Prueba de Caja Negra

"Proyecto Proformas Serviglas"

Integrantes: Mateo Neppas, Morrison Quillupangui, Mateo Arellano y Freddy Fuentes

Fecha: 16-01-2025

Versión Final

Historia de Revisión

Fecha	Versión	Descripción	Autores
10/01/2025	1	Versión inicial	Mateo Neppas, Morrison Quillupangui, Mateo Arellano y Freddy Fuentes
15/01/2025	2	Segunda Versión	Mateo Neppas, Morrison Quillupangui, Mateo Arellano y Freddy Fuentes
20/01/2025	3	Tercera versión	Mateo Neppas, Morrison Quillupangui, Mateo Arellano y Freddy Fuentes
12/02/2025	4	Cuarta versión	Mateo Neppas, Morrison Quillupangui, Mateo Arellano y Freddy Fuentes

1. REQOO1-REGISTRO DE USUARIO

1.1 Historia de Usuario

Campo	Descripcion
Número	REQ 001
Usuario	Administrador
Nombre de Historia	Registro de usuario
Prioridad en Negocio	Alta
Riesgo en Desarrollo	Medio
Iteración Asignada	No especificado
Programador Responsable	Mateo Arellanos
Descripción: - Ingresar al sistema. - Seleccionar la opción "Registro de Usuario". - Ingresar datos requeridos (nomb re de usuario, contraseña). - Confirmar registro. - El sistema valida la información y confirma el registro exitoso.	
Validación	El usuario debe ser registrado correctamente y poder iniciar sesión.

1.2 Particion de equivalencias

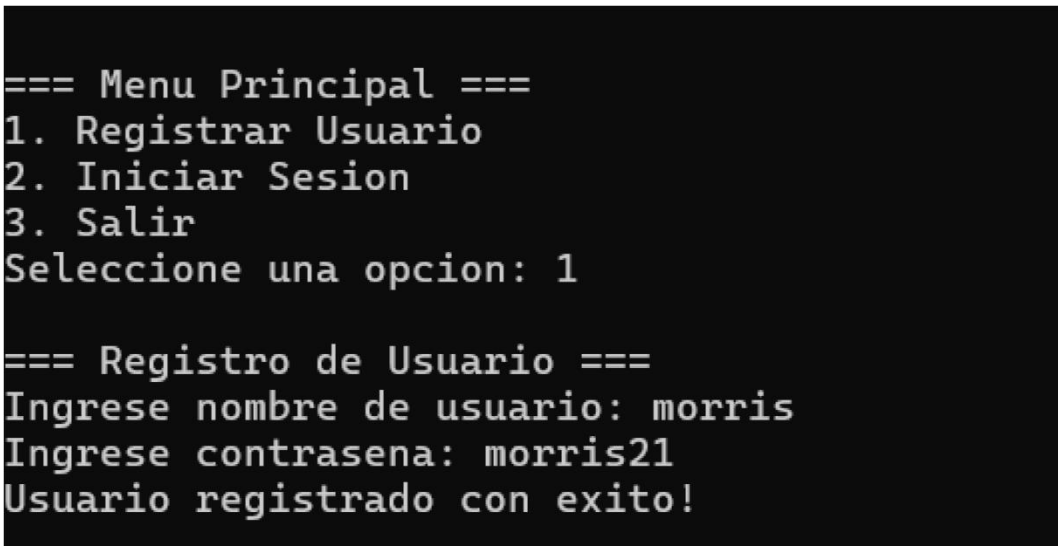
VARIABLE	**CLASE DE EQUIVALENCIA**	**ESTADO**	**REPRESENTANTE**
newUser.username	Ec1: “newUser.username”== " newUser.username "	Valido	usuario1
newUser.password	Ec1: “newUser.password”== “newUser.password”	Valido	12345

Descripción de la elución de acuerdo a los casos de prueba de la tabla 1, casos pass (Ok) y fail (no OK)

