Introducción
Les pedimos, para el buen desarrollo de esta evaluación, que respeten las siguientes reglas:

- Manténganse corteses, educados, respetuosos y constructivos en toda situación durante este intercambio. El vínculo de confianza entre la comunidad 42 y ustedes depende de ello.
- Señalen a la persona (o al grupo) evaluado cualquier posible malfuncionamiento del trabajo entregado, y tómense el tiempo para discutirlo y debatirlo.
- Acepten que a veces puede haber diferencias de interpretación sobre las demandas del tema o el alcance de las funcionalidades. Mantengan la mente abierta ante la visión del otro (¿tiene razón o no?) y evalúen lo más honestamente posible. La pedagogía de 42 solo tiene sentido si la evaluación entre pares se realiza de manera seria.

Pautas

- Solo deben evaluar lo que se encuentra en el repositorio Git de la entrega del estudiante o del grupo.
- Asegúrense de verificar que el repositorio Git sea el correspondiente al estudiante o grupo, y al proyecto.
- Verifiquen meticulosamente que no se haya utilizado ningún alias malicioso para inducirles a error y hacerles evaluar algo que no sea el contenido del repositorio oficial.
- Cualquier script que se proporcione para facilitar la evaluación debe ser rigurosamente verificado por ambas partes para evitar sorpresas.
- Si el estudiante corrector aún no ha hecho este proyecto, es obligatorio para él o ella leer todo el tema antes de comenzar esta evaluación.
- Utilicen las banderas disponibles en este baremo para señalar una entrega vacía, no funcional, un error de normas, un caso de trampa, etc. En ese caso, la evaluación finaliza y la nota es 0 (o -42 en el caso especial de trampa). No obstante, excepto en casos de trampa, se les anima a seguir discutiendo sobre el trabajo realizado (o no realizado) para identificar los problemas que causaron esta situación y evitarlos en la próxima entrega.
- Verifiquen cuidadosamente el código para asegurarse de que no se haya utilizado ninguna librería que facilite el cálculo o el análisis sintáctico.

Secciones
Parte preliminar
En esta parte solo se trata de verificar que la corrección no utiliza nada prohibido que pueda facilitar la realización del proyecto, ya sea con un tipo complejo en el lenguaje utilizado u otro.

Preliminares
Una vez clonado el repositorio, pidan a su corregido que configure el entorno de trabajo para ejecutar su entrega.
Aprovechen también para verificar que efectivamente existe código para gestionar los diferentes tipos de variables solicitadas, a saber:

- Números enteros naturales

- Números racionales
- Números complejos (con coeficientes racionales)
- Matrices
- Ecuaciones polinómicas de grado igual o inferior a 2

Verifiquen también que el programa compile y/o se ejecute correctamente.
Durante toda esta corrección, el programa NUNCA debe salir de forma imprevista (Segfault, error de interpretación, etc.).
Si alguna de estas etapas es incorrecta, el proyecto vale 0 y pueden detener la corrección.

Verificaciones usuales
Pidan al corregido que explique cómo gestiona el análisis sintáctico y los diferentes tipos (complejos, matrices). Si utiliza una librería que facilite alguno de estos puntos, la corrección se detiene, el proyecto vale 0 y pueden detener la corrección.

Parte de asignación
En esta parte se probarán todos los comportamientos relacionados con la asignación de una variable o una función. Se les invita a hacer muchas pruebas; la corrección solo les ofrece ideas para los tests.

Test de error elemental
Aquí probaremos errores básicos, como por ejemplo x == 2, o tecleos erráticos, sin sentido como x = 23edd23-+-+

Test de error semi-avanzado
Aquí será más engañoso, como por ejemplo = 2 o incluso 3 = 4, o x = g cuando g no está definido. Prueben sintaxis dudosas como f(x = 2 o x = [[4,2]].

