



unedl
Universidad Enrique Díaz de León

¡Innovamos por ti!

Inteligencia Artificial

-----Sistema Experto-----

**Ingeniería en Software
7º Semestre T/ Matutino Grupo:A**

Alumno:Jose Enrique Fernández Ortega

Introducción:

Este proyecto trata de un sistema experto hecho en python, que tiene una base de conocimiento, hechos(calificaciones), motor de inferencia(reglas que vamos a seguir) y usando la librería "Experta" (que permite crear sistemas expertos en Python de manera eficiente, utilizando hechos y reglas) que nos ayuda a proporcionar recomendaciones estudiantes basadas en sus calificaciones.

Guía:

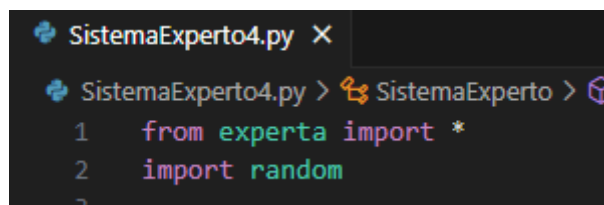
Software Sistema Experto de recomendaciones de calificaciones

Código por partes:

Librerías:

Experta: es una biblioteca de Python que facilita la creación de sistemas expertos. Permite definir reglas y hechos, y luego utilizar un motor de inferencia para tomar decisiones basadas en estas reglas

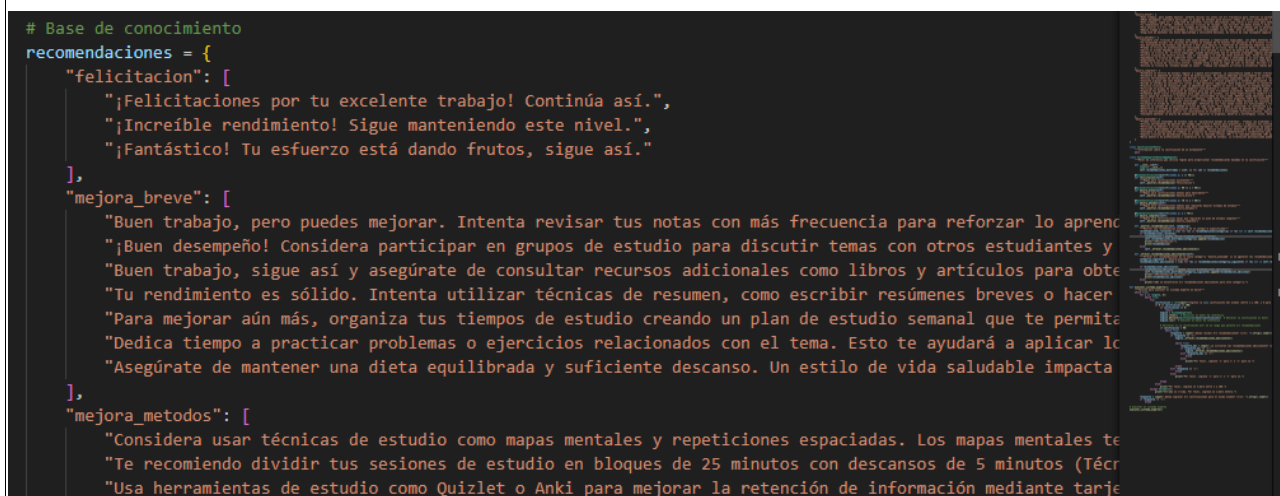
Random: es una biblioteca estándar de Python que proporciona herramientas para generar números aleatorios y realizar selecciones aleatorias. En el contexto del sistema experto, se utiliza para elegir recomendaciones de manera aleatoria para evitar respuestas repetitivas



```
SistemaExperto4.py X
SistemaExperto4.py > SistemaExperto >
1 from experta import *
2 import random
3
```

Base de conocimientos:

Es el repositorio donde se almacena la información y las reglas que el sistema utiliza para tomar decisiones y proporcionar recomendaciones. Esta información puede incluir datos, hechos, heurísticas, y reglas derivadas de la experiencia de expertos en un dominio específico.



```
# Base de conocimiento
recomendaciones = {
    "felicitacion": [
        "¡Felicitaciones por tu excelente trabajo! Continúa así.",
        "¡Increíble rendimiento! Sigue manteniendo este nivel.",
        "¡Fantástico! Tu esfuerzo está dando frutos, sigue así."
    ],
    "mejora_breve": [
        "Buen trabajo, pero puedes mejorar. Intenta revisar tus notas con más frecuencia para reforzar lo aprendido.",
        "¡Buen desempeño! Considera participar en grupos de estudio para discutir temas con otros estudiantes y reforzar tu comprensión.",
        "Buen trabajo, sigue así y asegúrate de consultar recursos adicionales como libros y artículos para obtener una perspectiva más amplia.",
        "Tu rendimiento es sólido. Intenta utilizar técnicas de resumen, como escribir resúmenes breves o hacer tarjetas de memoria.",
        "Para mejorar aún más, organiza tus tiempos de estudio creando un plan de estudio semanal que te permita mantener la consistencia.",
        "Dedica tiempo a practicar problemas o ejercicios relacionados con el tema. Esto te ayudará a aplicar lo que has aprendido.",
        "Asegúrate de mantener una dieta equilibrada y suficiente descanso. Un estilo de vida saludable impacta positivamente en tu rendimiento."
    ],
    "mejora_metodos": [
        "Considera usar técnicas de estudio como mapas mentales y repeticiones espaciadas. Los mapas mentales te ayudan a visualizar la información.",
        "Te recomiendo dividir tus sesiones de estudio en bloques de 25 minutos con descansos de 5 minutos (Técnica Pomodoro).",
        "Usa herramientas de estudio como Quizlet o Anki para mejorar la retención de información mediante tarjetas de memoria."
    ]
}
```

Hechos:

Son unidades de información básica en un sistema experto. Representan datos específicos sobre el problema o el dominio en cuestión. Los hechos pueden ser tanto datos estáticos, como conocimientos generales del dominio, o datos dinámicos, como información específica del caso que se está analizando.

En nuestro caso serán las calificaciones

```
class Calificacion(Fact):  
    """Información sobre la calificación de un estudiante"""  
    pass
```

Motor de inferencia:

Un motor de inferencia es el componente central de un sistema experto que procesa la base de conocimientos (hechos y reglas) para derivar conclusiones o tomar decisiones. Actúa como un mecanismo de razonamiento, aplicando las reglas a los hechos conocidos para generar nuevos hechos o acciones.

```
class SistemaExperto(KnowledgeEngine):  
    """Motor de inferencia que utiliza reglas para proporcionar recomendaciones basadas en la calificación"""  
  
    def __init__(self):  
        super().__init__()  
        self.recomendaciones_mostradas = {cat: [] for cat in recomendaciones}  
  
    @Rule(Calificacion(valor=P(lambda x: x >= 90)))  
    def felicitacion(self):  
        """Regla para calificaciones excelentes"""  
        self._mostrar_recomendacion('felicitacion')  
  
    @Rule(Calificacion(valor=P(lambda x: 80 <= x < 90)))  
    def mejora_breve(self):  
        """Regla para calificaciones buenas pero mejorables"""  
        self._mostrar_recomendacion('mejora_breve')  
  
    @Rule(Calificacion(valor=P(lambda x: 70 <= x < 80)))  
    def mejora_metodos(self):  
        """Regla para calificaciones medias que requieren mejorar métodos de estudio"""  
        self._mostrar_recomendacion('mejora_metodos')  
  
    @Rule(Calificacion(valor=P(lambda x: x < 70)))  
    def mejora_completa(self):
```

Función para ejecutar el sistema experto:

Con esta estructura, el código se organiza de manera clara y facilita la comprensión de cada componente del sistema experto.

```
def ejecutar_sistema_experto():
    """Función para ejecutar el sistema experto en bucle"""
    while True:
        for i in range(1, 6):
            while True:
                try:
                    calificacion = int(input(f"Ingrese la {i}ª calificación del alumno (entre 1 y 100, o 0 para salir): "))
                    if 0 <= calificacion <= 100:
                        if calificacion == 0:
                            return
                        engine = SistemaExperto()
                        engine.reset() # Reiniciar el motor de inferencia
                        engine.declare(Calificacion(valor=calificacion)) # Declarar la calificación al motor
                        engine.run() # Ejecutar el motor de inferencia

                        # Verificar si la calificación está en un rango que permite más recomendaciones
                        if calificacion < 90:
                            while True:
                                respuesta = input("¿Desea recibir más recomendaciones? (s/n): ").strip().lower()
                                if respuesta == 's':
                                    engine._ofrecer_recomendaciones_adicionales()

                                    while True:
                                        respuesta_mas = input("¿Le sirvieron las recomendaciones adicionales? (s/n): ")
                                        if respuesta_mas == 'n':
                                            break
                                        elif respuesta_mas == 's':
                                            engine._ofrecer_recomendaciones_adicionales()
                                else:
                                    break
                            else:
                                break
                    else:
                        print("Calificación no válida. Intente de nuevo.")
                except ValueError:
                    print("Error: Ingrese un número válido.")
```

Funcionamiento del sistema en general:

Este sistema experto está diseñado para proporcionar recomendaciones a estudiantes basadas en sus calificaciones. Utiliza el módulo “**experta**” para aplicar reglas basadas en el valor de las calificaciones ingresadas y ofrecer recomendaciones adaptadas a la situación académica del estudiante.

Componentes del Sistema:

Base de Conocimiento: La base de conocimiento contiene recomendaciones agrupadas en diferentes categorías:

Felicitación: Para calificaciones sobresalientes.

Mejora Breve: Para calificaciones buenas pero con margen de mejora.

Mejora Métodos: Para calificaciones medias que requieren mejorar métodos de estudio.

Mejora Completa: Para calificaciones bajas que requieren un plan de estudio completo.

Mejora Avanzada: Recomendaciones adicionales cuando las anteriores se han agotado.

Clases:

Calificación: Define la información sobre la calificación de un estudiante.

SistemaExperto: Motor de inferencia que aplica reglas para proporcionar recomendaciones

Reglas Aplicadas:

`felicitation()`: Para calificaciones ≥ 90 .

`mejora_breve()`: Para calificaciones entre 80 y 89.

`mejora_metodos()`: Para calificaciones entre 70 y 79.

`mejora_completa()`: Para calificaciones < 70 .

Métodos usados:

`_mostrar_recomendacion(categoria)`: Muestra una recomendación aleatoria de una categoría dada.

`_ofrecer_recomendaciones_adicionales()`: Ofrece recomendaciones de la categoría `mejora_avanzada` si se han agotado las recomendaciones anteriores.

Decisiones tomadas:

En el desarrollo del sistema experto de calificaciones abarcaron desde la definición del problema y la elección de herramientas hasta la implementación de reglas y la interacción con el usuario. Estas decisiones aseguraron la creación del sistema, que sea flexible y capaz de proporcionar recomendaciones útiles y variadas basadas en las calificaciones de los estudiantes.

Las decisiones fueron las siguientes:

1. Definición del Problema y dominio

- **Objetivo:** Crear un sistema experto que proporcione recomendaciones basadas en las calificaciones de un estudiante.
- **Dominio:** Calificaciones académicas y estrategias de estudio

2. Elección de la Herramienta de Desarrollo

- **Librería:** Se decidió utilizar `experta`, una biblioteca de Python para la creación de sistemas expertos.
- **Justificación:** `experta` proporciona una estructura clara para definir hechos y reglas, y un motor de inferencia para aplicar estas reglas, lo que simplifica el desarrollo del sistema experto.

3. Definición de Hechos

- **Hecho Principal:** La calificación del estudiante.

4. Definición de Reglas

- ⑩ Se establecieron reglas para diferentes rangos de calificaciones.

5.Implementación del Motor de Inferencia

- ⑩ El motor de inferencia se encarga de procesar las calificaciones y aplicar las reglas para proporcionar recomendaciones

6.Interacción con el Usuario

Se diseñó una interfaz de entrada que permite al usuario ingresar múltiples calificaciones y recibir recomendaciones correspondientes

Objetivos Alcanzados

Personalización de Recomendaciones: Se han implementado recomendaciones específicas basadas en el rango de calificaciones del estudiante.