1.3 Código

```
void registerUser() {
    clearScreen(); // Limpia la pantalla antes de registrar un usuario
    cout << "\n=== Registro de Usuario ===\n";
    User newUser;
    cout << "Ingrese nombre de usuario: ";
    cin >> newUser.username;
    clearInputBuffer(); // Limpia el buffer después de la entrada
    cout << "Ingrese contrasena: ";
    newUser.password = getPassword();
    users.push_back(newUser);
    saveData();
    cout << "Usuario registrado con exito!\n";
#ifdef _WIN32
    system("pause"); // Pausa en Windows
#else
    cout << "Presione Enter para continuar...";
    cin.get(); // Pausa en Linux/macOS
#endif
}
```

1.4 Ejecución



2. REQOO2-INICIO DE SESION

2.1 Historia de Usuario

Campo	Descripcion
Número	REQ 002
Usuario	Administrador
Nombre de Historia	Inicio de Sesion
Prioridad en Negocio	Alta
Riesgo en Desarrollo	Medio
Iteración Asignada	No especificado
Programador Responsable	Mateo Arellano
Descripción: * Seleccionar inicio de sesión * Ingresar usuario y contraseña re gistrado * Mensaje de inicio de sesión con éxito o fallido.	
Validación	Debe dar mensaje de inicio de sesión con éxito

2.2 Particion de equivalencias

VARIABLE	**CLASE DE EQUIVALENCIA**	**ESTADO**	**REPRESENTANTE**
username	Ec1: user.username == username	Valido	usuario1
	Ec2: user.username ¡= username	No valido	Usuario1 (cliente no registrado)
password	Ec1: user.password == password	Valido	12345
	Ec2: user.password ¡= password	No valido	12345 (cliente no registrado)

2.3 Código

```
bool loginUser() {
    clearScreen(); // Limpia la pantalla antes de iniciar sesión
    string username, password;
    cout << "\n=== Inicio de sesion ===\n";
    cout << "Nombre de usuario: ";
    cin >> username;
    clearInputBuffer(); // Limpia el buffer después de la entrada
    cout << "Contrasena: ";
    password = getPassword();
    for (const auto& user : users) {
        if (user.username == username && user.password == password) {
            cout << "\nInicio de sesion exitoso!\n";
#ifdef _WIN32
            system("pause"); // Pausa en Windows
#else
            cout << "Presione Enter para continuar...";
            cin.get(); // Pausa en Linux/macOS
#endif
            return true;
        }
    }
    cout << "\nUsuario o contrasena incorrectos.\n";
#ifdef _WIN32
    system("pause"); // Pausa en Windows
#else
    cout << "Presione Enter para continuar...";
    cin.get(); // Pausa en Linux/macOS
#endif
    return false;
}
```

2.4 Ejecución

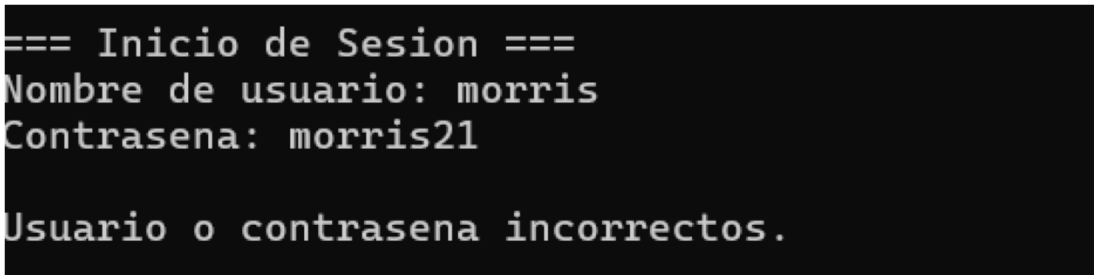
2.4.1 Caso Valido

```
=== Menu Principal ===
1. Registrar Usuario
2. Iniciar Sesion
3. Salir
Seleccione una opcion:
2

=== Inicio de Sesion ===
Nombre de usuario: morrs
Contrasena: morrs21

Inicio de sesion exitoso!
```

2.4.2 Caso Invalido



3. REQOO3-CREACION DE PROFORMAS

3.1 Historia de Usuario

Campo	Descripcion
Número	REQ 003
Usuario	Administrador
Nombre de Historia	Creación de proforma
Prioridad en Negocio	Alta
Riesgo en Desarrollo	Medio
Iteración Asignada	No especificado
Programador Responsable	Mateo Arellanos
Descripción: - Ingresar al sistema. - Seleccionar la opción "Crear Proforma". - Ingresar los datos del cliente y los materiales necesarios. - Validar que los datos sean correctos.	
Validación	La proforma debe generarse correctamente y almacenarse en el sistema.

