# GESTOR DE NOTAS ACADEMICAS

# MANUAL TECNICO

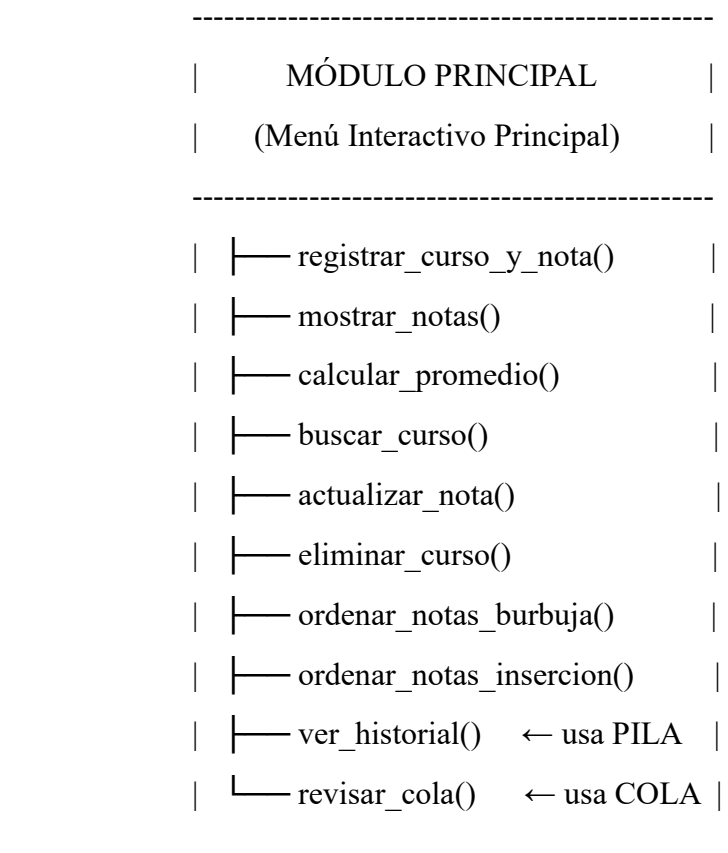
Mayli Fabiola Pirir

## 1. Descripcion tecnica general

El Sistema ‘Gestor de notas academicas’ esta desarrollado en Python y funciona completamente en consola, su propósito es gestionar el registro, modificación y analizar notas académicas asociadas a cursos. El sistema implementa estructuras de datos básicas (listas, pilas y colas) y algotitmos de ordenamiento.

## 2. Arquitectura y estructura general del código

El programa sigue un modelo modular, donde cada función tiene una responsabilidad especifica el flujo principal se controla mediante un menu interactivo que utiliza un bucle ‘while’ y condicionales para mantener la ejecución hasta que el usuario decida salir.




## 3. Uso de estructuras de datos

El sistema utiliza tres tipos de estructuras fundamentales:

- Lista principal: almacena los cursos y sus notas en forma de diccionario {‘curso’: nombre, ‘nota’: valor}.

- Pila (historial): guarda las acciones realizadas, su principio LIFO (ultimo en entrar, primero en salir) permite deshacer o visualizar los últimos cambios.
- Cola (revisión): gestiona los cursos pendientes de revisión en orden FIFO (primero en entrar, primero en salir).

Ejemplo de flujo de datos

[Usuario]

↓

[Menú Principal]

↓

[Acción Elegida]

↓

[Listado Principal]

↓

(Registra curso y nota)

↓

[Pila Historial]

↑ Guarda acción realizada

↓

[Cola de Revisión]

↑ Agrega cursos pendientes de revisión

## 4. Algoritmos de ordenamiento implementados

Se incluyeron dos algoritmos clásicos de ordenamiento, programados manualmente para fortalecer la comprensión del manejo de listas y bucles.

- Método Burbuja: recorre la lista comparando elementos adyacentes e intercambiándolos si están desordenados.
- Metodo de Inserción: construye una lista ordenada insertando elementos en la posición correcta uno a uno.

Ambos algoritmos se eligieron por su simplicidad y su valor didáctico, ya que permite observar como se mueven los elementos dentro de una lista sin usar funciones predefinidas de Python como sort() o sorted().

## 5. Documentación de funciones y modulos

- Registrar_curso_y_nota(): solicita datos y los agrega a la lista principal, validando entrada y rango.
- Mostrar_notas(): imprime todos los cursos registrados.
- Calcular_promedio(): recorre la lista sumando las notas y calcula el promedio general.
- Buscar_curso(): usa la búsqueda lineal para localizar un curso especifico
- Actualizar_nota(): permite modificar la nota de un curso existente
- Eliminar_curso(): borra un curso de la lista principal y registra la acción en la pila
- Ver_historial(): muestra las ultimas acciones realizadas (uso de pila).
- Revisar_cola(): procesa los cursos pendientes en orden de llegada (uso de cola).
- Ordenar_notas_burbuja() / ordenar_notas_insercion(): aplicn ordenamientos manuales a la lista principal.

## 6. Pseudocodigo principal

```
INICIO
  MIENTRAS (VERDADERO)
    IMPRIMIR "Menú principal"
    IMPRIMIR "1. Registrar curso y nota"
    IMPRIMIR "2. Mostrar notas"
    IMPRIMIR "3. Calcular promedio"
    IMPRIMIR "4. Buscar curso"
    IMPRIMIR "5. Actualizar nota"
    IMPRIMIR "6. Eliminar curso"
    IMPRIMIR "7. Ver historial"
    IMPRIMIR "8. Revisar cola"
    IMPRIMIR "9. Ordenar notas (burbuja)"
    IMPRIMIR "10. Ordenar notas (inserción)"
    IMPRIMIR "12. Salir"
    LEER opción
    SI opción == 1 ENTONCES
      registrar_curso_y_nota()
    SINO SI opción == 2 ENTONCES
      mostrar_notas()
    SINO SI opción == 3 ENTONCES
```

```
        calcular_promedio()
      SINO SI opción == 4 ENTONCES
        buscar_curso()
      SINO SI opción == 5 ENTONCES
        actualizar_nota()
      SINO SI opción == 6 ENTONCES
        eliminar_curso()
      SINO SI opción == 7 ENTONCES
        ver_historial()
      SINO SI opción == 8 ENTONCES
        revisar_cola()
      SINO SI opción == 9 ENTONCES
        ordenar_notas_burbuja()
      SINO SI opción == 10 ENTONCES
        ordenar_notas_insercion()
      SINO SI opción == 12 ENTONCES
        SALIR
      FIN_SI
    FIN_MIENTRAS
FIN
```