

# Prueba de Caja Negra

---

*Sistema de gestión de alícuotas para el condominio “La Primavera”*

## **Integrantes:**

Góngora Lucas,  
Manosalvas Gabriel,  
Molina Jairo,  
Vélez Yandry

## **Fecha**

2025-02-19

Sistema de gestión de alícuotas para el condominio “La Primavera”	Caja Negra	Versión: 4
		Página: 2
		Fecha: 18/02/2025

## ÍNDICE

<b>1. Historia de revisión</b>	<b>3</b>
<b>2. Desarrollo</b>	<b>4</b>
2.1. REQ 001 - Inicio de sesión seguro	4
2.1.1. Historia de usuario	4
2.1.2. Partición de clases equivalentes.	4
2.1.3. Código	5
2.1.4. Ejecución	5
2.2. REQ 002 - Registro de alícuota	6
2.2.1. Historia de usuario	6
2.2.2. Partición de clases equivalentes.	6
2.2.3. Código	10
2.2.4. Ejecución	11
2.3. REQ 003 - Listar alícuotas	13
2.3.1. Historia de usuario	13
2.3.2. Partición de clases equivalentes.	13
2.3.3. Código	14
2.3.4. Ejecución	14
2.4. REQ 004 - Actualizar alícuotas	15
2.4.1. Historia de usuario	15
2.4.2. Partición de clases equivalentes.	16
2.4.3. Código	16
2.4.4. Ejecución	17
2.5. REQ 005 - Eliminar alícuotas	18
2.5.1. Historia de usuario	18
2.5.2. Partición de clases equivalentes.	18
2.5.3. Código	19
2.5.4. Ejecución	20
2.6. REQ 006 - Filtrar alícuotas pendientes	20
2.6.1. Historia de usuario	20
2.6.2. Partición de clases equivalentes.	21
2.6.3. Código	22
2.6.4. Ejecución	22

Sistema de gestión de alícuotas para el condominio "La Primavera"	Caja Negra	Versión: 4
		Página: 3
		Fecha: 18/02/2025

## 1. Historia de revisión

Fecha	Versión	Descripción	Autores
09/01/2025	1	Creación del documento de cajas negras	Góngora Lucas, Manosalvas Gabriel, Molina Jairo, Vélez Yandry
16/01/2025	2	Modificación de validaciones de cajas negras	Góngora Lucas, Manosalvas Gabriel, Molina Jairo, Vélez Yandry
06/02/2025	3	Agregación de descripción de cada requisito.	Góngora Lucas, Manosalvas Gabriel, Molina Jairo, Vélez Yandry
19/02/2025	4	Modificación de validaciones de cajas negras.	Góngora Lucas, Manosalvas Gabriel, Molina Jairo, Vélez Yandry

Sistema de gestión de alícuotas para el condominio "La Primavera"	Caja Negra	Versión: 4
		Página: 4
		Fecha: 18/02/2025

## 2. Desarrollo

### 2.1. REQ 001 - Inicio de sesión seguro

#### 2.1.1. Historia de usuario

Historia de usuario	
Número: REQ 001	Usuario: Administrador
Nombre de la Historia: Iniciar sesión de forma segura	
Prioridad: Alta	
Programador responsable: Lucas Góngora	
Descripción: Se inicia sesión ingresando el nombre del usuario y la clave.	
Validación: Se valida el nombre de usuario y la clave y se indica en consola un mensaje de confirmación.	

