INFORME DE LABORATORIO

(formato estudiante)

INFORMACIÓN BÁSICA					
ASIGNATURA:	Programación WEB 2				
TÍTULO DE LA PRÁCTICA:	Python – Laboratorio 04				
NÚMERO DE PRÁCTICA:	04	**AÑO LECTIVO:**	2024	**NRO. SEMESTRE:**	II
FECHA DE PRESENTACIÓN	11/10/2024	**HORA DE PRESENTACIÓN**	23:59 PM		
INTEGRANTE (s) Christian Henry Casso Quispe				**NOTA (0-20)**	Nota colocada por el docente
DOCENTE(s): Lino Pinto Oppe					

RESULTADOS Y PRUEBAS

I. EJERCICIOS RESUELTOS:

Una empresa de transporte interprovincial desea registrar las ventas de pasajes, con las siguientes opciones:

El usuario podría escoger la primera opción, para el ingreso de datos, o la segunda opción, para ver los resultados de las ventas. Si desea continuar con otro registro de ventas puede volver a escoger la opción 1. Si no desea continuar el registro de ventas escoger la opción 2. Salir.

Opción 1:

a) En esta opción el usuario debe ingresar los siguientes datos, además debe realizar la validación de dato, con Do While.

- El ingreso del tipo de cliente, como número, de tipo int (1 o 2)
- La cantidad de pasajes, como número, de tipo int
- El ingreso del género del cliente, como letra, tipo char (M o m o F o f)
- El ingreso del tipo de servicio, como número, de tipo int (1-Economica / 2-Ejecutiva / 3-

Primera clase) Según el tipo de servicio, el precio es el siguiente:

Tipo de Pasaje	Precio Dolares($.)
1. Económica	70.00
2. Ejecutiva	140.00
3. Primera clase	280.00

Se aplica un porcentaje de descuento en función a la cantidad de pasajes.

Cantidad de Pasajes	Porcentaje de descuento
1	0%
2-5	5%
6-10	12%
11 a más	15%

b) Mostrar por cliente, lo siguiente:

- Importe Bruto = cantidad pasajes * precio.
- Monto de Descuento = Importe a Pagar * Porcentaje de Descuento
- Importe Neto = Importe Bruto – Monto de Descuento

Opción 2:

En esta opción se debe mostrar el resultado siguiente, pero debe tener en cuenta que el monto total de ventas es la suma de todos los importes a pagar de cada cliente:

- Cantidad de clientes de género masculino.
- Cantidad de ventas cuyo Importe Neto sea >=70 y <=500
- Cantidad de ventas de clientes de género femenino cuyo Importe Neto sea >=140 y <=1000
- El acumulado del Importe de Ventas.
- El acumulado del Importe Neto de clientes, de tipo 1.
- Promedio de Importe Neto, de clientes, de tipo 1.

Este código implementa un sistema de ventas para una empresa de transportes interprovincial, donde se pueden registrar ventas de boletos, aplicar descuentos, y generar un reporte de ventas. Está organizado en varias funciones que realizan validaciones y cálculos, y un bucle principal que muestra el menú y permite al usuario interactuar con el programa.

1. Funciones para validaciones

a) obtener_entero(mensaje, min_val=None, max_val=None)

Esta función solicita al usuario que ingrese un número entero, y realiza las siguientes validaciones:

- Asegura que el valor ingresado sea un número entero.
- Si se proporcionan valores mínimos y/o máximos (min_val y max_val), también valida que el número ingresado esté dentro de ese rango.
- Si el valor ingresado no es un número entero válido o está fuera de los límites, se muestra un mensaje de error y se vuelve a solicitar el ingreso.

```
# Funciones para validaciones
def obtener_entero(mensaje, min_val=None, max_val=None):
    while True:
```

```
        try:
            valor = int(input(mensaje))
            if (min_val is not None and valor < min_val) or (max_val
is not None and valor > max_val):
                print(f"Error: El valor debe estar entre {min_val} y
{max_val}.")
            else:
                return valor
        except ValueError:
            print("Error: Debe ingresar un número entero válido.")
```

b) obtener_genero(mensaje)

