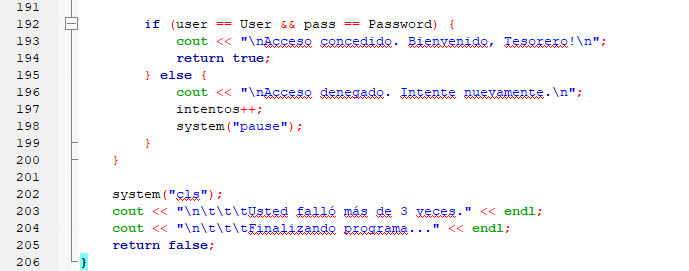
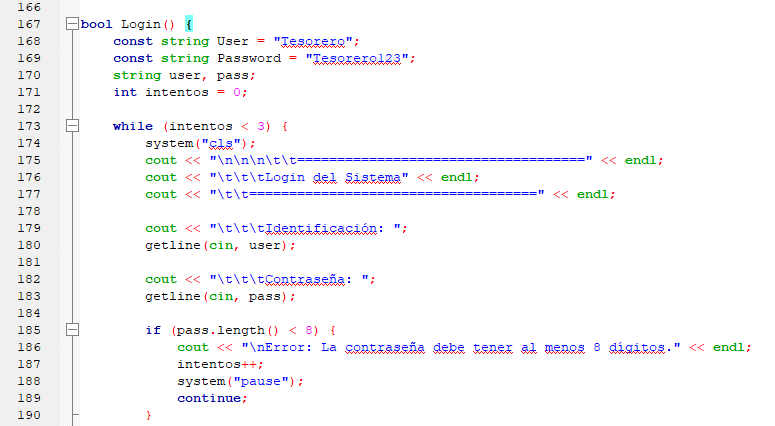
|  |
| --- |
|  |
| Prueba de Caja Blanca |
| ***“Aplicativo Tesorero”*** |
|  |
|  |
| **Integrantes: Paillacho Carlos, Sigsi David**  **Fecha 2025/01/21** |

**Prueba caja blanca de Requisito1 (Acceso al Sistema)**

1. **CÓDIGO FUENTE**



1. **Tabla de resumen**

|  |  |
| --- | --- |
| **Usuario** | Tesorero del condominio/conjunto residencial |
| **Procedimiento** | Acceder al sistema con credenciales, mensaje de error si la contraseña no tiene 8 dígitos, finalización del programa si excede 3 intentos fallidos |
| **Encargado** | **Desarrollador:** Carlos Paillacho |
| **Conclusiones** | Se implementó un sistema eficiente para el control del tesorero al sistema. |

