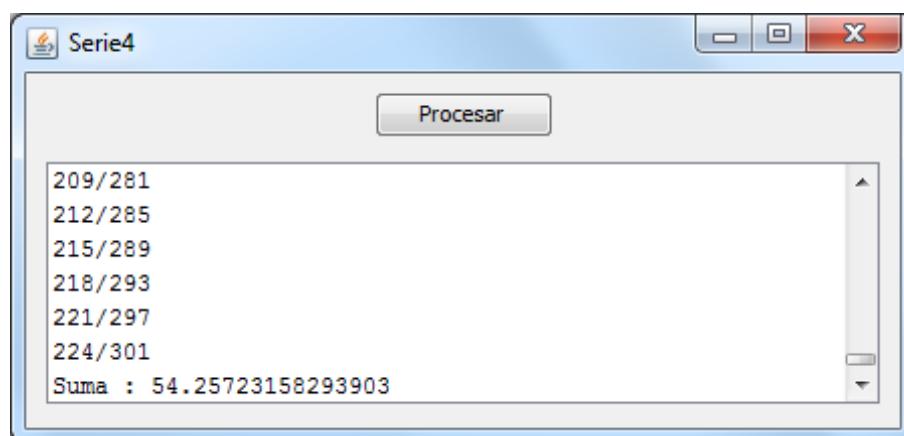
```

#### Problema 4

Diseñe un programa que imprima y sume 75 términos de la siguiente serie:

$$\frac{2}{5}, \frac{5}{9}, \frac{8}{13}, \frac{11}{17}, \dots$$

#### Interfaz Gráfica



#### Solución

```

package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

```

```
public class Serie4 extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;
    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Serie4 frame = new Serie4();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Serie4() {
        setTitle("Serie4");
        setBounds(100, 100, 450, 214);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        btnProcesar = new JButton("Procesar");
        btnProcesar.addActionListener(this);
        btnProcesar.setBounds(173, 9, 89, 23);
        getContentPane().add(btnProcesar);

        scpScroll = new JScrollPane();
        scpScroll.setBounds(10, 44, 414, 120);
        getContentPane().add(scpScroll);

        txtS = new JTextArea();
        txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
        scpScroll.setViewportView(txtS);
    }

    // Direcciona eventos de tipo ActionEvent
    public void actionPerformed(ActionEvent arg0) {
        if (arg0.getSource() == btnProcesar) {
            actionPerformedBtnProcesar(arg0);
        }
    }

    // Procesa la pulsación del botón Procesar
    protected void actionPerformedBtnProcesar(ActionEvent arg0) {
        // Declara e inicializa el término y la suma
        int num = 2, den = 5;
        double sum = 0;

        // Imprime y suma la serie
        for (int c = 0; c < 75; c++) {
            // Imprime el término actual
            txtS.append(num + "/" + den + "\n");

            // Suma el término actual
            sum += num * 1.0 / den;

            // Pasa al siguiente término
            num += 3;
            den += 4;
        }
    }
}
```

```

        // Imprime la suma
        txtS.append("Suma : " + sum);
    }
}

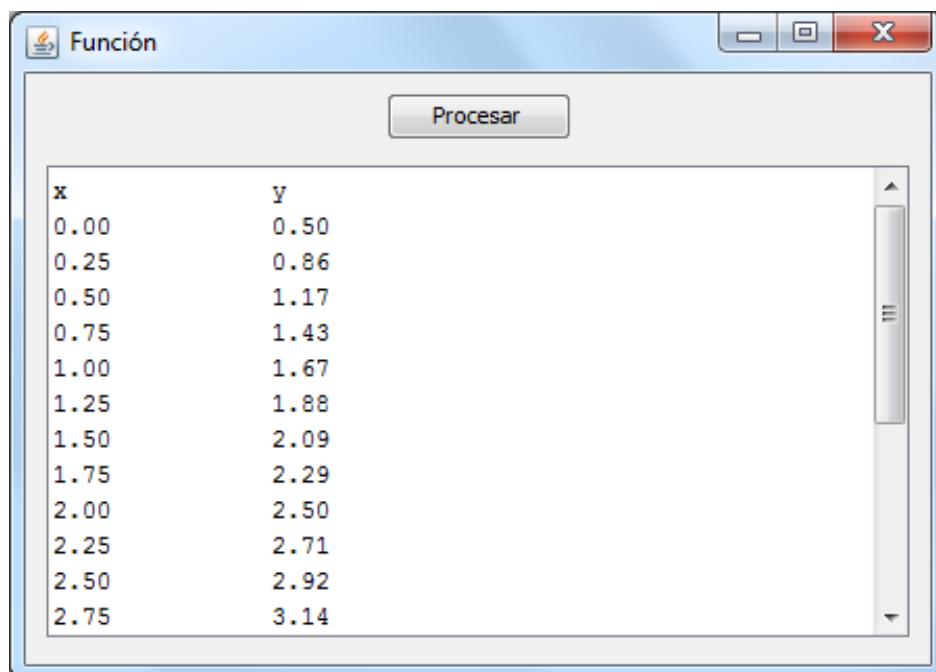
```

### Problema 5

Diseñe un programa que imprima una tabla de valores de x e y, para valores de x en el intervalo de 0 a 2.75 cada 0.25, siendo:

$$y = \frac{x^3 + 3x + 1}{x^2 + 2}$$

### Interfaz Gráfica



### Solución

```

package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.util.Locale;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

```

```

public class Funcion extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;
    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

```

```
// Lanza la aplicación
public static void main(String[] args) {
    try {
        UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
    } catch (Throwable e) {
        e.printStackTrace();
    }

    EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                Funcion frame = new Funcion();
                frame.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

// Crea la GUI
public Funcion() {
    setTitle("Función");
    setBounds(100, 100, 450, 321);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(173, 9, 89, 23);
    getContentPane().add(btnProcesar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 44, 414, 226);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declara e inicializa variables
    double y;

    // Imprime la cabecera
    txtS.append(String.format(Locale.US, "%-15.2s%-15.2s\n", "x", "y"));

    // Imprime la tabla
    for (double x = 0; x <= 2.75; x += 0.25) {

        // Calcula el valor de y para el valor actual de x
        y = (x * x * x + 3 * x + 1) / (x * x + 2);

