```
ouzo2:code$ ghci TestDataGenerators.hs
GHCi, version 8.2.2: http://www.haskell.org/ghc/  :? for help
Loaded GHCi configuration from /Users/hallgren/.ghci
[1 of 2] Compiling Overloading      ( Overloading.hs, interpreted )
[2 of 2] Compiling TestDataGenerators ( TestDataGenerators.hs, interpreted )

TestDataGenerators.hs:174:1: error:
    parse error (possibly incorrect indentation or mismatched brackets)
Failed, one module loaded.
*Overloading> :r
[2 of 2] Compiling TestDataGenerators ( TestDataGenerators.hs, interpreted )

TestDataGenerators.hs:159:22: error:
    • Couldn't match expected type ‘Property’ with actual type ‘Bool’
    • In the expression: isOrdered (insert x xs)
      In an equation for ‘prop_insert_1’:
          prop_insert_1 x xs = isOrdered (insert x xs)
    |
159 | prop_insert_1 x xs = isOrdered (insert x xs)
    |                      ^^^^^^^^^^^^^^^^^^^^^^
Failed, one module loaded.
*Overloading> :r
[2 of 2] Compiling TestDataGenerators ( TestDataGenerators.hs, interpreted )
Ok, two modules loaded.
*TestDataGenerators> quickCheck prop_insert_1
*** Failed! Falsifiable (after 4 tests and 5 shrinks):
0
[0,-1]
*TestDataGenerators> :r
[2 of 2] Compiling TestDataGenerators ( TestDataGenerators.hs, interpreted )

TestDataGenerators.hs:163:22: error:
    • Couldn't match expected type ‘Bool’ with actual type ‘Property’
    • In the expression: isOrdered xs ==> isOrdered (insert x xs)
      In an equation for ‘prop_insert_2’:
          prop_insert_2 x xs = isOrdered xs ==> isOrdered (insert x xs)
    |
163 | prop_insert_2 x xs = isOrdered xs ==> isOrdered (insert x xs)
    |                      ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
Failed, one module loaded.
*Overloading> quickCheck prop_insert_2

<interactive>:5:1: error:
    Variable not in scope: quickCheck :: t0 -> t

<interactive>:5:12: error: Variable not in scope: prop_insert_2
*Overloading> :r
[2 of 2] Compiling TestDataGenerators ( TestDataGenerators.hs, interpreted )
Ok, two modules loaded.
*TestDataGenerators> quickCheck prop_insert_2
*** Gave up! Passed only 81 tests.
*TestDataGenerators> stdArgs
Args {replay = Nothing, maxSuccess = 100, maxDiscardRatio = 10, maxSize = 100, c
hatty = True, maxShrinks = 9223372036854775807}
```

```
*TestDataGenerators> quickCheckWith stdArgs {maxDiscardRatio =20} prop_insert_2
+++ OK, passed 100 tests.
*TestDataGenerators> quickCheckWith stdArgs {maxDiscardRatio =20} prop_insert_2
+++ OK, passed 100 tests.
*TestDataGenerators> :r
[2 of 2] Compiling TestDataGenerators ( TestDataGenerators.hs, interpreted )
Ok, two modules loaded.
*TestDataGenerators> quickCheck prop_isOrdered
+++ OK, passed 100 tests:
87% False
13% True
*TestDataGenerators> quickCheck prop_isOrdered
+++ OK, passed 100 tests:
83% False
17% True
*TestDataGenerators> quickCheck prop_isOrdered
+++ OK, passed 100 tests:
85% False
15% True
*TestDataGenerators> :r
[2 of 2] Compiling TestDataGenerators ( TestDataGenerators.hs, interpreted )
Ok, two modules loaded.
*TestDataGenerators> quickCheck prop_insert_3
+++ OK, passed 100 tests.
*TestDataGenerators> sample or
or           orderedList
*TestDataGenerators> sample orderedList
[]
[(),()]
[(),(),(),()]
[]
[(),(),(),(),(),()]
[(),(),(),(),(),(),(),(),(),()]
[(),(),(),(),(),(),(),(),(),()]
[(),(),(),(),(),(),(),(),(),(),(),(),(),()]
[(),(),(),(),(),(),(),(),(),(),(),(),()]
[]
[(),(),(),(),(),(),(),()]
*TestDataGenerators> sample (orderedList ::Gen [Int])
[]
[]
[-3,4]
[]
[3,6]
[-6]
[-11,-5,-2]
[-14,0,3,6,8,8,9,13]
[-14,-11,0,7,14]
[-17,-9,-5,14]
[-20,-18,-16,-13,-10,-8,-2,8,14,15,16]
*TestDataGenerators> :i Ord
Ord          Ordered      OrderedList  Ordering
*TestDataGenerators> :i OrderedList
newtype OrderedList a = Ordered {getOrdered :: [a]}
```

```
        -- Defined in 'Test.QuickCheck.Modifiers'
instance Eq a => Eq (OrderedList a)
  -- Defined in 'Test.QuickCheck.Modifiers'
instance Functor OrderedList
  -- Defined in 'Test.QuickCheck.Modifiers'
instance Ord a => Ord (OrderedList a)
  -- Defined in 'Test.QuickCheck.Modifiers'
instance Show a => Show (OrderedList a)
  -- Defined in 'Test.QuickCheck.Modifiers'
instance Read a => Read (OrderedList a)
  -- Defined in 'Test.QuickCheck.Modifiers'
instance (Ord a, Arbitrary a) => Arbitrary (OrderedList a)
  -- Defined in 'Test.QuickCheck.Modifiers'
*TestDataGenerators> :r
[2 of 2] Compiling TestDataGenerators ( TestDataGenerators.hs, interpreted )
Ok, two modules loaded.
*TestDataGenerators> :t prop_insert_4
prop_insert_4 :: Int -> OrderedList Int -> Bool
*TestDataGenerators> quickCheck prop_insert_4
+++ OK, passed 100 tests.
*TestDataGenerators> :i NonNegative
newtype NonNegative a = NonNegative {getNonNegative :: a}
        -- Defined in 'Test.QuickCheck.Modifiers'
instance Eq a => Eq (NonNegative a)
  -- Defined in 'Test.QuickCheck.Modifiers'
instance Functor NonNegative
  -- Defined in 'Test.QuickCheck.Modifiers'
instance Ord a => Ord (NonNegative a)
  -- Defined in 'Test.QuickCheck.Modifiers'
instance Show a => Show (NonNegative a)
  -- Defined in 'Test.QuickCheck.Modifiers'
instance Read a => Read (NonNegative a)
  -- Defined in 'Test.QuickCheck.Modifiers'
instance Enum a => Enum (NonNegative a)
  -- Defined in 'Test.QuickCheck.Modifiers'
instance (Num a, Ord a, Arbitrary a) => Arbitrary (NonNegative a)
  -- Defined in 'Test.QuickCheck.Modifiers'
*TestDataGenerators> sample (arbitrary::Gen (NonNegative Int))
NonNegative {getNonNegative = 1}
NonNegative {getNonNegative = 2}
NonNegative {getNonNegative = 4}
NonNegative {getNonNegative = 3}
NonNegative {getNonNegative = 4}
NonNegative {getNonNegative = 0}
NonNegative {getNonNegative = 10}
NonNegative {getNonNegative = 10}
NonNegative {getNonNegative = 8}
NonNegative {getNonNegative = 12}
NonNegative {getNonNegative = 0}
*TestDataGenerators>
Leaving GHCi.
ouzo2:code$ ghci ArithmeticQuiz.hs
GHCi, version 8.2.2: http://www.haskell.org/ghc/  :? for help
Loaded GHCi configuration from /Users/hallgren/.ghci
```

```
[1 of 1] Compiling Main             ( ArithmeticQuiz.hs, interpreted )
Ok, one module loaded.
*Main> ex3
Mul (Add (Num 1) (Num 2)) (Num 3)
*Main> ex4
Add (Num 1) (Mul (Num 2) (Num 3))
*Main> :r
[1 of 1] Compiling Main             ( ArithmeticQuiz.hs, interpreted )
Ok, one module loaded.
*Main> eval ex1
2
*Main> eval ex2
3
*Main> eval ex3
9
*Main> eval ex4
7
*Main> :r
[1 of 1] Compiling Main             ( ArithmeticQuiz.hs, interpreted )
Ok, one module loaded.
*Main> showExpr e1

<interactive>:9:10: error:
    • Variable not in scope: e1 :: Expr
    • Perhaps you meant ‘ex1’ (line 26)
*Main> showExpr ex1
"2"
*Main> showExpr ex2
"1 + 2"
*Main> showExpr ex3
"1 + 2 * 3"
*Main> showExpr ex4
"1 + 2 * 3"
*Main> :r
[1 of 1] Compiling Main             ( ArithmeticQuiz.hs, interpreted )
Ok, one module loaded.
*Main> ex1
Num 2
*Main> showExpr ex1
"2"
*Main> showExpr ex2
"(1 + 2)"
*Main> showExpr ex3
"(1 + 2) * 3"
*Main> showExpr ex4
"(1 + 2 * 3)"
*Main> :r
[1 of 1] Compiling Main             ( ArithmeticQuiz.hs, interpreted )
Ok, one module loaded.
*Main> showExpr ex1
"2"
*Main> showExpr ex2
"1 + 2"
*Main> showExpr ex3
```

```
"(1 + 2) * 3"
*Main> showExpr ex4
"1 + 2 * 3"
*Main> :r
[1 of 1] Compiling Main             ( ArithmeticQuiz.hs, interpreted )
Ok, one module loaded.
*Main> showExpr ex1
"2"
*Main> showExpr ex2
"1 + 2"
*Main> showExpr ex3
"(1 + 2) * 3"
*Main> showExpr ex4
"1 + 2 * 3"
*Main> :t Add
Add :: Expr -> Expr -> Expr
*Main> :t elements
elements :: [a] -> Gen a
*Main> :r
[1 of 1] Compiling Main             ( ArithmeticQuiz.hs, interpreted )
Ok, one module loaded.
