Guía de Debugging con IA para Principiantes

🎯 Introducción

Esta guía te ayudará a resolver problemas en tu código usando herramientas de IA como ChatGPT, Claude, Gemini o Perplexity. Está diseñada especialmente para estudiantes que están comenzando a programar.
Ut

🏷️ Índice

1. Cuando el código no funciona

Paso 1: Recolectar la información

Antes de preguntar a la IA, junta:

- El código completo que no funciona
- El mensaje de error (si hay uno)
- Lo que esperabas que hiciera el código
- Lo que está haciendo en realidad

Paso 2: Estructurar tu pregunta

```
Tengo este código:
[PEGA TU CÓDIGO AQUÍ]

Cuando lo ejecuto, recibo este error:
[PEGA EL ERROR AQUÍ]

Lo que quiero que haga es:
[EXPLICA QUÉ DEBERÍA HACER EL CÓDIGO]
```

PROF

Paso 3: Preguntas útiles para la IA

- "¿Puedes explicarme qué significa este error?"
- "¿Puedes revisar mi código línea por línea y decirme dónde está el problema?"
- "¿Qué debo cambiar para que funcione?"

2. Cuando necesitas crear código nuevo

Paso 1: Describir el contexto

Explica a la IA:

- La tarea o ejercicio que necesitas resolver
- El lenguaje de programación que debes usar
- Si tienes algún código base o ejemplo de clase

Paso 2: Estructurar tu solicitud

```
Necesito crear un programa que:
[DESCRIBE LO QUE NECESITAS]

Debe ser en [LENGUAJE DE PROGRAMACIÓN]

En clase vimos este ejemplo similar:
[CÓDIGO DE EJEMPLO SI LO TIENES]
```

Paso 3: Preguntas para mejorar

- "¿Puedes explicarme cómo funciona cada parte del código?"
- "¿Cómo puedo modificar este código para [nuevo requisito]?"
- "¿Hay una forma más simple de hacer esto?"

3. Mejores Prácticas y Consejos

🎯 Sé específico

MAL: "Mi código no funciona"
BIEN: "Mi código de Python para calcular el promedio de una lista no funciona cuando la lista está vacía"

📝 Proporciona contexto

PROF

- Menciona qué versión del lenguaje usas
- Di qué editor o entorno de desarrollo usas
- Explica si es una tarea de clase o un proyecto personal

🔍 Aprende del proceso

- No solo copies y pegues la solución
- Pide explicaciones de por qué el error ocurrió
- Pregunta cómo evitar errores similares en el futuro

4. Ejemplos Prácticos

Ejemplo 1: Código que da error

```
```

```
PREGUNTA:
Tengo este código en Python:
numbers = [1, 2, 3, 4, 5]
for i in range(len(numbers)):
    print(numbers[i+1])

Me da este error:
IndexError: list index out of range

Debería imprimir todos los números de la lista.

RESPUESTA DE LA IA:
El error ocurre porque estás intentando acceder a una posición fuera de
la lista.
Veamos el problema:
1. Tu lista tiene índices del 0 al 4
2. En el loop, i va de 0 a 4
3. Cuando intentas numbers[i+1], buscarás las posiciones 1 a 5
4. ¡Pero la posición 5 no existe!

Solución:
numbers = [1, 2, 3, 4, 5]
for i in range(len(numbers)):
    print(numbers[i])
```

Ejemplo 2: Crear código nuevo

```
PREGUNTA:
Necesito hacer un programa en Python que:
- Pida al usuario su nombre
- Pida su edad
- Diga si es mayor de edad (18 años o más)

En clase vimos cómo pedir datos así:
name = input("Como te llamas?")

RESPUESTA DE LA IA:
Aquí tienes un programa basado en lo que viste en clase:

nombre = input("¿Cómo te llamas? ")
edad_texto = input("¿Cuántos años tienes? ")
edad = int(edad_texto)  # Convertimos el texto a número

if edad >= 18:
    print(f"Hola {nombre}, eres mayor de edad")
else:
    print(f"Hola {nombre}, eres menor de edad")
```

PROF

🚫 Errores comunes al usar IA para debugging

1. No proporcionar suficiente información
    - La IA no puede adivinar tu entorno o contexto
    - Siempre menciona el lenguaje y versión que usas
2. Copiar y pegar sin entender
    - Pide explicaciones de la solución
    - Pregunta qué causó el error original
3. No verificar las soluciones
    - La IA puede cometer errores
    - Siempre prueba el código que te da