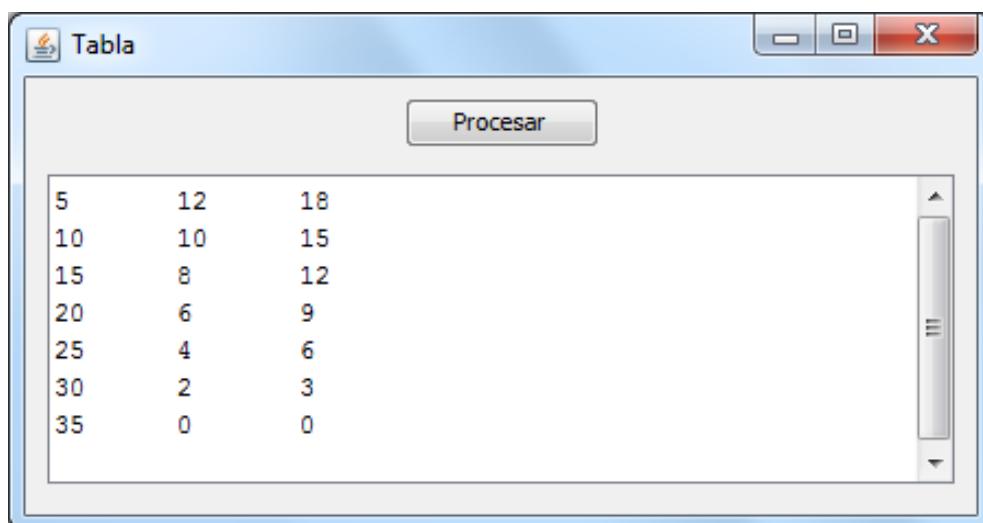
        // Imprime la pareja de valores x e y
        txtS.append(String.format(Locale.US, "%-15.2f%-15.2f\n", x, y));
    }
}
```

### Problema 6

Diseñe un programa que imprima la siguiente tabla de números:

5	12	18
10	10	15
15	8	12
20	6	9
25	4	6
30	2	3
35	0	0

### Interfaz Gráfica



### Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Tabla extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;
    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
    }
}
```

```
EventQueue.invokeLater(new Runnable() {
    public void run() {
        try {
            Tabla frame = new Tabla();
            frame.setVisible(true);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
});

// Crea la GUI
public Tabla() {
    setTitle("Tabla");
    setBounds(100, 100, 450, 237);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(173, 9, 89, 23);
    getContentPane().add(btnProcesar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 44, 414, 141);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declara e inicializa las columnas
    int colum1 = 5, colum2 = 12, colum3 = 18;

    // Imprime la tabla
    for (int veces = 0; veces < 7; veces++) {

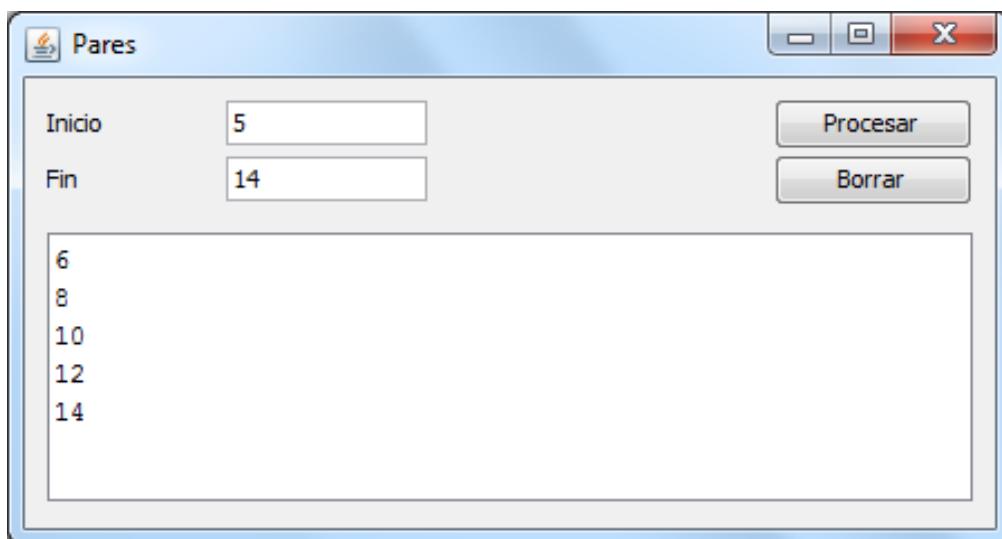
        // Imprime una fila de la tabla
        txtS.append(colum1 + "\t" + colum2 + "\t" + colum3 + "\n");

        // Pasa a la siguiente fila
        colum1 += 5;
        colum2 -= 2;
        colum3 -= 3;
    }
}
```

## Problema 7

Diseñe un programa que lea los extremos de un intervalo de números enteros e imprima todos los números pares del intervalo.

### Interfaz Gráfica



### Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
```

```
public class Pares extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblInicio;
    private JLabel lblFin;
    private JTextField txtInicio;
    private JTextField txtFin;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
    }
}
```

```
EventQueue.invokeLater(new Runnable() {
    public void run() {
        try {
            Pares frame = new Pares();
            frame.setVisible(true);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
});

// Crea la GUI
public Pares() {

    setTitle("Pares");
    setBounds(100, 100, 450, 239);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblInicio = new JLabel("Inicio");
    lblInicio.setBounds(10, 13, 80, 14);
    getContentPane().add(lblInicio);

    lblFin = new JLabel("Fin");
    lblFin.setBounds(10, 38, 80, 14);
    getContentPane().add(lblFin);

    txtInicio = new JTextField();
    txtInicio.setBounds(90, 10, 90, 20);
    getContentPane().add(txtInicio);
    txtInicio.setColumns(10);

    txtFin = new JTextField();
    txtFin.setBounds(90, 35, 90, 20);
    getContentPane().add(txtFin);
    txtFin.setColumns(10);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(335, 34, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 69, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {

    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}
```

```
// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declara variables
    int inicio, fin;

    // Ingresa los extremos del intervalo
    inicio = Integer.parseInt(txtInicio.getText());
    fin = Integer.parseInt(txtFin.getText());

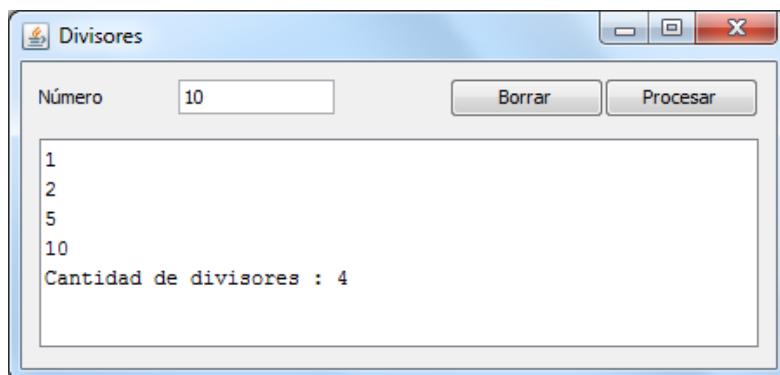
    // Imprime la lista de pares
    for (int numero = inicio; numero <= fin; numero++) {
        if (numero % 2 == 0)
            txtS.append(numero + "\n");
    }
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtInicio.setText("");
    txtFin.setText("");
    txtS.setText("");
    txtInicio.requestFocus();
}
```

### Problema 8

Diseñe un programa que imprima los divisores de un número natural y la cantidad de divisores encontrados.

### Interfaz Gráfica



### Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextArea;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextField;
```

```
public class Divisores extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblNumero;
    private JTextField txtNumero;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Divisores frame = new Divisores();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Divisores() {
        setTitle("Divisores");
        setBounds(100, 100, 450, 214);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        lblNumero = new JLabel("Número");
        lblNumero.setBounds(10, 13, 80, 14);
        getContentPane().add(lblNumero);

        txtNumero = new JTextField();
        txtNumero.setBounds(90, 10, 90, 20);
        getContentPane().add(txtNumero);
        txtNumero.setColumns(10);

        btnProcesar = new JButton("Procesar");
        btnProcesar.addActionListener(this);
        btnProcesar.setBounds(335, 9, 89, 23);
        getContentPane().add(btnProcesar);

        btnBorrar = new JButton("Borrar");
        btnBorrar.addActionListener(this);
        btnBorrar.setBounds(246, 9, 89, 23);
        getContentPane().add(btnBorrar);

        scpScroll = new JScrollPane();
        scpScroll.setBounds(10, 44, 414, 120);
        getContentPane().add(scpScroll);

        txtS = new JTextArea();
        txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
        scpScroll.setViewportView(txtS);
    }

    // Direcciona eventos de tipo ActionEvent
    public void actionPerformed(ActionEvent arg0) {
        if (arg0.getSource() == btnProcesar) {
            actionPerformedBtnProcesar(arg0);
        }
    }
}
```

```

        if (arg0.getSource() == btnBorrar) {
            actionPerformedBtnBorrar(arg0);
        }

        // Procesa la pulsación del botón Procesar
        protected void actionPerformedBtnProcesar(ActionEvent arg0) {
            // Declara variables
            int contadiv = 0, numero;

            // Ingresa el número
            numero = Integer.parseInt(txtNumero.getText());

            // Imprime los divisores del número
            for (int divisor = 1; divisor <= numero; divisor++) {

                // Si divisor divide a número de forma exacta, es un divisor
                if (numero % divisor == 0) {

                    // Imprime el divisor encontrado
                    txtS.append(divisor + "\n");

                    // Cuenta el divisor encontrado
                    contadiv++;
                }
            }

            // Imprime la cantidad de divisores
            txtS.append("Cantidad de divisores : " + contadiv);
        }

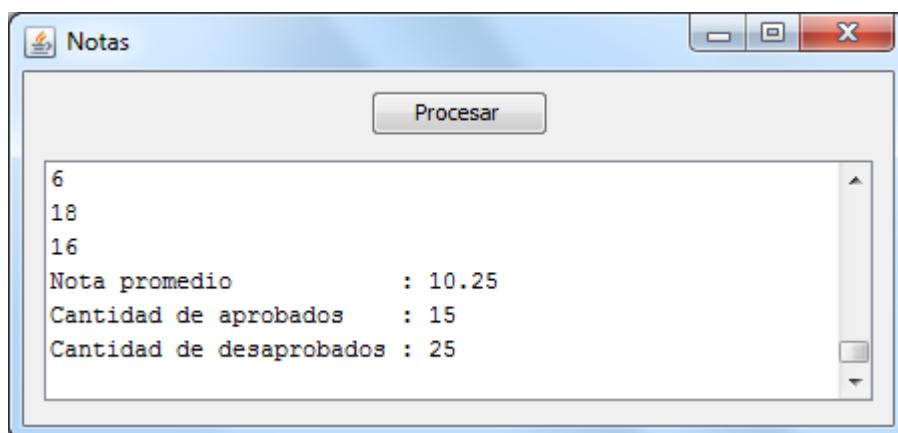
        // Procesa la pulsación del botón Borrar
        protected void actionPerformedBtnBorrar(ActionEvent arg0) {
            txtNumero.setText("");
            txtS.setText("");
            txtNumero.requestFocus();
        }
    }
}

```

### Problema 9

Diseñe un programa que genere aleatoriamente las notas de un examen de 40 estudiantes de una sección y determine la nota promedio, la cantidad de notas aprobatorias y la cantidad de notas desaprobatorias.