*Main> sample rExpr
Mul (Num 6) (Mul (Num 5) (Num 7))
Add (Add (Add (Add (Num 8) (Num 1)) (Add (Mul (Num 3) (Add (Num 6) (Num 8))) (Nu
m 9))) (Add (Num 10) (Mul (Add (Add (Add (Num 10) (Num 2)) (Mul (Mul (Num 5) (Ad
d (Num 6) (Mul (Mul (Mul (Num 2) (Mul (Num 3) (Num 4))) (Mul (Add (Num 6) (Add (
Num 1) (Num 3))) (Num 2))) (Num 3)))) (Add (Num 7) (Mul (Add (Num 2) (Mul (Add (
Num 1) (Mul (Add (Mul (Mul (Num 7) (Num 4)) (Num 7)) (Add (Num 4) (Num 3))) (Num
 10))) (Mul (Add (Mul (Num 4) (Add (Mul (Add (Num 7) (Mul (Mul (Mul (Add (Num 5)
 (Add (Add (Add (Mul (Num 2) (Add (Mul (Add (Num 2) (Mul (Add (Num 9) (Mul (Num
2) (Num 1))) (Mul (Mul (Add (Mul (Num 8) (Add (Mul (Num 1) (Mul (Mul (Add (Mul (
Num 10) (Add (Mul (Num 5) (Add (Num 2) (Num 3))) (Add (Mul (Num 6) (Mul (Add (Mu
l (Add (Mul (Num 10) (Mul (Mul (Add (Add (Num 3) (Mul (Num 7) (Num 4))) (Num 4))
 (Num 6)) (Num 9))) (Num 10)) (Mul (Mul (Num 4) (Add (Mul (Num 5) (Add (Add (Num
 7) (Add (Mul (Num 10) (Num 5)) (Add (Num 1) (Mul (Add (Add (Mul (Add (Num 9) (N
um 3)) (Mul (Mul (Num 5) (Num 9)) (Num 8))) (Add (Mul (Mul (Mul (Num 8) (Num 3))
 (Add (Mul (Num 10) (Mul (Mul (Num 5) (Num 10)) (Num 7))) (Mul (Num 7) (Add (Mul
 (Add (Mul (Num 9) (Num 9)) (Add (Num 2) (Num 1))) (Add (Num 2) (Num 8))) (Num 1
))))) (Num 8)) (Num 7))) (Mul (Mul (Mul (Add (Mul (Mul (Num 7) (Num 5)) (Add (Mu
l (Num 8) (Num 8)) (Mul (Num 1) (Add (Mul (Mul (Mul (Num 1) (Add (Mul (Mul (Add
(Add (Mul (Num 5) (Num 10)) (Mul (Num 2) (Num 7))) (Mul (Mul (Add (Add (Add (Num
 9) (Add (Num 8) (Num 8))) (Add (Add (Num 6) (Mul (Num 4) (Num 2))) (Num 10))) (
Num 9)) (Add (Mul (Mul (Num 10) (Num 4)) (Mul (Num 8) (Mul (Add (Num 10) (Num 1)
) (Add (Add (Add (Add (Add (Num 3) (Num 7)) (Num 6)) (Add (Num 10) (Num 7))) (Nu
m 7)) (Num 4))))) (Num 1))) (Num 4))) (Num 10)) (Num 9)) (Mul (Num 1) (Mul (Num
7) (Num 10))))) (Add (Num 9) (Mul (Num 5) (Num 4)))) (Num 1)) (Mul (Add (Add (Nu
m 2) (Num 10)) (Num 8)) (Num 1)))))) (Num 1)) (Num 9)) (Mul (Num 5) (Add (Add (N
um 7) (Mul (Add (Num 7) (Mul (Add (Num 6) (Add (Mul (Num 7) (Add (Num 7) (Num 1)
)) (Num 6))) (Add (Num 7) (Num 3)))) (Num 3))) (Num 1)))) (Add (Num 1) (Add (Mul
 (Num 1) (Num 9)) (Add (Num 9) (Add (Num 10) (Num 7)))))) (Num 5))))) (Mul (Num
 5) (Mul (Num 7) (Add (Num 1) (Num 7)))))) (Num 9))) (Add (Add (Num 8) (Num 8))
(Num 1)))) (Num 7)) (Num 8))) (Num 4)))) (Num 3)) (Num 9)) (Mul (Mul (Num 6) (Nu
m 7)) (Num 1)))) (Num 10))) (Num 6)) (Num 8)) (Mul (Mul (Num 5) (Add (Num 7) (Nu
m 9))) (Add (Num 3) (Add (Add (Mul (Num 9) (Add (Mul (Num 8) (Mul (Add (Num 3) (
```

```
Add (Num 2) (Mul (Add (Add (Add (Mul (Add (Add (Num 5) (Add (Num 6) (Num 10))) (
Add (Add (Num 7) (Add (Num 2) (Mul (Add (Num 8) (Mul (Num 5) (Add (Num 7) (Add (
Num 5) (Num 3))))) (Num 2)))) (Num 9))) (Mul (Num 9) (Mul (Mul (Mul (Num 7) (Add
 (Num 3) (Mul (Mul (Num 6) (Add (Num 4) (Num 6))) (Mul (Add (Num 8) (Num 6)) (Nu
m 1))))) (Mul (Num 4) (Mul (Mul (Mul (Mul (Mul (Mul (Add (Add (Mul (Num 9) (Num
6)) (Num 4)) (Add (Num 4) (Add (Add (Num 2) (Add (Add (Add (Num 9) (Mul (Num 6)
(Mul (Add (Mul (Num 5) (Num 7)) (Num 5)) (Mul (Num 5) (Num 8))))) (Num 7)) (Num
8))) (Num 10)))) (Mul (Num 7) (Num 3))) (Add (Num 5) (Num 1))) (Num 6)) (Num 6))
 (Add (Num 9) (Num 4))) (Num 4)))) (Mul (Mul (Mul (Add (Mul (Add (Add (Mul (Num
9) (Num 10)) (Num 1)) (Num 3)) (Add (Mul (Mul (Num 8) (Add (Num 8) (Mul (Num 5)
(Num 2)))) (Add (Num 3) (Mul (Mul (Num 9) (Add (Num 8) (Num 2))) (Num 10)))) (Nu
m 4))) (Num 1)) (Mul (Num 10) (Mul (Num 4) (Mul (Mul (Num 3) (Mul (Num 5) (Mul (
Num 6) (Num 1)))) (Mul (Mul (Num 3) (Num 10)) (Num 5)))))) (Num 4)) (Num 8))))) 
(Mul (Add (Num 10) (Num 4)) (Add (Add (Num 3) (Num 4)) (Mul (Num 5) (Add (Mul (M
ul (Num 4) (Mul (Num 2) (Mul (Mul (Num 10) (Add (Add (Num 4) (Num 6)) (Add (Num
7) (Num 3)))) (Num 4)))) (Add (Num 4) (Num 8))) (Add (Num 5) (Mul (Num 8) (Mul (
Num 5) (Num 2)))))))) (Mul (Add (Num 10) (Num 9)) (Mul (Mul (Num 10) (Add (Num
10) (Num 9))) (Add (Num 10) (Num 8))))) (Num 7)) (Mul (Num 8) (Num 3))))) (Mul (
Add (Mul (Mul (Num 8) (Num 4)) (Num 9)) (Add (Mul (Num 10) (Num 7)) (Num 4))) (A
dd (Mul (Num 7) (Mul (Add (Num 1) (Mul (Add (Add (Add (Add (Mul (Add (Add (Add (
Mul (Mul (Mul (Num 9) (Add (Num 4) (Num 6))) (Add (Mul (Num 6) (Mul (Add (Mul (A
dd (Num 9) (Num 8)) (Mul (Mul (Add (Num 7) (Num 3)) (Num 2)) (Mul (Num 5) (Num 1
)))) (Mul (Mul (Num 3) (Mul (Num 3) (Num 9))) (Add (Num 7) (Add (Add (Num 1) (Nu
m 8)) (Mul (Num 9) (Num 1)))))) (Add (Num 4) (Num 9))) (Num 10))) (Mul (Num 9) (
Num 7)))) (Num 4)) (Mul (Num 4) (Num 3))) (Num 7)) (Mul (Num 4) (Mul (Add (Num 9
) (Num 9)) (Num 8)))) (Num 10)) (Num 3)) (Add (Num 7) (Add (Mul (Num 10) (Mul (N
um 5) (Num 3))) (Num 9)))) (Num 4)) (Mul (Num 9) (Num 5)))) (Add (Mul (Num 8) (N
um 1)) (Mul (Num 3) (Num 3))))) (Num 1))))) (Add (Add (Num 2) (Mul (Add (Num 10)
 (Num 2)) (Mul (Num 7) (Add (Add (Num 4) (Mul (Num 5) (Mul (Num 6) (Add (Add (Nu
m 1) (Add (Add (Num 6) (Num 3)) (Num 9))) (Num 7))))) (Num 5))))) (Num 6)))) (Nu
m 8)) (Num 9))))))) (Num 5)) (Add (Add (Num 1) (Add (Mul (Num 2) (Mul (Num 5) (M
ul (Num 8) (Num 3)))) (Num 2))) (Num 10)))) (Add (Add (Num 10) (Num 7)) (Add (Nu
m 9) (Num 7)))) (Mul (Mul (Num 10) (Num 5)) (Num 4))) (Num 1))) (Num 10)) (Mul (
Num 9) (Mul (Num 7) (Num 8)))) (Num 3))) (Num 5)) (Add (Mul (Num 3) (Num 10)) (M
ul (Num 5) (Num 7))))) (Mul (Add (Num 5) (Mul (Mul (Num 2) (Add (Num 1) (Add (Ad
d (Add (Add (Num 9) (Add (Mul (Num 2) (Mul (Num 10) (Num 6))) (Num 1))) (Add (Nu
