

Prueba de Caja Blanca

“aplicación de gestión de comandas para una cevichería”

Integrantes:

Cristian Becerra
Kevin Castillo
Ismael Silva

Fecha: 2025-12-03

Prueba caja blanca de describa el requisito funcional

1. CÓDIGO FUENTE

Pegar el trozo de código fuente que se requiere para el caso de prueba

```
private void confirmarComanda() {
    if (cantidadesSeleccionadas.isEmpty()) {
        JOptionPane.showMessageDialog(this,
            "X Por favor, agregue al menos un producto al pedido",
            "Error",
            JOptionPane.ERROR_MESSAGE);
        return;
    }
    String cliente = txtCliente.getText().trim();
    if (cliente.isEmpty()) {
        JOptionPane.showMessageDialog(this,
            "X Por favor, ingrese el nombre del cliente",
            "Error",
            JOptionPane.ERROR_MESSAGE);
        txtCliente.requestFocus();
        return;
    }
    double total = 0;
    StringBuilder detalle = new StringBuilder();
    detalle.append("=== COMANDA CONFIRMADA ===\n\n");
    detalle.append("Cliente: ").append(cliente).append("\n");
    detalle.append("Fecha: ").append(new SimpleDateFormat("dd/MM/yyyy HH:mm:ss").format(new Date())).append("\n");
    detalle.append("Nº Pedido: #").append(String.format("%04d", new Random().nextInt(10000))).append("\n\n");
    detalle.append("DETAJE DEL PEDIDO:\n");
    detalle.append(" ".repeat(32)).append("\n");

    for (Map.Entry<Integer, Integer> entry : cantidadesSeleccionadas.entrySet()) {
        Producto producto = productosMap.get(entry.getKey());
        int cantidad = entry.getValue();
        double subtotal = producto.precio * cantidad;
        total += subtotal;

        detalle.append(String.format("• %s\n", producto.nombre));
        detalle.append(String.format("  %d x S/ %.2f = S/ %.2f\n", cantidad, producto.precio, subtotal));
    }
    detalle.append(" ".repeat(32)).append("\n");
    detalle.append(String.format("💰 TOTAL: S/ %.2f\n", total));

    if (!txtComentarios.getText().trim().isEmpty()) {
        detalle.append("💬 COMENTARIOS:\n");
        detalle.append(txtComentarios.getText()).append("\n\n");
    }
    detalle.append("✅ Comanda enviada a la cocina\n");
    detalle.append("🕒 Tiempo estimado: 20-30 minutos");

    JTextArea areaResumen = new JTextArea(detalle.toString());
    areaResumen.setFont(new Font("Monospaced", Font.PLAIN, 12));
    areaResumen.setEditable(false);

    JScrollPane scrollResumen = new JScrollPane(areaResumen);
    scrollResumen.setPreferredSize(new Dimension(500, 400));

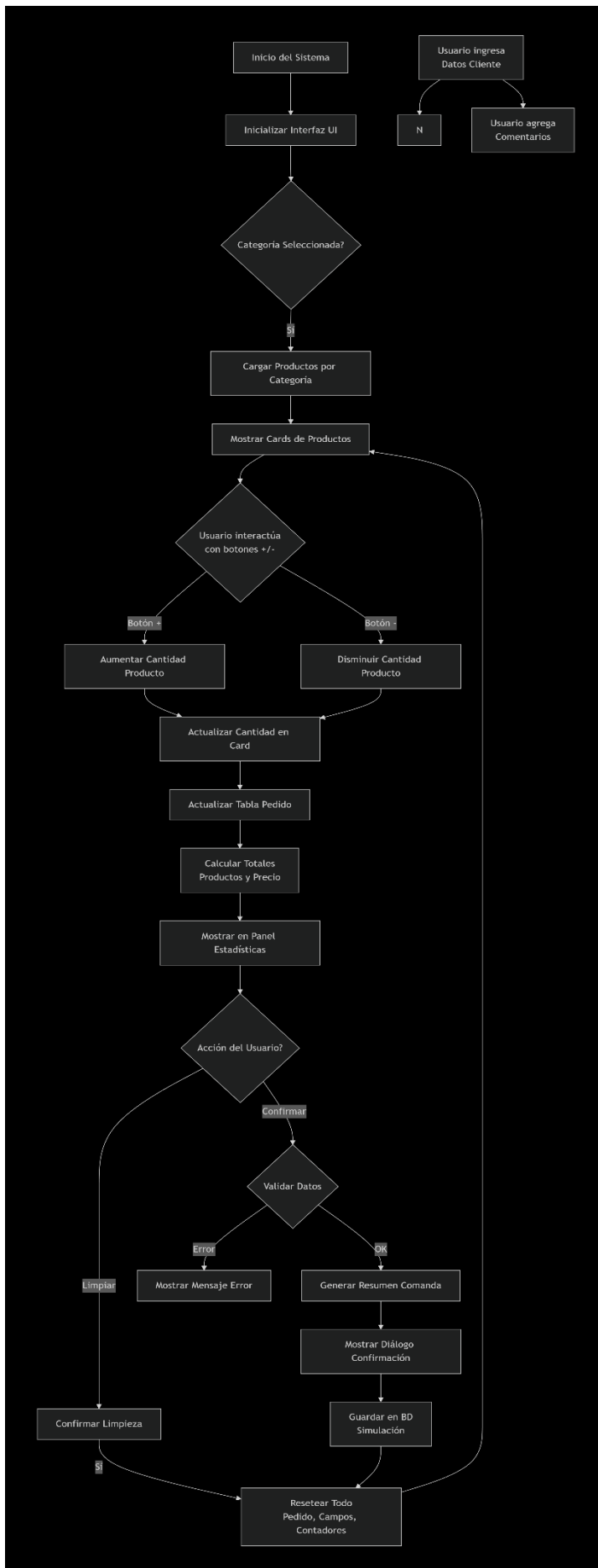
    JOptionPane.showMessageDialog(this,
        scrollResumen,
        "✅ Comanda Confirmada - #" + String.format("%04d", new Random().nextInt(10000)),
        JOptionPane.INFORMATION_MESSAGE);

    guardarComandaEnBD(cliente, total);

    limpiarPedido();
}
```

