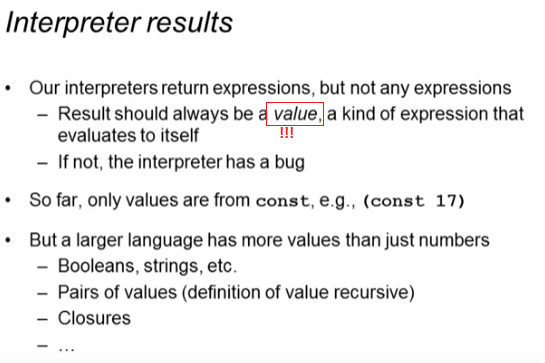
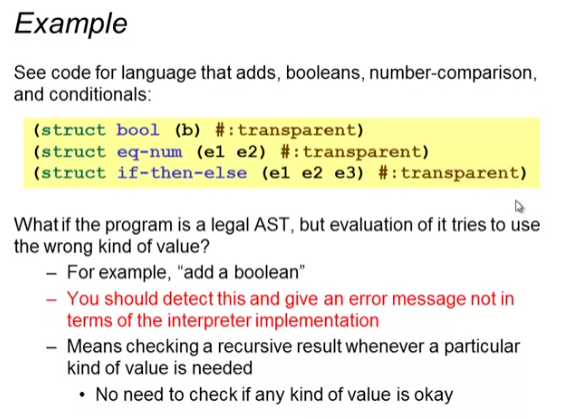
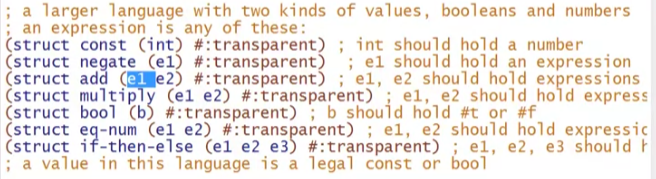
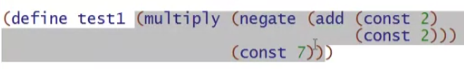


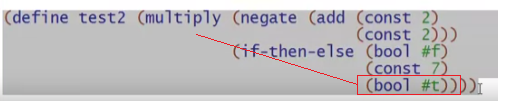
* We don’t need to worry about this! It’s okay to crash for this. No need to handle these kinds of type errors



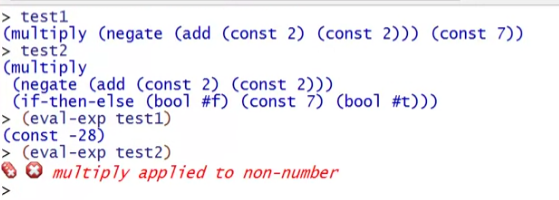






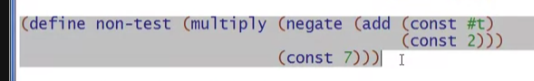


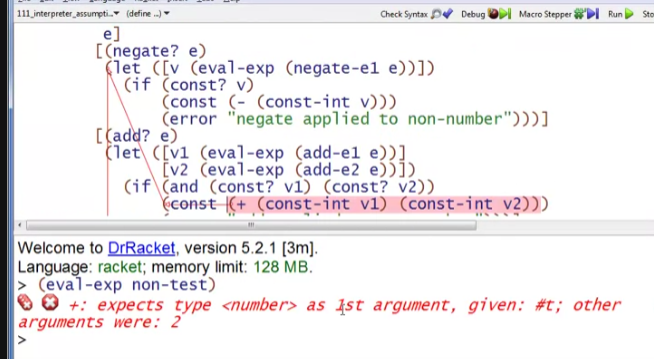
* This can end up multiplying a bool



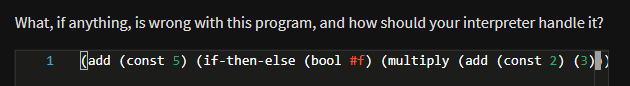
* We will only handle this if we have a legal AST all the way but the execution can result into type error values

But this kind we will not handle:

