MANUAL TÉCNICO

GESTOR DE NOTAS ACADÉMICAS

Ronal Emanuel Espinoza Joaquín

OCTUBRE 2025

Contenido

Descripción Técnica General Del Sistema

El Gestor de Notas Académicas es un sistema en consola diseñado para que estudiantes de cualquier nivel puedan registrar, organizar y analizar sus calificaciones de forma sencilla y eficiente. Su objetivo principal es ofrecer una herramienta clara, sin distracciones ni interfaces complejas, que permita al estudiante visualizar su rendimiento académico en cualquier momento.

A diferencia de métodos manuales (como apuntes en papel) o herramientas más complejas (como hojas de cálculo), este sistema no requiere conocimientos técnicos ni instalaciones adicionales. Funciona desde un menú interactivo en consola, sin necesidad de conexión a internet o programas externos, siendo ideal para entornos educativos con recursos limitados.

El sistema cubre necesidades clave como:

- Registro ordenado de asignaturas y calificaciones
- Calculo automático de promedio
- Conteo de cursos aprobados y reprobados
- Búsqueda, actualización y ordenamiento de información

Lenguaje Implementado: Python
Tipo de aplicación: Consola
Enfoque: Simplicidad, claridad y práctica educativa sobre estructuras de datos y algoritmos básicos.

Requisitos Del Sistema

Funcionales

- Registro de cursos y calificaciones: permite ingresar el nombre del curso y la nota (validando que esté entre 0 y 100).

- Visualización de notas: muestra todos los cursos registrados o un aviso si no hay datos.

- Cálculo del promedio general: promedia todas las calificaciones y muestra el resultado con dos decimales.

- Conteo de cursos aprobados y reprobados: indica cuántos cursos superan el umbral de aprobación (>=60).

- Búsqueda de cursos: localiza cursos por nombre sin importar si el usuario escribe en mayúsculas o minúsculas.

- Ordenamiento: permite ordenar los cursos por nota o por nombre, según elección del usuario.

- Historial de cambios: conserva en una pila (LIFO) los registros de modificaciones y eliminaciones.

- Cola de revisión: simula solicitudes de revisión académica (FIFO).

No Funcionales

- Ejecución en consola: el sistema no requiere interfaz gráfica, lo que lo hace ligero y portable.

- Lenguaje de programación: Python, elegido por su claridad y sintaxis simple.

- Restricciones: no utiliza librerías externas, solo funciones nativas del lenguaje.

- Estructura modular: cada funcionalidad se implementa como función independiente para mayor mantenimiento.

- Validación de datos: se incluyen controles para prevenir errores de entrada.

- Flujo controlado: se gestionan las operaciones mediante bucles y condicionales.

Estructura General Del Código

El código se compone de una serie de funciones modulares y dos listas globales principales:

- notas: lista de tuplas (curso, nota) que almacena los registros.
- historial_pila: lista utilizada como pila (LIFO) para registrar los cambios del usuario.

Funciones Principales

mostrarTitulo()

registrar_nuevo_curso()

mostrar_cursosYnotas()

promedio_de_notas()

contar_aprobadosYreprobados()

buscar_curso()

actualizar_nota()

eliminar_curso()

ordenar_por_nota()

ordenar_por_nombre()

cola_revision()

historial()

Un bucle while True actúa como menú principal, mostrando las opciones disponibles y llamando a la función correspondiente según la entrada del usuario.

Uso De Estructuras De Datos

Listas (notas): Contienen los cursos y notas; permiten inserciones, búsquedas, actualizaciones y ordenamientos.

Pilas (historial_pila): Registran los cambios realizados (actualizaciones o eliminaciones), mostrando primero los más recientes (LIFO).

Colas (cola_cursos): Simulan solicitudes de revisión académica, siguiendo un orden FIFO mediante append() y pop(0)

Justificación De Los Algoritmos De Ordenamiento

Burbuja (Bubble Sort) – ordenar_por_nota(): fácil de implementar, ideal para fines didácticos y listas pequeñas. Complejidad $O(n^2)$.

Inserción (Insertion Sort) – ordenar_por_nombre(): adecuado para listas cortas o parcialmente ordenadas. Complejidad $O(n^2)$.

Ambos algoritmos se eligieron por su simplicidad conceptual mostrando de esta manera los principios básicos de comparación e intercambio en estructuras de datos.