#### 2.1.2. Partición de clases equivalentes.

Sistema de gestión de alcuotas para el condominio "La Primavera"	Caja Negra	Versión: 4
		Página: 5
		Fecha: 18/02/2025

VARIABLE	CLASE DE EQUIVALENCIA	ESTADO	REPRESENTANTE
usuario	EC1: usuarioU == usuario	válido	Lucas
	EC2: usuarioU != usuario	no válido	Lucas123
	EC3: usuarioU != usuario	no válido	-----
	EC4: usuarioU != usuario	no válido	Lucas\$#"#
clave	EC1: claveU == clave	válido	lP2^\$sads
	EC2: claveU != clave	no válido	-----
	EC3: claveU != clave	no válido	lucas
	EC4: claveP != clave	no válido	lucas123

### 2.1.3. Código

```

validacion_username = r'^[a-zA-Z]+$'
validacion_clave = r'^(?=.*[!@#$%^&*() ,.?":{}|<>])(?=[8,})'

def obtener_credenciales():
    usuario = validar_entrada("username", validacion_username, "Ingresa solo letras")
    clave = validar_entrada("clave", validacion_clave, "Ingresa al menos 8 caracteres y al menos un caracter especial")
    return [usuario, clave]

def validar_entrada(tipo, patron, mensaje_error):
    while True:
        if tipo == "clave":
            entrada = input_password("Ingresa la clave: ")
        elif tipo == "username":
            entrada = input("Ingresa el nombre de usuario: ")
        else:
            print("Ingresa un tipo de entrada válido")
            break
        if re.match(patron, entrada):
            return entrada
        print(mensaje_error)

```

Fig 1. Código de python para la validación de nombre de usuario y clave.

Sistema de gestión de alícuotas para el condominio "La Primavera"	Caja Negra	Versión: 4
		Página: 6
		Fecha: 18/02/2025

2.1.4. Ejecución

2.1.4.1 Pass (OK)

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

      ----LOGIN----
Ingresar el nombre de usuario: Lucas
Ingresar la clave: lp2^$sads

Credenciales válidas
Presiona ENTER para pasar al menú ...
```

2.1.4.2. No Pass (No OK)

```
      ----LOGIN----
Ingresar el nombre de usuario: Lucas123
Ingresar solo letras
Ingresar el nombre de usuario:

Ingresar el nombre de usuario: Lucas
Ingresar la clave: Lucas
Ingresar al menos 8 caracteres y al menos un caracter especial
Ingresar la clave: Lucas123
Ingresar al menos 8 caracteres y al menos un caracter especial
Ingresar la clave:
Ingresar al menos 8 caracteres y al menos un caracter especial
Ingresar la clave:
```

2.2. REQ 002 - Registro de alícuota

2.2.1. Historia de usuario

Historia de usuario	
Número: REQ 002	Usuario: Administrador
Nombre de la Historia: Registrar alícuota	

Sistema de gestión de alcuotas para el condominio "La Primavera"	Caja Negra	Versión: 4
		Página: 7
		Fecha: 18/02/2025

<b>Prioridad:</b> Alta
<b>Programador responsable:</b> Gabriel Manosalvas
<b>Descripción:</b> Se ingresa la opción de registrar alcuota, seguido de los datos que se necesita ingresar para registrar alcuota.
<b>Validación:</b> Se registra la alcuota y se indica en consola un mensaje de confirmación.

### 2.2.2. Partición de clases equivalentes.

VARIABLE	CLASE DE EQUIVALENCIA	ESTADO	REPRESENTANTE
id	EC1: idU == id	válido	1
	EC2: idU != id	no válido	-----
	EC3: idU != id	no válido	id existente
residente	EC4: residenteU == residente	válido	Lucas
	EC5: residenteU != residente	no válido	Lucas123
	EC6: residenteU != residente	no válido	-----
	EC7: residenteU != residente	no válido	Lucas\$#"#
descripcion	EC8: descripcionU == descripcion	válido	Mantenimiento piscina.
	EC9: descripcionU != descripcion	no válido	-----

Sistema de gestión de alícuotas para el condominio "La Primavera"	Caja Negra	Versión: 4
		Página: 8
		Fecha: 18/02/2025

estado_pago	EC10: estado_pagoU == estado_pago	válido	pendiente
	EC11: estado_pagoU == estado_pago	válido	pagado
	EC12: estado_pagoU != estado_pago	no válido	-----
	EC13: estado_pagoU != estado_pago	no válido	debe pagar
	EC14: estado_pagoU != estado_pago	no válido	está pagado
porcentaje_alicuota	EC15: porcentaje_alicuota U == porcentaje_alicuota	válido	0.12
	EC16: porcentaje_alicuota U != porcentaje_alicuota	no válido	0,12
	EC17: porcentaje_alicuota U != porcentaje_alicuota	no válido	-----
	EC18: porcentaje_alicuota U != porcentaje_alicuota	no válido	2.0
	EC19: porcentaje_alicuota U != porcentaje_alicuota	no válido	dos



Sistema de gestión de alícuotas para el condominio "La Primavera"	Caja Negra	Versión: 4
		Página: 9
		Fecha: 18/02/2025

base_imponible	EC20: porcentaje_alicuota U == porcentaje_alicuota	válido	300
	EC21: porcentaje_alicuota U != porcentaje_alicuota	no válido	trescientos
	EC22: porcentaje_alicuota U != porcentaje_alicuota	no válido	-----
base_imponible	EC23: base_imponibleU == base_imponible	válido	300
	EC24: base_imponibleU != base_imponible	no válido	trescientos
	EC25: base_imponibleU != base_imponible	no válido	-----
multa	EC26: multaU == multa	válido	300
	EC27: multaU != multa	no válido	trescientos
	EC28: multaU != multa	no válido	-----
descuento	EC29: descuentoU == descuento	válido	300
	EC30: descuentoU != descuento	no válido	trescientos
	EC31: descuentoU != descuento	no válido	-----
ajuste_extraordinario	EC32: ajuste_extraordinarioU ==	válido	300

Sistema de gestión de alcuotas para el condominio "La Primavera"	Caja Negra	Versión: 4
		Página: 10
		Fecha: 18/02/2025

	ajuste_extraordinario		
	EC33: ajuste_extraordinarioU != ajuste_extraordinario	no válido	trescientos
	EC34: ajuste_extraordinarioU != ajuste_extraordinario	no válido	-----

### 2.2.3. Código