m 2) (Mul (Add (Num 4) (Num 3)) (Add (Mul (Num 7) (Add (Mul (Mul (Num 7) (Num 6)
) (Num 9)) (Add (Mul (Num 10) (Add (Mul (Num 7) (Num 7)) (Mul (Mul (Add (Add (Nu
m 7) (Mul (Num 7) (Add (Num 8) (Mul (Add (Num 9) (Add (Mul (Num 1) (Mul (Add (Mu
l (Mul (Mul (Num 9) (Num 6)) (Num 4)) (Mul (Num 1) (Mul (Num 1) (Num 10)))) (Num
 1)) (Add (Add (Num 1) (Num 8)) (Num 10)))) (Num 9))) (Mul (Num 3) (Num 3)))))) 
(Add (Num 8) (Num 10))) (Mul (Mul (Mul (Mul (Num 2) (Num 8)) (Add (Num 3) (Num 1
))) (Num 10)) (Add (Num 5) (Mul (Add (Num 5) (Num 5)) (Mul (Mul (Add (Mul (Add (
Num 7) (Add (Num 9) (Add (Mul (Add (Add (Add (Add (Num 7) (Add (Add (Add (Add (Num 3)
 (Mul (Num 7) (Num 2))) (Add (Add (Mul (Add (Mul (Num 3) (Num 5)) (Add (Add (Mul
 (Num 3) (Num 3)) (Mul (Num 7) (Num 7))) (Num 3))) (Num 6)) (Add (Num 10) (Num 2
))) (Num 3))) (Num 2)) (Num 10))) (Num 9)) (Num 3)) (Add (Add (Num 3) (Num 1)) (
Mul (Add (Mul (Num 2) (Num 5)) (Add (Mul (Add (Num 6) (Num 2)) (Num 5)) (Num 5))
) (Mul (Mul (Num 1) (Mul (Num 7) (Num 7))) (Num 5))))) (Add (Mul (Num 5) (Num 1)
) (Num 6))))) (Num 8)) (Mul (Add (Num 4) (Num 10)) (Num 7))) (Num 10)) (Num 5)))
))) (Add (Num 7) (Num 10))))) (Num 5)))) (Num 8))))) (Mul (Add (Num 3) (Num 6)) 
(Mul (Add (Mul (Num 7) (Mul (Add (Mul (Num 2) (Mul (Mul (Num 2) (Mul (Num 3) (Nu
m 9))) (Num 3))) (Num 3)) (Add (Mul (Num 3) (Num 1)) (Add (Add (Add (Mul (Mul (N
um 6) (Add (Num 9) (Num 1))) (Num 3)) (Add (Mul (Add (Num 10) (Num 3)) (Mul (Add
```

(Add (Add (Num 1) (Add (Mul (Add (Mul (Num 4) (Num 7)) (Add (Num 10) (Num 6))) (Num 5)) (Num 7))) (Add (Mul (Mul (Num 3) (Mul (Add (Num 8) (Num 6)) (Num 4))) (Add (Num 1) (Num 7))) (Mul (Num 5) (Num 4)))) (Add (Mul (Num 3) (Num 4)) (Add (Mul (Mul (Add (Num 4) (Mul (Num 1) (Num 3))) (Mul (Num 10) (Num 2))) (Num 4)) (Num 9)))) (Num 3))) (Num 1))) (Num 2)) (Num 9))))) (Add (Add (Add (Num 9) (Num 5)) (Add (Num 7) (Num 10))) (Num 10))) (Num 2)))) (Add (Num 1) (Num 3))))) (Num 10))) (Mul (Num 4) (Add (Add (Mul (Num 4) (Num 8)) (Add (Add (Num 8) (Num 5)) (Num 9))) (Num 10))))) (Mul (Num 8) (Num 1))))) (Num 7))))) (Num 9)) (Add (Add (Num 2) (Num 4)) (Mul (Add (Add (Mul (Add (Add (Num 8) (Num 7)) (Num 2)) (Add (Mul (Add (Add (Add (Mul (Mul (Num 5) (Mul (Mul (Num 3) (Mul (Num 3) (Num 10))) (Mul (Num 2) (Mul (Num 10) (Mul (Mul (Num 1) (Num 10)) (Num 3))))))) (Mul (Num 6) (Num 2))) (Add (Add (Num 1) (Mul (Mul (Num 7) (Mul (Num 9) (Mul (Mul (Mul (Num 2) (Add (Num 8) (Mul (Num 3) (Mul (Mul (Add (Mul (Num 2) (Num 8)) (Num 6)) (Num 1)) (Num 1))))) (Num 3)) (Mul (Num 8) (Mul (Mul (Mul (Num 4) (Add (Add (Add (Num 3) (Mul (Mul (Num 8) (Mul (Num 3) (Num 6))) (Num 8))) (Num 4)) (Add (Mul (Add (Mul (Mul (Add (Num 3) (Num 2)) (Num 4)) (Num 2)) (Num 2)) (Num 2)) (Add (Add (Num 9) (Mul (Add (Mul (Num 4) (Num 6)) (Num 3)) (Add (Num 8) (Num 9)))) (Add (Num 4) (Add (Mul (Num 3) (Mul (Num 3) (Add (Mul (Num 7) (Num 6)) (Mul (Num 6) (Mul (Mul (Add (Mul (Add (Num 9) (Add (Num 9) (Add (Add (Add (Mul (Num 1) (Num 6)) (Add (Num 9) (Num 4))) (Num 3)) (Mul (Num 8) (Add (Mul (Add (Num 5) (Num 5)) (Num 5)) (Num 3)))))) (Num 4)) (Add (Num 1) (Add (Num 6) (Add (Num 5) (Mul (Mul (Mul (Add (Mul (Add (Num 5) (Num 6)) (Mul (Add (Num 1) (Num 7)) (Num 8))) (Num 6)) (Num 8)) (Num 1)) (Num 4)))))) (Num 7)) (Add (Num 9) (Add (Num 10) (Num 10)))))))) (Num 5)))))) (Num 9)) (Num 2)))))) (Num 6))) (Num 9))) (Mul (Num 3) (Add (Num 2) (Num 6)))) (Num 1)) (Add (Mul (Mul (Add (Num 6) (Num 6)) (Num 7)) (Num 3)) (Num 3))) (Mul (Add (Add (Mul (Mul (Num 1) (Mul (Mul (Num 1) (Add (Num 6) (Num 10))) (Add (Num 6) (Mul (Mul (Mul (Mul (Mul (Mul (Num 5) (Mul (Add (Add (Num 7) (Add (Add (Mul (Num 2) (Add (Add (Add (Add (Mul (Num 4) (Add (Num 4) (Num 3))) (Add (Add (Add (Mul (Num 9) (Add (Add (Add (Mul (Mul (Num 9) (Mul (Num 4) (Add (Num 5) (Add (Num 2) (Num 6))))) (Num 10)) (Mul (Mul (Mul (Mul (Add (Add (Num 9) (Mul (Num 2) (Num 5))) (Num 1)) (Add (Num 4) (Num 4))) (Mul (Add (Add (Num 3) (Mul (Num 5) (Mul (Mul (Num 10) (Num 1)) (Add (Mul (Add (Add (Num 1) (Add (Num 7) (Mul (Mul (Mul (Num 3) (Num 9)) (Num 8)) (Num 7)))) (Add (Num 6) (Num 5))) (Mul (Mul (Add (Add (Num 1) (Add (Add (Add (Add (Mul (Num 6) (Add (Num 5) (Mul (Num 3) (Num 3)))) (Num 4)) (Add (Num 9) (Add (Mul (Num 5) (Num 4)) (Mul (Mul (Add (Num 8) (Mul (Num 4) (Num 5))) (Num 1)) (Num 8))))) (Num 6)) (Num 3))) (Add (Add (Num 2) (Num 9)) (Num 7))) (Mul (Num 7) (Add (Num 6) (Num 10)))) (Num 7))) (Add (Mul (Num 10) (Num 9)) (Num 2)))))) (Num 9)) (Add (Mul (Add (Num 4) (Mul (Num 3) (Num 4))) (Num 6)) (Num 10)))) (Mul (Num 2) (Num 10))) (Add (Mul (Add (Mul (Mul (Add (Num 5) (Num 3)) (Mul (Mul (Num 5) (Mul (Num 7) (Num 3))) (Num 5))) (Num 2)) (Add (Num 3) (Add (Mul (Num 8) (Add (Mul (Add (Num 7) (Num 7)) (Mul (Mul (Add (Num 1) (Add (Mul (Num 3) (Num 1)) (Mul (Num 3) (Num 10)))) (Num 2)) (Num 8))) (Num 3))) (Num 1)))) (Num 4)) (Num 7)))) (Add (Num 9) (Num 1))) (Num 8))) (Num 7)) (Add (Mul (Add (Mul (Add (Num 9) (Num 4)) (Add (Num 3) (Num 7))) (Mul (Num 3) (Num 1))) (Num 3)) (Num 4))) (Num 8))) (Num 5)) (Num 7)) (Num 10))) (Num 5)) (Num 3))) (Num 9)) (Mul (Mul (Add (Add (Add (Num 6) (Num 10)) (Num 6)) (Num 3)) (Mul (Mul (Mul (Num 1) (Num 3)) (Mul (Num 2) (Add (Num 6) (Num 8)))) (Add (Num 7) (Add (Add (Num 6) (Num 6)) (Num 2))))) (Num 3)))) (Num 2)) (Add (Num 7) (Num 6))) (Mul (Add (Mul (Num 8) (Num 9)) (Add (Num 6) (Num 1))) (Num 7))) (Add (Mul (Num 4) (Mul (Mul (Num 2) (Add (Num 6) (Add (Mul (Mul (Num 8) (Add (Add (Num 5) (Add (Mul (Num 2) (Num 9)) (Mul (Add (Num 1) (Num 1)) (Mul (Add (Num 10) (Add (Add (Add (Num 8) (Num 8)) (Num 1)) (Mul (Num 8) (Mul (Num 5) (Add (Num 1) (Num 8)))))) (Num 8))))) (Mul (Num 8) (Num 2)))) (Add (Mul (Num 10) (Add (Num 3) (Mul (Num 5) (Num 8)))) (Num 3))) (Mul (Num 10) (Num 1))))) (Mul (Mul (Num 7) (Num 4)) (Mul (Num 6) (Mul