Ampliación de Recomendaciones: Se han añadido recomendaciones avanzadas para ofrecer más opciones cuando se agotan las anteriores.

```
"mejora_avanzada": [  
    "Prueba técnicas avanzadas de estudio como el 'aprendizaje basado en problemas'. Trabaja con problemas c  
    "Utiliza herramientas de gestión del tiempo como Trello o Notion para planificar y organizar tus activid  
    "Participa en webinars y cursos en línea para complementar tus estudios y obtener perspectivas de expert  
    "Desarrolla habilidades de autoevaluación avanzadas. Realiza evaluaciones detalladas de tu propio progre  
    "Considera la posibilidad de trabajar en proyectos de investigación independientes o colaborativos para  
    "Explora técnicas de aprendizaje inmersivo como la realidad virtual (VR) o la realidad aumentada (AR) pa  
    "Busca mentoría de profesionales o académicos en tu campo de estudio. La orientación personalizada puede
```

Interactividad: Se permite al usuario solicitar más recomendaciones y validar su utilidad, proporcionando una experiencia más dinámica.

Desafíos Enfrentados

Diversidad en Recomendaciones: Asegurar que las recomendaciones sean variadas y útiles en función del rango de calificaciones.

Gestión de Recomendaciones Agotadas: Implementar una estrategia para ofrecer recomendaciones adicionales cuando las recomendaciones principales se han agotado

```
def _ofrecer_recomendaciones_adicionales(self):  
    """Ofrecer más recomendaciones de la nueva categoría 'mejora_avanzada' si se agotaron las recomendaciones  
    categoria_siguiente = 'mejora_avanzada'  
    recomendaciones_adicionales = [rec for rec in recomendaciones[categoria_siguiente] if rec not in self.recomendaciones_mostradas]  
  
    if recomendaciones_adicionales:  
        recomendacion_adicional = random.choice(recomendaciones_adicionales)  
        self.recomendaciones_mostradas[categoria_siguiente].append(recomendacion_adicional)  
        print("\nRecomendación adicional:")  
        print(recomendacion_adicional)
```

Validación de Entradas: Manejar entradas inválidas o fuera de rango para evitar errores durante la ejecución.

Aprendizajes Obtenidos

Aprendí a utilizar el módulo “experta” para construir un sistema experto y aplicar reglas de inferencia y a instalarla desde cmd.

La Incorporación de una base de conocimiento que es la que va alimentar las sugerecia del motor de inferencia

Aplicar la reglas en el motor de inferencia para que de las recomendaciones según la calificación.

Aplicaciones Potenciales

Herramienta de Apoyo Académico: Para proporcionar retroalimentación y recomendaciones personalizadas a los estudiantes.

Educación Personalizada: Ofrecer recomendaciones que el estudiantes no habría tenido en cuenta y que sirvan en diferentes niveles educativos, para usar una variedad de métodos de estudios que le ayude a subir la calificación.

Conclusión:

Con este sistema experto queremos llegar a que los estudiantes tenga una forma de aumentar su rendimiento académico, mediante recomendaciones personalizadas en sus calificaciones donde tengan algún problema.

