UNIVERSIDAD MARIANO GÁLVEZ DE GUATEMALA

Nombre: Susana Julissa López Vásquez

Curso: Algoritmos

CATEDRÁTICO: ING. MIGUEL CATALAN

Nombre del proyecto: Gestor de notas académicas

Lenguaje de programación: Python

Guatemala, Octubre 2025

UNIVERSIDAD MARIANO
GÁLVEZ DE GUATEMALA

ALGORITMOS

MANUAL TÉCNICO

GESTOR DE NOTAS ACADÉMICAS

INGENIERÍA EN SISTEMAS

conoceréis la verdad y la verdad os hará libres

Juan 8: 32

Gestor de notas académicas

Descripción técnica general del sistema.
El Gestor de notas académicas es un programa desarrollado en Python, utilizando el entorno de desarrollo Visual Studio.

El sistema funciona completamente en consola, sin emplear interfaces gráficas. Está diseñado para ayudar a un estudiante a registrar, consultar, modificar, eliminar las notas y cursos de forma interactiva.

El programa se controla mediante un menú repetitivo implementado con un ciclo while True, que muestra distintas opciones y se ejecuta hasta que el usuario decide salir.

Cada curso se guarda como un diccionario con el nombre y la nota, dentro de una lista principal llamada cursos. Además se utilizan otras estructuras de datos como pilas y colas para registrar los cambios realizados y simular revisiones.

El código está dividido en funciones modulares, cada una encargada de una tarea específica, se utilizan estructuras if, for, while, validaciones de entradas y algoritmos de búsqueda lineal y binaria también ordenamiento burbuja e inserción para garantizar el correcto funcionamiento del sistema.

Estructura general del código.

El código fuente está organizado así:

cursos = [ ] (Lista principal que guarda todos los cursos y notas)
historial =[ ] (Lista secundaria usada como pila para registrar los cambios realizados registro,actualización y eliminación)
Funciones del sistema
registrar_cursos()
mostrar_curso()
calcular_promedio()
contar_Aprobadas_y_Reprobadas()
Buscar_curso()
Actualizar_nota()
eliminar_curso()
ordenamiento_de_notas_burbuja()
ordenamiento_de_curso_insercion()
busqueda_binaria()
mostrar_historial()
cola_revision()
bienvenida_al_usuario()
menu()

Llama a la funcion bienvenida_al_usuario() para mostrar un mensaje inicial, luego ejecuta menu() para controlar todo el flujo del programa.

Explicación del uso de listas, y pilas, colas

Listas.
En el desarrollo del programa Gestor de Notas se implementaron tres estructuras de datos fundamentales: listas, pilas y colas. Estas estructuras fueron utilizadas para organizar y manipular la información ingresada por el usuario de manera eficiente dentro del sistema.

- Lista principal: cursos
  En el programa, cursos es la estructura central, la cual almacena todos los cursos y notas registradas por el usuario. Cada elemento de esta lista es un diccionario que contiene dos claves:
  “curso” nombre del curso ingresado
  “nota” calificación numérica obtenida.

  Cuando el usuario utiliza la opción REGISTRAR NUEVO CURSO el programa solicita el nombre y la nota, valida los datos y luego ejecuta las siguientes instrucciones:

cursos.append({“curso”: nombre, “nota”: nota})
Esto agrega un nuevo diccionario a la lista cursos, permitiendo así registrar múltiples cursos.

posteriormente, la funcion mostrar_curso() recorre la lista utilizando la instrucción:
for i, item in enumerate(cursos):
    print(f”[{i}] -> {item[‘curso’]} - Notas: {item[nota]}”)
De esta manera, el programa muestra todos los cursos registrados junto con su índice, nombre y nota correspondiente.

El uso de lista resulta adecuado porque permite agregar, recorrer, modificar y eliminar elementos fácilmente, lo cual se aplica en diferentes partes del sistema:

En Actualizar_nota() se modifica el valor de una nota existente.

En eliminar_curso() se remueve un curso con el método .pop(i).

En Calcular_promedio() se suman todas las notas y se divide entre la cantidad total de cursos.

Esta estructura facilita la administración dinámica de los datos sin necesidad de usar archivos o bases de datos externas.

PILA de historial

La lista historial se utiliza como una pila, una estructura de tipo LIFO que significa que el último elemento agregado es el primero en ser mostrado o registrado.