(Num 10) (Add (Mul (Add (Mul (Mul (Num 7) (Num 3)) (Num 8)) (Num 3)) (Num 2)) (Num 10))))))) (Add (Add (Num 1) (Mul (Num 9) (Num 7))) (Num 2)))) (Num 8))))) (Num 2)) (Num 3)) (Mul (Num 9) (Num 3))) (Mul (Num 4) (Mul (Num 1) (Mul (Add (Add (Mul (Add (Num 8) (Mul (Num 10) (Num 6))) (Num 2)) (Add (Add (Add (Add (Add (Add (Add (Add (Add (Mul (Mul (Num 7) (Mul (Mul (Mul (Num 4) (Add (Add (Add (Add (Mul (Num 2) (Num 8)) (Add (Num 8) (Num 1))) (Add (Add (Num 6) (Num 6)) (Mul (Num 8) (Num 3)))) (Num 1)) (Add (Mul (Mul (Mul (Num 5) (Add (Mul (Num 5) (Num 4)) (Num 5))) (Add (Num 6) (Mul (Mul (Mul (Mul (Num 5) (Num 9)) (Num 10)) (Mul (Add (Add (Num 7) (Num 3)) (Num 8)) (Num 9))) (Num 4)))) (Num 7)) (Add (Num 9) (Num 6))))) (Num 2)) (Mul (Num 3) (Num 4)))) (Mul (Add (Add (Add (Add (Num 10) (Num 7)) (Num 4)) (Add (Mul (Add (Add (Num 6) (Mul (Num 7) (Add (Add (Mul (Add (Mul (Mul (Num 10) (Mul (Add (Mul (Add (Mul (Mul (Mul (Mul (Num 10) (Mul (Num 1) (Mul (Num 5) (Num 9)))) (Num 2)) (Num 9)) (Num 7)) (Num 2)) (Add (Num 2) (Mul (Num 2) (Num 10)))) (Add (Add (Num 7) (Num 4)) (Num 8))) (Add (Add (Num 3) (Num 9)) (Num 1)))) (Mul (Add (Mul (Add (Num 10) (Add (Mul (Num 2) (Mul (Num 6) (Num 8))) (Add (Add (Add (Num 2) (Num 3)) (Mul (Num 6) (Num 2))) (Mul (Num 1) (Add (Mul (Num 9) (Num 4)) (Num 8)))))) (Mul (Num 8) (Mul (Mul (Mul (Add (Num 9) (Mul (Num 4) (Num 3))) (Num 1)) (Add (Mul (Num 3) (Num 10)) (Num 10))) (Add (Add (Mul (Mul (Num 5) (Num 8)) (Add (Num 6) (Mul (Add (Mul (Num 10) (Mul (Mul (Mul (Num 3) (Mul (Add (Add (Num 6) (Mul (Num 9) (Add (Num 1) (Num 4)))) (Num 3)) (Num 2))) (Add (Add (Add (Num 1) (Add (Num 2) (Mul (Add (Add (Add (Mul (Add (Add (Num 2) (Num 5)) (Num 8)) (Mul (Add (Mul (Num 9) (Num 2)) (Num 5)) (Num 3))) (Add (Mul (Num 9) (Num 5)) (Add (Num 2) (Num 10)))) (Num 7)) (Add (Mul (Num 7) (Mul (Mul (Add (Num 1) (Num 4)) (Mul (Add (Add (Num 6) (Mul (Mul (Add (Num 3) (Add (Add (Num 6) (Num 1)) (Mul (Num 10) (Mul (Num 4) (Num 9))))) (Mul (Add (Num 1) (Num 9)) (Num 2))) (Add (Mul (Num 1) (Mul (Mul (Num 10) (Num 7)) (Add (Num 3) (Num 6)))) (Add (Num 9) (Num 10))) (Mul (Mul (Num 5) (Mul (Num 1) (Num 5))) (Num 6))))) (Num 9)) (Num 6)) (Num 7))) (Num 3))) (Add (Num 1) (Num 8))))) (Mul (Add (Add (Num 7) (Add (Num 9) (Num 5))) (Mul (Add (Num 8) (Mul (Num 3) (Mul (Num 3) (Add (Add (Add (Num 1) (Num 5)) (Mul (Num 9) (Num 4))) (Num 8))))) (Num 1))) (Mul (Num 10) (Num 3)))) (Add (Num 9) (Mul (Mul (Mul (Num 4) (Num 7)) (Num 2)) (Num 3))))) (Num 5))) (Mul (Num 2) (Add (Num 2) (Num 5)))) (Num 1)))) (Num 2)) (Add (Num 1) (Num 1)))))) (Num 1)) (Num 3))) (Num 6)) (Add (Num 4) (Mul (Add (Mul (Mul (Num 7) (Num 2)) (Add (Add (Num 9) (Add (Add (Num 9) (Num 4)) (Num 4))) (Num 2))) (Add (Mul (Mul (Mul (Num 6) (Num 7)) (Num 9)) (Mul (Add (Num 10) (Num 1)) (Mul (Num 6) (Num 3)))) (Add (Add (Num 8) (Num 3)) (Add (Mul (Num 2) (Mul (Num 7) (Num 8))) (Add (Add (Add (Num 10) (Num 9)) (Num 3)) (Num 5)))))) (Mul (Num 7) (Num 1))))) (Mul (Mul (Num 8) (Num 1)) (Add (Add (Num 6) (Mul (Add (Add (Num 1) (Num 8)) (Num 6)) (Num 7))) (Add (Num 7) (Num 3))))) (Num 5)))) (Num 1)) (Mul (Add (Num 5) (Mul (Num 8) (Add (Mul (Mul (Add (Add (Mul (Add (Add (Num 3) (Mul (Num 2) (Add (Add (Num 8) (Mul (Mul (Add (Num 2) (Add (Add (Add (Add (Num 9) (Num 8)) (Mul (Add (Num 10) (Num 7)) (Num 5))) (Num 5)) (Num 8))) (Num 4)) (Num 10))) (Add (Mul (Num 4) (Add (Num 8) (Add (Num 4) (Mul (Num 10) (Add (Num 8) (Num 7)))))) (Mul (Add (Num 1) (Num 8)) (Mul (Num 8) (Num 10))))))) (Add (Num 4) (Num 6))) (Mul (Num 6) (Add (Num 8) (Num 3)))) (Add (Add (Add (Num 5) (Mul (Num 4) (Num 4))) (Mul (Num 8) (Add (Num 10) (Add (Mul (Num 10) (Num 8)) (Num 7))))) (Add (Num 6) (Num 7)))) (Mul (Add (Add (Add (Num 10) (Add (Add (Num 5) (Num 3)) (Add (Num 3) (Num 4)))) (Num 1)) (Mul (Num 9) (Num 1))) (Num 2))) (Mul (Add (Mul (Num 9) (Mul (Mul (Num 9) (Num 10)) (Num 1))) (Mul (Num 2) (Mul (Add (Mul (Num 7) (Add (Num 5) (Mul (Num 10) (Mul (Add (Num 6) (Num 5)) (Mul (Mul (Add (Mul (Num 9) (Add (Add (Mul (Num 3) (Num 3)) (Add (Mul (Mul (Mul (Mul (Add (Num 5) (Add (Num 6) (Num 9))) (Num 2)) (Mul (Add (Num 1) (Num 4)) (Add (Num 8) (Num 6)))) (Add (Add (Num 1) (Num 1)) (Add (Num 6) (Add (Mul (Mul (Num 2) (Mul (Mul (Add (Num 10) (Num 1)) (Num 3)) (Num 9))) (Mul (Mul (Mul (Mul (Add (Num 10) (Num 3)) (Num 6)) (Num 9)) (Num 3)) (Num 7)))

(Mul (Num 1) (Mul (Add (Num 9) (Mul (Add (Mul (Mul (Add (Add (Num 4) (Num 4)) (
Num 2)) (Mul (Num 9) (Num 4))) (Add (Add (Mul (Add (Num 4) (Add (Add (Add (Mul (
Num 6) (Num 10)) (Mul (Mul (Num 8) (Num 7)) (Num 9))) (Mul (Mul (Add (Mul (Mul (
Num 7) (Add (Num 3) (Num 6))) (Mul (Add (Mul (Num 6) (Mul (Num 4) (Num 9))) (Mul
 (Num 8) (Num 3))) (Mul (Mul (Add (Num 1) (Add (Num 4) (Num 1))) (Add (Num 3) (M
ul (Num 10) (Num 1)))) (Num 3)))) (Mul (Num 5) (Add (Add (Add (Num 2) (Num 2)) (
Add (Mul (Num 5) (Mul (Num 10) (Num 3))) (Num 6))) (Num 8)))) (Num 2)) (Num 3)))
 (Add (Num 3) (Mul (Num 10) (Num 1))))) (Mul (Mul (Mul (Mul (Num 1) (Mul (Add (M
ul (Add (Mul (Add (Num 2) (Num 9)) (Num 3)) (Add (Mul (Add (Num 3) (Num 3)) (Num
 2)) (Mul (Num 5) (Num 5)))) (Mul (Num 2) (Num 7))) (Num 9)) (Mul (Num 4) (Num 9
)))) (Mul (Add (Num 7) (Num 6)) (Num 3))) (Num 2)) (Add (Num 3) (Num 7)))) (Add
(Num 4) (Num 3))) (Num 10))) (Num 6)) (Num 9))) (Num 8)))))) (Mul (Num 5) (Add