1. **DIAGRAMA DE FLUJO (DF)**

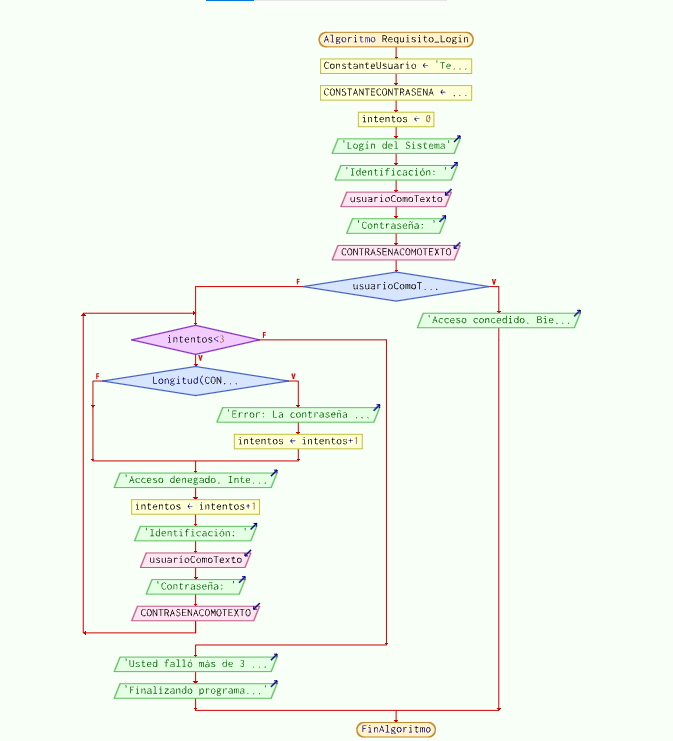
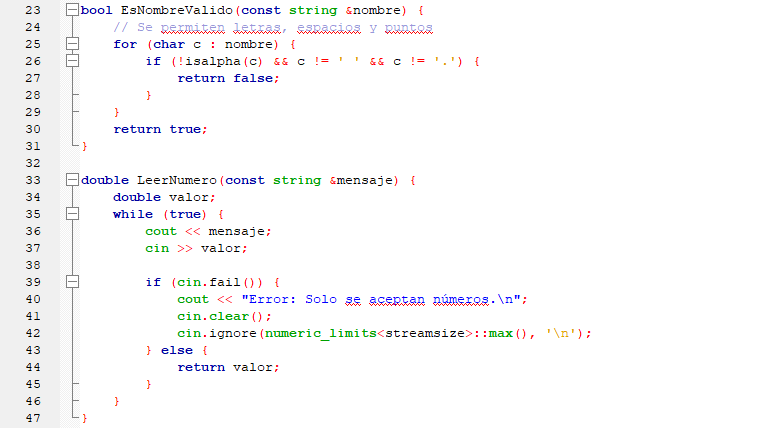
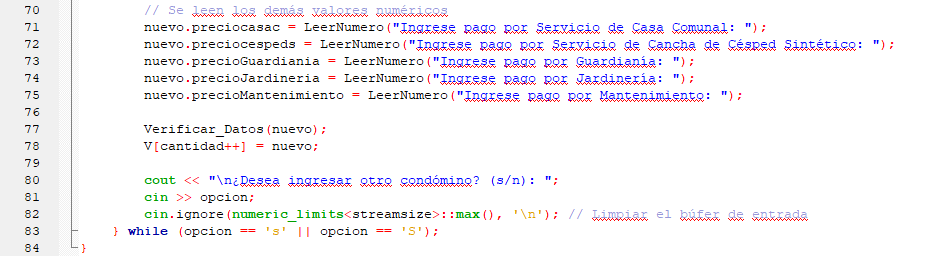
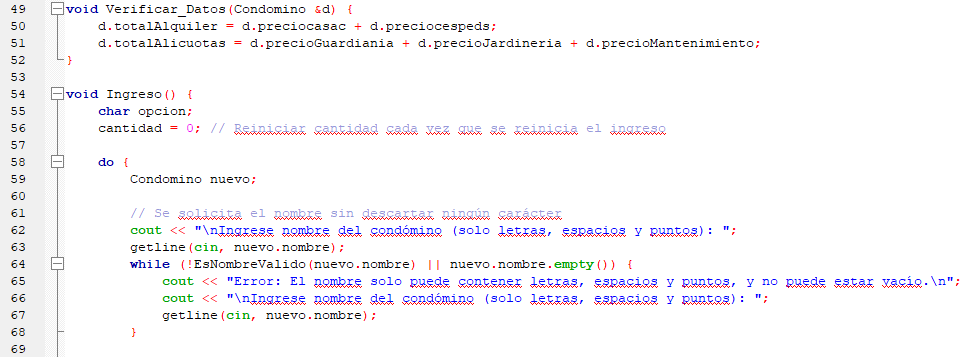


Figura 1 Nota: Se adaptó el código C++ para funcionar en PseInt

**Prueba caja blanca de Requisito 2 (Registros de pagos)**

1. ***CÓDIGO FUENTE***

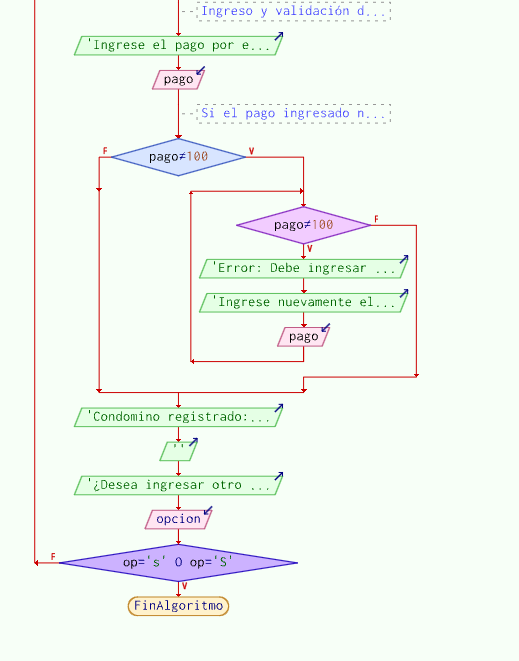
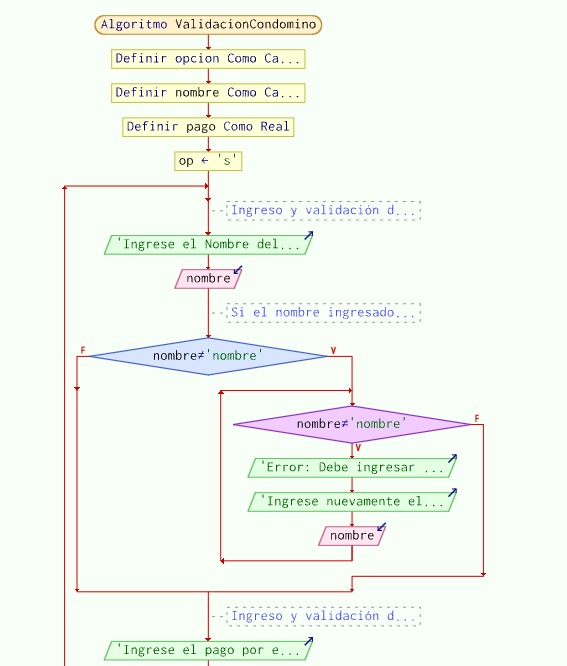