Esta función solicita al usuario que ingrese el género del cliente ('M' para masculino o 'F' para femenino). Convierte la entrada a mayúsculas para hacer la validación. Si la entrada es incorrecta, sigue solicitando hasta obtener un valor válido.

```
def obtener_genero(mensaje):
    while True:
        genero = input(mensaje).upper()
        if genero in ['M', 'F']:
            return genero
        else:
            print("Error: Ingrese 'M' para masculino o 'F' para
femenino.")
```

2. Cálculos de precios y descuentos

a) calcular_precio_servicio(tipo_servicio)

Esta función retorna el precio correspondiente al tipo de servicio seleccionado. Los precios son:

- Económico: $70
- Ejecutivo: $140
- Primera Clase: $280

El tipo de servicio se pasa como parámetro y se utiliza un diccionario para asignar los precios.

```
# Función para calcular el precio según el tipo de servicio
def calcular_precio_servicio(tipo_servicio):
```

```
    precios = {1: 70.00, 2: 140.00, 3: 280.00}
    return precios.get(tipo_servicio, 0)
```

b) calcular_descuento(cantidad_pasajes)

Calcula el porcentaje de descuento en función de la cantidad de pasajes comprados:

- 1 pasaje: 0% de descuento.
- 2 a 5 pasajes: 5% de descuento.
- 6 a 10 pasajes: 12% de descuento.
- Más de 10 pasajes: 15% de descuento.

```
# Función para calcular el descuento según la cantidad de pasajes
def calcular_descuento(cantidad_pasajes):
    if cantidad_pasajes == 1:
        return 0.00
    elif 2 <= cantidad_pasajes <= 5:
        return 0.05
    elif 6 <= cantidad_pasajes <= 10:
        return 0.12
    else:
        return 0.15
```

3. Registro de ventas

registrar_venta()

Esta función procesa la venta de pasajes, solicitando varios datos al usuario:

1. **Tipo de cliente** (1 o 2).
2. **Cantidad de pasajes**.
3. **Género del cliente** (M o F).
4. **Tipo de servicio** (económico, ejecutivo, primera clase).

Luego, se realizan los siguientes cálculos:

- El precio bruto de la venta (cantidad de pasajes × precio por servicio).
- El monto del descuento.
- El importe neto (precio bruto - descuento).

También actualiza las estadísticas globales, como el total acumulado de ventas y la cantidad de ventas de clientes masculinos, femeninos, y en ciertos rangos de precios.

```
# Función para procesar una venta
def registrar_venta():
    global genero_m, ventas_femeninas, ventas_rango,
acumulado_importe_ventas, tipo_cliente_1,
acumulado_importe_neto_tipo_1

    tipo_cliente = obtener_entero("Ingrese tipo de cliente (1 o 2):
", 1, 2)
    cantidad_pasajes = obtener_entero("Ingrese cantidad de pasajes:
", 1)
    genero_cliente = obtener_genero("Ingrese género del cliente
(M/F): ")
    tipo_servicio = obtener_entero("Ingrese tipo de servicio (1:
Económico, 2: Ejecutivo, 3: Primera Clase): ", 1, 3)

    # Cálculos
    precio = calcular_precio_servicio(tipo_servicio)
    descuento = calcular_descuento(cantidad_pasajes)
    importe_bruto = cantidad_pasajes * precio
    monto_descuento = importe_bruto * descuento
    importe_neto = importe_bruto - monto_descuento

    # Mostrar resultados de la venta
    print(f"\nImporte Bruto: ${importe_bruto:.2f}")
    print(f"Descuento aplicado: ${monto_descuento:.2f}")
    print(f"Importe Neto: ${importe_neto:.2f}\n")

    # Actualización de estadísticas
    acumulado_importe_ventas += importe_neto
    if genero_cliente == 'M':
        genero_m += 1
    if genero_cliente == 'F' and 140 <= importe_neto <= 1000:
        ventas_femeninas += 1
    if 70 <= importe_neto <= 500:
        ventas_rango += 1
```

```
    if tipo_cliente == 1:
        tipo_cliente_1 += 1
        acumulado_importe_neto_tipo_1 += importe_neto
```