(Num 1) (Num 3)))) (Add (Add (Num 2) (Num 1)) (Num 9)))) (Mul (Num 7) (Add (Num
6) (Num 9))))) (Num 5)) (Num 1)) (Add (Mul (Add (Mul (Num 3) (Num 9)) (Mul (Num
5) (Num 7))) (Add (Num 8) (Mul (Add (Num 10) (Num 9)) (Num 5)))) (Mul (Num 3) (M
ul (Num 3) (Num 6)))))))) (Num 2)) (Num 2)))) (Mul (Num 6) (Num 6)))) (Num 1))
(Add (Mul (Num 8) (Add (Num 1) (Add (Num 7) (Mul (Mul (Num 2) (Num 1)) (Num 5)))
)) (Add (Mul (Mul (Add (Add (Num 1) (Num 1)) (Num 5)) (Num 5)) (Mul (Add (Num 3)
 (Num 3)) (Add (Add (Mul (Add (Add (Num 7) (Num 6)) (Num 1)) (Num 2)) (Add (Add
(Mul (Add (Mul (Num 8) (Num 8)) (Num 5)) (Mul (Num 7) (Add (Num 10) (Mul (Mul (N
um 10) (Add (Num 3) (Num 3))) (Add (Add (Num 3) (Num 10)) (Add (Num 3) (Mul (Num
 9) (Num 9)))))))) (Mul (Num 5) (Num 8))) (Num 6))) (Num 4)))) (Mul (Num 4) (Num
 6))))))) (Num 7))) (Add (Mul (Num 4) (Mul (Num 6) (Num 6))) (Mul (Add (Num 3) (
Num 4)) (Mul (Num 9) (Mul (Num 4) (Num 5))))))) (Add (Add (Add (Num 8) (Num 1))
(Num 8)) (Num 10))) (Num 6))) (Add (Num 1) (Num 10))) (Add (Mul (Num 1) (Mul (Ad
d (Num 2) (Num 3)) (Add (Add (Mul (Num 2) (Num 7)) (Mul (Num 4) (Mul (Add (Add (
Mul (Num 1) (Num 1)) (Add (Mul (Num 3) (Num 2)) (Num 2))) (Num 6)) (Mul (Num 2)
(Num 6))))) (Mul (Mul (Num 9) (Num 8)) (Num 3))))) (Num 1))) (Num 10)) (Mul (Add
 (Num 4) (Num 10)) (Num 3))) (Mul (Add (Num 5) (Add (Add (Add (Num 3) (Mul (Add
(Num 4) (Num 6)) (Num 5))) (Add (Mul (Num 2) (Add (Num 6) (Num 4))) (Num 9))) (N
um 9))) (Add (Mul (Num 10) (Mul (Add (Add (Mul (Add (Mul (Mul (Mul (Mul (Num 7)
(Add (Add (Num 8) (Mul (Num 10) (Mul (Num 3) (Add (Add (Add (Num 10) (Mul (Num 5
) (Mul (Add (Num 6) (Add (Num 9) (Num 5))) (Num 3)))) (Num 7)) (Add (Mul (Add (N
um 6) (Num 3)) (Mul (Mul (Num 2) (Num 7)) (Add (Mul (Num 10) (Mul (Num 2) (Num 1
0))) (Mul (Mul (Mul (Num 2) (Num 9)) (Mul (Mul (Mul (Add (Add (Add (Num 3) (Add
(Num 9) (Mul (Num 6) (Add (Mul (Num 8) (Num 2)) (Mul (Add (Num 1) (Mul (Add (Mul
 (Add (Mul (Mul (Mul (Num 3) (Mul (Num 1) (Mul (Num 2) (Num 4)))) (Num 7)) (Num
9)) (Add (Num 9) (Num 10))) (Mul (Add (Num 1) (Num 8)) (Add (Num 4) (Mul (Add (N
um 2) (Num 10)) (Num 10))))) (Num 5)) (Mul (Add (Mul (Num 3) (Num 10)) (Num 2))
(Num 2)))) (Add (Num 7) (Num 2))))))) (Num 8)) (Add (Num 10) (Mul (Num 9) (Num 6
)))) (Add (Mul (Add (Num 1) (Num 3)) (Num 10)) (Mul (Num 9) (Num 3)))) (Num 10))
 (Num 1))) (Num 7))))) (Mul (Num 1) (Mul (Add (Add (Add (Num 10) (Mul (Num 5) (M
ul (Add (Num 6) (Add (Num 6) (Num 1))) (Mul (Num 8) (Num 1))))) (Mul (Add (Mul (
Num 1) (Num 4)) (Num 9)) (Mul (Mul (Num 5) (Mul (Num 10) (Mul (Mul (Add (Add (Ad
d (Num 3) (Add (Mul (Num 5) (Mul (Mul (Add (Num 6) (Add (Num 4) (Add (Mul (Num 5
) (Num 4)) (Num 4)))) (Num 8)) (Add (Mul (Num 2) (Mul (Add (Num 6) (Num 1)) (Mul
 (Add (Add (Num 7) (Num 1)) (Add (Mul (Add (Add (Num 2) (Add (Num 4) (Mul (Num 3
) (Mul (Mul (Mul (Num 10) (Mul (Add (Num 2) (Add (Mul (Add (Mul (Add (Num 7) (Nu
m 6)) (Num 10)) (Num 10)) (Mul (Num 6) (Num 3))) (Add (Num 8) (Add (Num 5) (Num
10))))) (Add (Num 7) (Num 4)))) (Add (Num 10) (Add (Add (Num 1) (Mul (Mul (Num 4
) (Add (Num 10) (Mul (Num 7) (Num 2)))) (Num 10))) (Add (Num 2) (Num 1))))) (Num
 5))))) (Mul (Num 3) (Num 3))) (Num 5)) (Mul (Num 3) (Add (Add (Num 5) (Num 10))
 (Add (Mul (Mul (Add (Num 4) (Num 6)) (Add (Add (Mul (Num 1) (Add (Mul (Add (Add
 (Num 5) (Mul (Mul (Add (Num 9) (Add (Num 2) (Mul (Add (Num 10) (Add (Mul (Num 7

) (Mul (Mul (Mul (Num 3) (Num 10)) (Add (Num 9) (Add (Num 5) (Add (Add (Add (Mul
 (Num 7) (Mul (Num 3) (Add (Num 5) (Add (Num 8) (Num 4))))) (Mul (Num 2) (Num 5)
)) (Add (Mul (Add (Add (Mul (Num 4) (Mul (Add (Mul (Num 2) (Num 2)) (Mul (Num 7)
 (Num 3))) (Num 2))) (Add (Mul (Add (Mul (Add (Mul (Num 6) (Num 10)) (Add (Add (
Add (Add (Add (Mul (Num 4) (Num 5)) (Num 5)) (Mul (Num 2) (Num 6))) (Num 1)) (Nu
m 7)) (Add (Num 9) (Num 9)))) (Num 1)) (Mul (Add (Add (Num 2) (Mul (Mul (Add (Ad
d (Add (Num 8) (Num 10)) (Num 8)) (Mul (Num 8) (Mul (Num 1) (Mul (Mul (Add (Num
2) (Num 3)) (Num 10)) (Add (Num 1) (Num 3)))))) (Add (Num 3) (Add (Add (Mul (Num
 3) (Num 6)) (Num 6)) (Num 6)))) (Mul (Num 1) (Num 3)))) (Num 9)) (Num 9))) (Num
 5)) (Num 10))) (Num 3)) (Num 8)) (Add (Mul (Mul (Add (Num 5) (Mul (Mul (Num 9)
(Num 8)) (Add (Mul (Mul (Add (Num 5) (Add (Add (Add (Mul (Num 10) (Num 4)) (Num
10)) (Num 5)) (Num 10))) (Num 6)) (Num 2)) (Add (Num 10) (Add (Mul (Num 1) (Num
6)) (Num 3)))))) (Num 5)) (Num 7)) (Num 1)))) (Mul (Num 8) (Add (Mul (Num 8) (Nu
m 6)) (Num 3))))))) (Num 2))) (Num 2))) (Mul (Num 6) (Add (Add (Num 8) (Num 9))
(Mul (Add (Add (Num 2) (Num 10)) (Num 2)) (Add (Add (Mul (Num 9) (Num 5)) (Num 7
)) (Add (Add (Num 10) (Mul (Add (Mul (Mul (Num 7) (Mul (Mul (Num 8) (Add (Mul (A
dd (Num 3) (Mul (Mul (Num 3) (Add (Num 1) (Num 5))) (Add (Num 8) (Num 9)))) (Add
 (Mul (Mul (Num 4) (Num 10)) (Mul (Add (Add (Num 3) (Mul (Add (Add (Num 4) (Mul
(Num 6) (Num 3))) (Mul (Mul (Add (Add (Num 3) (Mul (Add (Num 7) (Mul (Add (Mul (
Add (Num 9) (Add (Add (Mul (Mul (Num 8) (Mul (Add (Add (Add (Mul (Mul (Mul (Add
(Add (Num 1) (Add (Num 8) (Add (Num 2) (Num 7)))) (Add (Add (Mul (Num 4) (Add (A
dd (Num 8) (Mul (Num 8) (Add (Add (Mul (Mul (Add (Add (Num 7) (Num 10)) (Num 2))
 (Add (Mul (Num 1) (Add (Add (Add (Num 9) (Num 5)) (Num 3)) (Num 8))) (Add (Mul
(Add (Num 3) (Num 5)) (Num 3)) (Num 10)))) (Mul (Num 5) (Num 6))) (Num 10)) (Add
 (Num 8) (Num 10))))) (Num 4))) (Num 6)) (Add (Add (Mul (Mul (Mul (Add (Mul (Num
 1) (Add (Num 4) (Num 6))) (Num 2)) (Mul (Mul (Mul (Mul (Mul (Mul (Num 1) (Add (