```
def solicitar_datos_alicuota(con_id = True):
    if con_id:
        id = input("ID de la alicuota: ")

    residente = input("Nombre del residente: ")

    validacion_estado_pago = r"^(pendiente|pagado)$"
    mensaje_error_estado_pago = "El estado de pago debe ser 'pendiente' o 'pagado'."
    estado_pago = validar_pregunta("Estado de pago (pendiente/pagado): ", validacion_estado_pago, mensaje_error_estado_pago)

    validacion_porcentaje = r"^(0(\.\d+)?|1(\.\d*)?)$"
    mensaje_error_porcentaje = "debe ser un valor decimal entre 0 y 1"
    porcentaje_alicuota = validar_pregunta("Porcentaje de alicuota: ", validacion_porcentaje, mensaje_error_porcentaje)

    base_imponible = validar_numero_decimal("Base imponible: ")

    descripcion = input("Descripción: ")

    multa = validar_numero_decimal("Monto de la multa: ")

    descuento = validar_numero_decimal("Monto de descuento: ")

    ajuste_extraordinario = validar_numero_decimal("Ajuste extraordinario: ")
    if con_id:
        return id, residente, estado_pago, porcentaje_alicuota, base_imponible, descripcion, multa, descuento, ajuste_extraordinario
    else:
        return residente, estado_pago, porcentaje_alicuota, base_imponible, descripcion, multa, descuento, ajuste_extraordinario

def validar_numero_decimal(pregunta):
    validacion_numero_decimal = r"^(?=[.\d])(?=[.\d])?.+?"
    mensaje_error_numero_decimal = "Por favor, ingresa un numero válido."

    valor = float(validar_pregunta(pregunta, validacion_numero_decimal, mensaje_error_numero_decimal))

    return valor
```

Sistema de gestión de alícuotas para el condominio "La Primavera"	Caja Negra	Versión: 4
		Página: 11
		Fecha: 18/02/2025