### Interfaz Gráfica



### Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Notas extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;
    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Notas frame = new Notas();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Notas() {
        setTitle("Notas");
        setBounds(100, 100, 450, 214);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        btnProcesar = new JButton("Procesar");
        btnProcesar.addActionListener(this);
        btnProcesar.setBounds(173, 9, 89, 23);
        getContentPane().add(btnProcesar);

        scpScroll = new JScrollPane();
        scpScroll.setBounds(10, 44, 414, 120);
        getContentPane().add(scpScroll);

        txtS = new JTextArea();
        txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
        scpScroll.setViewportView(txtS);
    }

    // Direcciona eventos de tipo ActionEvent
    public void actionPerformed(ActionEvent arg0) {
        if (arg0.getSource() == btnProcesar) {
            actionPerformedBtnProcesar(arg0);
        }
    }

    // Procesa la pulsación del botón Procesar
    protected void actionPerformedBtnProcesar(ActionEvent arg0) {
        // Declara e inicializa variables
        int nota, capro = 0, cdesa = 0, suma = 0;
        double pro;
```

```
// Proceso repetitivo
for (int c = 0; c < 40; c++) {

    // Genera una nota
    nota = (int) (21 * Math.random());

    // Imprime la nota generada
    txtS.append(nota + "\n");

    // Suma la nota
    suma += nota;

    // Contabiliza la nota como aprobatoria o desaprobatoria
    if (nota >= 13)
        capro++;
    else
        cdesa++;
}

// Determina la nota promedio
pro = suma * 1.0 / 25;

// Imprime el reporte final
txtS.append("Nota promedio: " + pro + "\n");
txtS.append("Cantidad de aprobados : " + capro + "\n");
txtS.append("Cantidad de desaprobados : " + cdesa + "\n");
}
```

# Problemas propuestos

## Actividad

- Diseñe un programa que imprima y sume 50 términos de la siguiente serie. Los términos serán mostrados en una columna a razón de un término por fila.

3, 10, 17, 24, 31, ...

- Diseñe un programa que imprima y sume 100 términos de la siguiente serie. Los términos serán mostrados en una columna a razón de un término por fila.

$\frac{3}{2}, \frac{7}{5}, \frac{11}{8}, \frac{15}{11}, \dots$

- Diseñe un programa que imprima y sume 75 términos de la siguiente serie. Los términos serán mostrados en una columna a razón de un término por fila.

$\frac{2}{3}, \frac{5}{5}, \frac{8}{7}, \frac{11}{9}, \dots$

- Diseñe un programa que genere aleatoriamente los sueldos de 30 empleados de una empresa con valores del intervalo 2500 a 3500 y determine:

- El sueldo promedio
- La cantidad de empleados que ganan más de S/.3000
- La cantidad de empleados que ganan menos o igual que S/.3000

- Diseñe un programa que genere aleatoriamente las edades de 45 personas con valores del intervalo 20 a 70 y determine:

- La edad promedio
- La cantidad de personas menores de edad
- La cantidad de personas mayores de edad

- Diseñe un programa que genere aleatoriamente el stock de 100 productos con valores del intervalo 0 a 200 y determine:

- La cantidad de productos cuyo stock es < 50
- La cantidad de productos cuyo stock es  $\geq 50$  pero < 100
- La cantidad de productos cuyo stock es  $\geq 100$  pero < 150
- La cantidad de productos cuyo stock es  $\geq 150$

- Diseñe un programa que determine la cantidad de números múltiplos de 3 que se encuentran en el intervalo de números enteros 100 a 900

- Diseñe un programa que determine la suma de los números múltiplos de 4 y la suma de los números múltiplos de 5 que se encuentran en el intervalo de números enteros 5 a 125.

## Autoevaluación

- Diseñe un programa que imprima la siguiente serie en una columna a razón de un término por fila.

3, 7, 11, 15, 19, 23, ... , 199

- Diseñe un programa que imprima y sume 50 términos de la siguiente serie. Los términos serán mostrados en una columna a razón de un término por fila.

5, 8, 11, 14, 17, 20, 23, ...

- Diseñe un programa que imprima y sume 100 términos de la siguiente serie. Los términos serán mostrados en una columna a razón de un término por fila.

$\frac{3}{4}, \frac{5}{7}, \frac{7}{10}, \frac{9}{13}, \frac{11}{16}, \dots$

- Diseñe un programa que imprima y sume n términos de la siguiente serie. Los términos serán mostrados en una columna a razón de un término por fila.

$\frac{1}{2}, \frac{4}{4}, \frac{7}{6}, \frac{10}{8}, \dots$

- Diseñe un programa que halle la suma de n términos de la siguiente serie:

$\frac{1}{1}, \frac{6}{5}, \frac{11}{9}, \frac{16}{13}, \dots$

- Diseñe un programa que genere aleatoriamente el número de hijos de 75 personas con valores del intervalo 1 a 5 y determine:

- La cantidad de personas que tienen dos hijos
- La cantidad de personas que tienen tres hijos
- La cantidad de personas que tienen cuatro hijos
- La cantidad de personas que tienen cinco hijos

- Diseñe un programa que genere aleatoriamente los sueldos de 150 empleados de una empresa con valores enteros del intervalo 2000 a 4000 y determine:

- La suma de los sueldos de todos los empleados
- La cantidad de empleados con sueldos < 2500
- La cantidad de empleados con sueldos  $\geq 2500$  pero  $< 3000$
- La cantidad de empleados con sueldos  $\geq 3000$  pero  $< 3500$
- La cantidad de empleados con sueldos  $\geq 3500$

# Para recordar

- La estructura **for** es ideal para bucles en los que se conoce el número de iteraciones.
- Si la condición del **for** resulta **falsa** la primera vez que se evalúa su condición de control, el **for** no efectuará ninguna iteración.

### 6.1.3. Estructura de repetición do – while

La estructura **do – while** repite una acción o un conjunto de acciones mientras sea verdadera una determinada condición, para lo cual, primero, ejecuta la acción y, luego, verifica la condición. La acción puede ser una **acción simple** o una **acción compuesta** (varias acciones). En la figura 1, se muestra el diagrama de flujo de la estructura hacer- mientras.

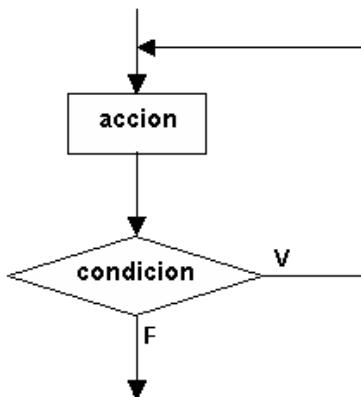


Figura 1 Diagrama de flujo de la estructura **do – while**

La sintaxis de la estructura de repetición **do – while** es la siguiente:

```

do{
    accion1;
    accion2;
    .
    .
    .
    accion3;
} while (condición);
  
```

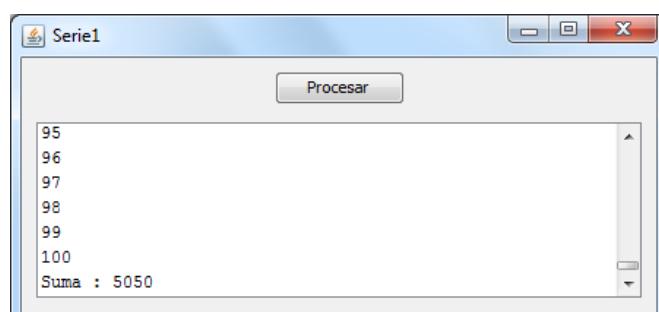
#### Problemas resueltos

##### Problema 1

Diseñe un programa que imprima y sume la siguiente serie:

1, 2, 3, ... , 100

#### Interfaz Gráfica



## Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Serie1 extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;
    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Serie1 frame = new Serie1();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Serie1() {
        setTitle("Serie1");
        setBounds(100, 100, 450, 214);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        btnProcesar = new JButton("Procesar");
        btnProcesar.addActionListener(this);
        btnProcesar.setBounds(173, 9, 89, 23);
        getContentPane().add(btnProcesar);

        scpScroll = new JScrollPane();
        scpScroll.setBounds(10, 44, 414, 120);
        getContentPane().add(scpScroll);

        txtS = new JTextArea();
        txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
        scpScroll.setViewportView(txtS);
    }

    // Direcciona eventos de tipo ActionEvent
    public void actionPerformed(ActionEvent arg0) {
        if (arg0.getSource() == btnProcesar) {
            actionPerformedBtnProcesar(arg0);
        }
    }
}
```

```

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declara e inicializa variables
    int ter = 1, sum = 0;

    // Limpia la pantalla
    txtS.setText("");

    // Genera los términos de la serie
    do {
        // Imprime el término actual
        txtS.append(ter + "\n");

        // Suma el término actual
        sum += ter;

        // Genera el siguiente término
        ter++;
    } while (ter <= 100);

    // Imprime la suma de la serie
    txtS.append("Suma : " + sum);
}
}

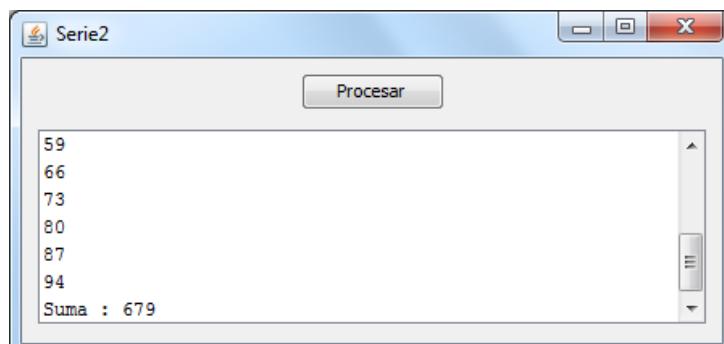
```

## Problema 2

Diseñe un programa que imprima y sume la siguiente serie:

3, 10, 17, 24, 31, ... , 94

### Interfaz Gráfica



### Solución

```

package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

```

```
public class Serie2 extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;
    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Serie2 frame = new Serie2();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Serie2() {
        setTitle("Serie2");
        setBounds(100, 100, 450, 214);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        btnProcesar = new JButton("Procesar");
        btnProcesar.addActionListener(this);
        btnProcesar.setBounds(173, 9, 89, 23);
        getContentPane().add(btnProcesar);

        scpScroll = new JScrollPane();
        scpScroll.setBounds(10, 44, 414, 120);
        getContentPane().add(scpScroll);

        txtS = new JTextArea();
        txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
        scpScroll.setViewportView(txtS);
    }

    // Direcciona eventos de tipo ActionEvent
    public void actionPerformed(ActionEvent arg0) {
        if (arg0.getSource() == btnProcesar) {
            actionPerformedBtnProcesar(arg0);
        }
    }

    // Procesa la pulsación del botón Procesar
    protected void actionPerformedBtnProcesar(ActionEvent arg0) {
        // Declara e inicializa variables
        int ter = 3, sum = 0;

        // Limpia la pantalla
        txtS.setText("");

        // Genera los términos de la serie
        do {
            // Imprime el término actual
            txtS.append(ter + "\n");

            // Suma el término actual
            sum += ter;
        }
    }
}
```

```

        // Genera el siguiente término
        ter += 7;

    } while (ter <= 94);

    // Imprime la suma de la serie
    txtS.append("Suma : " + sum);
}
}

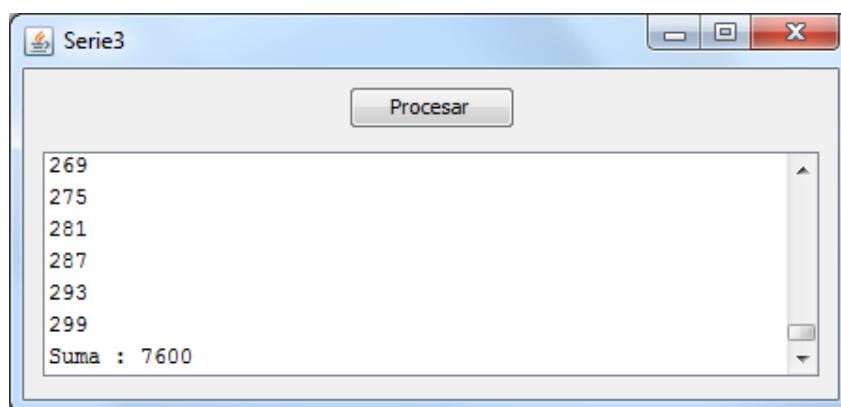
```

### Problema 3

Diseñe un programa que imprima y sume 50 términos de la siguiente serie:

5, 11, 17, 23, 29, 35, ...

### Interfaz Gráfica



### Solución

```

package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Serie3 extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;
    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
    }
}

```

```
EventQueue.invokeLater(new Runnable() {
    public void run() {
        try {
            Serie3 frame = new Serie3();
            frame.setVisible(true);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
});

// Crea la GUI
public Serie3() {
    setTitle("Serie3");
    setBounds(100, 100, 450, 214);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(173, 9, 89, 23);
    getContentPane().add(btnProcesar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 44, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declara e inicializa variables
    int ter = 5, sum = 0, c = 0;

    // Limpia la pantalla
    txtS.setText("");

    // Genera los términos de la serie
    do {
        // Imprime el término actual
        txtS.append(ter + "\n");

        // Suma el término actual
        sum += ter;

        // Genera el siguiente término
        ter += 6;

        // Cuenta la cantidad de términos
        c++;
    } while (c < 50);

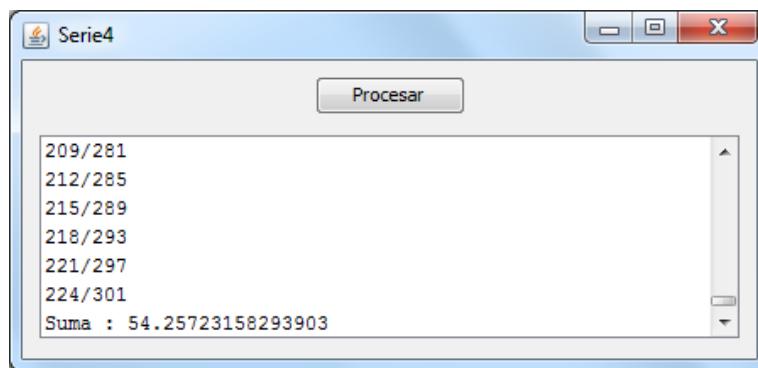
    // Imprime la suma de la serie
    txtS.append("Suma : " + sum);
}
}
```

## Problema 4

Diseñe un programa que imprima y sume 75 términos de la siguiente serie:

$$\frac{2}{5}, \frac{5}{9}, \frac{8}{13}, \frac{11}{17}, \dots$$

### Interfaz Gráfica



### Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Serie4 extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;
    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Serie4 frame = new Serie4();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}
```

```
// Crea la GUI
public Serie4() {
    setTitle("Serie4");
    setBounds(100, 100, 450, 214);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(173, 9, 89, 23);
    getContentPane().add(btnProcesar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 44, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declara e inicializa variables
    int num = 2, den = 5, c = 0;
    double sum = 0;

    // Limpia la pantalla
    txtS.setText("");

    // Genera los términos de la serie
    do {
        // Imprime el término actual
        txtS.append(num + "/" + den + "\n");

        // Suma el término actual
        sum += num * 1.0 / den;

        // Genera el siguiente término
        num += 3;
        den += 4;

        // Cuenta la cantidad de términos
        c++;
    } while (c < 75);