3.2 Partición de Equivalencia

1. newProforma .clientName	Ec1: “newProforma.clientName”	Valido	Cliente: "Empresa X",
	Ec1: “newProforma.clientName” ¡= “newProforma.clientName”	No valido	Cliente: "Empresa X",

3.3 Código

```
void createProforma() {
    clearScreen(); // Limpia la pantalla antes de crear una proforma
    if (materials.empty()) {
        cout << "\nNo hay materiales disponibles para crear una proforma.\n";
#ifdef _WIN32
        system("pause"); // Pausa en Windows
#else
        cout << "Presione Enter para continuar...";
        cin.get(); // Pausa en Linux/macOS
#endif
        return;
    }

    Proforma newProforma;
    cout << "\n=== Creacion de Proforma ===\n";
    cout << "Ingrese el nombre del cliente: ";
    clearInputBuffer(); // Limpia el buffer antes de usar getline
    getline(cin, newProforma.clientName); // Lee el nombre del cliente

    // Asignar fecha actual
    time_t now = time(0);
    tm* localTime = localtime(&now);
    char buffer[80];
    strftime(buffer, 80, "%Y-%m-%d", localTime);
    newProforma.date = buffer;

    float totalCost = 0;
    int materialChoice;
    do {
        cout << "\nMateriales disponibles:\n";
        for (size_t i = 0; i < materials.size(); i++) {
            cout << i + 1 << ". " << materials[i].name << " - $" << materials[i].price << "\n";
        }
        cout << materials.size() + 1 << ". Terminar seleccion\n";
        materialChoice = getValidOption(1, materials.size() + 1);
        if (materialChoice > 0 && materialChoice <= materials.size()) {
            int quantity;
            cout << "Ingrese la cantidad: ";
            quantity = getValidOption(1, 1000, false); // No mostrar "Seleccione una opcion"
```

3.4 Ejecución

3.4.1 Caso Valido

```
Ingrese el nombre del cliente: serviglas

Materiales disponibles:
1. cal - $12
2. arena - $10
3. Terminar seleccion
Seleccione el numero del material: 1
Ingrese la cantidad: 4

Materiales disponibles:
1. cal - $12
2. arena - $10
3. Terminar seleccion
Seleccione el numero del material: 2
Ingrese la cantidad: 3

Materiales disponibles:
1. cal - $12
2. arena - $10
3. Terminar seleccion
Seleccione el numero del material: 3
Proforma creada con exito!
```

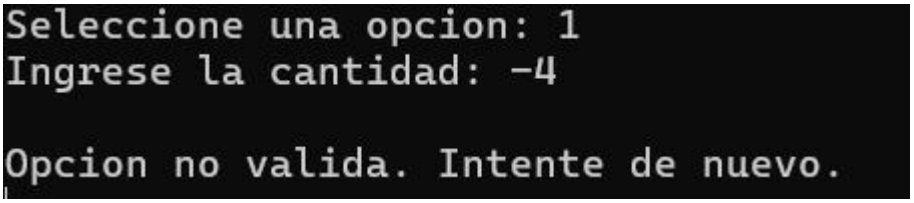
3.4.2 Caso Invalido

```
=== Ingreso de Material ===
Cuantos materiales desea ingresar? as

Opcion no valida. Intente de nuevo.
```

```
Opcion no valida. Intente de nuevo.
0

Opcion no valida. Intente de nuevo.
```



4. REQOO4-AGREGAR MATERIALES

4.1 Historia de Usuario

Campo	Descripcion
Número	REQ 004
Usuario	Administrador
Nombre de Historia	Ingreso de materiales
Prioridad en Negocio	Alta
Riesgo en Desarrollo	Medio
Iteración Asignada	No especificado
Programador Responsable	Mateo Arellanos
Descripción: - Ingresar al sistema. - Seleccionar la opción "Ingreso de materiales". - Ingresar los los materiales necesarios. - Validar que los datos sean correctos.	
Validación	Los materiales debe generarse correctamente y almacenarse en el sistema.