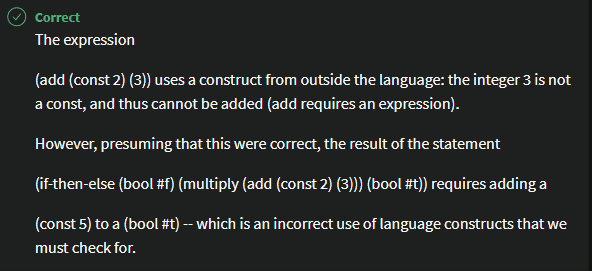


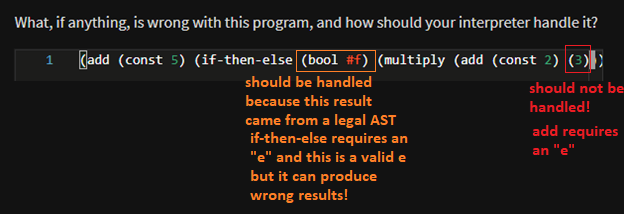


* This is because we directly use (const #t) here in the add struct
* Unlike the other before this, it only became an error because of a return value in if then else statements

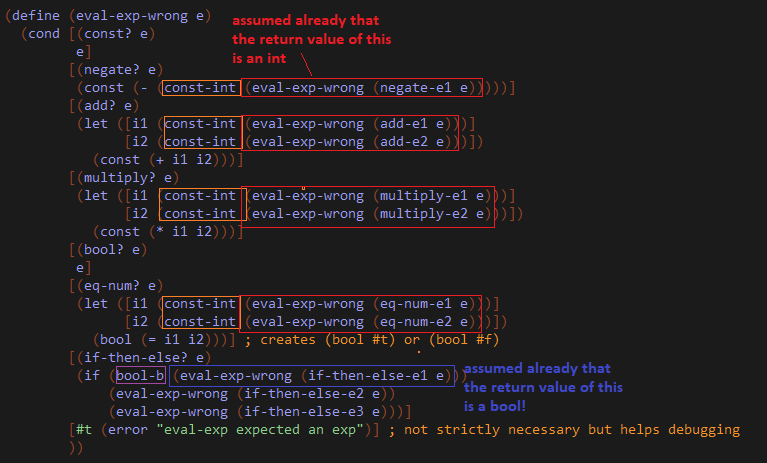




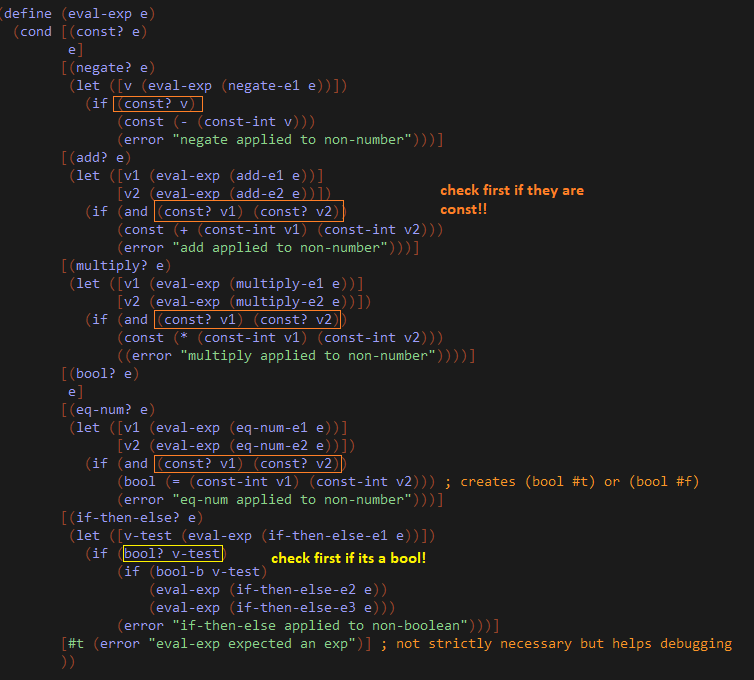




Wrong interpreter



Better and correct



* Basically just checking the types of the RETURN values of our expressions
  + Because as we said earlier we will handle the type checking of the RETURN values but not the INPUTTED values in our structs