Num 2) (Mul (Add (Num 7) (Add (Num 1) (Add (Num 9) (Add (Mul (Add (Mul (Num 2) (
Add (Add (Mul (Mul (Mul (Num 10) (Add (Num 4) (Num 3))) (Num 10)) (Mul (Mul (Num
 4) (Add (Mul (Num 2) (Num 2)) (Add (Add (Add (Num 10) (Num 2)) (Num 5)) (Add (M
ul (Num 1) (Num 8)) (Add (Num 6) (Mul (Num 6) (Num 1))))))) (Num 6))) (Num 7)) (
Add (Mul (Num 9) (Mul (Add (Num 2) (Num 10)) (Add (Add (Num 9) (Add (Add (Mul (N
um 6) (Num 10)) (Num 1)) (Mul (Mul (Add (Add (Mul (Num 9) (Num 5)) (Mul (Num 8)
(Num 4))) (Add (Mul (Num 7) (Add (Mul (Num 9) (Num 8)) (Add (Num 1) (Add (Add (M
ul (Add (Num 6) (Num 3)) (Num 8)) (Num 9)) (Num 1))))) (Mul (Mul (Num 2) (Num 10
)) (Add (Num 4) (Num 5))))) (Num 7)) (Mul (Add (Add (Mul (Add (Num 4) (Mul (Num
10) (Num 4))) (Mul (Mul (Add (Add (Mul (Num 10) (Mul (Num 8) (Num 4))) (Num 7))
(Num 10)) (Num 5)) (Num 2))) (Num 2)) (Num 4)) (Add (Num 10) (Mul (Num 7) (Mul (
Add (Num 9) (Mul (Add (Num 2) (Num 4)) (Num 9))) (Add (Add (Num 4) (Num 10)) (Mu
l (Num 1) (Mul (Num 6) (Add (Add (Add (Mul (Mul (Add (Add (Num 6) (Mul (Num 5) (
Num 6))) (Num 7)) (Add (Add (Num 4) (Mul (Num 6) (Num 3))) (Num 7))) (Num 7)) (A
dd (Num 8) (Mul (Mul (Num 1) (Mul (Num 5) (Mul (Mul (Num 2) (Num 6)) (Add (Mul (
Num 4) (Num 6)) (Num 6))))) (Mul (Add (Num 3) (Mul (Mul (Add (Mul (Num 9) (Num 3
)) (Num 8)) (Num 7)) (Mul (Num 9) (Num 8)))) (Num 4))))) (Num 5)) (Num 2))))))))
)))) (Mul (Add (Num 9) (Add (Num 1) (Num 8))) (Num 6))))) (Num 10)))) (Num 10))
(Num 4)) (Num 7))))) (Num 10)))) (Add (Num 7) (Mul (Num 9) (Num 6)))) (Add (Num
3) (Num 9))) (Mul (Add (Add (Add (Add (Num 6) (Add (Num 3) (Num 1))) (Num 7)) (Num 7)
) (Mul (Add (Add (Mul (Mul (Add (Add (Mul (Num 7) (Num 10)) (Num 7)) (Mul (Num 8
) (Num 2))) (Num 6)) (Num 6)) (Num 7)) (Mul (Add (Num 6) (Num 1)) (Mul (Num 5) (
Num 6)))) (Add (Num 1) (Mul (Num 5) (Num 1)))))) (Num 8)) (Num 8))) (Mul (Mul (A
dd (Add (Add (Add (Num 1) (Num 3)) (Num 2)) (Add (Num 10) (Num 5))) (Mul (Add (Mul (N
um 5) (Num 8)) (Mul (Add (Num 6) (Num 1)) (Num 3))) (Mul (Num 2) (Num 10)))) (Nu
m 4))) (Mul (Add (Add (Add (Add (Add (Mul (Num 9) (Num 2)) (Add (Num 9) (Num 6))) (Nu
m 4)) (Add (Num 9) (Mul (Num 9) (Num 4)))) (Add (Add (Num 2) (Mul (Mul (Num 1) (
Add (Num 10) (Num 1))) (Num 6))) (Add (Add (Add (Num 3) (Add (Mul (Num 3) (Num 2

)) (Num 8))) (Num 1)) (Add (Num 2) (Add (Num 9) (Num 5)))))) (Mul (Add (Mul (Num
 2) (Num 2)) (Num 9)) (Mul (Add (Num 8) (Num 10)) (Num 4))))) (Mul (Num 10) (Add
 (Mul (Num 5) (Add (Add (Mul (Mul (Add (Mul (Num 2) (Add (Add (Num 4) (Mul (Add
(Num 9) (Add (Mul (Num 1) (Num 2)) (Num 1))) (Num 6))) (Mul (Num 5) (Num 8)))) (
Num 9)) (Num 5)) (Mul (Mul (Num 8) (Mul (Num 5) (Num 9))) (Num 5))) (Num 2)) (Mu
l (Add (Num 4) (Mul (Num 6) (Add (Num 5) (Num 3)))) (Mul (Num 5) (Mul (Num 8) (N
um 5)))))) (Num 4)))) (Add (Num 1) (Num 10))))) (Add (Add (Num 7) (Num 9)) (Num
6))) (Mul (Num 6) (Num 10))) (Num 2)) (Num 8)) (Num 9)) (Num 7)) (Mul (Add (Num
4) (Num 2)) (Mul (Add (Num 9) (Num 3)) (Add (Num 10) (Num 5)))))) (Num 5)) (Add
(Add (Num 1) (Num 6)) (Mul (Add (Mul (Num 8) (Num 2)) (Mul (Mul (Num 10) (Mul (N
um 3) (Num 3))) (Num 1))) (Mul (Num 2) (Add (Num 4) (Mul (Add (Num 4) (Num 10))
(Num 6)))))) (Num 6))) (Num 5)) (Mul (Add (Num 4) (Mul (Add (Num 1) (Mul (Mul (
Mul (Add (Mul (Mul (Num 5) (Mul (Num 2) (Mul (Num 5) (Num 8)))) (Num 3)) (Add (N
um 8) (Num 9))) (Num 6)) (Num 6)) (Add (Mul (Add (Num 2) (Mul (Mul (Add (Num 6)
(Num 8)) (Add (Mul (Num 1) (Num 6)) (Mul (Add (Add (Num 8) (Add (Num 4) (Num 7))
) (Add (Mul (Num 5) (Num 7)) (Num 1))) (Mul (Mul (Add (Mul (Mul (Num 2) (Add (Mu
l (Add (Num 9) (Mul (Add (Add (Num 9) (Num 10)) (Num 2)) (Add (Mul (Mul (Mul (Nu
m 9) (Add (Add (Num 9) (Mul (Num 7) (Mul (Num 5) (Add (Mul (Num 6) (Add (Num 1)
(Num 8))) (Num 4))))) (Num 6))) (Add (Num 8) (Num 4))) (Add (Mul (Add (Num 8) (A
dd (Num 8) (Num 7))) (Mul (Num 10) (Mul (Mul (Num 5) (Num 2)) (Num 7)))) (Num 3)
)) (Num 3)))) (Num 8)) (Add (Num 9) (Num 2)))) (Add (Mul (Mul (Num 7) (Add (Mul
(Add (Add (Num 1) (Add (Num 5) (Add (Num 8) (Mul (Mul (Mul (Num 2) (Num 4)) (Mul
 (Num 10) (Num 9))) (Num 7))))) (Num 10)) (Mul (Num 9) (Num 7))) (Add (Num 3) (N
um 3)))) (Add (Add (Mul (Num 10) (Mul (Num 2) (Num 9))) (Num 4)) (Add (Mul (Add
(Mul (Num 4) (Add (Num 10) (Mul (Add (Num 1) (Num 3)) (Mul (Num 6) (Num 10))))))
(Num 6)) (Num 3)) (Num 7)))) (Num 4))) (Num 9)) (Mul (Num 5) (Num 1))) (Num 3)))
)) (Num 2))) (Num 10)) (Num 4)))) (Add (Num 3) (Add (Num 2) (Num 7))))) (Num 3))
) (Num 10))) (Num 1))) (Add (Add (Num 1) (Add (Add (Mul (Num 7) (Num 10)) (Mul (
Add (Num 1) (Num 8)) (Num 7))) (Mul (Num 1) (Mul (Num 1) (Mul (Add (Num 3) (Num
5)) (Add (Mul (Num 7) (Num 6)) (Num 3)))))) (Num 6))) (Num 7)) (Num 5))) (Num 6
))) (Mul (Num 3) (Num 7))) (Add (Mul (Num 8) (Mul (Mul (Add (Num 10) (Add (Num 9
) (Add (Num 3) (Num 9)))) (Num 3)) (Num 5))) (Mul (Num 9) (Add (Mul (Num 3) (Add
 (Add (Num 9) (Mul (Mul (Add (Add (Num 6) (Add (Num 6) (Num 5))) (Num 3)) (Mul (
Num 9) (Add (Add (Add (Mul (Mul (Mul (Num 6) (Add (Num 8) (Num 9))) (Num 5)) (Nu
m 8)) (Add (Num 5) (Num 6))) (Num 2)) (Add (Add (Num 2) (Mul (Mul (Num 1) (Mul (
Num 6) (Add (Mul (Num 7) (Add (Num 3) (Num 9))) (Add (Num 1) (Mul (Mul (Mul (Mul
 (Add (Mul (Add (Num 9) (Num 3)) (Num 1)) (Add (Num 5) (Num 5))) (Add (Num 7) (M
ul (Mul (Add (Mul (Add (Num 8) (Num 5)) (Add (Num 3) (Num 9))) (Num 9)) (Num 5))
 (Mul (Num 4) (Num 3))))) (Mul (Mul (Num 9) (Mul (Num 3) (Add (Num 9) (Num 5))))
 (Num 3))) (Add (Num 5) (Add (Mul (Num 8) (Mul (Num 5) (Add (Num 1) (Add (Mul (N