4.2 Partición de Equivalencia

VARIABLE	**CLASE DE EQUIVALENCIA**	**ESTADO**	**REPRESENTANTE**
2.newMaterial.name 2.1 materialChoicE 2.2 newMaterial. price	Ec2: “newMaterial.name”== “newMaterial.name” Ec2.1: “materialChoice “== ”materialChoice“ Ec2.2: “newMaterial.price”== “newMaterial.price”	Valido	Material: "Cemento"

	Ec1: “newProforma.clientName” ¡= “newProforma.clientName”	No valido	Material: ninguno
	Ec2: “newMaterial.name”!= “newMaterial.name” Ec2.1: materialChoice != materials.size() + 1 Ec 2.2: newMaterial.price!= “newMaterial.price”		

4.3 Código

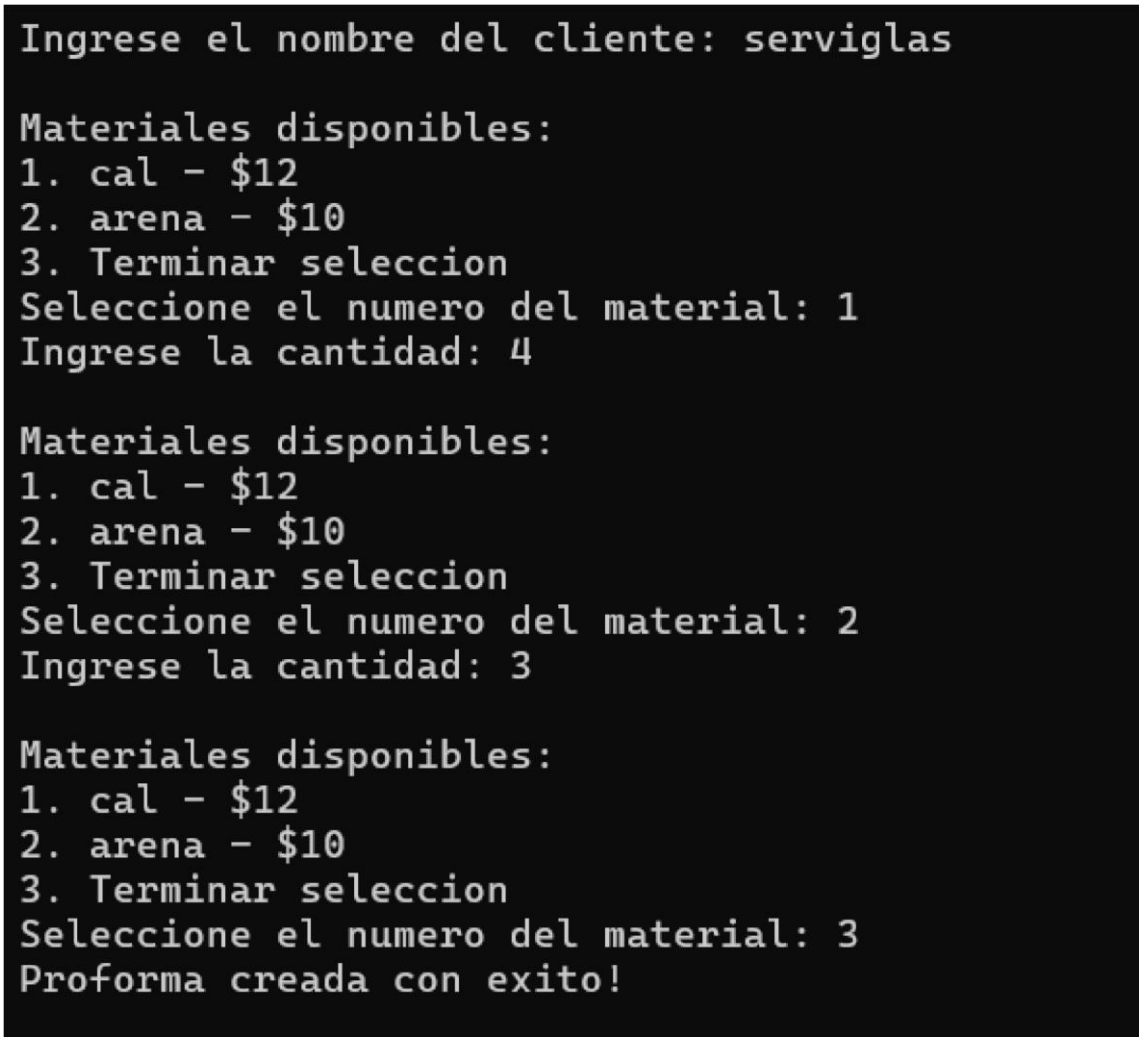
```
void addMaterial() {
    clearScreen(); // Limpia la pantalla antes de agregar materiales
    cout << "\n=== Ingreso de Material ===\n";
    int materialCount;
    do {
        cout << "Cuantos materiales desea ingresar? ";
        materialCount = getValidOption(1, 100, false); // No mostrar "Seleccione una opcion"
    } while (materialCount <= 0);

    for (int i = 0; i < materialCount; ++i) {
        Material newMaterial;
        cout << "Nombre del material: ";
        cin >> newMaterial.name;
        clearInputBuffer(); // Limpia el buffer después de la entrada
        do {
            cout << "Precio del material ($): ";
            cin >> newMaterial.price;
            if (cin.fail() || newMaterial.price <= 0) {
                cin.clear();
                cin.ignore(numeric_limits<streamsize>::max(), '\n');
                cout << "El precio debe ser un numero mayor que 0. Intente de nuevo.\n";
            }
        } while (cin.fail() || newMaterial.price <= 0);
        materials.push_back(newMaterial);
    }
    saveData();
    cout << "Materiales ingresados con exito!\n";
#ifdef _WIN32
    system("pause"); // Pausa en Windows
#else
    cout << "Presione Enter para continuar...";
    cin.get(); // Pausa en Linux/macOS
#endif
}
```

