eval-exp: comprehensive test with mupl-length function (MUPL call applied to non-closure #(struct:call #(struct:var "f") #(struct:aunit))) [error]

~~eval-exp: isaunit evaluates its expression ((isaunit (fst (apair (aunit) (aunit)))) should evaluate to (int 1) but evaluated to (int 0)) [incorrect answer]~~

eval-exp: recursive function call (MUPL call applied to non-closure #(struct:call #(struct:fun "sum-to" "n" #(struct:ifgreater #(struct:var "n") #(struct:int 1) #(struct:add #(struct:var "n") #(struct:call #(struct:var "sum-to") #(struct:add #(struct:var "n") #(struct:int -1)))) #(struct:int 1))) #(struct:int 5))) [error]

eval-exp: complex call expression (MUPL call applied to non-closure #(struct:call #(struct:fun #f "x" #(struct:add #(struct:var "x") #(struct:int 2))) #(struct:int 4))) [error]

eval-exp: evaluating a fun results in a correct closure [incorrect answer]

~~ifaunit: first expression is aunit ('(ifaunit (fst (apair (aunit) (int 0))) (int 4) (int 10)) should result in MUPL that evaluates to (int 4) but resulted in (int 10)) [incorrect answer]~~

mlet\*: sequential bindings (unbound variable during evaluation "x") [error]

ifeq: resulting expression uses \_x and \_y variables (couldn't find binding of \_x) [incorrect answer]

~~ifeq: resulting expression evaluates to e3 if e1 and e2 evaluate to equal integers ('(ifeq (add (int 0) (int 5)) (int 5) (int 0) (int 1)) should result in MUPL that evaluates to (int 0) but resulted in (int 1)) [incorrect answer]~~

~~ifeq: resulting expression evaluates e1 and e2 exactly once (Found e1 to be evaluated more/less than once) [incorrect answer]~~

mupl-map: multiple element list (fun-formal: contract violation expected: fun? given: (var "f")) [error]

mupl-map: single element list (fun-formal: contract violation expected: fun? given: (var "f")) [error]

mupl-map: empty list (MUPL fst applied to non-apair) [error]

mupl-mapAddN: multiple element list (found bad MUPL expression: "map") [error]

mupl-mapAddN: single element list (found bad MUPL expression: "map") [error]

mupl-mapAddN: empty list (found bad MUPL expression: "map") [error]

compute-free-vars: correctly computes free vars [incorrect answer]

compute-free-vars: no free vars case [incorrect answer]

eval-under-env-c: correctly filters closure environments [incorrect answer]