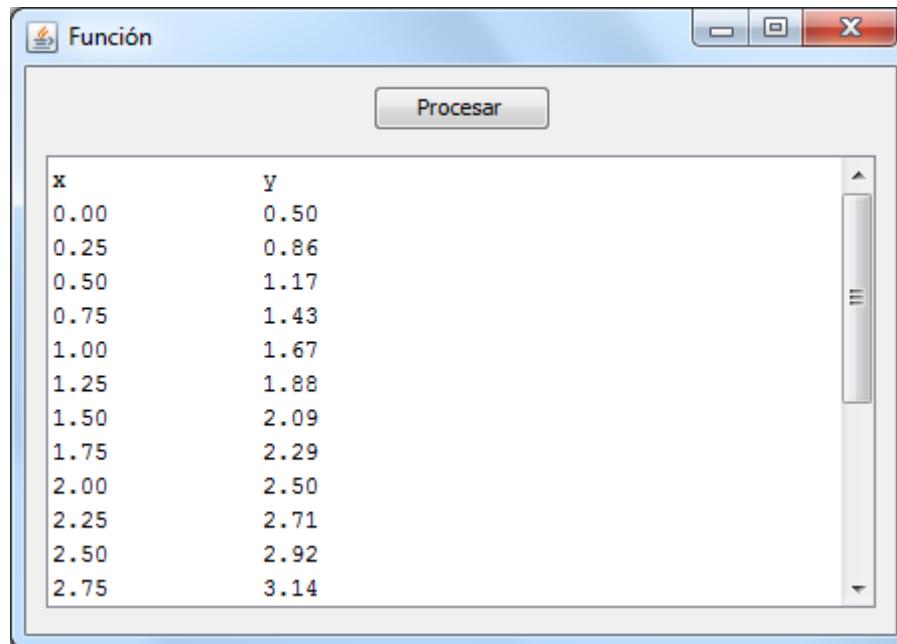
    // Imprime la suma de la serie
    txtS.append("Suma : " + sum);
}
}
```

## Problema 5

Diseñe un programa que imprima una tabla de valores de x e y, para valores de x en el intervalo de 0 a 2.75 cada 0.25, siendo:

$$y = \frac{x^3 + 3x + 1}{x^2 + 2}$$

### Interfaz Gráfica



### Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import java.util.Locale;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Funcion extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;
    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
    }
}
```

```
EventQueue.invokeLater(new Runnable() {
    public void run() {
        try {
            Funcion frame = new Funcion();
            frame.setVisible(true);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
});

// Crea la GUI
public Funcion() {
    setTitle("Función");
    setBounds(100, 100, 450, 321);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(173, 9, 89, 23);
    getContentPane().add(btnProcesar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 44, 414, 226);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declara e inicializa variables
    double x = 0, y;

    // Imprime la cabecera
    txtS.setText(String.format(Locale.US, "%-15.2s%-15.2s\n", "x", "y"));

    // Imprime la tabla
    do {
        // Calcula el valor de y para el valor actual de x
        y = (x * x * x + 3 * x + 1) / (x * x + 2);

        // Imprime la pareja de valores x e y
        txtS.append(String.format(Locale.US, "%-15.2f%-15.2f\n", x, y));

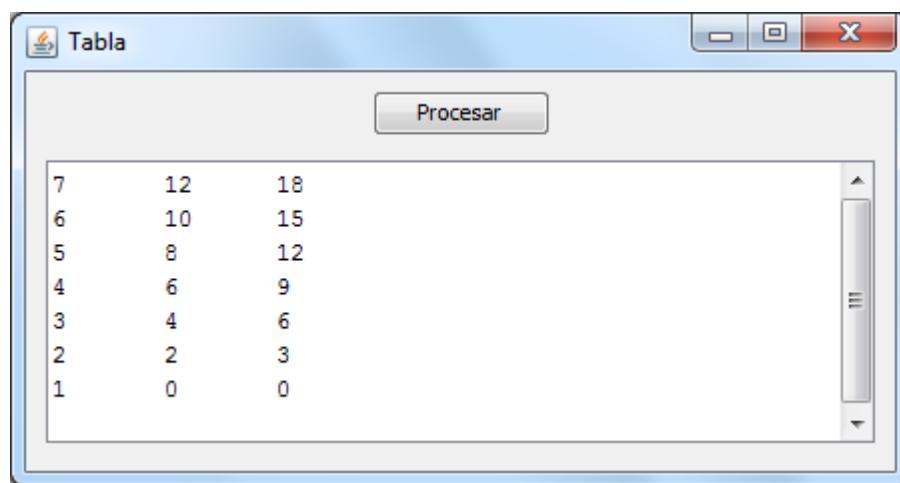
        // Genera el siguiente valor de x
        x += 0.25;
    } while (x <= 2.75);
}
```

## Problema 6

Diseñe un programa que imprima la siguiente tabla de números:

7	12	18
6	10	15
5	8	12
4	6	9
3	4	6
2	2	3
1	0	0

## Interfaz Gráfica



## Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Tabla extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;
    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
    }
}
```

```
EventQueue.invokeLater(new Runnable() {
    public void run() {
        try {
            Tabla frame = new Tabla();
            frame.setVisible(true);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
});

// Crea la GUI
public Tabla() {
    setTitle("Tabla");
    setBounds(100, 100, 450, 237);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(173, 9, 89, 23);
    getContentPane().add(btnProcesar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 44, 414, 141);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declara e inicializa variables
    int a = 7, b = 12, c = 18;

    // Limpia la pantalla
    txtS.setText("");

    // Imprime la tabla
    do {

        // Imprime un elemento de cada columna
        txtS.append(a + "\t" + b + "\t" + c + "\n");

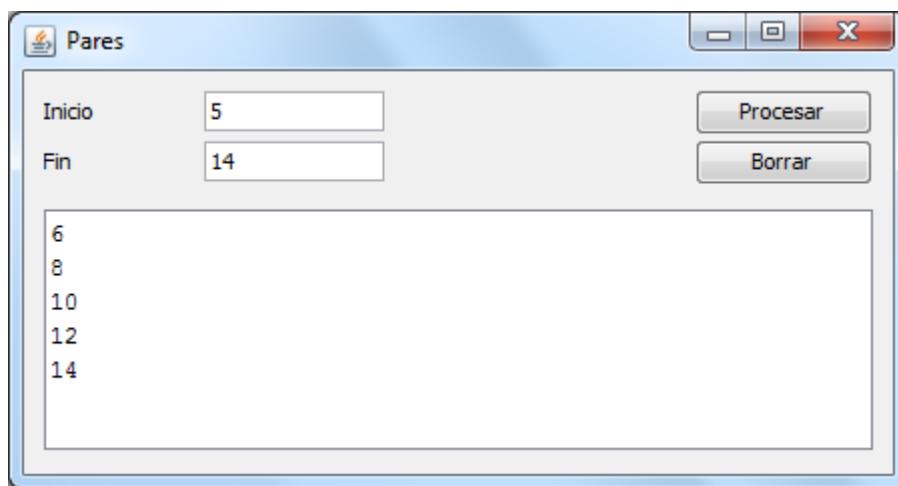
        // Genera el siguiente elemento de cada columna
        a--;
        b -= 2;
        c -= 3;

    } while (a >= 1);
}
```

## Problema 7

Diseñe un programa que lea los extremos de un intervalo de números enteros e imprima todos los números pares del intervalo.

### Interfaz Gráfica



### Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
```

```
public class Pares extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblInicio;
    private JLabel lblFin;
    private JTextField txtInicio;
    private JTextField txtFin;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
    }
}
```

```
EventQueue.invokeLater(new Runnable() {
    public void run() {
        try {
            Pares frame = new Pares();
            frame.setVisible(true);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
});

// Crea la GUI
public Pares() {
    setTitle("Pares");
    setBounds(100, 100, 450, 239);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblInicio = new JLabel("Inicio");
    lblInicio.setBounds(10, 13, 80, 14);
    getContentPane().add(lblInicio);

    lblFin = new JLabel("Fin");
    lblFin.setBounds(10, 38, 80, 14);
    getContentPane().add(lblFin);

    txtInicio = new JTextField();
    txtInicio.setBounds(90, 10, 90, 20);
    getContentPane().add(txtInicio);
    txtInicio.setColumns(10);

    txtFin = new JTextField();
    txtFin.setBounds(90, 35, 90, 20);
    getContentPane().add(txtFin);
    txtFin.setColumns(10);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(335, 34, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 69, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {

        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}
```

```
// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declara variables
    int inicio, fin, numero;

    // Ingresa los extremos del intervalo
    inicio = Integer.parseInt(txtInicio.getText());
    fin = Integer.parseInt(txtFin.getText());

    // Inicializa el número
    numero = inicio;

    // Imprime la lista de números pares
    do {
        if (numero % 2 == 0)
            txtS.append(numero + "\n");

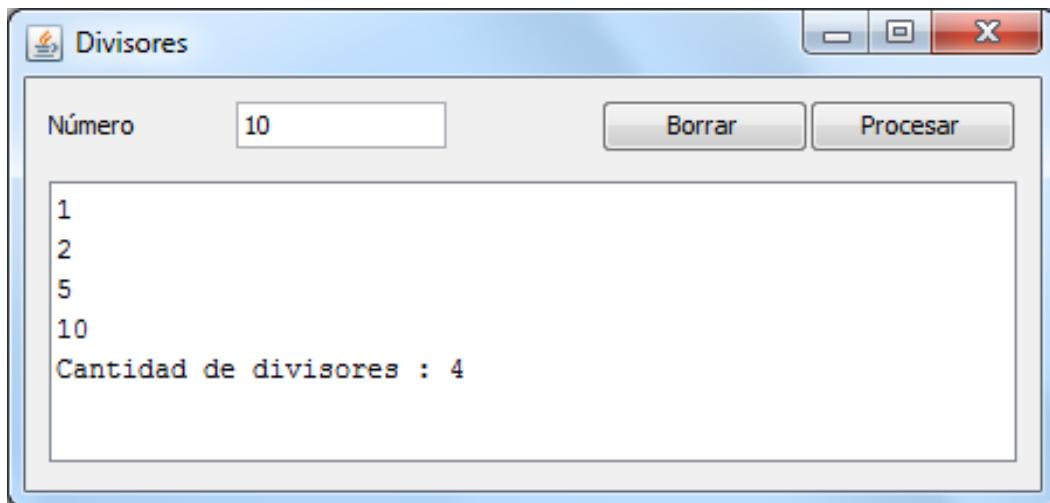
        numero++;
    } while (numero <= fin);
}

// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtInicio.setText("");
    txtFin.setText("");
    txtS.setText("");
    txtInicio.requestFocus();
}
```