Documentación De Cada Función

mostrarTitulo()

- Propósito: imprimir título y menú principal.
- Parámetros: ninguno.
- Retorno: None.

registrar_nuevo_curso()

- Propósito: solicitar nombre y nota, validar la nota se encuentre entre el rango permitido (0–100) y agregar (curso, nota) a notas.
- Validaciones: la nota debe ser numérica y estar en [0,100].
- Efecto: modifica notas.

mostrar_cursosYnotas()

- Propósito: mostrar todos los pares almacenados en notas.
- Comportamiento: avisa si notas está vacío.

promedio_de_notas()

- Propósito: calcular y mostrar el promedio de las notas en notas.
- Manejo: imprime del promedio con 2 decimales (preferencias del formato)

contar_aprobadosYreprobados()

Propósito: contar notas >=60 (aprobados) y <60 (reprobados) e imprimir totales.

buscar_curso()

Propósito: buscar un curso por nombre (comparación case-insensitive) e imprimir resultado.

actualizar_nota()

- propósito: localizar un curso, pedir nueva nota (validada entre el rango), registrar el cambio en historial_pila y actualizar notas.
- Efectos: modifica notas y historial_pila

eliminar_curso()

- Propósito: eliminar un curso tras confirmación del usuario y registrar la eliminación en historial_pila.
- Comportamiento: busca por coincidencia case-insensitive y remueve la tupla si existe.

ordenar_por_nota()

Propósito: ordenar notas por el campo numérico de la nota usando burbuja (menor a mayor). Retorna notas ordenada.

ordenar_por_nombre()

Propósito: ordenar notas por el nombre del curso usando inserción alfabética.

cola_revision()

- Propósito: simular una cola de solicitudes de revisión con comandos AGREGAR, REVISAR y FIN.
- Implementación: lista simple con append y pop(0).

historial()

Propósito: imprimir cambios registrados en historial_pila en orden LIFO (más reciente primero).

Bucle principal (menú)

Propósito: ciclo while True que muestra opciones y llama a funciones según la entrada del usuario; incluye opción de salida.

Diagrama / Pseudocódigo Principal

```
INICIO

    Definir opcion Como Entero // Usamos Entero para la opción de 1 a 12

    // Inicialización del título (llamando a la función MostrarTitulo)

    Llamar MostrarTitulo()

    // Bucle principal: MIENTRAS VERDADERO (bucle infinito hasta que se ejecute FIN en la opción
12)

    MIENTRAS VERDADERO HACER

        // Mostrar Menú Completo

        IMPRIMIR "1. REGISTRAR NUEVO CURSO"

        IMPRIMIR "2. MOSTRAR TODOS LOS CURSOS Y NOTAS"

        IMPRIMIR "3. CALCULAR PROMEDIO GENERAL"

        IMPRIMIR "4. CONTAR CURSOS APROBADOS Y REPROBADOS"

        IMPRIMIR "5. BUSCAR CURSO POR NOMBRE"

        IMPRIMIR "6. ACTUALIZAR NOTA DE UN CURSO"

        IMPRIMIR "7. ELIMINAR UN CURSO"

        IMPRIMIR "8. ORDENAR CURSOS POR NOTA"

        IMPRIMIR "9. ORDENAR CURSOS POR NOMBRE"

        IMPRIMIR "10. REVISION DE CURSOS (COLA)"

        IMPRIMIR "11. HISTORIAL DE CAMBIOS (PILA)"

        IMPRIMIR "12. SALIR"

        IMPRIMIR "POR FAVOR, SELECCIONE UNA OPCIÓN

        LEER opcion

        SI opcion == 1 ENTONCES

            Llamar registrar_nuevo_curso()

        SINO SI opcion == 2 ENTONCES

            Llamar mostrar_cursosYnotas()

        SINO SI opcion == 3 ENTONCES
```

```
            Llamar promedio_de_notas()
        SINO SI opcion == 4 ENTONCES
            Llamar contar_aprobadosYreprobados()
        SINO SI opcion == 5 ENTONCES
            Llamar buscar_curso()
        SINO SI opcion == 6 ENTONCES
            Llamar actualizar_nota()
        SINO SI opcion == 7 ENTONCES
            Llamar eliminar_curso()
        SINO SI opcion == 8 ENTONCES
            Llamar ordenar_por_nota()
        SINO SI opcion == 9 ENTONCES
            Llamar ordenar_por_nombre()
        SINO SI opcion == 10 ENTONCES
            Llamar cola_revision()
        SINO SI opcion == 11 ENTONCES
            Llamar historial()
        SINO SI opcion == 12 ENTONCES
            IMPRIMIR "SALIENDO DEL PROGRAMA... ¡HASTA PRONTO!"
            FIN // Terminar ejecución del programa (equivale a 'break' del 'while True')
        SINO
            IMPRIMIR "Error: Opción inválida. Por favor, intente con una opción del 1 al 12."
        FIN_SI

    FIN_MIENTRAS

FIN
```