4. Generar reporte de ventas

mostrar_reporte()

Genera un informe de las ventas acumuladas. Incluye:

- Número de clientes masculinos.
- Ventas entre $70 y $500.
- Ventas a clientas femeninas entre $140 y $1000.
- Total acumulado de ventas.
- Promedio de importe neto para los clientes tipo 1 (si existen).

```
# Función para mostrar el reporte de ventas
def mostrar_reporte():
    print("\nReporte de Ventas:")
    print(f"Cantidad de clientes de género masculino: {genero_m}")
    print(f"Cantidad de ventas con Importe Neto entre $70 y $500:
{ventas_rango}")
    print(f"Cantidad de ventas de clientes femeninos con Importe Neto
entre $140 y $1000: {ventas_femeninas}")
    print(f"Acumulado total de Importe de Ventas:
${acumulado_importe_ventas:.2f}")

    if tipo_cliente_1 > 0:
        promedio_importe_neto_tipo_1 = acumulado_importe_neto_tipo_1
/ tipo_cliente_1
        print(f"Acumulado de Importe Neto de clientes de tipo 1:
${acumulado_importe_neto_tipo_1:.2f}")
        print(f"Promedio de Importe Neto de clientes de tipo 1:
${promedio_importe_neto_tipo_1:.2f}")
    else:
        print("No se han registrado clientes de tipo 1.")
```

5. Variables acumulativas globales

Estas variables mantienen estadísticas globales de las ventas:

- genero_m: Número de clientes masculinos.
- ventas_rango: Ventas con importe neto entre $70 y $500.
- ventas_femeninas: Ventas de clientas femeninas con importe neto entre $140 y $1000.
- acumulado_importe_ventas: Total acumulado de las ventas.
- tipo_cliente_1: Número de clientes del tipo 1.
- acumulado_importe_neto_tipo_1: Importe neto acumulado para los clientes tipo 1.

```
# Variables acumulativas globales
genero_m = 0
ventas_rango = 0
ventas_femeninas = 0
acumulado_importe_ventas = 0
tipo_cliente_1 = 0
acumulado_importe_neto_tipo_1 = 0
```

6. Bucle principal del menú

El menú se ejecuta dentro de un bucle while que permite realizar varias operaciones hasta que el usuario elija salir. Aquí se utilizan try y except para manejar errores al ingresar valores no válidos.

a) Opción 1: Registrar una venta

Llama a la función registrar_venta() para registrar una venta.

b) Opción 2: Mostrar el reporte de ventas

Llama a la función mostrar_reporte() para mostrar las estadísticas.

c) Opción 3: Salir

Finaliza el programa al poner verificar_menu = False.

d) Validación de opción no válida

Si el usuario ingresa una opción no válida, se muestra un mensaje de error y el programa espera 3 segundos antes de continuar.

```
import time
# Bucle principal
verificar_menu = True
while verificar_menu:
    print("BIENVENIDO A MI PROGRAMA DE EMPRESA DE TRANSPORTES
INTERPROVINCIAL\n")
    print("\nMENU DE OPCIONES\n[1] REGISTRAR VENTA DE PASAJE\n[2]
REPORTAR VENTAS\n[3] SALIR")

    opcion = None  # Inicializamos 'opcion' para la validación

    # Bucle para asegurar que se ingresa un número entero válido
    while opcion is None:
        try:
            opcion = int(input("Seleccione una opción: "))
        except ValueError:
            print("Error: Debe ingresar un número entero.")

    # Verificamos las opciones una vez que 'opcion' es válida
    if opcion == 1:
        registrar_venta()
        # Opción para seguir registrando ventas o volver al menú
principal
        while True:
            otra_opcion = input("\n¿Desea registrar otra venta?
(S/N): ").upper()
            if otra_opcion == 'S':
                registrar_venta()
            elif otra_opcion == 'N':
                break
            else:
                print("Opción no válida. Ingrese 'S' o 'N'.")

    elif opcion == 2:
        mostrar_reporte()

    elif opcion == 3:
```

```
        verificar_menu = False
        print("Saliendo del programa...")

    else:
        print("Error: Opción no válida. Intente de nuevo.")
        time.sleep(3)  # Pausa de 3 segundos antes de reiniciar el
bucle
```

