Carrera:

Ingeniería en sistemas de la información 4to semestre

Materia:

Programación IV

Docente:

Ing. Edison Meneses Mg.

Estudiante:

Alberto Falconí

Tema:

Sistema de Gestión de Consultas Médicas

Fecha:

25 de abril de 2025

Contenido

1. Objetivo del Proyecto 3

2. Contexto del Sistema 3

3. Desarrollo 3

3.1 Estructura del Proyecto 3

3.2 Diseño de la clase Paciente 4

3.3 Funcionamiento del Menú y Funciones 4

3.4 Capturas de pantalla 5

4. Conclusiones 8

4.1 Aprendizajes 8

4.2 Dificultades e Intereses 9

4.3 Mejoras futuras 9

5. Anexos 9

Informe Técnico: Sistema de Gestión de Consultas Médicas

1. Objetivo del Proyecto

El propósito de este proyecto se basa en desarrollar un sistema con modularización en lenguaje Python permitiendo registrar, consultar y gestionar pacientes y sus historiales médicos de manera funcional y básica. Se tiene como enfoque a consultorios pequeños brindando una herramienta que automatice diferentes procesos y disminuyendo el tiempo de trabajo.

2. Contexto del Sistema

Actualmente, existen centros médicos que desperdician el tiempo en procesos manuales como la gestión de sus pacientes y otras funcionalidades propias del ámbito. Con este sistema se ofrece una solución eficaz, simple y funcional para digitalizar dichos procesos, optimizando el tiempo de atención y mejorando el acceso a la información del paciente.

3. Desarrollo

3.1 Estructura del Proyecto

El sistema está dividido en tres archivos principales para una mejor organización:

- **paciente.py**: Contiene la clase Paciente, que representa la información de cada paciente y su historial de consultas.
- **funciones.py**: Incluye funciones relacionadas con el manejo de pacientes como registro, búsqueda, visualización, y almacenamiento de consultas.
- **main.py**: Archivo principal desde el cual se ejecuta el programa. Contiene el menú interactivo que permite al usuario acceder a las distintas funcionalidades.

3.2 Diseño de la clase Paciente

La clase está diseñada para almacenar los datos personales de un paciente junto a su historial de consultas. Incluye métodos para agregar consultas y mostrar el historial completo.

```
class Paciente:
    def __init__(self,cedula, nombre, edad, tipo_sangre):
        #Datos del paciente
        self.cedula=cedula
        self.nombre=nombre
        self.edad=edad
        self.tipo_sangre=tipo_sangre

        #Las consultas son una lista de objetos de la clase Consulta
        self.consultas=[]

    def agregar_consulta(self, fecha, diagnostico, tratamiento):
        consulta=[fecha, diagnostico, tratamiento]
        self.consultas.append(consulta)

    def mostrar_historial(self):
        print(f"Historial de consultas de {self.nombre}:")
        for consulta in self.consultas:
            print(f"Fecha: {consulta[0]}, Diagnóstico: {consulta[1]},
Tratamiento: {consulta[2]}")
```

3.3 Funcionamiento del Menú y Funciones

El menú se ejecuta desde main.py, con opciones claras y validaciones básicas. Algunas de las funciones incluidas:

- Registrar nuevo paciente (verifica que la cédula no esté repetida)
- Registrar consulta a paciente existente
- Buscar paciente por cédula y mostrar historial
- Mostrar todos los pacientes registrados
- Salir del sistema

Ejemplo de menú:

```
from funciones import *

# Definición de la función principal
def menu():
    while True:
        print("\n\t\tMENU PRINCIPAL")
        print("1. Registrar nuevo paciente")
        print("2. Registrar consulta")
        print("3. Mostrar datos del paciente")
        print("4. Mostrar Pacientes Registrados")
        print("5. Salir")

        opcion = int(input("Seleccione una opción: "))

        if opcion == 1:
            registrar_paciente()
        elif opcion == 2:
            registrar_consulta()
        elif opcion == 3:
            mostrar_paciente()
        elif opcion == 4:
            mostrar_todos()
        elif opcion == 5:
            print("Saliendo del programa...")
            break
        else:
            print("Opción inválida. Intente nuevamente.")

if __name__ == "__main__":
    menu()
```