Cada vez que el usuario realiza una accion importante como registrar, actualizar o eliminar un curso—, el sistema guarda una descripción textual de la acción dentro del historial mediante el método .append().
Por ejemplo:

historial.append(f"SE REGISTRO EL CURSO '{nombre}' CON NOTA {nota}")
Mas adelante la funcion mostrar_historial() recorre la lista en orden inverso usando:
for i, evento in enumerate(reversed(historial), start=1):
    print(f"{i}. {evento}")

Esto permite mostrar primero las acciones más recientes, simulando el comportamiento natural de una pila.
El historial sirve como registro de actividades realizadas por el usuario, proporcionando trazabilidad dentro del sistema.

En términos prácticos, la pila historial permite revisar los cambios más recientes y comprender la secuencia de operaciones realizadas durante la sesión.

COLA de revision

La función cola_revision() implementa una cola simulada mediante una lista, donde los cursos se procesan en el mismo orden en que fueron ingresados.
Este tipo de estructura se denomina FIFO (First In, First Out), es decir, el primero en entrar es el primero en salir.

Durante la ejecución, el programa solicita al usuario que ingrese los nombres de los cursos que desea enviar a revisión.
Cada nombre se agrega al final de la cola mediante la instrucción:

```
cola_revision.append(nombre)
```

Cuando el usuario escribe la palabra “fin”, el ciclo termina y se recorren los elementos en orden de ingreso:

```
for curso in cola_revision:
    print(f"EL CURSO '{curso}' ESTA SIENDO REVISADO")
```

Este procedimiento imita el comportamiento de una cola de atención, donde los cursos se atienden en secuencia.

Aunque en este caso se trata de una simulación, la estructura resulta útil para comprender el flujo ordenado de procesamiento en los programas.

Justificación de los algoritmos de ordenamientos

Las estructuras anteriores trabajan en conjunto con diferentes algoritmos implementados dentro del sistema:

Búsqueda lineal:
Se aplica en la función Buscar_curso().
El programa recorre la lista cursos comparando el texto ingresado con los nombres almacenados, permitiendo coincidencias parciales sin distinguir mayúsculas ni minúsculas.

Búsqueda binaria:
Utilizada en busqueda_binaria(), requiere que la lista esté ordenada alfabéticamente.
Permite encontrar un curso más rápidamente dividiendo la lista en mitades

sucesivas hasta hallar la coincidencia.

Ordenamiento burbuja:
Implementado en ordenamiento_de_notas_burbuja(), ordena los cursos de mayor a menor nota comparando pares consecutivos de elementos.

Ordenamiento por inserción:
Aplicado en ordenamiento_de_curso_insercion(), organiza los cursos alfabéticamente por nombre moviendo cada elemento hasta su posición correcta.

Estos algoritmos se apoyan directamente en las listas, demostrando cómo las estructuras básicas del lenguaje permiten desarrollar funcionalidades más avanzadas sin depender de librerías externas.

El uso combinado de listas, pilas y colas dentro del Gestor de Notas Académicas demuestra el manejo de estructuras de datos fundamentales en Python.
Cada una cumple un propósito específico:

La lista cursos centraliza la información principal.

La pila historial conserva el registro de las operaciones recientes.

La cola de revisión modela un proceso de atención secuencial.

Estas estructuras, junto con los algoritmos de búsqueda y ordenamiento, garantizan que el sistema funcione de forma ordenada, eficiente y completamente modular, cumpliendo con los requerimientos del proyecto académico.

Documentación de cada función o módulo

Cursos = []
Esta es la lista principal que almacena todos los cursos y notas registradas, se usa una lista porque permite agregar, recorrer, modificar y eliminar fácilmente los elementos.

historial = []
Esta es la lista secundaria que funciona como pila, aqui se registran las acciones realizadas por el usuario: Registro, actualizaciones y eliminación de cursos.

Registrar_cursos()

Esta función se encarga de registrar cursos y notas.
Primero le pide al usuario que escriba el nombre del curso y después le solicita la nota. si el usuario escribe la palabra fin el registro se detiene y la función termina.

Entrada:
1. Nombre del curso
2. Nota del curso

Proceso técnico
1. Se ejecuta un ciclo while True que permite registrar varios cursos
2. Se solicita el nombre del curso con input()
3. Si el usuario escribe fin se rompe el ciclo
4. Si el nombre está vacío, se muestra un mensaje de error
5. Después de haber ingresado el nombre del curso se pide que ingrese la nota
6. Hace validaciones:
    - Que la nota sea numérica
    - Que esté dentro del rango de 0 a 100
    - El nombre no puede estar vacío
7. Si las validaciones son correctas, se crea un diccionario con el curso y la nota
8. Se guarda en la lista curso y se agrega una descripción al historial .