II. PRUEBAS

Primera Prueba:

```
C:\PWEB2-2024B\2024B\PWII-2024B\lab04>python Ejercicio.py
BIENVENIDO A MI PROGRAMA DE EMPRESA DE TRANSPORTES INTERPROVINCIAL


MENU DE OPCIONES
[1] REGISTRAR VENTA DE PASAJE
[2] REPORTAR VENTAS
[3] SALIR
Seleccione una opción: 1
Ingrese tipo de cliente (1 o 2): 1
Ingrese cantidad de pasajes: 6
Ingrese género del cliente (M/F): m
Ingrese tipo de servicio (1: Económico, 2: Ejecutivo, 3: Primera Clase): 2

Importe Bruto: $840.00
Descuento aplicado: $100.80
Importe Neto: $739.20


¿Desea registrar otra venta? (S/N): s
Ingrese tipo de cliente (1 o 2): 1
Ingrese cantidad de pasajes: 5
Ingrese género del cliente (M/F): f
Ingrese tipo de servicio (1: Económico, 2: Ejecutivo, 3: Primera Clase): 2

Importe Bruto: $700.00
Descuento aplicado: $35.00
Importe Neto: $665.00


¿Desea registrar otra venta? (S/N): n
BIENVENIDO A MI PROGRAMA DE EMPRESA DE TRANSPORTES INTERPROVINCIAL


MENU DE OPCIONES
[1] REGISTRAR VENTA DE PASAJE
[2] REPORTAR VENTAS
[3] SALIR
Seleccione una opción: 2

Reporte de Ventas:
Cantidad de clientes de género masculino: 1
Cantidad de ventas con Importe Neto entre $70 y $500: 0
Cantidad de ventas de clientes femeninos con Importe Neto entre $140 y $1000: 1
Acumulado total de Importe de Ventas: $1404.20
Acumulado de Importe Neto de clientes de tipo 1: $1404.20
Promedio de Importe Neto de clientes de tipo 1: $702.10
BIENVENIDO A MI PROGRAMA DE EMPRESA DE TRANSPORTES INTERPROVINCIAL
```

```
MENU DE OPCIONES
[1] REGISTRAR VENTA DE PASAJE
[2] REPORTAR VENTAS
[3] SALIR
Seleccione una opción: 3
Saliendo del programa...
```

En la primera prueba, comencé ejecutando el programa y se me mostró el menú principal con tres opciones: registrar una venta, reportar las ventas o salir. Decidí seleccionar la opción 1 para registrar una venta de pasaje.

Para la **primera venta**, ingresé los siguientes datos:

- Tipo de cliente: seleccioné el tipo 1.
- Cantidad de pasajes: ingresé 6.
- Género del cliente: elegí masculino.
- Tipo de servicio: opté por el servicio ejecutivo.

El programa calculó el importe bruto, que fue $840, y aplicó un descuento del 12%, lo que resultó en un descuento de $100.80. Finalmente, el importe neto a pagar fue de $739.20. Después de ver los resultados, me preguntó si quería registrar otra venta, a lo cual respondí que sí.

En la **segunda venta**, ingresé nuevos datos:

- Tipo de cliente: otra vez, seleccioné el tipo 1.
- Cantidad de pasajes: ingresé 5.
- Género del cliente: elegí femenino.
- Tipo de servicio: nuevamente, el servicio ejecutivo.

El importe bruto fue de $700, con un descuento del 5%, es decir, $35, lo que dejó el importe neto en $665. Después de esta venta, decidí no registrar más ventas.