Código completo:

```
from experta import *
import random

# Base de conocimiento
recomendaciones = {
    "felicitacion": [
        "¡Felicitaciones por tu excelente trabajo! Continúa así.",
        "¡Increíble rendimiento! Sigue manteniendo este nivel.",
        "¡Fantástico! Tu esfuerzo está dando frutos, sigue así."
    ],
    "mejora_breve": [
        "Buen trabajo, pero puedes mejorar. Intenta revisar tus notas con más frecuencia para reforzar lo aprendido y evitar olvidos.",
        "¡Buen desempeño! Considera participar en grupos de estudio para discutir temas con otros estudiantes y afianzar tu conocimiento a través del intercambio de ideas.",
        "Buen trabajo, sigue así y asegúrate de consultar recursos adicionales como libros y artículos para obtener perspectivas diferentes y una comprensión más profunda.",
        "Tu rendimiento es sólido. Intenta utilizar técnicas de resumen, como escribir resúmenes breves o hacer esquemas, para consolidar mejor la información y facilitar la revisión.",
        "Para mejorar aún más, organiza tus tiempos de estudio creando un plan de estudio semanal que te permita abordar todos los temas de manera equilibrada y sistemática.",
        "Dedica tiempo a practicar problemas o ejercicios relacionados con el tema. Esto te ayudará a aplicar lo que has aprendido y a identificar áreas que necesitan más atención.",
        "Asegúrate de mantener una dieta equilibrada y suficiente descanso. Un estilo de vida saludable impacta positivamente en tu capacidad de concentración y rendimiento académico."
    ],
    "mejora_metodos": [
        "Considera usar técnicas de estudio como mapas mentales y repeticiones espaciadas. Los mapas mentales te ayudan a organizar y conectar información visualmente, mientras que las repeticiones espaciadas mejoran la retención a largo plazo. Dedica al menos 1 hora diaria al estudio para aplicar estas técnicas de manera efectiva.",
        "Te recomiendo dividir tus sesiones de estudio en bloques de 25 minutos con descansos de 5 minutos (Técnica Pomodoro). Esto ayuda a mantener la concentración y evitar el agotamiento mental al estudiar en intervalos cortos y gestionados.",
        "Usa herramientas de estudio como Quizlet o Anki para mejorar la retención de información mediante tarjetas de memoria digitales y repeticiones programadas. Intenta estudiar en un ambiente libre de distracciones para maximizar la eficacia del estudio.",
        "Prueba la técnica de autoevaluación realizando exámenes de práctica. Esto te permite medir tu comprensión del material y ajustar tu enfoque de estudio en función de las áreas que necesiten más atención.",
        "Aplica el método Feynman: enseña el material a otra persona o escríbelo de manera que puedas entenderlo fácilmente. Este enfoque te obliga a simplificar y clarificar tus conocimientos, ayudando a identificar lagunas en tu comprensión.",
        "Integra la técnica de 'práctica espaciada' usando aplicaciones que te permitan programar repasos periódicos de la información, lo que favorece la retención a largo plazo.",
        "Utiliza técnicas de lectura activa, como el subrayado y la toma de notas mientras lees. Estas prácticas mantienen tu mente involucrada y facilitan la comprensión y retención del material."
```


"Implementa la técnica de 'inmersión total', dedicando sesiones largas y enfocadas en un solo tema para profundizar en el conocimiento antes de cambiar a otro tema. Esto te ayuda a comprender el material a fondo.",

"Experimenta con la técnica de 'revisión intercalada', alternando entre diferentes materias o temas durante las sesiones de estudio. Esto ayuda a mejorar la transferencia de aprendizaje y la aplicación de conocimientos en contextos diversos.",

"Prueba la técnica de 'escritura activa', tomando notas a mano en lugar de usar un teclado. Escribir a mano favorece la retención y comprensión del material debido a la mayor implicación cognitiva.",

"Utiliza la técnica de 'estudio basado en casos'. Trabaja con ejemplos prácticos y escenarios reales para aplicar lo que has aprendido, lo que te permite ver la relevancia y aplicación práctica del material."

],

"mejora_completa": [

"Establece un horario de estudio regular y síguelo estrictamente. La consistencia ayuda a formar hábitos de estudio y asegura que cubras todos los temas necesarios de manera oportuna.",

"Encuentra un ambiente de estudio libre de distracciones. Un espacio dedicado exclusivamente al estudio te permitirá concentrarte mejor y ser más productivo.",

"Utiliza herramientas de estudio como Quizlet o Anki para repeticiones espaciadas, que te ayudarán a recordar la información a largo plazo mediante revisiones programadas.",

"Consulta páginas web como Google Scholar para acceder a artículos académicos y estudios adicionales. Esto te permitirá obtener una comprensión más profunda y actualizada de los temas que estás estudiando.",

"Divide el estudio en sesiones más cortas y toma descansos regulares. Esto ayuda a mantener la concentración y evita la fatiga mental, mejorando tu capacidad para retener información.",

"Practica técnicas de relajación como la meditación para reducir el estrés y mejorar tu enfoque durante el estudio. La reducción del estrés puede aumentar tu capacidad para procesar y recordar información.",

