TABLE OF ERRORS

Spec	How ChatGPT Ran	Error Message
addr1	(3) No fault location (No Bug / Fix comment and Loc) [Asked to fix assertion]	Type Error Message Observed != can be used only between 2 expressions of the same arity, or between 2 integer expressions. Left type = {this/Address} Right type = {PrimitiveBoolean}

arr1	(3) No fault location
	(No Bug / Fix comment and Loc)
	[Asked to fix assertion]
		Type Error Message Observed
		in can be used only between 2 expressions of the same arity.
		Left type = {none->none}
		Right type = {this/Element}

	(4) No indication of bug location nor asking to pass the assertions	Type Error Message Observed This must be an integer expression. Instead, it has the following possible type(s): {this/Element}

arr2	(3) No fault location
	(No Bug / Fix comment and Loc)
	[Asked to fix assertion]
		Type Error Message Observed
		in can be used only between 2 expressions of the same arity.
		Left type = {Int}
		Right type = {Int->this/Element}

	(4) No indication of bug location nor asking to pass the assertions	Type Error Message Observed Analysis cannot be performed since it requires higher-order quantification that could not be skolemized.

balancedBST1	(5) With bug location but w/o fix comments [Mention of passing assertion]	Type Error Message Observed This must be a formula expression. Instead, it has the following possible type(s): {this/Node}
balancedBST2_1	(3) No fault location (No Bug / Fix comment and Loc) [Asked to fix assertion]	Type Error Message Observed This must be a formula expression. Instead, it has the following possible type(s): {none}
balancedBST2_2	(3) No fault location (No Bug / Fix comment and Loc) [Asked to fix assertion]	Type Error Message Observed ^ can be used only with a binary relation. Instead, its possible type(s) are: {this/Node}
balancedBST2_3	(5) With bug location but w/o fix comments [Mention of passing assertion]	Syntax Error The name "abs" cannot be found.
bempl1	(5) With bug location but w/o fix comments [Mention of passing assertion]	Syntax error at line 17 column 8: There are 1 possible tokens that can appear here:[
cd1_2	(5) With bug location but w/o fix comments [Mention of passing assertion]	Type Error Message Observed This cannot be a legal relational join where left hand side is c . ~ (this/Class <: ext) (type = {this/Class}) right hand side is this/Object (type = {this/Object})

ctree1	(2) With bug location but w/o fix comments [No mention of passing assertion]	Type Error Message Observed This must be a set or relation. Instead, it has the following possible type(s): {PrimitiveBoolean}
dll1_2	(5) With bug location but w/o fix comments [Mention of passing assertion]	Syntax error The name "null" cannot be found.
dll2_2	(5) With bug location but w/o fix comments [Mention of passing assertion]	Syntax error at line 33 column 32:
farmer1	(5) With bug location but w/o fix comments [Mention of passing assertion]	Syntax Error The name "item" cannot be found.
fsm1_2	(5) With bug location but w/o fix comments [Mention of passing assertion]	Syntax error at line 46 column 1:
student2_2	(5) With bug location but w/o fix comments [Mention of passing assertion]	Type Error Message Observed = can be used only between 2 expressions of the same arity, or between 2 integer expressions. Left type = {this/Boolean} Right type = {PrimitiveBoolean}
student4	(5) With bug location but w/o fix comments [Mention of passing assertion]	Type Error Message Observed = can be used only between 2 expressions of the same arity, or between 2 integer expressions. Left type = {this/Boolean} Right type = {PrimitiveBoolean}

Student6_1	(5) With bug location but w/o fix comments
	[Mention of passing assertion]
		Type Error Message Observed
		= can be used only between 2 expressions of the same arity, or between 2 integer expressions.
		Left type = {this/Boolean}
		Right type = {PrimitiveBoolean}

student7_2	(3) No fault location
	(No Bug / Fix comment and Loc)
	[Asked to fix assertion]
		Type Error Message Observed
		= can be used only between 2 expressions of the same arity, or between 2 integer expressions.
		Left type = {Int}
		Right type = {PrimitiveBoolean}

	5) With bug location but w/o fix comments
	[Mention of passing assertion]
		Type Error Message Observed
		= can be used only between 2 expressions of the same arity, or between 2 integer expressions.
		Left type = {Int}
		Right type = {PrimitiveBoolean}

student8	5) With bug location but w/o fix comments
	[Mention of passing assertion]
		Type Error Message Observed
		= can be used only between 2 expressions of the same arity, or between 2 integer expressions.
		Left type = {Int}
		Right type = {PrimitiveBoolean}

student9	(3) No fault location
	(No Bug / Fix comment and Loc)
	[Asked to fix assertion]
		Type Error Message Observed
		= can be used only between 2 expressions of the same arity, or between 2 integer expressions.
		Left type = {PrimitiveBoolean}
		Right type = {this/True}

student12.1	(2) With bug location but w/o fix comments
	[No mention of passing assertion]
		Type Error Message Observed
		= can be used only between 2 expressions of the same arity, or between 2 integer expressions.
		Left type = {this/Boolean}
		Right type = {PrimitiveBoolean}

student14_1	(5) With bug location but w/o fix comments [Mention of passing assertion]	Type Error Message Observed This must be a set or relation. Instead, it has the following possible type(s): {PrimitiveBoolean}
student16	(5) With bug location but w/o fix comments [Mention of passing assertion]	Syntax error at line 45 column 18: There are 1 possible tokens that can appear here:)
student17_1	(5) With bug location but w/o fix comments [Mention of passing assertion]	Syntax error at line 20 column 15: There are 1 possible tokens that can appear here: /

student18_3	(2) With bug location but w/o fix comments [No mention of passing assertion]	Type Error Message Observed This must be a set or relation. Instead, it has the following possible type(s): {PrimitiveBoolean}

student19_2	(5) With bug location but w/o fix comments
	[Mention of passing assertion]
		Type Error Message Observed
		& can be used only between 2 expressions of the same arity.
		Left type = {this/Node->this/Node}
		Right type = {this/Node}