Luego, volví al menú principal y seleccioné la opción 2 para generar el **reporte de ventas**. El programa me mostró que hubo 1 cliente masculino, ninguna venta con un importe neto entre $70 y $500, y 1 venta de cliente femenino con un importe entre $140 y $1000. El acumulado total de ventas fue $1404.20, y el promedio de ventas para los clientes de tipo 1 fue de $702.10.

Finalmente, seleccioné la opción 3 para salir del programa.

SEGUNDA PRUEBA:

```
C:\PWEB2-2024B\2024B\PWII-2024B\lab04>python Ejercicio.py
BIENVENIDO A MI PROGRAMA DE EMPRESA DE TRANSPORTES INTERPROVINCIAL


MENU DE OPCIONES
[1] REGISTRAR VENTA DE PASAJE
[2] REPORTAR VENTAS
[3] SALIR
Seleccione una opción: |
Error: Debe ingresar un número entero.
Seleccione una opción: 2.5
Error: Debe ingresar un número entero.
Seleccione una opción: 4
Error: Opción no válida. Intente de nuevo.
BIENVENIDO A MI PROGRAMA DE EMPRESA DE TRANSPORTES INTERPROVINCIAL


MENU DE OPCIONES
[1] REGISTRAR VENTA DE PASAJE
[2] REPORTAR VENTAS
[3] SALIR
Seleccione una opción: 1
Ingrese tipo de cliente (1 o 2): 2
Ingrese cantidad de pasajes: 3
Ingrese género del cliente (M/F): m
Ingrese tipo de servicio (1: Económico, 2: Ejecutivo, 3: Primera Clase): 1

Importe Bruto: $210.00
Descuento aplicado: $10.50
Importe Neto: $199.50


¿Desea registrar otra venta? (S/N): y
Opción no válida. Ingrese 'S' o 'N'.

¿Desea registrar otra venta? (S/N): s
Ingrese tipo de cliente (1 o 2): 1
Ingrese cantidad de pasajes: 4
Ingrese género del cliente (M/F): f
Ingrese tipo de servicio (1: Económico, 2: Ejecutivo, 3: Primera Clase): 3

Importe Bruto: $1120.00
Descuento aplicado: $56.00
Importe Neto: $1064.00
```

```
¿Desea registrar otra venta? (S/N): s
Ingrese tipo de cliente (1 o 2): 1
Ingrese cantidad de pasajes: 10
Ingrese género del cliente (M/F): 4
Error: Ingrese 'M' para masculino o 'F' para femenino.
Ingrese género del cliente (M/F): f
Ingrese tipo de servicio (1: Económico, 2: Ejecutivo, 3: Primera Clase): 4
Error: El valor debe estar entre 1 y 3.
Ingrese tipo de servicio (1: Económico, 2: Ejecutivo, 3: Primera Clase): 2

Importe Bruto: $1400.00
Descuento aplicado: $168.00
Importe Neto: $1232.00


¿Desea registrar otra venta? (S/N): n
BIENVENIDO A MI PROGRAMA DE EMPRESA DE TRANSPORTES INTERPROVINCIAL


MENU DE OPCIONES
[1] REGISTRAR VENTA DE PASAJE
[2] REPORTAR VENTAS
[3] SALIR
Seleccione una opción: 2

Reporte de Ventas:
Cantidad de clientes de género masculino: 1
Cantidad de ventas con Importe Neto entre $70 y $500: 1
Cantidad de ventas de clientes femeninos con Importe Neto entre $140 y $1000: 0
Acumulado total de Importe de Ventas: $2495.50
Acumulado de Importe Neto de clientes de tipo 1: $2296.00
Promedio de Importe Neto de clientes de tipo 1: $1148.00
BIENVENIDO A MI PROGRAMA DE EMPRESA DE TRANSPORTES INTERPROVINCIAL


MENU DE OPCIONES
[1] REGISTRAR VENTA DE PASAJE
[2] REPORTAR VENTAS
[3] SALIR
Seleccione una opción: 3
Saliendo del programa...

C:\PWEB2-2024B\2024B\PWII-2024B\lab04>
```

En esta segunda prueba, verifico si funcionan las validaciones y los errores. Y como se puede apreciar funciona todo correctamente. Por ejemplo en la primera parte quiero verificar los errores cuando se ingresa un no entero (mensaje: “Debe ingresar un numero entero”) y después que pasa si se ingresa un numero entero pero no válido (mensaje: “Número no válido, vuelva a intentarlo….”).