1. ***Tabla de resumen***

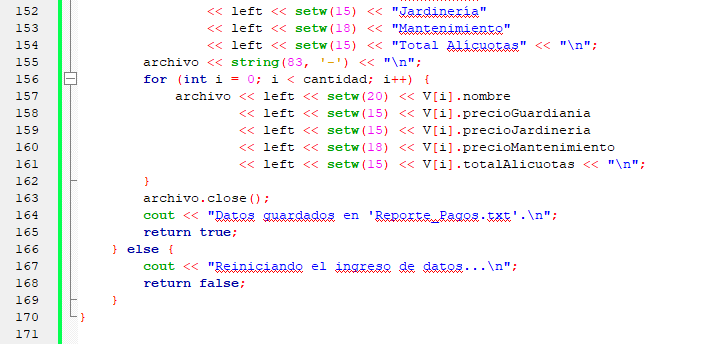
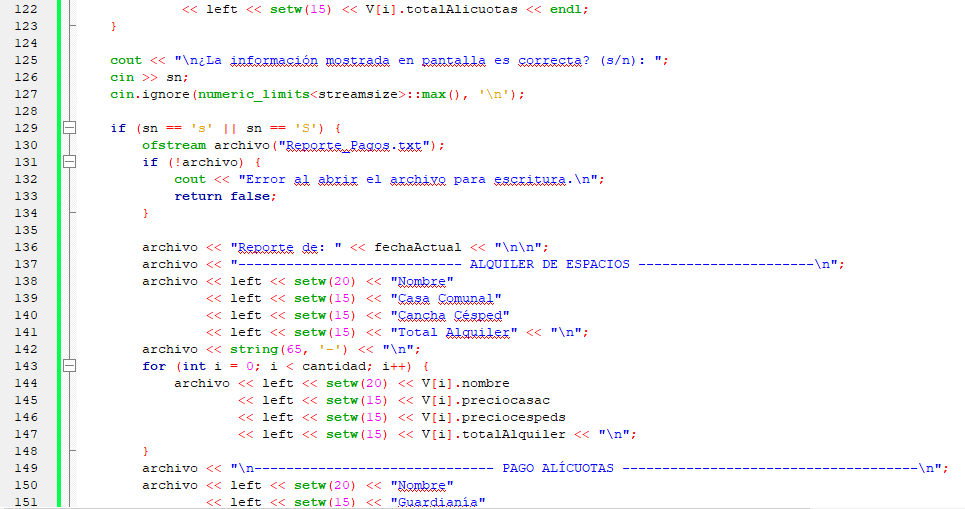
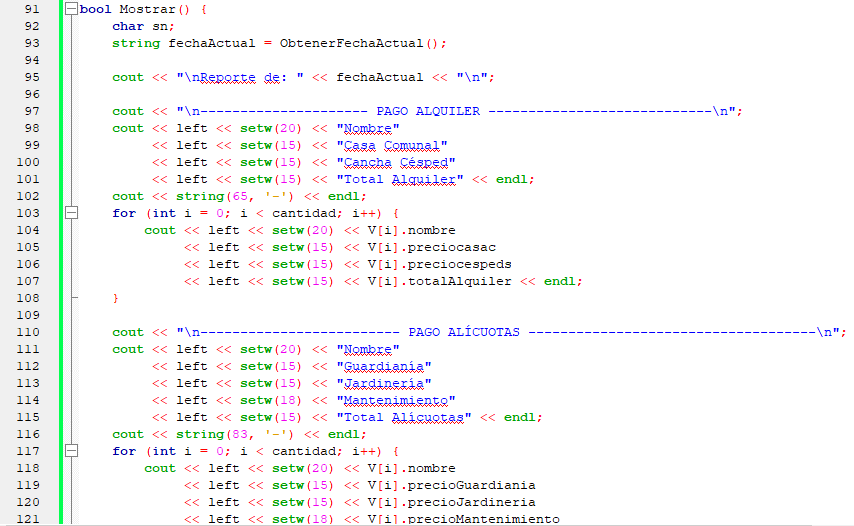
|  |  |
| --- | --- |
| **Usuario** | Tesorero del condominio/conjunto residencial |
| **Procedimiento** | Acceder al sistema con credenciales, ingresar información de condóminos, verificar cálculos y generar facturas. |
| **Encargado** | **Desarrollador:** Carlos Paillacho |
| **Conclusiones** | Se implementó un sistema eficiente para el control de pagos y alícuotas, asegurando precisión y facilidad de uso para el tesorero. |