um 2) (Add (Num 6) (Mul (Add (Num 5) (Num 8)) (Mul (Num 7) (Num 3))))) (Add (Num
 8) (Num 6)))))) (Num 5)))) (Num 9)))))) (Mul (Num 4) (Num 7)))) (Num 8))))) (Nu
m 2))) (Num 9))) (Add (Mul (Mul (Mul (Mul (Num 2) (Mul (Mul (Num 2) (Mul (Num 5)
 (Mul (Add (Mul (Add (Add (Add (Mul (Num 4) (Num 6)) (Mul (Num 5) (Add (Num 4) (Num 6
)))) (Add (Mul (Num 4) (Add (Mul (Num 5) (Mul (Add (Add (Add (Num 8) (Mul (Add (
Num 9) (Num 8)) (Add (Add (Num 5) (Num 10)) (Mul (Mul (Mul (Add (Add (Num 1) (Nu
m 4)) (Num 8)) (Mul (Mul (Num 10) (Num 1)) (Add (Add (Mul (Num 4) (Num 3)) (Num
1)) (Add (Num 4) (Num 4))))) (Num 2)) (Mul (Num 9) (Add (Mul (Mul (Num 10) (Add
(Num 8) (Num 6))) (Num 2)) (Add (Num 6) (Num 9))))))))) (Mul (Num 2) (Add (Add (A
dd (Add (Add (Num 1) (Num 7)) (Num 1)) (Num 3)) (Mul (Num 4) (Num 3))) (Add (Add
 (Mul (Num 2) (Num 3)) (Num 8)) (Mul (Mul (Num 3) (Num 5)) (Mul (Add (Num 3) (Nu
m 5)) (Num 9)))))))) (Num 3)) (Mul (Add (Add (Add (Mul (Num 9) (Num 9)) (Num 1))
(Add (Add (Num 9) (Num 5)) (Num 3))) (Num 6)) (Mul (Num 8) (Num 8))))) (Mul (Mul
 (Add (Num 7) (Num 5)) (Num 3)) (Add (Num 8) (Add (Add (Num 3) (Mul (Mul (Mul (N

um 1) (Num 1)) (Num 5)) (Add (Mul (Num 8) (Num 7)) (Num 5)))) (Mul (Add (Num 5) (Add (Mul (Mul (Add (Mul (Num 6) (Num 1)) (Num 8)) (Add (Mul (Num 7) (Add (Num 6) (Add (Num 5) (Mul (Num 10) (Mul (Num 5) (Mul (Mul (Mul (Mul (Mul (Mul (Mul (Mul (Add (Mul (Mul (Mul (Mul (Add (Mul (Add (Mul (Mul (Num 4) (Add (Mul (Num 7) (Mul (Num 8) (Mul (Num 8) (Num 2)))) (Num 7))) (Num 3)) (Mul (Num 9) (Mul (Mul (Num 8) (Num 1)) (Num 8)))) (Num 9)) (Add (Mul (Add (Add (Add (Add (Add (Add (Mul (Num 4) (Num 6)) (Mul (Num 1) (Num 4))) (Num 2)) (Mul (Add (Num 10) (Num 7)) (Num 5))) (Num 7)) (Num 8)) (Num 6)) (Mul (Mul (Add (Num 2) (Num 6)) (Mul (Num 10) (Mul (Num 3) (Num 10)))) (Num 2)))) (Add (Num 1) (Mul (Mul (Add (Add (Add (Num 10) (Num 3)) (Num 5)) (Num 10)) (Add (Num 7) (Mul (Add (Mul (Num 9) (Num 9)) (Add (Num 2) (Num 2))) (Num 1)))) (Add (Num 8) (Add (Mul (Num 4) (Num 10)) (Mul (Num 7) (Mul (Num 7) (Num 10)))))))) (Num 1)) (Mul (Mul (Num 8) (Num 9)) (Mul (Mul (Mul (Add (Num 4) (Add (Num 8) (Num 10))) (Mul (Num 5) (Mul (Num 10) (Mul (Add (Num 1) (Add (Mul (Num 1) (Num 3)) (Add (Mul (Num 7) (Mul (Add (Mul (Num 8) (Mul (Num 1) (Num 2))) (Mul (Mul (Num 6) (Add (Num 2) (Add (Num 7) (Add (Add (Num 7) (Num 10)) (Num 4))))) (Add (Num 10) (Mul (Mul (Num 8) (Mul (Num 7) (Mul (Num 9) (Mul (Num 4) (Num 10))))) (Num 10))))) (Mul (Mul (Num 8) (Mul (Mul (Num 9) (Num 5)) (Add (Num 8) (Mul (Mul (Num 6) (Num 6)) (Add (Add (Num 10) (Num 10)) (Add (Num 9) (Num 9))))))) (Num 3)))) (Num 1)))) (Num 2))))) (Num 9)) (Num 7)))) (Num 5)) (Num 4)) (Mul (Num 9) (Num 4))) (Add (Add (Num 1) (Add (Mul (Num 8) (Add (Num 1) (Mul (Num 10) (Num 9)))) (Add (Num 10) (Mul (Mul (Num 10) (Num 10)) (Num 9))))) (Num 7))) (Mul (Num 8) (Num 4))) (Add (Num 9) (Mul (Num 2) (Mul (Add (Num 3) (Num 8)) (Num 6))))) (Add (Num 8) (Num 4))) (Num 5)) (Mul (Num 4) (Mul (Add (Num 9) (Add (Mul (Add (Add (Num 1) (Num 4)) (Num 2)) (Num 6)) (Add (Mul (Mul (Num 2) (Num 1)) (Mul (Num 3) (Num 2))) (Add (Num 2) (Mul (Mul (Num 9) (Add (Mul (Add (Num 3) (Mul (Num 7) (Mul (Mul (Add (Mul (Num 7) (Add (Num 1) (Mul (Num 7) (Num 7)))) (Num 3)) (Add (Mul (Mul (Add (Mul (Num 10) (Mul (Num 6) (Mul (Num 3) (Add (Num 10) (Add (Add (Add (Num 5) (Add (Add (Mul (Num 2) (Num 1)) (Mul (Num 1) (Num 4))) (Add (Add (Mul (Num 8) (Num 4)) (Mul (Mul (Add (Add (Num 2) (Num 2)) (Add (Num 5) (Num 8))) (Mul (Num 4) (Add (Add (Num 3) (Num 8)) (Num 2)))) (Add (Num 3) (Add (Mul (Mul (Mul (Num 3) (Num 9)) (Mul (Add (Num 5) (Mul (Add (Add (Num 8) (Num 10)) (Add (Mul (Add (Num 10) (Num 3)) (Num 6)) (Num 9))) (Num 8))) (Add (Add (Num 8) (Add (Add (Add (Add (Mul (Mul (Mul (Add (Mul (Add (Num 3) (Num 8)) (Add (Add (Add (Num 2) (Add (Add (Add (Mul (Mul (Add (Num 5) (Num 9)) (Num 6)) (Add (Num 7) (Num 6))) (Num 2)) (Mul (Num 7) (Num 3))) (Num 6))) (Add (Num 5) (Add (Num 3) (Add (Num 9) (Num 7))))) (Add (Num 9) (Num 5)))) (Mul (Add (Num 7) (Add (Add (Add (Num 3) (Num 7)) (Num 7)) (Mul (Add (Num 4) (Mul (Num 10) (Num 9))) (Mul (Mul (Num 5) (Num 2)) (Num 2))))) (Mul (Add (Num 2) (Mul (Num 9) (Add (Mul (Num 10) (Mul (Num 2) (Mul (Add (Num 2) (Mul (Num 8) (Num 9))) (Num 10)))) (Add (Num 3) (Num 9))))) (Num 9)))) (Num 2)) (Add (Add (Num 9) (Num 1)) (Add (Num 9) (Add (Add (Mul (Add (Add (Num 10) (Mul (Add (Add (Num 3) (Num 5)) (Num 5)) (Num 1))) (Num 1)) (Add (Mul (Add (Mul (Num 3) (Num 6)) (Num 5)) (Num 9)) (Num 9))) (Mul (Num 9) (Mul (Mul (Add (Num 9) (Num 8)) (Mul (Num 7) (Mul (Num 1) (Mul (Mul (Num 2) (Add (Mul (Num 1) (Mul (Add (Add (Num 4) (Num 9)) (Add (Num 2) (Num 7))) (Mul (Num 8) (Mul (Num 4) (Num 2))))) (Add (Mul (Num 1) (Add (Num 9) (Num 2))) (Mul (Mul (Num 10) (Num 7)) (Add (Num 2) (Num 4)))))) (Num 5))))) (Num 1)))) (Num 4))))) (Num 7)) (Mul (Mul (Num 9) (Mul (Num 8) (Num 3))) (Mul (Mul (Mul (Num 1) (Add (Num 2) (Num 2))) (Mul (Num 4) (Add (Mul (Add (Mul (Mul (Num 2) (Num 5)) (Add (Mul (Num 4) (Num 10)) (Add (Add (Num 2) (Mul (Mul (Num 7) (Add (Num 7) (Mul (Add (Add (Mul (Num 10) (Mul (Add (Add (Mul (Add (Add (Num 1) (Mul (Num 1) (Mul (Num 2) (Add (Num 9) (Num 8))))) (Num 8)) (Num 8)) (Add (Add (Add (Add (Num 4) (Num 3)) (Mul (Mul (Num 10) (Add (Num 5) (Add (Num 8) (Mul (Add (Num 9) (Add (Num 8) (Num 10))) (Num 8))))) (Num 4))) (Num 6)) (Num 7))) (Num 10)) (Num 7))) (Num 8)) (Mul (Add (Mul (Mul (Num 9) (Num 3)) (Add (Mul (Add (Add (Num 1) (Num 6)) (Num 6)

) (Mul (Num 4) (Num 7))) (Add (Num 4) (Num 8)))) (Add (Mul (Add (Add (Add (Num 9
