Algunas de las pocas cosas que pude ver, dado el estado del compilador:  
  
El assembler lo tendrían que generar en la misma carpeta que el archivo de entrada, con el mismo nombre y extensión .asm, no .txt.  
  
El scrollpane de los números de línea no puede ser independiente del scrollpane del código.  
  
El menú de nuevo no invoca el File Chooser.  
  
Las sentencias en los case no terminan con punto?  
  
El código es: var2 = var1 - aa(). Pero en errores dice que falta paréntesis (.  
  
Marca error de re-declaración de variables cuando no hay ninguna re-declarada. Internamente usan nombres de variables que no permiten al usuario SIN avisarle. Los nombres de variables DEBEN permitir números. Y en el caso en el que no lo permitan, tienen que notificarle al usuario qué es lo que está mal, no decirle que la variable está re declarada porque ustedes decidieron truncar los números al final de los nombres.  
  
Las invocaciones a funciones solo pueden usarse en donde se puedan utilizar las expresiones, por lo tanto no pueden utilizarse como sentencias individuales.  
  
El programa sigue teniendo problemas cuando se cargan archivos por separado. Es decir, sigue reportando los errores de los archivos leídos anteriormente y por eso no permite generar código, aun cuando el último archivo cargado no tiene errores.  
  
bbb = aaa - aa(). Genera como error: ERROR Linea 16: Falta '(' en llamado a Funcion. No entiendo el error, cuando copio el aa() a una línea separada no genera ningún error. Las expresiones tienen que soportar funciones.  
  
De los pocos tercetos que pude ver, los de la forma (Label, Label12, ) que generaron no corresponden.  
  
En el assembler pusieron el código de los tercetos. No corresponde. Así nunca se va a poder ensamblar el archivo.  
  
Considerando su ejemplo, las sentencias correspondientes a la función no se encuentran delimitados, es decir, no está marcado cuál es el final de la función. Por ejemplo:  
  
LONG FUNCTION aa {  
ccc : LONG.  
fff = 3.  
ccc = 50.  
OUT("Invocacion de la funcion").  
RETURN (var3).  
}  
  
Genera:  
0--> (Label, Label0, )  
1--> (=, fff, 3)  
2--> (=, ccc, 50)  
3--> (OUT, "Invocacion de la funcion", )  
4--> ....  
  
El assembler de invocación a funciones está mal. El call se usa para invocar. El ret se usa como última instrucción de la función.  
En el assembler hacen call a labels que no existen.  
Las cadenas a imprimir por el out cuando las declarar no las encierran entre comillas. Las declaraciones están duplicadas. Por ejemplo:  
Imprimo DB Imprimo, 0

Los saltos por overflow no deben realizarse de esa forma. Ya hay instrucciones de Jump específicas que se activan con los diferentes flags.  
  
Para c = a - b tienen:  
MOV R1,\_a  
SUB R1,\_b  
MOV #aux4,R1  
MOV R1,#aux4  
MOV \_c,R1  
Hay mov innecesarios.  
  
Las operaciones con floats no corresponden. No se pueden hacer con registros. Se hacen en el co-procesador. Eso está explicado en las filminas.  
  
Código del if:  
IF( 3 < 4) THEN  
OUT("MENOR").  
ELSE  
OUT("MAYOR").           
END\_IF.  
  
Genera:  
CMP 3,4  
JL direccion\_falso  
JLE Label4  
invoke MessageBox, NULL, addr MENOR, addr MENOR, MB\_OK  
JMP Label6  
Label4:  
invoke MessageBox, NULL, addr MAYOR, addr MAYOR, MB\_OK  
Label6:  
invoke ExitProcess, 0  
  
El primer error es que CMP no acepta dos operadores inmediatos. Después fíjense que dirección\_falso NO existe. Por qué tienen dos sentencias de J seguidas?

Su archivo de pruebas no tiene el LET ni conversiones. Debería.

El switch directamente no funciona.

Por favor, vayan corrigiendo esto.