```
def validar_texto(pregunta):
    validacion_texto = r"^[a-zA-Z]+$"
    mensaje_error = "Debes ingresar solo letras"
    validar_pregunta(pregunta,validacion_texto,mensaje_error)

def validar_pregunta(pregunta,validacion,mensaje_error):
    while True:
        valor = input(pregunta)
        if re.match(validacion,valor):
            return valor
        print(mensaje_error)
        input("\n Aplasta Enter para volver a rellenar el campo de forma correcta`\n")

def ingresar_validar_id ():
    validacion_id = r"^[a-zA-Z0-9]+$"
    while True:
        id = validar_pregunta("ID de la alícuota: ",validacion_id, "Ingresa numeros o letras")
        if not validar_id(id):
            return id
        print("el id ya existe")
        input("\n Aplasta Enter para volver a rellenar el campo de forma correcta\n")
```

Fig 2. Código de python para la validación de datos de alícuota.

#### 2.2.4. Ejecución

##### 2.2.4.1 Pass (OK)

Sistema de gestión de alícuotas para el condominio "La Primavera"	Caja Negra	Versión: 4
		Página: 12
		Fecha: 18/02/2025

```

-----MENU-----
1. Registrar alicuota
2. Mostrar alicuotas
3. Buscar alicuota
4. Actualizar alicuota
5. Mostrar alicuotas pendientes por pagar
6. Borrar alicuota
7. Salir del programa
Ingresa tu opcion: 1
Registrar una alícuota:
ID de la alícuota: 1
Nombre del residente: Lucas
Estado de pago (pendiente/pagado): pendiente
Porcentaje de alícuota: 0.12
Base imponible: 300
Descripción: Pagos de mantenimiento de piscina
Monto de la multa: 34
Monto de descuento: 25
Ajuste extraordinario: 0
Alicuota registrada con exito en './databases/alicuota.csv'
Alícuota registrada correctamente.
Presiona ENTER PARA VOLVER AL MENU PRINCIPAL

```

#### 2.2.4.2. No Pass (No OK)

```

-----MENU-----
1. Registrar alicuota
2. Mostrar alicuotas
3. Buscar alicuota
4. Actualizar alicuota
5. Mostrar alicuotas pendientes por pagar
6. Borrar alicuota
7. Salir del programa
Ingresa tu opcion: 1
Registrar una alícuota:
ID de la alícuota: 1
el id ya existe

```

Sistema de gestión de alícuotas para el condominio "La Primavera"	Caja Negra	Versión: 4
		Página: 13
		Fecha: 18/02/2025

Registrar una alícuota:  
ID de la alícuota: 2  
Nombre del residente: 123  
Debes ingresar solo letras

Aplasta Enter para volver a rellenar el campo de forma correcta`

Nombre del residente: Lucas  
Estado de pago (pendiente/pagado): debe pagar  
El estado de pago debe ser 'pendiente' o 'pagado'.

Aplasta Enter para volver a rellenar el campo de forma correcta`

Estado de pago (pendiente/pagado): pagado  
Porcentaje de alícuota: 2.0  
debe ser un valor decimal entre 0 y 1

Aplasta Enter para volver a rellenar el campo de forma correcta`

Porcentaje de alícuota: 0.12  
Base imponible: trescientos  
Por favor, ingresa un numero válido.

Aplasta Enter para volver a rellenar el campo de forma correcta`

Base imponible: 300  
Descripción: 123  
Debes ingresar solo letras

Aplasta Enter para volver a rellenar el campo de forma correcta`

Descripción: Mantenimiento  
Monto de la multa: veinte  
Por favor, ingresa un numero válido.

Aplasta Enter para volver a rellenar el campo de forma correcta`

Monto de la multa: 20  
Monto de descuento:  
Por favor, ingresa un numero válido.

Aplasta Enter para volver a rellenar el campo de forma correcta`

Sistema de gestión de alícuotas para el condominio "La Primavera"	Caja Negra	Versión: 4
		Página: 14
		Fecha: 18/02/2025

## 2.3. REQ 003 - Listar alícuotas

### 2.3.1. Historia de usuario

Historia de usuario	
<b>Número:</b> REQ 003	<b>Usuario:</b> Administrador
<b>Nombre de la Historia:</b> Listar alícuotas	
<b>Prioridad:</b> Alta	
<b>Programador responsable:</b> Yandry Velez	
<b>Descripción:</b> Crear función para leer CSV y mostrar en tabla	
<b>Validación:</b> Se valida el nombre de usuario y la clave y se indica en consola un mensaje de confirmación.	

### 2.3.2. Partición de clases equivalentes.

VARIABLE	CLASE DE EQUIVALENCIA	ESTADO	REPRESENTANTE
ID	EC1: idU == id	válido	1
	EC2: idU != id	no válido	-----
	EC3: idU != id	no válido	id existente
multa	EC1: multaU == multa	válido	300
	EC2: multaU != multa	no válido	trescientos
	EC3: multaU != multa	no válido	-----



Sistema de gestión de alícuotas para el condominio "La Primavera"	Caja Negra	Versión: 4
		Página: 16
		Fecha: 18/02/2025

