UNIDAD DE APRENDIZAJE
3

SEMANA
7

Estructura de selección múltiple switch

LOGRO DE LA UNIDAD DE APRENDIZAJE

Al finalizar la unidad, el alumno, mediante el uso de estructuras de selección, diseña algoritmos que involucren procesos selectivos.

TEMARIO

1. Estructura de selección múltiple switch

ACTIVIDADES

Los alumnos desarrollan algoritmos que involucren estructuras de selección múltiple.

1. Estructura de selección múltiple switch

La e*structura 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.

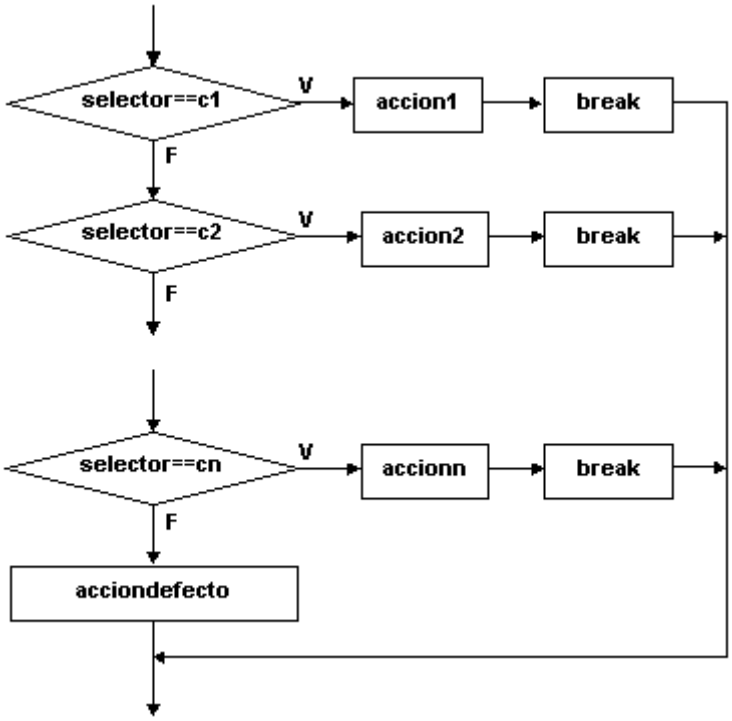


Figura 1 Estructura de Selección Múltiple **switch**

En la tabla que sigue, se muestra el código y el pseudocódigo de la estructura switch.

Código Java	Pseudocódigo

`switch( selector ){`
`  case c1:`
`  accion1;`
`  break;`
`  case c2:`
`  accion2;`
`  break;`
`  .`
`  .`
`  .`
`  case cn:`
`  accionn;`
`  break;`
`  default :`
`  acciondefecto;`
`}`
	`segun( selector ){`
	`  caso c1:`
	`  accion1`
	`  caso c2:`
	`  accion2`
	`  .`
	`  .`
	`  .`
	`  caso cn:`
	`  accionn`
	`  defecto:`
	`  acciondefecto`
	`}`

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.

2. 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

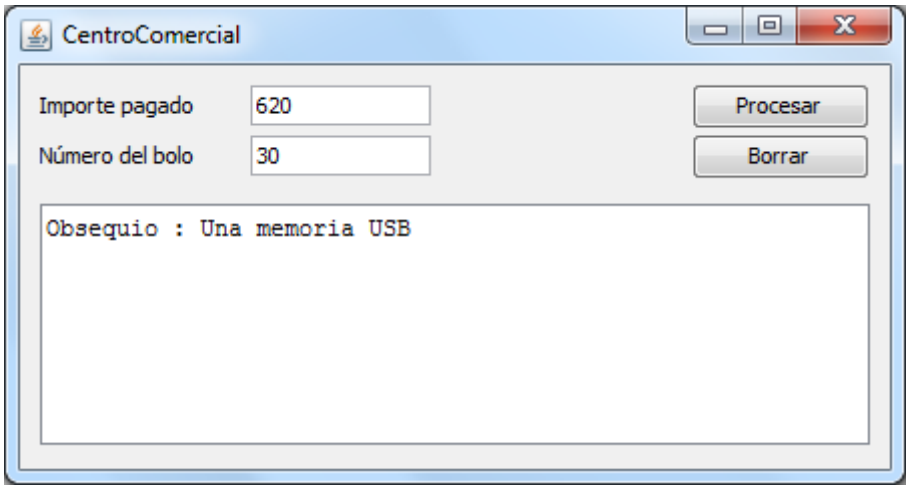
Dado el importe total pagado y el número del bolo, diseñe un programa que determine el obsequio correspondiente.

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
```

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("N\u00FAmero 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 imptotpag;
        String obsequio;

        // Entrada de datos
        imptotpag = 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

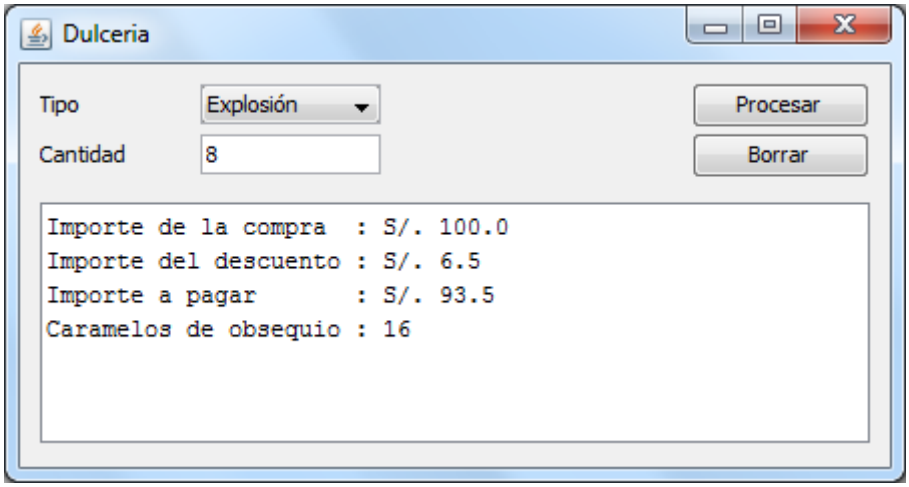
```
Inicio
    // Declaración de variables
    entero tipo, cantidad, caramelos
    real impcom, impdes, imppag