"Participa en grupos de estudio para compartir conocimientos y resolver dudas. El intercambio de ideas y la discusión con otros estudiantes pueden proporcionar nuevas perspectivas y ayudar a consolidar el aprendizaje.",

"Organiza tus apuntes y materiales de estudio para que sean fácilmente accesibles. Mantener todo bien organizado te ayudará a encontrar rápidamente la información cuando la necesites.",

"Establece metas claras y alcanzables para cada sesión de estudio. Las metas específicas te ayudarán a mantener el enfoque y medir tu progreso, haciendo que el estudio sea más estructurado y efectivo.",

"Busca apoyo de profesores o tutores si tienes dificultades en algún tema. Obtener ayuda adicional puede aclarar dudas y proporcionar orientación para superar obstáculos en tu aprendizaje.",

"Utiliza la técnica de la 'lectura activa'. Subraya, toma notas y formula preguntas sobre el contenido para mantenerte comprometido y mejorar la comprensión y retención del material.",

"Integra la técnica de 'espaciado intercalado', alternando entre diferentes temas durante las sesiones de estudio. Esto mejora la transferencia de aprendizaje y te prepara para aplicar conocimientos en contextos diversos.",

"Incorpora el método de 'práctica deliberada'. Enfócate en las áreas donde tienes más dificultades y trabaja intensamente para mejorar esas habilidades específicas, maximizando tu progreso.",

"Realiza un resumen de lo aprendido al final de cada sesión. Esto ayuda a consolidar la información y a identificar áreas que pueden necesitar repaso adicional.",

"Desarrolla un sistema de organización personal, como una agenda o aplicación de gestión de tareas, para planificar y hacer seguimiento de tus estudios. Esto te ayudará a mantenerte en camino y asegurarte de cubrir todos los temas necesarios.",

"Busca oportunidades para aplicar tus conocimientos en proyectos o actividades extracurriculares. Aplicar lo aprendido en contextos prácticos refuerza tu comprensión y te brinda experiencia valiosa.",

"Considera mantener un diario de estudio para registrar tu progreso, desafíos y estrategias útiles. Esto te permitirá reflexionar sobre tus métodos de estudio y hacer ajustes para mejorar tu enfoque."

],

"mejora_avanzada": [

"Prueba técnicas avanzadas de estudio como el 'aprendizaje basado en problemas'. Trabaja con problemas complejos y desafiantes para mejorar tus habilidades de resolución de problemas y aplicación práctica del conocimiento.",

"Utiliza herramientas de gestión del tiempo como Trello o Notion para planificar y organizar tus actividades académicas y personales. Estas herramientas te ayudarán a mantenerte organizado y enfocado en tus objetivos.",

"Participa en webinars y cursos en línea para complementar tus estudios y obtener perspectivas de expertos en el campo. Esta exposición adicional puede mejorar tu comprensión y proporcionar nuevas ideas.",

"Desarrolla habilidades de autoevaluación avanzadas. Realiza evaluaciones detalladas de tu propio progreso y identifica áreas específicas donde puedes mejorar.",

"Considera la posibilidad de trabajar en proyectos de investigación independientes o colaborativos para aplicar y profundizar tus conocimientos en áreas de interés particular.",

"Explora técnicas de aprendizaje inmersivo como la realidad virtual (VR) o la realidad aumentada (AR) para un aprendizaje interactivo y envolvente en materias específicas.",

"Busca mentoría de profesionales o académicos en tu campo de estudio. La orientación personalizada puede proporcionarte una visión valiosa y ayudarte a alcanzar tus metas académicas y profesionales."