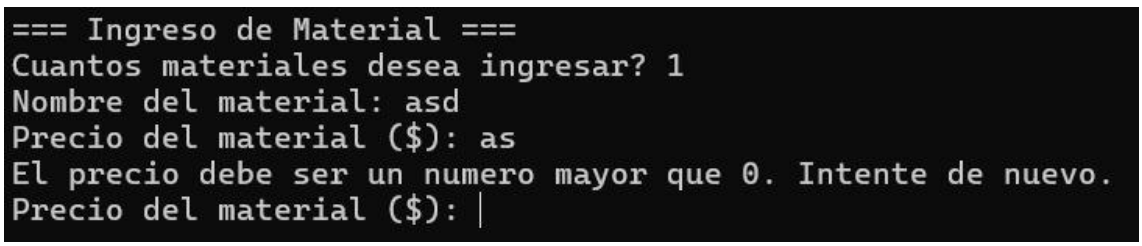
4.4 Ejecución

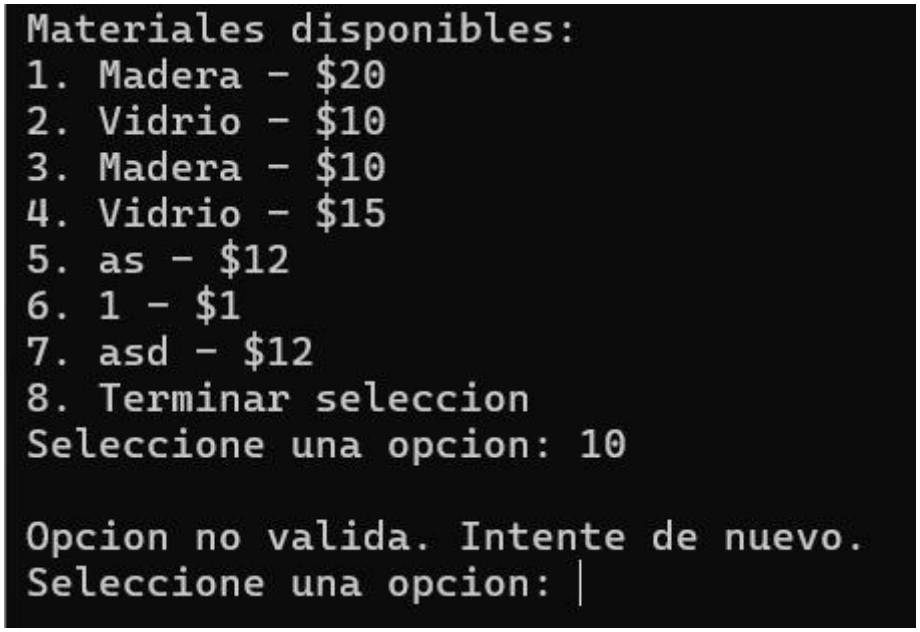
4.4.1 Caso Valido

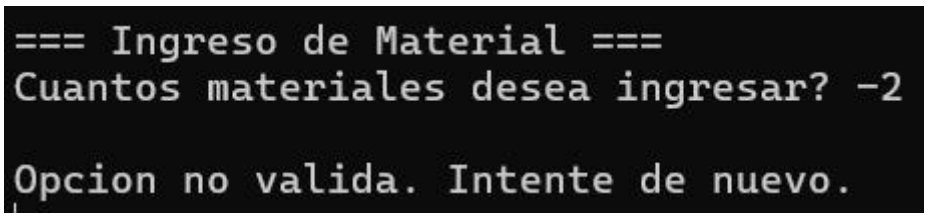
```
=== Ingreso de Material ===
Cuantos materiales desea ingresar? 2
Nombre del material: cal
Precio del material: 12
Nombre del material: arena
Precio del material: 10
Materiales ingresados con exito!
```



4.4.2 Caso Invalido




Materiales disponibles:
1. Madera - $20
2. Vidrio - $10
3. Madera - $10
4. Vidrio - $15
5. as - $12
6. 1 - $1
7. asd - $12
8. Terminar seleccion
Seleccione una opcion: 10
Opcion no valida. Intente de nuevo.
Seleccione una opcion:


=== Ingreso de Material ===
Cuantos materiales desea ingresar? -2
Opcion no valida. Intente de nuevo.

5. REQOO5-FINALIZACION PROFORMAS

5.1 Historia de Usuario

Campo	Historia de usuario
Número	REQ 005
Usuario	Administrador
Nombre de Historia	Eliminacion de proformas
Prioridad en Negocio	Alta
Riesgo en Desarrollo	Medio
Iteración Asignada	No especificado
Programador Responsable	Mateo Arellano
Descripción: • Ingresar opción eliminar p roformas • Seleccionar la proforma a eliminar. • Confirmar la eliminación. • El sistema elimina la profo rma y actualiza la base de datos.	
Validación	La proforma debe ser eliminada correctamente

5.2 Participación de Equivalencias

VARIABLE	CLASE DE EQUIVALENCIA	ESTADO	REPRESENTANTE
deleteProforma	EC1: “deleteProformasList"== “deleteProformasList"	Valido	Confirmar eliminacion
	Ec2: “deleteProformasList"!=“deleteProformasList"	valido	Cancelar eliminacion