3.4 Capturas de pantalla

Clase Paciente

```
4. class Paciente:
5.     def __init__(self,cedula, nombre, edad, tipo_sangre):
6.         #Datos del paciente
7.         self.cedula=cedula
8.         self.nombre=nombre
9.         self.edad=edad
10.        self.tipo_sangre=tipo_sangre
11.
12.        #Las consultas son una lista de objetos de la clase Consulta
13.        self.consultas=[]
14.
```

```
15.     def agregar_consulta(self, fecha, diagnostico, tratamiento):
16.         consulta=[fecha, diagnostico, tratamiento]
17.         self.consultas.append(consulta)
18.
19.     def mostrar_historial(self):
20.         print(f"Historial de consultas de {self.nombre}:")
21.         for consulta in self.consultas:
22.             print(f"Fecha: {consulta[0]}, Diagnóstico: {consulta[1]},
    Tratamiento: {consulta[2]}")
23.
```

Funciones del Sistema

```
from paciente import Paciente

#Lista para almacenar los pacientes
lista_pacientes = []

def buscar_paciente(cedula):
    for paciente in lista_pacientes:
        if paciente.cedula == cedula:
            return paciente
    return None

def registrar_paciente():
    print("\n\t\tREGISTRAR NUEVO PACIENTE")
    cedula = input("Ingrese la cédula del paciente: ")
    while len(cedula) != 10 or not cedula.isdigit():
        print("Cédula inválida. Debe tener 10 dígitos.")
        cedula = input("Ingrese la cédula del paciente: ")

    if buscar_paciente(cedula):
        print("El paciente ya está registrado.")
        return

    nombre = input("Ingrese el nombre del paciente: ")
    edad = input("Ingrese la edad del paciente: ")
    while not edad.isdigit() or int(edad)<=0:
        print("Edad inválida. Debe ser un número positivo.")
        edad = input("Ingrese la edad del paciente: ")

    tipo_sangre = input("Ingrese el tipo de sangre del paciente: ")
    while tipo_sangre not in ["A+", "A-", "B+", "B-", "AB+", "AB-", "O+", "O-
"]:
        print("Tipo de sangre inválido. Debe ser uno de los siguientes: A+, A-
, B+, B-, AB+, AB-, O+, O-.")
        tipo_sangre = input("Ingrese el tipo de sangre del paciente: ")

    nuevo_paciente = Paciente(cedula, nombre, edad, tipo_sangre)
```

```
    lista_pacientes.append(nuevo_paciente)
    print(f"Paciente {nombre} registrado exitosamente.")

def registrar_consulta():
    print("\n\t\tREGISTRAR CONSULTA")
    cedula = input("Ingrese la cédula del paciente: ")
    while len(cedula) != 10 or not cedula.isdigit():
        print("Cédula inválida. Debe tener 10 dígitos.")
        cedula = input("Ingrese la cédula del paciente: ")
    paciente = buscar_paciente(cedula)

    if not paciente:
        print("Paciente no encontrado.")
        return

    fecha = input("Ingrese la fecha de la consulta (DD/MM/AAAA): ")
    diagnostico = input("Ingrese el diagnóstico: ")
    tratamiento = input("Ingrese el tratamiento: ")

    paciente.agregar_consulta(fecha, diagnostico, tratamiento)
    print(f"Consulta registrada para {paciente.nombre}.")

def mostrar_paciente():
    print("\n\t\tDATOS DEL PACIENTE")
    cedula = input("Ingrese la cédula del paciente: ")
    while len(cedula) != 10 or not cedula.isdigit():
        print("Cédula inválida. Debe tener 10 dígitos.")
        cedula = input("Ingrese la cédula del paciente: ")
    paciente = buscar_paciente(cedula)

    if not paciente:
        print("Paciente no encontrado.")
        return

    print(f"Nombre: {paciente.nombre}, Edad: {paciente.edad}, Tipo de Sangre:
{paciente.tipo_sangre}")
    paciente.mostrar_historial()

def mostrar_todos():
    print("\n\t\tPACIENTES REGISTRADOS")
    if not lista_pacientes:
        print("No hay pacientes registrados.")
        return

    for paciente in lista_pacientes:
        print(f"Cédula: {paciente.cedula}, Nombre: {paciente.nombre}, Edad:
{paciente.edad}, Tipo de Sangre: {paciente.tipo_sangre}")
        paciente.mostrar_historial()
```

Menú

```python
from funciones import *

# Definición de la función principal
def menu():
    while True:
        print("\n\t\tMENU PRINCIPAL")
        print("1. Registrar nuevo paciente")
        print("2. Registrar consulta")
        print("3. Mostrar datos del paciente")
        print("4. Mostrar historial de consultas")
        print("5. Salir")

        opcion = int(input("Seleccione una opción: "))

        if opcion == 1:
            registrar_paciente()
        elif opcion == 2:
            registrar_consulta()
        elif opcion == 3:
            mostrar_paciente()
        elif opcion == 4:
            mostrar_todos()
        elif opcion == 5:
            print("Saliendo del programa...")
            break
        else:
            print("Opción inválida. Intente nuevamente.")

if __name__ == "__main__":
    menu()
```

4. Conclusiones

4.1 Aprendizajes

Los aprendizajes que fueron implementados para el desarrollo del sistema básico comprenden temas como la Programación orientada a objetos para la parte de los datos y métodos del Paciente. La Modularización fue clase al momento de desarrollar el sistema ya que brinda un entendimiento más claro de cada funcionalidad del código.