    // 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
```

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

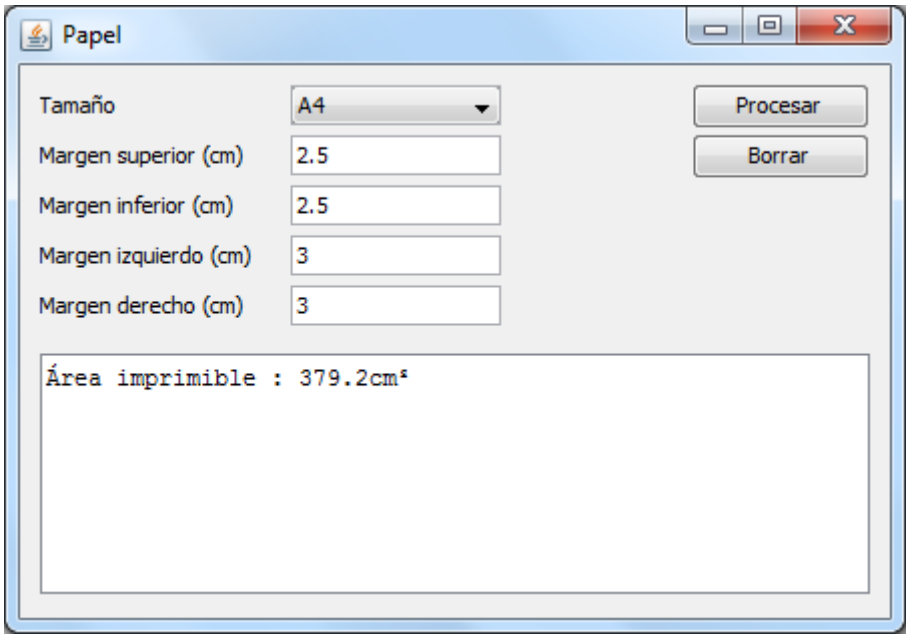
Se conoce como área imprimible al área que queda libre luego de descontar los márgenes superior, inferior, izquierdo y derecho. Dado el tamaño del papel y los márgenes superior, inferior, izquierdo y derecho en centímetros, diseñe un programa que determine el área imprimible en cm^2.

Algoritmo

```
Inicio
    // Declaración de variables
    entero tamaño
    real mrgsup, mrginf, mrgder, mrgizq, ancho, alto, area

    // Entrada de datos
    Leer 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 area imprimible
    area = (ancho-mrgizq-mrgder)*(alto-mrgsup-mrginf)

    // Muestra al área imprimible
    Imprimir area
Fin
```

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 ade 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 area 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:

	Precio del boleto	
Calidad	**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.

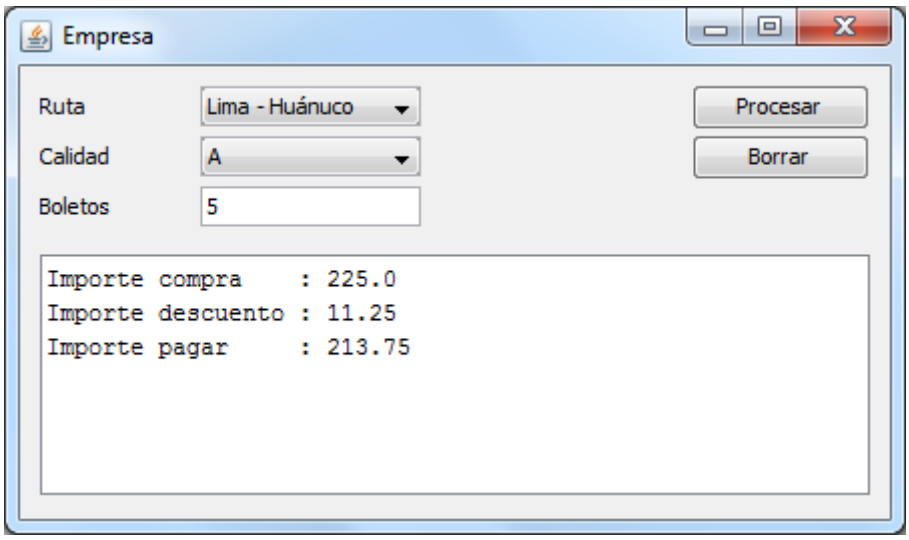
Dada la ruta elegida, la calidad del servicio y la cantidad de boletos adquiridos, diseñe un programa que determine el importe de la compra, el importe del descuento y el importe a pagar.

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
```

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

Reemplaze 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;
    case 1:
        precio = 15;
        break;
    case 2:
```

```
        precio = 10;
        break;
    default:
        precio = 12;
}
```

Problema 6

Reemplaze 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

El equivalente **switch** es el siguiente:

```
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

Reemplaze 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;
}
```