```

-----MENU-----
1. Registrar alicuota
2. Mostrar alicuotas
3. Buscar alicuota
4. Actualizar alicuota
5. Mostrar alicuotas pendientes por pagar
6. Borrar alicuota
7. Salir del programa
Ingresa tu opcion: 8
Ingresa una opcion del 1 al 7
Presiona ENTER para volver al menu

```

## 2.4. REQ 004 - Actualizar alícuotas

### 2.4.1. Historia de usuario

Historia de usuario	
Número: REQ 004	Usuario: Administrador
Nombre de la Historia: Actualizar alícuotas	
Prioridad: Alta	
Programador responsable: Gabriel Manosalvas	
Descripción: Poder modificar la alícuota.	
Validación: Crear función para buscar por ID, editar y guardar cambios.	

### 2.4.2. Partición de clases equivalentes.

VARIABLE	CLASE DE EQUIVALENCIA	ESTADO	REPRESENTANTE
id	EC1: idU == id	válido	1
	EC2: idU != id	no válido	-----



Sistema de gestión de alícuotas para el condominio "La Primavera"	Caja Negra	Versión: 4
		Página: 17
		Fecha: 18/02/2025

	EC2: idU != id	no válido	id existente
estado_pago	EC1: estado_pagoU == estado_pago	válido	pendiente
	EC2: estado_pagoU == estado_pago	válido	pagado
	EC3: estado_pagoU != estado_pago	no válido	-----
	EC4: estado_pagoU != estado_pago	no válido	debe pagar
	EC5: estado_pagoU != estado_pago	no válido	está pagado

### 2.4.3. Código

```
def opt_actualizar_alicuota():

    print("Actualizar una alícuota:")

    id = input("ID de la alícuota: ")

    alicuota_actual = buscar_por_id("./databases/alicuota.csv", id)

    if alicuota_actual:
        print("Datos actuales de la alícuota:")
        print(alicuota_actual)
        con_id = False
        datos = solicitar_datos_alicuota(con_id)
        if datos:
            actualizar_alicuota(id,*datos)
            print("Alícuota actualizada correctamente.")
    else:
        print("Alícuota no encontrada.")
    input("Presiona ENTER PARA VOLVER AL MENU PRINCIPAL")
    limpiar_pantalla()
```

Sistema de gestión de alícuotas para el condominio "La Primavera"	Caja Negra	Versión: 4
		Página: 18
		Fecha: 18/02/2025

Fig 4. Código de python.

#### 2.4.4. Ejecución

##### 2.4.4.1 Pass (OK)

```

-----MENU-----
1. Registrar alicuota
2. Mostrar alicuotas
3. Buscar alicuota
4. Actualizar alicuota
5. Mostrar alicuotas pendientes por pagar
6. Borrar alicuota
7. Salir del programa
Ingresa tu opcion: 4
Actualizar una alícuota:
ID de la alícuota: 2
Datos actuales de la alícuota:
['2', 'Pedro', 'pagado', '0.12', '300.0', '', '2025-02-19', '20.0', '10.0', '0.0', '46.0']
Nombre del residente: Lucas
Estado de pago (pendiente/pagado): pagado
Porcentaje de alícuota: 0.15
Base imponible: 300

```

##### 2.4.4.2. No Pass (No OK)