]

}

```
class Calificacion(Fact):
```

```
    """Información sobre la calificación de un estudiante"""
```

```
    pass
```

```
class SistemaExperto(KnowledgeEngine):
```

```
    """Motor de inferencia que utiliza reglas para proporcionar recomendaciones basadas en la calificación"""
```

```
    def __init__(self):
```

```
        super().__init__()
```

```
        self.recomendaciones_mostradas = {cat: [] for cat in recomendaciones}
```

```
    @Rule(Calificacion(valor=P(lambda x: x >= 90)))
```

```
    def felicitacion(self):
```

```
        """Regla para calificaciones excelentes"""
```

```
        self._mostrar_recomendacion('felicitacion')
```

```
    @Rule(Calificacion(valor=P(lambda x: 80 <= x < 90)))
```

```
    def mejora_breve(self):
```

```
        """Regla para calificaciones buenas pero mejorables"""
```

```
        self._mostrar_recomendacion('mejora_breve')
```

```
    @Rule(Calificacion(valor=P(lambda x: 70 <= x < 80)))
```

```

def mejora_metodos(self):
    """Regla para calificaciones medias que requieren mejorar métodos de estudio"""
    self._mostrar_recomendacion('mejora_metodos')

@Rule(Calificacion(valor=P(lambda x: x < 70)))
def mejora_completa(self):
    """Regla para calificaciones bajas que requieren un plan de estudio completo"""
    self._mostrar_recomendacion('mejora_completa')

def _mostrar_recomendacion(self, categoria):
    """Mostrar una recomendación aleatoria de la categoría especificada"""
    recomendaciones_restantes = [rec for rec in recomendaciones[categoria] if rec not in
self.recomendaciones_mostradas[categoria]]
    if recomendaciones_restantes:
        recomendacion = random.choice(recomendaciones_restantes)
        self.recomendaciones_mostradas[categoria].append(recomendacion)
        print("\nRecomendación:")
        print(recomendacion)
    else:
        self._ofrecer_recomendaciones_adicionales()

def _ofrecer_recomendaciones_adicionales(self):
    """Ofrecer más recomendaciones de la nueva categoría 'mejora_avanzada' si se agotaron las
recomendaciones de las categorías anteriores"""
    categoria_siguiente = 'mejora_avanzada'
    recomendaciones_adicionales = [rec for rec in recomendaciones[categoria_siguiente] if rec
not in self.recomendaciones_mostradas[categoria_siguiente]]
    if recomendaciones_adicionales:
        recomendacion_adicional = random.choice(recomendaciones_adicionales)
        self.recomendaciones_mostradas[categoria_siguiente].append(recomendacion_adicional)
        print("\nRecomendación adicional:")
        print(recomendacion_adicional)
    else:
        print("\nNo se encontraron más recomendaciones adicionales para esta categoría.")

def ejecutar_sistema_experto():
    """Función para ejecutar el sistema experto en bucle"""
    while True:
        for i in range(1, 6):
            while True:
                try:
                    calificacion = int(input(f"Ingrese la {i}ª calificación del alumno (entre 1 y 100, o 0
para finalizar): "))
                    if 0 <= calificacion <= 100:
                        if calificacion == 0:
                            return
                        engine = SistemaExperto()
                        engine.reset() # Reiniciar el motor de inferencia
                        engine.declare(Calificacion(valor=calificacion)) # Declarar la calificación al motor
                        engine.run() # Ejecutar el motor de inferencia

```

```

        # Verificar si la calificación está en un rango que permite más recomendaciones
        if calificacion < 90:
            while True:
                respuesta = input("¿Desea recibir más recomendaciones? (s/n):")
                respuesta = respuesta.strip().lower()
                if respuesta == 's':
                    engine._ofrecer_recomendaciones_adicionales()
                    while True:
                        respuesta_mas = input("¿Le sirvieron las recomendaciones adicionales? (s/n): ")
                        respuesta_mas = respuesta_mas.strip().lower()
                        if respuesta_mas == 'n':
                            engine._ofrecer_recomendaciones_adicionales()
                        elif respuesta_mas == 's':
                            break
                        else:
                            print("Por favor, ingrese 's' para sí o 'n' para no.")
                    break
                elif respuesta == 'n':
                    break
                else:
                    print("Por favor, ingrese 's' para sí o 'n' para no.")
            break
        else:
            print("Por favor, ingrese un número entre 1 y 100.")
    except ValueError:
        print("Entrada no válida. Por favor, ingrese un número entero.")

    respuesta = input("¿Desea ingresar más calificaciones para el mismo alumno? (s/n):")
    respuesta = respuesta.strip().lower()
    if respuesta != 's':
        break

# Ejecutar el sistema experto
ejecutar_sistema_experto()

```