Salida:

Mensaje confirmando que el curso fue registrado con exito.

```
SELECCIONE UNA OPCION DEL 1 AL 13: 1

INGRESE EL NOMBRE DEL CURSO QUE DESEA REGISTRAR (O ESCRIBA 'FIN' PARA TERMINAR): MICROECONOMIA
INGRESE LA NOTA DEL CURSO (0 A 100): 90
CURSO 'MICROECONOMIA' ¡REGISTRADO CON EXITO!
```

mostrar_curso()

Muestra todos los cursos y notas que han sido registrados

Proceso técnico:
1. Verifica si la lista cursos tiene elementos

2. Si está vacía, muestra un mensaje informando que no hay cursos registrados
3. Si hay cursos, usa enumerate() para mostrar el índice, el nombre y la nota de cada curso

Salida:
Muestra una lista en consola con el indice, curso y nota

```
SELECCIONE UNA OPCION DEL 1 AL 13: 2

CURSOS REGISTRADOS (CON INDICE):
[0] -> microeconomia - Notas: 90
```

calcular_promedio()

Calcula y muestra el promedio general de todas las notas registradas.

Proceso técnico:

- Si no hay cursos registrados (len(cursos) == 0) se imprime un mensaje y se sale con return
- Si hay cursos registrados el programa suma todas las notas y divide el total entre el número de cursos para calcular el promedio.

Salida:
Muestra en consola el promedio general

```
SELECCIONE UNA OPCION DEL 1 AL 13: 3

PROMEDIO GENERAL: 95.00
```

contar_Aprobados_y_Reprobados()

- Cuenta cuantos cursos están aprobados y cuantos estan reprobados

Cursos aprobados ≥ 60
Cursos reprobados < 60

- El programa recorre la lista de cursos y cuenta cuantos tienen una nota ≥ 60 considerándolos como aprobados

Salida:

Muestra un mensaje en consola indicando la cantidad de cursos aprobados y reprobados.

```
SELECCIONE UNA OPCION DEL 1 AL 13: 4

CURSOS APROBADOS: 2
CURSOS REPROBADOS: 0
```

Buscar_curso()

Busca curso usando coincidencias parciales (Búsqueda lineal)

Entrada:
Nombre del curso que desea buscar.

Proceso técnico:

1. Solicita el nombre del curso que desea buscar
2. Convierte el texto para evitar problemas de minusculas y mayusculas
3. Recorre la lista cursos comparando cada elemento con el texto ingresado
4. si encuentra coincidencias las imprime
5. si no muestra mensaje que no se encontro

Validaciones
Si el usuario no ingresó nada muestra: “DEBE DE INGRESAR UN NOMBRE PARA REALIZAR LA BÚSQUEDA”

```
SELECCIONE UNA OPCION DEL 1 AL 13: 5

INGRESE EL NOMBRE DEL CURSO QUE BUSCA: ESTADISTICA
CURSO ENCONTRADO: ESTADISTICA, NOTA: 100
```

```
SELECCIONE UNA OPCION DEL 1 AL 13: 5

INGRESE EL NOMBRE DEL CURSO QUE BUSCA: MICRO
CURSO ENCONTRADO: MICROECONOMIA, NOTA: 90
```

Actualizar_nota()

Modifica la nota de un curso ya existente

Entrada:

- Nombre del curso
- Nota nueva (Numeros dentro del rango 0 a 100)

Proceso técnico:

1. Busca el curso usando comparaciones sin distinguir mayusculas

2. Si lo encuentra, pide que ingrese la nueva nota
3. Valida que la nota sea numérica y este dentro del rango 0 a 100
4. Actualiza la nota en la lista y registra el cambio en historial

Salida:
Muestra un mensaje en consola de confirmación que la nota fue actualizada correctamente.

```
SELECCIONE UNA OPCION DEL 1 AL 13: 6

INGRESE EL NOMBRE DEL CURSO A ACTUALIZAR:  MICROECONOMIA
INGRESE LA NUEVA NOTA (0 A 100): 100
¡NOTA ACTUALIZADA CON EXITO!
```

eliminar_curso()