### Problema 8

Diseñe un programa que imprima los divisores de un número natural y la cantidad de divisores encontrados.

#### Interfaz Gráfica



#### Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import javax.swing.JLabel;
import javax.swing.JTextField;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Divisores extends JFrame implements ActionListener {

    private static final long serialVersionUID = 1L;

    // Declaración de variables
    private JLabel lblNumero;
    private JTextField txtNumero;
    private JButton btnProcesar;
    private JButton btnBorrar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
    }
}
```

```
EventQueue.invokeLater(new Runnable() {
    public void run() {
        try {
            Divisores frame = new Divisores();
            frame.setVisible(true);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
});

// Crea la GUI
public Divisores() {

    setTitle("Divisores");
    setBounds(100, 100, 450, 214);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    lblNumero = new JLabel("Número");
    lblNumero.setBounds(10, 13, 80, 14);
    getContentPane().add(lblNumero);

    txtNumero = new JTextField();
    txtNumero.setBounds(90, 10, 90, 20);
    getContentPane().add(txtNumero);
    txtNumero.setColumns(10);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(335, 9, 89, 23);
    getContentPane().add(btnProcesar);

    btnBorrar = new JButton("Borrar");
    btnBorrar.addActionListener(this);
    btnBorrar.setBounds(246, 9, 89, 23);
    getContentPane().add(btnBorrar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 44, 414, 120);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {

    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }

    if (arg0.getSource() == btnBorrar) {
        actionPerformedBtnBorrar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declara e inicializa variables
    int contadiv = 0, div = 1, num;

    // Ingresá el número
    num = Integer.parseInt(txtNumero.getText());
}
```

```
// Busca los divisores del número
do {
    // Verifica si div es un divisor
    if (num % div == 0) {

        // Imprime el divisor encontrado
        txtS.append(div + "\n");

        // Cuenta el divisor encontrado
        contadiv++;
    }

    // Genera el siguiente posible divisor
    div++;
} while (div <= num);

// Imprime la cantidad de divisores encontrados
txtS.append("Cantidad de divisores : " + contadiv);
}

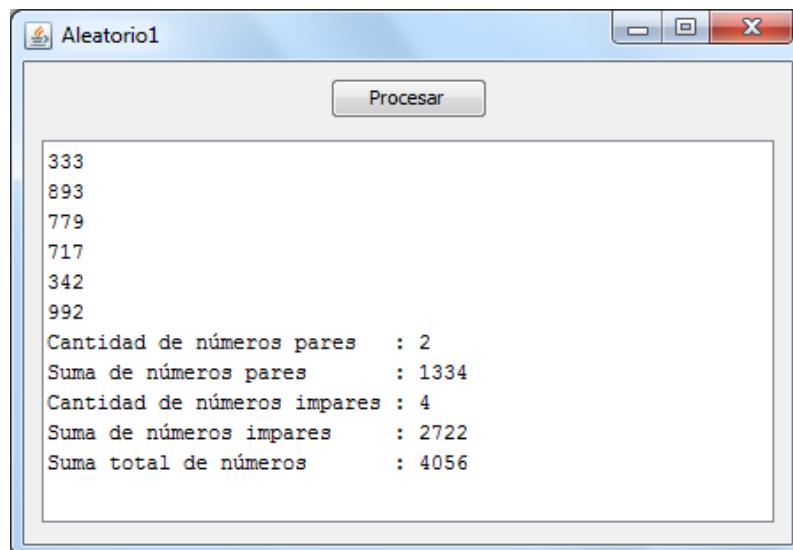
// Procesa la pulsación del botón Borrar
protected void actionPerformedBtnBorrar(ActionEvent arg0) {
    txtNumero.setText("");
    txtS.setText("");
    txtNumero.requestFocus();
}
```

## Problema 9

Diseñe un programa que genere números aleatorios enteros del intervalo 100 a 999 hasta obtener un número par mayor o igual a 500. Imprima lo números generados y determine:

- La suma de los números generados
- La cantidad de números pares generados
- La cantidad de números impares generados
- La suma de los números pares generados
- La suma de los números impares generados

### Interfaz Gráfica



### Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Aleatorio1 extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;
    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
    }
}
```

```

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Aleatorio1 frame = new Aleatorio1();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }

    // Crea la GUI
    public Aleatorio1() {
        setTitle("Aleatorio1");
        setBounds(100, 100, 450, 310);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        getContentPane().setLayout(null);

        btnProcesar = new JButton("Procesar");
        btnProcesar.addActionListener(this);
        btnProcesar.setBounds(173, 9, 89, 23);
        getContentPane().add(btnProcesar);

        scpScroll = new JScrollPane();
        scpScroll.setBounds(10, 44, 414, 216);
        getContentPane().add(scpScroll);

        txtS = new JTextArea();
        txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
        scpScroll.setViewportView(txtS);
    }

    // Direcciona eventos de tipo ActionEvent
    public void actionPerformed(ActionEvent arg0) {
        if (arg0.getSource() == btnProcesar) {
            actionPerformedBtnProcesar(arg0);
        }
    }

    // Procesa la pulsación del botón Procesar
    protected void actionPerformedBtnProcesar(ActionEvent arg0) {

        // Declara e inicializa variables
        int sumtot = 0, canpar = 0, sumpar = 0, canimp = 0, sumimp = 0, n;

        // Limpia la pantalla
        txtS.setText("");

        // Genera los números aleatorios
        do {
            // Genera un número aleatorio de 100 a 999
            n = (int) ((999 - 100 + 1) * Math.random() + 100);

            // Imprime el número generado
            txtS.append(n + "\n");

            // Suma el número generado
            sumtot += n;

            // Cuenta y suma el número según sea par o impar
            if (n % 2 == 0) {
                canpar++;
                sumpar += n;
            } else {
                canimp++;
                sumimp += n;
            }
        } while (! (n % 2 == 0 && n >= 500));
    }
}

```

```

        // Imprime el reporte solicitado
        txtS.append("Cantidad de números pares : " + canpar + "\n");
        txtS.append("Suma de números pares : " + sumpar + "\n");
        txtS.append("Cantidad de números impares : " + canimp + "\n");
        txtS.append("Suma de números impares : " + sumimp + "\n");
        txtS.append("Suma total de números : " + sumtot);
    }
}

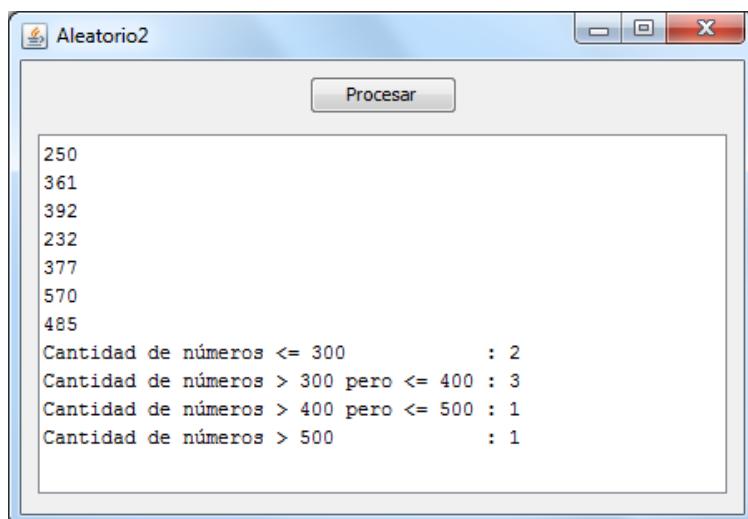
```