***3. DIAGRAMA DE FLUJO (DF)***



**Prueba de Caja Blanca de Requisito 3 (Impresión de Factura)**

1. ***CÓDIGO FUENTE***

******

1. ***Tabla de resumen***

|  |  |
| --- | --- |
| **Usuario** | Tesorero del condominio/conjunto residencial |
| **Procedimiento** | Se imprime la matriz de Alquiler y la matriz de Alícuotas. Se pide al usuario confirme con “s” si la información es correcta. En caso de negativa el usuario puede reingresar los datos. Una vez aceptados los datos con “s”, el programa crea un bloc de notas con la factura. |
| **Encargado** | **Desarrollador:** Carlos Paillacho |
| **Conclusiones** | El requisito funcional es que el usuario tenga la posibilidad de cambiar los datos en caso de haber cometido una equivocación durante el proceso; consideramos con esto haber logrado crear un sistema amigable al usuario. |

1. **DIAGRAMA DE FLUJO (DF)**
2. **GRAFO DE FLUJO (GF)**

Realizar un GF en base al DF del numeral 2

Patrón de fondo

Descripción generada automáticamente

[1] 🡪 Inicio

[2] 🡪**Login**

[3] 🡪 validación de credenciales.

[4] 🡪 sí// mensaje de validación Ingreso del nombre del condómino.

[5] 🡪 no//Mensaje de error.

[6]🡪 Si // regresa al nodo [2]  
 No// solo tiene 3 intentos.

[7] 🡪 Ingresar los valores pagados

[8] 🡪 Cálculo de valores .

[9] 🡪 pregunta si se desea ingresar otro condómino.

[10] 🡪 no//repetición del bucle.

[11] 🡪 sí //Impresión de la factura

[12] 🡪 Fin.

1. **IDENTIFIACCIÒN DE LAS RUTAS**

**🔹 #1: Acceso exitoso y flujo normal**

R1: 1 → 2 → 3 → 5 → 6 → 7 → 8 → 9 → 10 → 11 → 12 → 13 → 14 → 15 → 16 → 17

🔹 **#2: Acceso fallido por intentos agotados**

R2: 1 → 2 → 3 → 4 → 2 → 3 → 4 → 2 → 3 → 4 → 18

**🔹 #3: Nombre inválido en el bucle de validación**

R3: 1 → 2 → 3 → 5 → 6 → 7 → 8 → 9 → 10 → 9 → 10 → 9 → 11 → 12 → 13 → 14 → 15 → 16 → 17

**🔹 #4: Ingreso de múltiples condóminos**

R4: 1 → 2 → 3 → 5 → 6 → 7 → 8 → 9 → 10 → 11 → 12 → 13 → 14 → 8 → 9 → 10 → 11 → 12 → 13 → 14 → 15 → 16 → 17

1. **COMPLEJIDAD CICLOMÁTICA**

**Se puede calcular de las siguientes formas:**

**Fórmula 1: V(G) = A - N + 2P**

* A (Aristas): Ahora tenemos 14 aristas.
* N (Nodos): Son 12 nodos en total.
* P (Nodos de decisión): Son 3 nodos de decisión.

**Entonces:**

V(G)=A−N+2P=14−12+2(3)=14−12+6=4  
V(G) = A - N + 2P = 14 - 12 + 2(3) = 14 - 12 + 6 = 4  
V(G)=A−N+2P=14−12+2(3)=14−12+6=4

**Fórmula 2: V(G) = P + 1**

* P (Nodos de decisión): Son 3.

**Entonces:**

V(G)=P+1=3+1=4V(G) = P + 1 = 3 + 1 = 4V(G)=P+1=3+1=4