```

-----MENU-----
1. Registrar alicuota
2. Mostrar alicuotas
3. Buscar alicuota
4. Actualizar alicuota
5. Mostrar alicuotas pendientes por pagar
6. Borrar alicuota
7. Salir del programa
Ingresa tu opcion: 4
Actualizar una alícuota:
ID de la alícuota: 2
Datos actuales de la alícuota:
['2', 'Pedro', 'pagado', '0.12', '300.0', '', '2025-02-19', '20.0', '10.0', '0.0', '46.0']
Nombre del residente: Lucas
Estado de pago (pendiente/pagado): pagado
Porcentaje de alícuota: 0.15
Base imponible: 300
Descripción: "
Debes ingresar solo letras

Aplasta Enter para volver a rellenar el campo de forma correcta

```

#### 2.5. REQ 005 - Eliminar alícuotas

##### 2.5.1. Historia de usuario

Sistema de gestión de alícuotas para el condominio "La Primavera"	Caja Negra	Versión: 4
		Página: 19
		Fecha: 18/02/2025

Historia de usuario	
<b>Número:</b> REQ 005	<b>Usuario:</b> Administrador
<b>Nombre de la Historia:</b> Eliminar alícuotas	
<b>Prioridad:</b> Alta	
<b>Programador responsable:</b> Jairo Molina	
<b>Descripción:</b> Se permite eliminar alícuotas registradas	
<b>Validación:</b> Se elimina la alícuota y verificar que no aparezca en la base de datos	

### 2.5.2. Partición de clases equivalentes.

VARIABLE	CLASE DE EQUIVALENCIA	ESTADO	REPRESENTANTE
id	EC1: idU == id	válido	1
	EC2: idU != id	no válido	-----
	EC2: idU != id	no válido	id existente
	EC1: idU == id	válido	1
clave	EC1: claveU == clave	válido	lp2^\$sads
	EC2: claveU != clave	no válido	-----
	EC3: claveU != clave	no válido	lucas
	EC4: claveP != clave	no válido	lucas123

Sistema de gestión de alícuotas para el condominio "La Primavera"	Caja Negra	Versión: 4
		Página: 20
		Fecha: 18/02/2025

### 2.5.3. Código

```
def eliminar_alicuota(id):
    eliminar(database, id)

def eliminar(path_archivo_csv, criterio_de_búsqueda):
    if not verificar_existencia_archivo(path_archivo_csv):
        print(f"El archivo '{path_archivo_csv}' no existe.")
        return

    with open(path_archivo_csv, mode='r', newline='') as file:
        rows = list(csv.reader(file))

    nuevas_filas = [fila for fila in rows if criterio_de_búsqueda not in
fila]

    if len(nuevas_filas) != len(rows):
        with open(path_archivo_csv, mode='w', newline='') as file:
            writer = csv.writer(file)
            writer.writerows(nuevas_filas)
        print(f"Registro con criterio '{criterio_de_búsqueda}'
eliminado.")
    else:
        print(f"No se encontraron registros para eliminar con el criterio
'{criterio_de_búsqueda}'.")
```

Fig 5. Código de python.

### 2.5.4. Ejecución

#### 2.3.4.1 Pass (OK)

```
5. Mostrar alícuotas pendientes por pagar
6. Borrar alícuota
7. Salir del programa
Ingresa tu opción: 6
Eliminar una alícuota por ID:
ID de la alícuota: 2
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
ID | Residente | Estado de Pago | Porcentaje Alícuota | Base Imponible | Descripción | Fecha de Registro | Multa | Descuento | Adicional | Total
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
2  | Pedro     | pagado         | 0.12                | 300.0          |              | 2025-02-19       | 20.0 | 10.0      | 0.0       | 46.0
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
Estas seguro de eliminar esta alícuota (sí/no):
```

Sistema de gestión de alícuotas para el condominio "La Primavera"	Caja Negra	Versión: 4
		Página: 21
		Fecha: 18/02/2025

#### 2.5.4.2. No Pass (No OK)