🌟 Consejos finales

1. Aprende de los errores
    - Guarda las soluciones que te funcionaron
    - Crea un documento con errores comunes y sus soluciones
2. Mejora tus preguntas
    - Sé más específico cada vez
    - Incluye más contexto relevante
3. No te desanimes
    - Los errores son normales al programar
    - Cada error es una oportunidad de aprendizaje

🎓 Recuerda

PROF

- La IA es una herramienta de ayuda, no una solución mágica
- El objetivo es aprender, no solo obtener código que funcione
- Practica explicar tus problemas claramente
- Toma notas de las soluciones y explicaciones

Aquí tienes un documento comparativo de herramientas IA para diferentes casos de uso:

Guía Rápida: Claude vs Gemini vs ChatGPT vs Perplexity

Elección inteligente según tu necesidad

---

1. Claude AI (Anthropic)

Fortalezas:

- 🛠 **Razonamiento técnico avanzado**: Ideal para depurar código y explicar conceptos complejos
- 📑 **Análisis de documentos largos** (hasta 75k palabras)
- 🤖 **Conversaciones estructuradas**: Mantiene mejor el hilo en diálogos extensos

Mejor usa cuando:

- Necesitas soluciones detalladas para problemas de programación
- Trabajas con documentos técnicos extensos (manuales, especificaciones)
- Prefieres respuestas menos verbosas y más directas

Limitaciones:

- Dificultad para captar sarcasmo/referencias culturales
- No genera imágenes ni analiza datos complejos

---

2. Gemini (Google)

Fortalezas:

- 🌐**Multimodalidad avanzada**: Mejor integración con herramientas Google (Workspace, Colab)
- 📊 **Análisis de datos**: Útil para procesar grandes conjuntos de información
- 🎯 **Precisión factual**: Menos alucinaciones que otras IA

Mejor usa cuando:

- Trabajas con datos estructurados (tablas, hojas cálculo)
- Necesitas integración con ecosistema Google
- Desarrollas contenido multilingüe

Limitaciones:

- Lentitud en respuestas (10+ segundos)
- Dificultad con comandos básicos tipo asistente
- Restricciones en imágenes con personas

PROF

---

3. ChatGPT (OpenAI)

Fortalezas:

- 🎭 **Versatilidad creativa**: Mejor para narrativas, contenido literario
- ⚡ **Velocidad**: Respuestas inmediatas para consultas simples
- 🤝 **Integraciones**: Amplio ecosistema de plugins y APIs

Mejor usa cuando:

- Generas contenido creativo (historias, copywriting)
- Necesitas prototipado rápido de ideas
- Trabajas con plugins especializados (DALL-E, Wolfram)

Limitaciones:

- Mayor tasa de alucinaciones
- Sesgos más pronunciados en respuestas
- Límite de contexto (~8k tokens)

---

4. Perplexity

Fortalezas:

- 🔍 **Precisión investigativa**: Respuestas con fuentes citadas [Search Knowledge]
- 📚 **Actualización en tiempo real**: Acceso a información reciente
- 🎓 **Estilo académico**: Ideal para papers/revisiones técnicas

Mejor usa cuando:

- Investigas temas especializados
- Necesitas referencias verificables
- Priorizas exactitud sobre creatividad

Limitaciones:

- Menos capacidad de diálogo extendido
- Opciones limitadas de personalización

Tabla Comparativa

Criterio	Claude	Gemini	ChatGPT	Perplexity
Técnico/Código	⭐⭐⭐⭐⭐	⭐⭐⭐	⭐⭐⭐⭐	⭐⭐
Creatividad	⭐⭐⭐	⭐⭐	⭐⭐⭐⭐⭐	⭐
Precisión	⭐⭐⭐⭐	⭐⭐⭐⭐⭐	⭐⭐⭐	⭐⭐⭐⭐⭐
Velocidad	⭐⭐⭐⭐	⭐⭐	⭐⭐⭐⭐⭐	⭐⭐⭐⭐
Multimodalidad	⭐⭐	⭐⭐⭐⭐⭐	⭐⭐⭐	⭐

PROF

Consejos de Selección

1. **Programación**: Claude + ChatGPT (combinar precisión y creatividad)
2. **Análisis de datos**: Gemini + Perplexity (datos + verificación)
3. **Contenido creativo**: ChatGPT + Gemini (narrativa + precisión)
4. **Investigación**: Perplexity + Claude (fuentes + profundidad)

Regla de oro: Para resultados óptimos, usa 2 IA simultáneamente y compara sus salidas.