Además, se usaron validaciones para el ingreso tanto de edad como de cédula, permitiendo que el usuario pueda comprender el error al momento del ingreso de datos.

4.2 Dificultades e Intereses

La parte más interesante del sistema es la Modularización y las diferentes funciones por parte del archivo “funciones.py” que comprende el comportamiento principal del sistema.

Por otro lado, lo más desafiante fue la parte del menú al momento de diseñar para que el usuario pueda entender de mejor manera las opciones. Las importaciones de las funciones, métodos y atributos de cada uno de los archivos también tuvieron su grado de complejidad.

4.3 Mejoras futuras

Para futuras modificaciones, agregar un algoritmo que permita validar que la cédula ingresada exista y sea correcta. Implementar otras librerías para un manejo de exportación de archivos que contenga la información de todos los pacientes registrados, de tal manera que se pueda generar informes y mejorar la atención. Por último, tomar a consideración la implementación de una interfaz gráfica intuitiva y visual con la ayuda de librerías como Tkinter propia del lenguaje Python.

5. Anexos

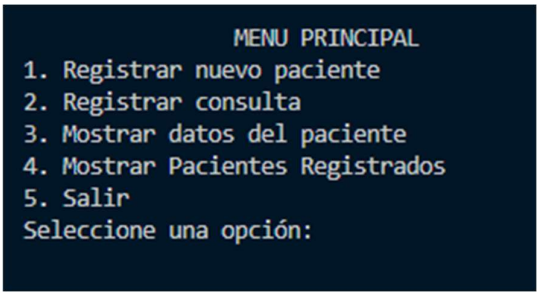

Ilustración 1 Menú del Sistema

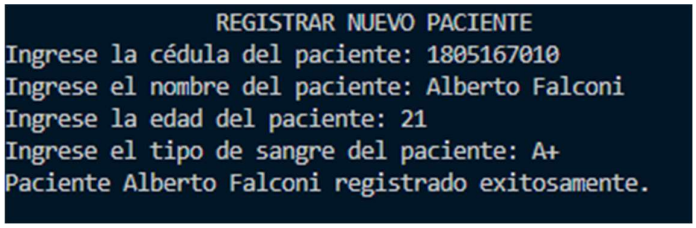

Ilustración 2 Opción 1 - Registro Paciente

```
REGISTRAR CONSULTA
Ingrese la cédula del paciente: 1805167010
Ingrese la fecha de la consulta (DD/MM/AAAA): 12-02-205
Ingrese el diagnóstico: Gripe
Ingrese el tratamiento: Ibuprofeno 8 horas
Consulta registrada para Alberto Falconi.
```

Ilustración 3 Opción 2 - Registro Consulta

```
DATOS DEL PACIENTE
Ingrese la cédula del paciente: 1805167010
Nombre: Alberto Falconi, Edad: 21, Tipo de Sangre: A+
Historial de consultas de Alberto Falconi:
Fecha: 12-02-205, Diagnóstico: Gripe, Tratamiento: Ibuprofeno 8 horas
```

Ilustración 4 Opción 4 - Mostrar Datos Paciente

```
PACIENTES REGISTRADOS
Cédula: 1805167010, Nombre: Alberto Falconi, Edad: 21, Tipo de Sangre: A+
Historial de consultas de Alberto Falconi:
Fecha: 12-02-205, Diagnóstico: Gripe, Tratamiento: Ibuprofeno 8 horas
```

Ilustración 5 Opción 4 - Mostrar Pacientes Registrados

```
MENU PRINCIPAL
1. Registrar nuevo paciente
2. Registrar consulta
3. Mostrar datos del paciente
4. Mostrar Pacientes Registrados
5. Salir
Seleccione una opción: 6
Opción inválida. Intente nuevamente.
```

Ilustración 6 Elección Errónea

```
MENU PRINCIPAL
1. Registrar nuevo paciente
2. Registrar consulta
3. Mostrar datos del paciente
4. Mostrar Pacientes Registrados
5. Salir
Seleccione una opción: 5
Saliendo del programa...
```

Ilustración 7 Opción 5 - Salir