```

PROBLEMAS  SALIDA  CONSOLA DE DEPURACIÓN  TERMINAL  PUERTOS  COMENTARIOS
1. Registrar alicuota
2. Mostrar alicuotas
3. Buscar alicuota
4. Actualizar alicuota
5. Mostrar alicuotas pendientes por pagar
6. Borrar alicuota
7. Salir del programa
Ingresar tu opción: 6
Eliminar una alicuota por ID:
ID de la alicuota: 9
No se encontraron resultados para el criterio '9'.
  
```

## 2.6. REQ 006 - Filtrar alícuotas pendientes

### 2.6.1. Historia de usuario

Historia de usuario	
<b>Número:</b> REQ 006	<b>Usuario:</b> Administrador
<b>Nombre de la Historia:</b> Filtrar alícuotas pendientes	
<b>Prioridad:</b> Alta	
<b>Programador responsable:</b> Yandry Velez	
<b>Descripción:</b> Poder visualizar los residentes que faltan por pagar	
<b>Validación:</b> Crear función para filtrar por estado "pendiente" y mostrar lista	

### 2.6.2. Partición de clases equivalentes.

VARIABLE	CLASE DE EQUIVALENCIA	ESTADO	REPRESENTANTE
----------	-----------------------	--------	---------------

Sistema de gestión de alícuotas para el condominio "La Primavera"	Caja Negra	Versión: 4
		Página: 22
		Fecha: 18/02/2025

residente	EC4: residenteU == residente	válido	Lucas
	EC5: residenteU !=residente	no válido	Lucas123
	EC6: residenteU !=residente	no válido	-----
	EC7: residenteU != residente	no válido	Lucas\$#"#
base_imponible	EC20: porcentaje_alicuota U == porcentaje_alicuota	válido	300
	EC21: porcentaje_alicuota U != porcentaje_alicuota	no válido	trescientos
	EC22: porcentaje_alicuota U != porcentaje_alicuota	no válido	-----
	EC20: porcentaje_alicuota U == porcentaje_alicuota	válido	300

### 2.6.3. Código

```
def buscar_alicuotas_pendientes():
    alicuotas = buscar(database, "pendiente")
    imprimir_alicuotas(alicuotas)
```

Fig 6. Código de python.

Sistema de gestión de alícuotas para el condominio “La Primavera”	Caja Negra	Versión: 4
		Página: 23
		Fecha: 18/02/2025

### 2.6.4. Ejecución

#### 2.6.4.1 Pass (OK)

```
-----MENU-----
1. Registrar alicuota
2. Mostrar alicuotas
3. Buscar alicuota
4. Actualizar alicuota
5. Mostrar alicuotas pendientes por pagar
6. Borrar alicuota
7. Salir del programa
Ingresa tu opcion: 5
Mostrando alicuotas pendientes por pagar:
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
ID | Residente | Estado de Pago | Porcentaje Alicuota | Base Imponible | Descripción | Fecha de Registro | Multa | Descuento | Adicional | Total
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1 | Lucas | pendiente | 0.12 | 300.0 | Pagos de mantenimiento de piscina | 2025-02-19 | 34.0 | 25.0 | 0.0 | 45.0
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
Presiona ENTER PARA VOLVER AL MENU PRINCIPAL
```

#### 2.6.4.2. No Pass (No OK)

```
-----MENU-----
1. Registrar alicuota
2. Mostrar alicuotas
3. Buscar alicuota
4. Actualizar alicuota
5. Mostrar alicuotas pendientes por pagar
6. Borrar alicuota
7. Salir del programa
Ingresa tu opcion: 9
Ingresa una opcion del 1 al 7
Presiona ENTER para volver al menu
```