Elimina el curso registrada con confirmación del usuario

Entrada:
Nombre del curso

Proceso técnico:
1. Recorre la lista y busca coincidencias exactas sin importar mayúsculas
2. Si encuentra el curso pide la confirmación del usuario
3. Si el usuario responde “si” elimina el curso con .pop(i) y lo registra en historial
4. Si responde “no” se cancela la operación

Salida:
Mensaje confirmando la eliminación o cancelación del curso.

```
SELECCIONE UNA OPCION DEL 1 AL 13: 7

INGRESE EL NOMBRE DEL CURSO A ELIMINAR: ESTADISTICA
ESTA SEGURO DE QUE QUIERE ELIMINAR ESTE CURSO 'ESTADISTICA' (SI/NO): NO
ELIMINACION CANCELADA
```

```
SELECCIONE UNA OPCION DEL 1 AL 13: 7

INGRESE EL NOMBRE DEL CURSO A ELIMINAR: ESTADISTICA
ESTA SEGURO DE QUE QUIERE ELIMINAR ESTE CURSO 'ESTADISTICA' (SI/NO): SI
CURSO ELIMINADO CORRECTAMENTE
```

ordenamiento_de_notas_burbuja()

Ordena los cursos de mayor a menor nota con el método burbuja

Proceso técnico:

- Usa dos bucles anidados para recorrer la lista
- Compara pares consecutivos de notas y los intercambia si estan en el orden incorrecto
- Al finalizar imprime los cursos ordenados

```
CURSOS ORDENADOS POR NOTA
1. ESTADISTICA - Nota: 100
2. MICROECONOMIA - Nota: 90
```

ordenamiento_de_curso_insercion()

ordena los cursos alfabéticamente según su nombre con el método de inserción.

Proceso técnico:

- Recorre la lista desde la segunda posición (i = 1)
- Compara el curso actual con los anteriores
- Mueve los elementos hacia la derecha hasta ubicar en su posición correcta

  Salida:

  Muestra en consola la lista ordenada.

```
SELECCIONE UNA OPCION DEL 1 AL 13: 9

CURSOS ORDENADOS POR NOMBRE
1. ALGEBRA - Nota: 90
2. ESTADISTICA - Nota: 100
3. MICROECONOMIA - Nota: 90
```

busqueda_binaria()

Buscar un curso en una lista ordenada alfabéticamente usando búsqueda binaria

Proceso técnico:

- Verificar si la lista cursos tiene elementos
- ordenar los cursos por nombre
- Define los índices inicio y fin

- Calcula el punto medio (medio = (inicio + fin) // 2)
- Compara el curso del medio con el nombre buscado
- Ajusta el rango de búsqueda hasta encontrar el curso o agotar las opciones

Salida:
Muestra el curso encontrado o un mensaje si no existe.

```
SELECCIONE UNA OPCION DEL 1 AL 13: 10

INGRESE EL NOMBRE DEL CURSO A BUSCAR: ALGEBRA
CURSO ENCONTRADO: ALGEBRA - Nota: 90
```

mostra_historial()

Muestra todas las acciones registradas en la pila historial de la más reciente a la más antigua.

Proceso técnico:

- La función verifica si la lista historial está vacia
- Si hay registros después de registrar, actualizar o eliminar cursos recorre el historial en orden inverso

Salida:
Si no hay registros imprime “HISTORIAL VACIO”
Si hay acciones realizadas muestra en consola.

```
HISTORIAL DE CAMBIOS
1. SE REGISTRO EL CURSO 'ALGEBRA' CON NOTA 90
2. SE REGISTRO EL CURSO 'ESTADISTICA' CON NOTA 100
3. SE ELIMINO: MICROECONOMIA - Nota: 100
4. SE REGISTRO EL CURSO 'MICROECONOMIA' CON NOTA 90
5. SE ELIMINO: ESTADISTICA - Nota: 100
6. SE ACTUALIZO: MICROECONOMIA - Nota ANTERIOR: 90 NOTA NUEVA: 100
7. SE REGISTRO EL CURSO 'ESTADISTICA' CON NOTA 100
8. SE REGISTRO EL CURSO 'MICROECONOMIA' CON NOTA 90
```

cola_revision()

Simula una cola de atención para procesar solicitudes de revisión de cursos.
Esta función representa el funcionamiento de una estructura de datos tipo cola donde el primer curso que se agrega es el primero en ser atendido

Proceso técnico:

- Se crea una lista vacía llamada cola_revision = [ ], que actúa como la cola donde se almacenarán los cursos a revisar.
- El programa entra en un bucle while True donde se le solicita al usuario que escriba los nombres de los cursos, cada nombre ingresado se agrega al final de la cola
- Si el usuario escribe fin el ciclo se detiene y se procede a procesar la cola.