Luego, seleccioné la opción "1" para registrar una venta. Ingresé todos los datos requeridos correctamente, y el programa realizó los cálculos pertinentes, mostrando el importe bruto, el descuento aplicado y el importe neto de la venta.

A continuación, al solicitar otra venta, ingresé "y" en vez de "S" o "N" para confirmar si quería registrar otra venta. El programa respondió: “Opción no válida. Ingrese 'S' o 'N'.” Esto demuestra que el manejo de errores está bien implementado, permitiéndome corregir mi entrada.

Posteriormente, al registrar una nueva venta, cometí un error al ingresar el género como "4". El programa me advirtió: “Error: Ingrese 'M' para masculino o 'F' para femenino.” Después, ingresé correctamente "F", pero al ingresar el tipo de servicio, volví a cometer un error al poner "4", que estaba fuera del rango válido (1 a 3). Nuevamente, el programa mostró un mensaje de error: “Error: El valor debe estar entre 1 y 3.” Esto confirmó que la validación de rangos también funciona bien.

Finalmente, el programa me permitió completar el registro de ventas y generar un reporte. Al seleccionar la opción para reportar ventas, se presentaron las estadísticas de manera clara y precisa, reflejando todas las validaciones y cálculos realizados.

En conclusión, esta prueba me permitió verificar que todas las validaciones y manejos de errores del programa funcionan adecuadamente, mejorando la experiencia del usuario y asegurando que los datos ingresados sean válidos.

III. CUESTIONARIO:

¿QUÉ FUE LO NUEVO QUE APRENDI EN ESTE LABORATORIO?

En este laboratorio, aprendí varias cosas nuevas que mejoraron mi comprensión y habilidades de programación:

1. **Validación de Entradas:** Aprendí a implementar validaciones efectivas para asegurar que los usuarios ingresen datos correctos, utilizando estructuras de control como bucles while y declaraciones try-except. Esto garantiza que el programa no falle debido a entradas inválidas y mejora la robustez del software.
2. **Manejo de Errores:** A través de la implementación de mensajes de error claros y específicos, comprendí la importancia de guiar al usuario en la corrección de errores. Esto no solo mejora la experiencia del usuario, sino que también hace que el programa sea más fácil de usar.
3. **Cálculos Dinámicos:** Aprendí a realizar cálculos basados en las entradas del usuario, como el cálculo del importe bruto, descuentos y el importe neto de las ventas. Esto me permitió entender cómo manejar datos y realizar operaciones matemáticas en tiempo real.

4. **Uso de Funciones:** El laboratorio me ayudó a profundizar en el uso de funciones para estructurar mejor el código, lo que facilita la legibilidad y el mantenimiento del mismo. Aprendí a dividir el programa en partes más manejables, como las funciones para registrar ventas y mostrar reportes.
5. **Acumulación de Datos:** Aprendí a utilizar variables acumulativas para llevar un seguimiento de estadísticas como el número de ventas y los ingresos totales. Esto es crucial para generar reportes significativos que reflejen el desempeño del negocio.
6. **Interacción con el Usuario:** Me familiaricé con la creación de un menú interactivo, permitiendo que los usuarios seleccionen opciones. Esto me ayudó a entender cómo diseñar la interfaz de usuario de manera efectiva y cómo guiar al usuario a través de diferentes funcionalidades del programa.

CONCLUSIONES

- Aprendí la importancia de validar las entradas del usuario, lo que mejoró la precisión del programa al asegurar que los datos ingresados fueran correctos y coherentes.
- Me di cuenta de que proporcionar mensajes de error claros y específicos ayuda a los usuarios a entender mejor sus errores y corregirlos, mejorando la interacción del usuario y la experiencia general.
- Al utilizar funciones para organizar el código, me sentí más cómodo y seguro al desarrollar el programa, lo que facilita la lectura, mantenimiento y actualización del código en el futuro.