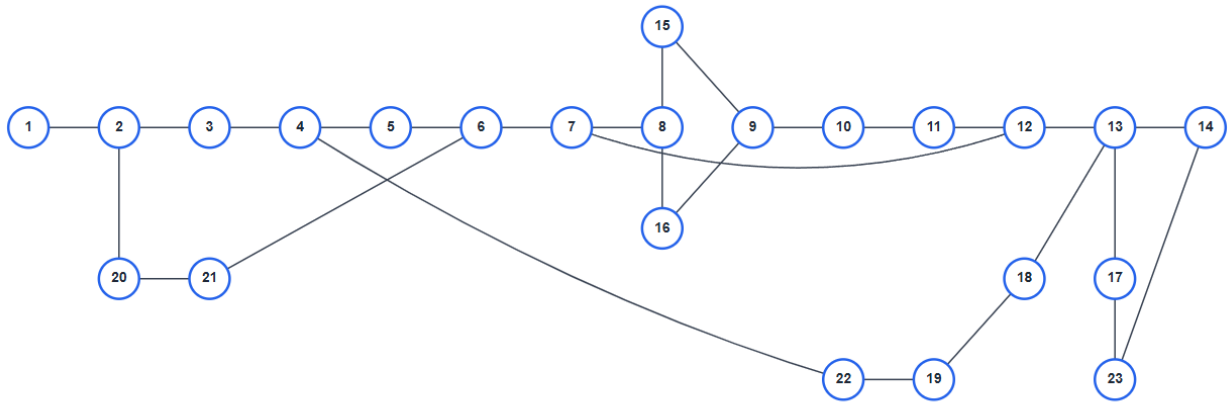
2. DIAGRAMA DE FLUJO (DF)

Realizar un DF del código fuente del numeral 1



3. GRAFO DE FLUJO (GF)

Realizar un GF en base al DF del numeral



4. IDENTIFICACIÓN DE LAS RUTAS (Camino basico)

Determinar en base al GF del numeral 4
RUTAS

R1: N1 → N2 → N3 → N4 → N5 → N23

R2: N1 → N2 → N3 → N4 → N5 → N6 → N7 → N9 → N10 → N11 → N12 → N15 → N23

R3: N1 → N2 → N3 → N4 → N5 → N13 → N15 → N18 → N20 → N21 → N22 → N17 → N5 → N23

R4: N1 → N2 → N3 → N4 → N5 → N13 → N15 → N18 → N19 → N13 → N15 → N18 → N20 → N21 → N22 → N17 → N5 → N23

R5: N1 → N2 → N3 → N4 → N5 → N6 → N8 → N9 → N10 → N11 → N12 → N15 → N16 → N15 → N23

R6: N1 → N2 → N3 → N4 → N5 → N13 → N15 → N16 → N17 → N5 → N23

R7: N1 → N2 → N3 → N4 → N5 → N6 → N15 → N23

5. COMPLEJIDAD CICLOMÁTICA

Se puede calcular de las siguientes formas:

Por fórmula de McCabe:

$$V(G) = A - N + 2P$$

$$V(G) = 29 - 23 + 2(1)$$

$$V(G) = 29 - 23 + 2$$

$$V(G) = 8$$

Por nodos de decisión:

$$V(G) = P + 1$$

Donde P = número de nodos predicado

$$V(G) = 5 + 1$$

$$V(G) = 6$$

Por regiones del grafo:

Contando regiones en el grafo plano:

- Región exterior: 1
- Regiones interiores: 7

Total regiones: 8

Resultado Final:

$$V(G) = 8$$

DONDE:

P: Número de nodos predicado

A: Número de aristas

N: Número de nodos