Salida:
Al finalizar la revisión de todos los elementos el programa imprime.

```
SELECCIONE UNA OPCION DEL 1 AL 13: 12

INGRESE LOS CURSOS QUE DESEA REVISAR
ESCRIBA 'fin' PARA TERMINAR.

CURSOS A REVISAR: ALGEBRA
CURSOS A REVISAR: ESTADISTICA
CURSOS A REVISAR: MICROECONOMIA
CURSOS A REVISAR: FIN

PROCESANDO SOLICITUD DE REVISION
EL CURSO 'ALGEBRA' ESTA SIENDO REVISADO
EL CURSO 'ESTADISTICA' ESTA SIENDO REVISADO
EL CURSO 'MICROECONOMIA' ESTA SIENDO REVISADO
TODAS LAS SOLICITUDES HAN SIDO ATENDIDAS
```

Salir (Finalización del programa)

Permite al usuario salir del programa de forma segura, mostrando un mensaje de despedida y finalizando la ejecución del menú principal.

```
SELECCIONE UNA OPCION DEL 1 AL 13: 13

¡GRACIAS POR USAR EL GESTOR DE NOTAS ACADEMICAS HASTA PRONTO!
```

Diagrama general del sistema o pseudocódigo principal

```
INICIO

cursos <- [ ]
historial <- [ ]
bienvenida_al_usuario()

MIENTRAS VERDADERO HACER
   MOSTRAR “=====GESTOR DE NOTAS ACADEMICAS 2025 =====”
   MOSTRAR “1. REGISTRAR NUEVO CURSO”
   MOSTRAR “2. MOSTRAR TODOS LOS CURSOS Y NOTAS”
   MOSTRAR “3. CALCULAR PROMEDIO GENERAL”
   MOSTRAR “4. CONTAR LOS CURSOS APROBADOS Y REPROBADOS”
   MOSTRAR “5. ”BUSCAR CURSO POR NOMBRE (BUSQUEDA LINEAL)
   MOSTRAR  “6. ACTUALIZAR NOTA DE UN CURSO”
   MOSTRAR  “7. ELIMINAR UN CURSO”
   MOSTRAR  “8. ORDENAR CURSO POR NOTA (BURBUJA)”
   MOSTRAR  “9- ORDENAR CURSOS POR NOMBRE (INSERCION)”
   MOSTRAR  “10. BUSCAR CURSO POR NOMBRE (BUSQUEDA BINARIA)”
   MOSTRAR  “11. MOSTRAR HISTORIAL DE CAMBIOS”
   MOSTRAR  “12. SOLICITAR REVISION DE CURSOS”
   MOSTRAR  “13. SALIR”

   LEER OPCION
   SI opcion == 1 ENTONCES
      registrar_cursos()
   SINO SI opcion == 2 ENTONCES
      mostrar_curso()
   SINO SI opcion == 3 ENTONCES
      Calcular_promedio()
   SINO SI opcion == 4 ENTONCES
      Contar_Aprobadas_y_Reprobadas()
   SINO SI opcion == 5 ENTONCES
      Buscar_curso()
   SINO SI opcion == 6 ENTONCES
      Actualizar_nota()
   SINO SI opcion == 7 ENTONCES
      eliminar_curso()
   SINO SI opcion == 8 ENTONCES
      ordenamiento_de_notas_burbuja()
   SINO SI opcion == 9 ENTONCES
      ordenamiento_de_curso_insercion()
   SINO SI opcion == 10 ENTONCES
```

```
        busqueda_binaria()
    SINO SI opcion == 11 ENTONCES
        mostrar_historial()
    SINO SI opcion == 12 ENTONCES
        cola_revision()
    SINO SI opcion == 13 ENTONCES
        MOSTRAR "Gracias por usar el Gestor de Notas Académicas. ¡Hasta pronto!"
        SALIR DEL PROGRAMA
    SINO
        MOSTRAR "Opción inválida. Intente nuevamente."
    FIN SI

FIN MIENTRAS

FIN
```