### Problema 10

Diseñe un programa que genere números aleatorios enteros del intervalo 200 a 600 hasta obtener un número impar mayor de 400 pero menor de 500. Imprima los números generados y determine:

- La cantidad de números generados ≤ 300
- La cantidad de números generados > 300 pero ≤ 400
- La cantidad de números generados > 400 pero ≤ 500
- La cantidad de números generados > 500

### Interfaz Gráfica



### Solución

```

package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Aleatorio2 extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;
    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

```

```
// Lanza la aplicación
public static void main(String[] args) {
    try {
        UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
    } catch (Throwable e) {
        e.printStackTrace();
    }

    EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                Aleatorio2 frame = new Aleatorio2();
                frame.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

// Crea la GUI
public Aleatorio2() {

    setTitle("Aleatorio2");
    setBounds(100, 100, 450, 310);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(173, 9, 89, 23);
    getContentPane().add(btnProcesar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 44, 414, 216);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {

    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declara e inicializa variables
    int c1 = 0, c2 = 0, c3 = 0, c4 = 0, n;

    // Limpia la pantalla
    txtS.setText("");

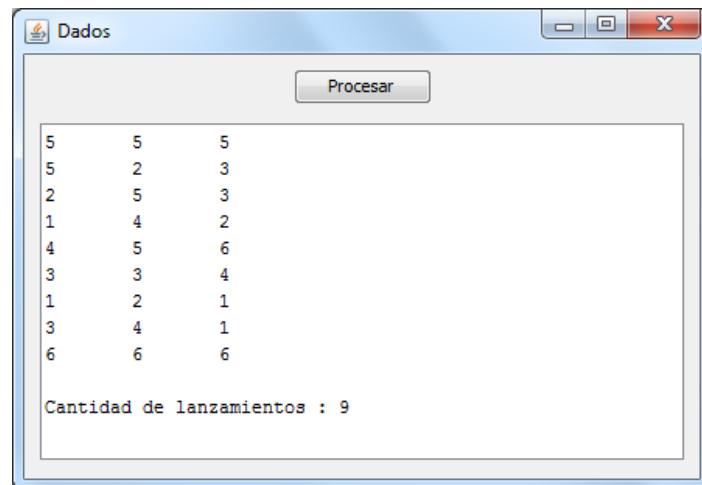
    // Genera los números aleatorios
    do {
        // Genera un número aleatorio de 200 a 600
        n = (int) ((600 - 200 + 1) * Math.random() + 200);

        // Imprime el número generado
        txtS.append(n + "\n");
    }
}
```

```
// Cuenta el número generado
if (n <= 300)
    c1++;
else if (n <= 400)
    c2++;
else if (n <= 500)
    c3++;
else
    c4++;
} while (!(n % 2 != 0 && n > 400 && n < 500));
// Imprime el reporte solicitado
txtS.append("Cantidad de números <= 300 : " + c1 + "\n");
txtS.append("Cantidad de números > 300 pero <= 400 : " + c2 + "\n");
txtS.append("Cantidad de números > 400 pero <= 500 : " + c3 + "\n");
txtS.append("Cantidad de números > 500 : " + c4);
}
```

**Problema 11**

Diseñe un programa que lance tres dados aleatoriamente hasta obtener 6 en los tres dados. Imprima los puntajes obtenidos en cada dado y la cantidad de lanzamientos efectuados.

**Interfaz Gráfica****Solución**

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Dados extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;
    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Dados frame = new Dados();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}
```

```
// Crea la GUI
public Dados() {
    setTitle("Datos");
    setBounds(100, 100, 450, 310);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(173, 9, 89, 23);
    getContentPane().add(btnProcesar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 44, 414, 216);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declara e inicializa variables
    int c = 0, dado1, dado2, dado3;

    // Limpia la pantalla
    txtS.setText("");

    // Simula el lanzamiento de los tres dados
    do {
        // Genera los puntajes de los tres dados
        dado1 = (int) (6 * Math.random() + 1);
        dado2 = (int) (6 * Math.random() + 1);
        dado3 = (int) (6 * Math.random() + 1);

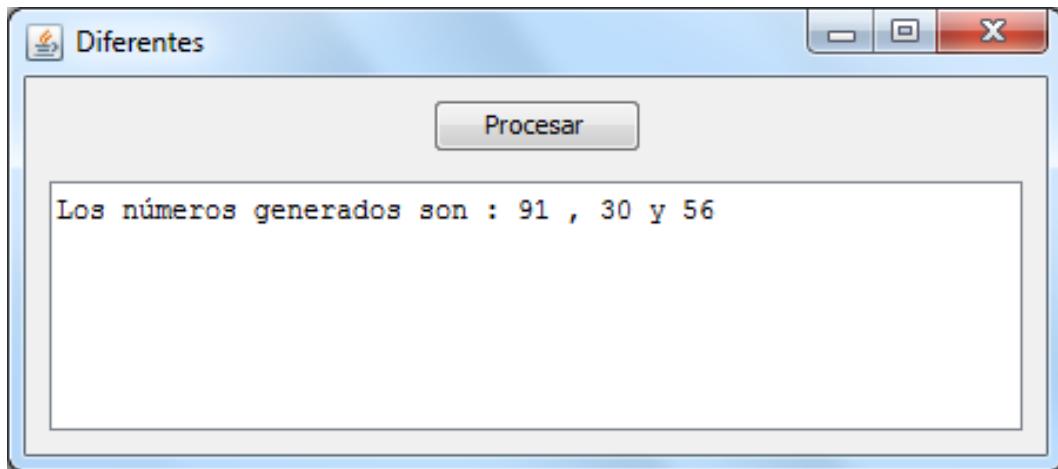
        // Imprime los puntajes de los dados
        txtS.append(dado1 + "\t" + dado2 + "\t" + dado3 + "\n");

        // Cuenta el lanzamiento efectuado c++;
    } while (!(dado1 == 6 && dado2 == 6 && dado3 == 6));

    // Imprime la cantidad de lanzamientos
    txtS.append("\nCantidad de lanzamientos : " + c);
}
}
```

**Problema 12**

Diseñe un programa que genere tres números aleatorios enteros diferentes de dos cifras.

**Interfaz Gráfica****Solución**

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Diferentes extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;
    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    Diferentes frame = new Diferentes();
                    frame.setVisible(true);
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        });
    }
}
```

```
// Crea la GUI
public Diferentes() {
    setTitle("Diferentes");
    setBounds(100, 100, 450, 198);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(173, 9, 89, 23);
    getContentPane().add(btnProcesar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 44, 414, 106);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {
    // Declara e inicializa variables
    int n1, n2, n3;

    // Limpia la pantalla
    txtS.setText("");

    // Genera los números aleatorios
    do {
        n1 = aleatorio();
        n2 = aleatorio();
        n3 = aleatorio();
    } while (n1 == n2 || n2 == n3 || n1 == n3);

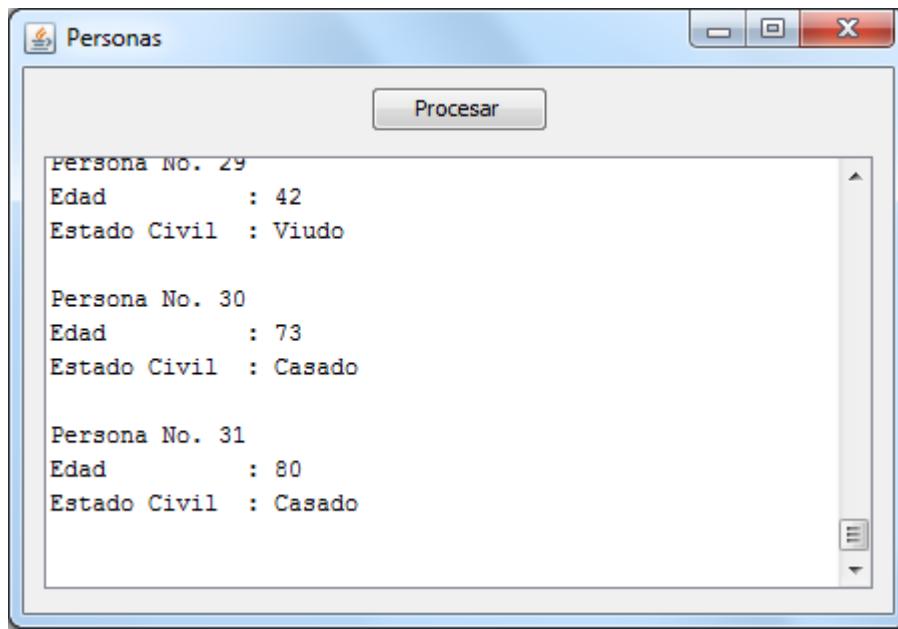
    // Imprime el reporte solicitado
    txtS.setText("Los números generados son : ");
    txtS.append(n1 + " , " + n2 + " y " + n3);
}

// Genera y retorna un número aleatorio de dos cifras
int aleatorio() {
    return (int) ((99 - 10 + 1) * Math.random() + 10);
}
```