5.3 Código

```
void deleteProforma() {
    clearScreen(); // Limpia la pantalla antes de eliminar proformas
    if (proformas.empty()) {
        cout << "\nNo hay proformas disponibles para eliminar.\n";
#ifdef _WIN32
        system("pause"); // Pausa en Windows
#else
        cout << "Presione Enter para continuar...";
        cin.get(); // Pausa en Linux/macOS
#endif
        return;
    }

    cout << "\n=== Eliminar Proforma ===\n";
    viewProformas(); // Mostrar las proformas disponibles

    int proformaChoice;
    do {
        cout << "Seleccione el numero de la proforma que desea eliminar (1-" << proformas.size() << "): ";
        proformaChoice = getValidOption(1, proformas.size(), false); // No mostrar "Seleccione una opcion"
    } while (proformaChoice < 1 || proformaChoice > proformas.size());

    proformas.erase(proformas.begin() + proformaChoice - 1);
    saveData();
    cout << "Proforma eliminada con exito!\n";
#ifdef _WIN32
    system("pause"); // Pausa en Windows
#else
    cout << "Presione Enter para continuar...";
    cin.get(); // Pausa en Linux/macOS
#endif
}
```

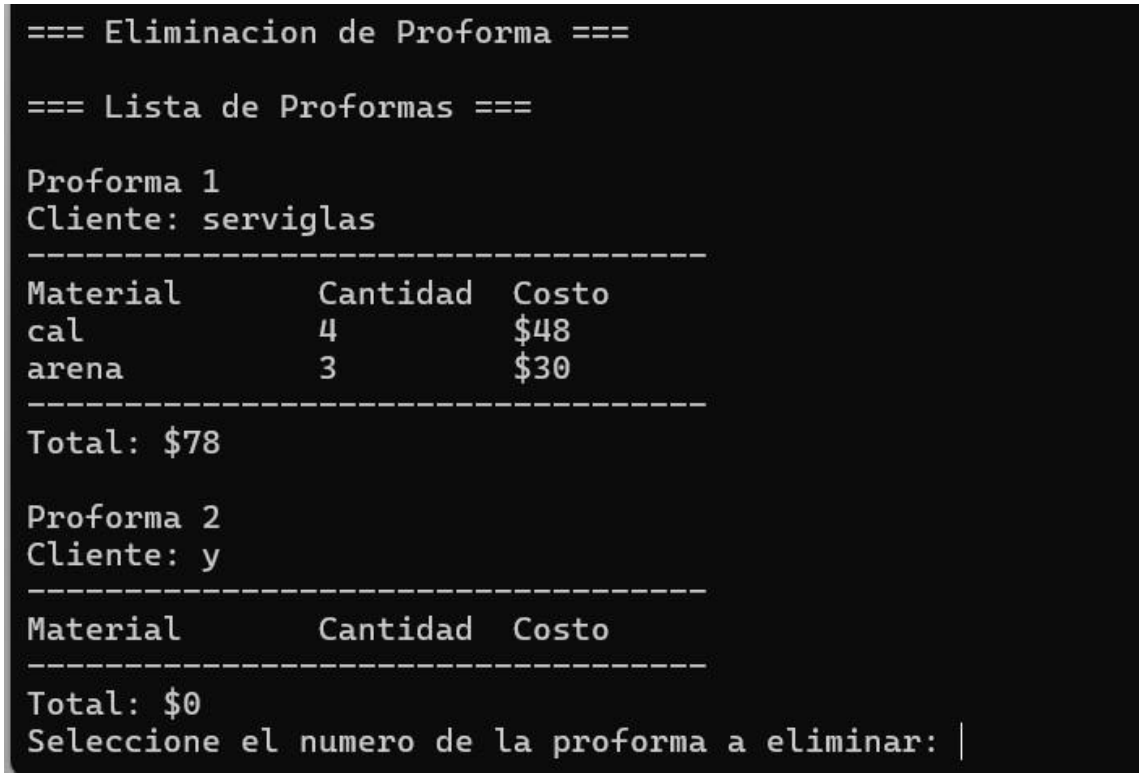
5.4 Ejecución

5.4.1 Caso Valido

```
-----------------------------------
Material        Cantidad  Costo
cal             4         $48
arena           3         $30
-----------------------------------
Total: $78

Proforma 2
Cliente: x
-----------------------------------
Material        Cantidad  Costo
-----------------------------------
Total: $0

Proforma 3
Cliente: y
-----------------------------------
Material        Cantidad  Costo
-----------------------------------
Total: $0
Seleccione el numero de la proforma a eliminar: 2
Proforma eliminada con exito!
```



5.4.2 Caso Invalido