METODOLOGÍA DE TRABAJO

1. **Definición del Problema**:

   Identifiqué el objetivo del laboratorio, que era desarrollar un programa para gestionar las ventas de pasajes en una empresa de transportes interprovincial.

2. **Investigación Preliminar**:

Realicé una revisión de conceptos básicos sobre programación en Python, especialmente sobre el manejo de entradas del usuario, validaciones y cálculos matemáticos.

3. **Diseño del Programa**:

   Esquematicé las funciones necesarias, como la gestión de ventas, la generación de reportes y la validación de datos. Decidí cómo se estructuraría el flujo del programa, incluyendo los menús y las interacciones del usuario.

4. **Desarrollo del Código**:

   Comencé a programar, implementando cada función de manera modular. Me aseguré de seguir las mejores prácticas de codificación para mejorar la legibilidad y el mantenimiento del código.

5. **Pruebas y Validación**:

   Realicé múltiples pruebas para verificar el funcionamiento correcto del programa. Ingresé diferentes tipos de datos para asegurarme de que las validaciones funcionaran adecuadamente y se manejara correctamente cualquier error.

6. **Documentación**:

   Documenté el código para explicar la lógica detrás de cada función y el flujo del programa, así como los mensajes de error que se mostraban al usuario.

7. **Revisión y Mejora**:

   Después de las pruebas iniciales, identifiqué áreas de mejora en el código y realicé ajustes para optimizar la funcionalidad y la experiencia del usuario.

8. **Conclusiones**:

   Reflexioné sobre lo aprendido durante el proceso y cómo aplicar este conocimiento en futuros proyectos. También consideré la importancia de la validación de datos y la experiencia del usuario en el desarrollo de software.

RUBRICA:

- El alumno deberá autocalificarse, marcando o dejando en blanco las celdas de la columna **Chec- klist**, deacuerdo a si cumplió o no con el ítem correspondiente.
- Si un alumno supera la fecha de entrega, su calificación siempre será sobre la nota mínima

aprobatoria, siempre y cuando cumpla con todos lo items. (Máximo 24 horas)

- El alumno debe autocalificarse en la columna **Estudiante** de acuerdo a la tabla de calificación de niveles de desempeño:

Tabla 2: Niveles de desempeño

	Nivel			
Puntos	Insatisfactorio 25 %	En Proceso 50 %	Satisfactorio 75 %	Sobresaliente 100 %
2.0	0.5	1.0	1.5	2.0
4.0	1.0	2.0	3.0	4.0

Tabla 3: Rúbrica para contenido del Informe y evidencias

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
1. GitHub	Repositorio se pudo clonar y se evidencia la estructura adecuada para revisar los entregables. (Se descontará puntos por error u observación)	4	X	X	
2. Commits	Hay porciones de código fuente asociado a los commits planificados con explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4		X	
3. Ejecución	Se incluyen comandos para ejecuciones y pruebas del código fuente explicadas gradualmente que permitirían replicar el proyecto. (Se des- contará puntos por cada omisión)	4		X	
4. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2		X	
7. Ortografía	El documento no muestra errores ortográficos. (Se descontará puntos por error encontrado)	2		X	
8. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente con explicaciones puntuales pero precisas, agregando diagramas generados a partir del código fuente y refleja un acabado impecable. (El profesor puede preguntar para refrendar calificación).	4		X	
Total		20		16	

REFERENCIAS Y BIBLIOGRAFÍA

- Python Software Foundation. (2024). *The Python Standard Library*. https://docs.python.org/3/library/
- Sweigart, A. (2020). *Automate the Boring Stuff with Python* (2nd ed.). No Starch Press. https://automatetheboringstuff.com/
- van Rossum, G., & Drake, F. L. (2009). *The Python Language Reference*. Python Software Foundation. https://docs.python.org/3/reference/