Test de error avanzado
Intenten los casos más absurdos que puedan imaginar, como por ejemplo x = --2, f(x) = x * 2 luego t = f(x) (lo cual no es posible), i = 2 (sabiendo que está prohibido dejar que el usuario asigne la variable i). No duden en probar lo que se les ocurra.

Test válido elemental
En las siguientes pruebas, usen "nombreDeLaVariable = ?" para conocer el valor atribuido a la variable en el contexto del programa. Por ejemplo, si ingresan x = 2, pueden hacer x = ? y deberían ver 2 en la línea de la interfaz del programa. En esta parte se trata de probar operaciones de la forma x = 2, y = 4i, z = [[2,3]; [3,5]].

Test válido semi-avanzado
Aquí se probará la asignación de funciones y entre variables. No duden en jugar con los espacios y tabulaciones, que deben ser gestionados. Prueben x = 2 luego y = x, luego y = ?. También prueben x = 2, luego x = 5, luego x = ?, si x no vale 5, es 0. Pueden intentar lo mismo con matrices o números imaginarios, como A = [[2,3]], luego B = A, si B = ? no muestra A, es 0.

Test válido avanzado
Para esta pregunta, se probarán asignaciones que mezclen muchos elementos. Comiencen con x = 2, y = x * [[4,2]], f(z) = z * y, si f(z) = ? no muestra z * [[8,4]], es 0. También prueben x = 2, luego f(x) = x * 5, si f(x) = ? no muestra 10 (o algo similar como 2 * 5), es 0 también. No duden

en probar todo lo que puedan imaginar en términos de asignación, mezclando todos los tipos, desde matrices hasta números imaginarios, etc., siempre que tenga sentido matemático.

Parte de cálculo
En esta parte se probarán todos los comportamientos relacionados con el cálculo y la evaluación de una función. Se les invita a hacer muchas pruebas; la corrección solo les ofrece ideas para los tests.

Test válido elemental
Aquí se les pide probar cálculos muy simples, como 2 + 2 = ?, 3 * 4 = ?, x = 2 luego x + 2 = ?. Del mismo modo, pueden intentar la división por 0, como con 2 / 0 = ?. Prueben también la gestión de números decimales, como con 1.5 + 1 = ?

Test válido semi-avanzado
Aquí los cálculos serán un poco más complejos, como por ejemplo x = 2 * i, luego x ^ 2 = ?, si el resultado no es -2i, es 0. Prueben también una multiplicación de matrices, como A = [[2,3];[3,4]] y B = [[1,0];[0,1]], luego A ** B = ?. Deberían ver la matriz A, de lo contrario es 0. Prueben también entradas como f(x) = x + 2, p = 4, f(p) = ? Si el resultado no es 6, es 0.

Test válido avanzado
Prueben cálculos complejos como 4 - 3 - (2 * 3) ^ 2 * (2 - 4) + 4 = ? o f(x) = 2*(x + 3*(x - 4)), luego p = 2, luego f(3) - f(p) + 2= ? y el resultado esperado es 10. Pueden mezclar números complejos con funciones como f(x) = 2 * x * i, luego f(2) = ? (el resultado esperado es 4i). Igual con matrices, no duden en intentarlo, usen el programa como si estuvieran usando una calculadora.

Bonus
Recuerden: Si en algún momento el programa no reacciona correctamente (error de bus, segfault, etc.), la evaluación finaliza y la nota es 0. Usen las banderas correspondientes. Esta instrucción está activa durante toda la evaluación. Los bonus solo deben ser evaluados si, y solo si, la parte obligatoria es PERFECTA. Por PERFECTA, se entiende que está completamente realizada y que no es posible encontrar fallos en su comportamiento, incluso en caso de error, por muy engañoso que sea o de mal uso, etc. Concretamente, esto significa que si la parte obligatoria no ha obtenido TODOS los puntos durante esta evaluación, los bonus deben ser IGNORADOS por completo.

Bonus
Déjense guiar por su corregido en cuanto a los bonus implementados. La evaluación de los bonus queda a su libre discreción.

4o