### Problema 13

Diseñe un programa que genere aleatoriamente la edad y el estado civil de un conjunto de personas hasta obtener una edad igual a 80. La edad será obtenida del intervalo 25 a 80. El estado civil será soltero, casado, viudo o divorciado.

### Interfaz Gráfica



### Solución

```
package cibertec;

import java.awt.EventQueue;
import java.awt.Font;
import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.UIManager;
import java.awt.event.ActionListener;
import java.awt.event.ActionEvent;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;

public class Personas extends JFrame implements ActionListener {
    private static final long serialVersionUID = 1L;

    private JButton btnProcesar;
    private JScrollPane scpScroll;
    private JTextArea txtS;

    // Lanza la aplicación
    public static void main(String[] args) {
        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (Throwable e) {
            e.printStackTrace();
        }
    }
}
```

```
EventQueue.invokeLater(new Runnable() {
    public void run() {
        try {
            Personas frame = new Personas();
            frame.setVisible(true);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
});

// Crea la GUI
public Personas() {
    setTitle("Personas");
    setBounds(100, 100, 450, 310);
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    getContentPane().setLayout(null);

    btnProcesar = new JButton("Procesar");
    btnProcesar.addActionListener(this);
    btnProcesar.setBounds(173, 9, 89, 23);
    getContentPane().add(btnProcesar);

    scpScroll = new JScrollPane();
    scpScroll.setBounds(10, 44, 414, 216);
    getContentPane().add(scpScroll);

    txtS = new JTextArea();
    txtS.setFont(new Font("Monospaced", Font.PLAIN, 12));
    scpScroll.setViewportView(txtS);
}

// Direcciona eventos de tipo ActionEvent
public void actionPerformed(ActionEvent arg0) {
    if (arg0.getSource() == btnProcesar) {
        actionPerformedBtnProcesar(arg0);
    }
}

// Procesa la pulsación del botón Procesar
protected void actionPerformedBtnProcesar(ActionEvent arg0) {

    // Declara e inicializa variables
    int numper = 1; // Número de persona
    int edad; // Edad de la persona
    String estado; // Estado civil de la persona

    // Limpia la pantalla
    txtS.setText("");

    // Repite hasta que la edad sea 80
    do {

        // Genera los datos de la persona
        edad = generaEdad();
        estado = generaEstadoCivil();

        // Imprime los datos de la persona
        txtS.append("Persona No. " + numper + "\n");
        txtS.append("Edad : " + edad + "\n");
        txtS.append("Estado Civil : " + estado + "\n\n");

        // Incrementa el número de persona
        numper++;

    } while (edad != 80);
}
```

```
// Genera y retorna la edad de una persona del intervalo 25 a 80
int generaEdad() {
    return (int) ((80 - 25 + 1) * Math.random() + 25);
}

// Genera y retorna el estado civil de una persona
String generaEstadoCivil() {
    // Genera un número aleatorio del intervalo 1 a 4
    int e = (int) ((4 - 1 + 1) * Math.random() + 1);

    // Obtiene y retorna el estado civil
    switch (e) {
        case 1:
            return "Soltero";
        case 2:
            return "Casado";
        case 3:
            return "Viudo";
        default:
            return "Divorciado";
    }
}
```

# Problemas propuestos

## Actividad

1. Diseñe un programa que simule varios lanzamientos de un dado hasta obtener un seis. Muestre los puntajes del dado conforme se vayan generando y muestre al final cuántos lanzamientos fueron necesarios efectuar
  
2. Diseñe un programa que genere números aleatorios del intervalo 10 a 90 hasta obtener un número par mayor de 45 pero menor de 55. El programa mostrará:
  - Los números generados.
  - La cantidad de números generados
  - La suma de todos los números generados.
  - La cantidad de números generados del intervalo 10 a 30
  - La cantidad de números generados del intervalo 31 a 50
  - La cantidad de números generados del intervalo 51 a 70
  - La cantidad de números generados del intervalo 71 a 90
  
3. Diseñe un programa que genere números aleatorios del intervalo 200 a 800 hasta obtener un número impar mayor de 500. El programa mostrará:
  - Los números generados
  - La suma de todos los números generados.
  - La cantidad de números generados del intervalo 200 a 400
  - La cantidad de números generados del intervalo 401 a 600
  - La cantidad de números generados del intervalo 601 a 800
  
4. Diseñe un programa que genere sueldos aleatorios enteros con valores del intervalo 2500 a 3500 hasta obtener un sueldo igual a 3500 o igual a 2500. Imprima los sueldos generados y determine:
  - El sueldo promedio
  - La cantidad de sueldos  $> S/.3000$
  - La cantidad de sueldos  $\leq S/.3000$

## Autoevaluación

1. Diseñe un programa de que genere edades aleatorias enteras con valores del intervalo 20 a 70 hasta obtener una edad mayor de 40 pero menor de 50. Imprima las edades generadas y determine:
  - La edad promedio
  - La cantidad de personas menores de edad
  - La cantidad de personas mayores de edad
2. Diseñe un programa que genere stocks aleatorios con valores del intervalo 0 a 200 hasta obtener un stock igual a 0. Imprima los stocks generados y determine:
  - La cantidad de stocks < 50
  - La cantidad de stocks  $\geq 50$  pero  $< 100$
  - La cantidad de stocks  $\geq 100$  pero  $< 150$
  - La cantidad de stocks  $\geq 150$
3. Diseñe un programa que genere sueldos aleatorios enteros con valores del intervalo 2000 a 4000 hasta obtener un sueldo mayor de 2500 pero menor de 3500. Imprima los sueldos generados y determine:
  - La suma de los sueldos generados
  - La cantidad de sueldos < 2500
  - La cantidad de sueldos  $\geq 2500$  pero  $< 3000$
  - La cantidad de sueldos  $\geq 3000$  pero  $< 3500$
  - La cantidad de sueldos  $\geq 3500$
4. Diseñe un programa que genere números aleatorios enteros del intervalo 1 a 10000 hasta obtener un número igual a 500. Imprima los números generados y determine:
  - La cantidad de números generados
  - La cantidad de números generados de una cifra
  - La cantidad de números generados de dos cifras
  - La cantidad de números generados de tres cifras
  - La cantidad de números generados de cuatro cifras

## Para recordar

- La estructura **do-while** evalúa su condición de control luego de ejecutar su cuerpo de acciones, a diferencia de la estructura **while** que, primero, prueba su condición de control y, luego, ejecuta su cuerpo de acciones.
- El cuerpo de acciones de la estructura **do-while** se ejecutará por lo menos una vez, a diferencia de la estructura **while** que podría no ejecutar su cuerpo de acciones.

# Bibliografía

- Deitel, P. J. (2016). *Java: cómo programar.* 10a ed. Pearson Educación.  
Centro de Información: Código 005.133J DEIT 2016
- Hinojosa Lazo, H. A. (2012). *Fundamentos de programación.* UNMSM, Fondo Editorial.  
Centro de Información: Código 005.1 HINO
- Joyanes Aguilar, L. (2013). *Fundamentos generales de programación.* McGraw-Hill.  
Centro de Información: Código 005.1 JOYA/U
- Marcelo Villalobos, R. W. (2016). *Fundamentos de programación Java con más de 100 algoritmos.* Macro.  
Centro de Información: Código 005.133J MARC/F
- Sznajdleder, P. A. (2016). *Java a fondo: curso de programación.* 3a ed. Alfaomega.  
Centro de Información: Código 005.133J SZNA 2016