) (Add (Add (Num 10) (Num 2)) (Num 9))) (Mul (Add (Add (Add (Mul (Num 8) (Num 2)
) (Num 10)) (Add (Mul (Mul (Add (Num 8) (Num 4)) (Num 8)) (Num 1)) (Mul (Num 9)
(Mul (Num 3) (Add (Num 2) (Mul (Num 5) (Mul (Mul (Add (Mul (Mul (Num 8) (Add (Nu
m 1) (Add (Mul (Num 3) (Mul (Num 3) (Num 2))) (Add (Num 5) (Num 9))))) (Add (Num
 3) (Num 2))) (Num 3)) (Add (Num 5) (Num 8))) (Num 8)))))))) (Num 2)) (Num 1)))
(Add (Num 1) (Add (Num 6) (Num 3)))) (Num 7)) (Add (Num 5) (Add (Mul (Num 5) (Ad
d (Num 1) (Mul (Num 6) (Mul (Num 1) (Mul (Num 7) (Num 10)))))) (Add (Num 4) (Mul
 (Num 4) (Num 2))))))) (Mul (Mul (Num 3) (Add (Num 5) (Add (Num 7) (Num 10)))) (
Num 4)))) (Mul (Add (Num 9) (Num 4)) (Mul (Mul (Num 2) (Mul (Num 8) (Num 6))) (M
ul (Add (Add (Add (Num 8) (Num 3)) (Add (Add (Mul (Add (Mul (Num 4) (Add (Num 1)
 (Num 5))) (Add (Num 6) (Add (Mul (Num 4) (Num 6)) (Mul (Num 5) (Num 7))))) (Mul
 (Mul (Num 8) (Add (Add (Add (Mul (Num 8) (Mul (Num 4) (Mul (Mul (Add (Num 7) (M
ul (Mul (Num 3) (Add (Num 3) (Num 6))) (Num 7))) (Add (Num 3) (Add (Num 5) (Num
4)))) (Mul (Add (Mul (Add (Mul (Mul (Add (Num 1) (Num 9)) (Num 3)) (Add (Num 8)
(Mul (Num 8) (Num 9)))) (Add (Num 8) (Num 7))) (Mul (Mul (Add (Add (Mul (Num 10)
 (Num 4)) (Add (Add (Num 7) (Num 5)) (Num 3))) (Add (Num 6) (Num 4))) (Add (Mul
(Add (Num 4) (Mul (Num 5) (Mul (Num 10) (Mul (Add (Num 7) (Num 7)) (Add (Add (Mu
l (Mul (Mul (Mul (Num 4) (Mul (Add (Mul (Add (Add (Mul (Add (Num 1) (Num 9)) (Nu
m 7)) (Add (Num 2) (Add (Num 2) (Mul (Num 1) (Num 10))))) (Num 7)) (Mul (Mul (Ad
d (Mul (Num 8) (Add (Num 7) (Add (Num 10) (Add (Add (Add (Num 3) (Add (Num 1) (N
um 1))) (Num 5)) (Add (Num 3) (Num 4)))))) (Add (Mul (Mul (Mul (Mul (Num 1) (Mul
 (Num 5) (Num 2))) (Num 5)) (Num 2)) (Num 10)) (Mul (Mul (Num 1) (Mul (Add (Num
2) (Add (Num 5) (Num 2))) (Num 6))) (Num 10)))) (Num 1)) (Add (Num 7) (Mul (Num
4) (Mul (Add (Mul (Num 5) (Num 9)) (Mul (Num 2) (Mul (Num 5) (Num 3)))) (Mul (Ad
d (Num 9) (Add (Num 8) (Add (Num 2) (Add (Mul (Mul (Num 1) (Num 4)) (Num 5)) (Nu
m 6))))) (Num 7))))))) (Add (Mul (Mul (Num 7) (Num 1)) (Num 6)) (Add (Num 4) (Nu
m 9)))) (Mul (Num 5) (Num 6)))) (Num 3)) (Num 4)) (Mul (Mul (Add (Num 8) (Num 9)
) (Num 10)) (Num 5))) (Num 9)) (Add (Num 9) (Num 4))))))) (Num 8)) (Num 2))) (Ad
d (Mul (Num 5) (Mul (Mul (Num 5) (Mul (Num 2) (Mul (Num 2) (Mul (Num 9) (Num 4))
))) (Num 3))) (Num 1)))) (Num 3)) (Add (Num 6) (Num 3)))))) (Add (Num 2) (Mul (N
um 7) (Add (Num 1) (Num 9))))) (Add (Num 7) (Mul (Add (Add (Add (Mul (Num 3) (Nu
m 7)) (Add (Num 3) (Mul (Add (Add (Add (Num 5) (Add (Num 8) (Add (Add (Add (Num
6) (Mul (Mul (Mul (Add (Add (Num 5) (Num 8)) (Add (Num 8) (Add (Mul (Add (Num 3)
 (Num 4)) (Mul (Num 9) (Add (Num 2) (Num 10)))) (Num 2)))) (Num 1)) (Add (Num 7)
 (Mul (Num 1) (Num 9)))) (Num 3))) (Num 3)) (Num 7)))) (Add (Mul (Add (Mul (Num
8) (Mul (Num 8) (Add (Mul (Mul (Num 8) (Num 10)) (Num 5)) (Add (Add (Add (Add (M
ul (Num 1) (Num 1)) (Num 6)) (Num 4)) (Num 4)) (Add (Num 9) (Num 2)))))) (Num 6)
) (Mul (Add (Add (Num 3) (Add (Num 2) (Num 4))) (Num 7)) (Mul (Num 5) (Num 2))))
 (Num 8))) (Num 10)) (Add (Num 2) (Add (Add (Mul (Add (Num 10) (Num 10)) (Num 5)
) (Add (Num 9) (Num 6))) (Mul (Num 9) (Num 4))))))) (Mul (Add (Mul (Mul (Num 10)
 (Add (Num 1) (Num 9))) (Add (Add (Add (Mul (Num 3) (Add (Num 6) (Num 6))) (Num
1)) (Mul (Num 10) (Num 7))) (Add (Num 10) (Num 9)))) (Mul (Mul (Add (Add (Add (N
um 5) (Mul (Add (Num 8) (Num 5)) (Num 2))) (Add (Num 3) (Add (Num 10) (Num 2))))
 (Num 6)) (Num 9)) (Add (Num 6) (Num 5)))) (Add (Mul (Num 10) (Num 3)) (Mul (Mul
 (Mul (Mul (Add (Mul (Num 2) (Num 8)) (Add (Num 7) (Mul (Add (Num 7) (Add (Num 8
) (Add (Num 4) (Num 8)))) (Mul (Num 8) (Mul (Num 2) (Num 1)))))) (Mul (Num 3) (N
um 5))) (Num 7)) (Num 9)) (Num 1))))) (Num 4)) (Mul (Num 9) (Add (Add (Add (Mul
(Num 10) (Num 2)) (Mul (Mul (Add (Mul (Num 8) (Num 7)) (Num 10)) (Add (Mul (Add
(Add (Num 3) (Mul (Mul (Mul (Num 4) (Add (Num 4) (Mul (Num 7) (Num 1)))) (Num 4)
) (Num 9))) (Add (Mul (Add (Mul (Num 3) (Num 9)) (Add (Mul (Num 7) (Num 4)) (Mul
 (Num 9) (Num 1)))) (Num 3)) (Num 4))) (Num 10)) (Add (Num 9) (Num 9)))) (Mul (N
um 8) (Num 5)))) (Mul (Num 1) (Num 10))) (Add (Add (Mul (Add (Num 4) (Add (Num 1
) (Add (Num 1) (Add (Mul (Num 3) (Add (Mul (Num 3) (Mul (Num 1) (Num 5))) (Num 6

))) (Num 2))))) (Mul (Num 9) (Num 3))) (Num 6)) (Mul (Num 6) (Num 10)))))))) (Nu
m 9))) (Mul (Mul (Num 2) (Num 5)) (Add (Mul (Num 7) (Add (Num 2) (Mul (Num 2) (N
um 10)))) (Mul (Add (Add (Add (Add (Num 3) (Num 10)) (Num 4)) (Num 6)) (Num 3))
(Add (Add (Num 6) (Add (Mul (Mul (Num 7) (Add (Num 5) (Mul (Mul (Add (Num 8) (Ad
d (Add (Add (Num 10) (Num 8)) (Num 10)) (Add (Mul (Num 4) (Num 6)) (Mul (Num 8)
(Num 8))))) (Num 6)) (Num 10)))) (Num 8)) (Add (Mul (Num 10) (Num 1)) (Mul (Num
2) (Num 3))))) (Num 7)))))))) (Num 5)) (Mul (Add (Num 3) (Add (Num 3) (Mul (Num 1
0) (Num 6)))) (Num 1)))) (Add (Num 6) (Add (Num 10) (Add (Mul (Mul (Num 7) (Num
5)) (Num 1)) (Add (Num 7) (Num 1)))))) (Num 3))))))) (Mul (Num 2) (Add (Add (Num
 8) (Mul (Add (Num 4) (Mul (Mul (Num 2) (Num 3)) (Add (Mul (Num 6) (Num 10)) (Mu
l (Num 3) (Num 1))))) (Num 10))) (Num 6))))) (Mul (Mul (Add (Add (Mul (Mul (Add
(Num 2) (Num 3)) (Num 5)) (Mul (Add (Add (Mul (Mul (Num 4) (Mul (Mul (Num 3) (Ad
d (Mul (Add (Mul (Num 3) (Mul (Add (Num 5) (Add (Mul (Mul (Add (Num 1) (Num 4))
(Num 8)) (Add (Mul (Num 7) (Mul (Num 1) (Add (Num 9) (Num 4)))) (Add (Num 6) (Mu
l (Num 10) (Add (Mul (Mul (Num 1) (Mul (Num 8) (Num 10))) (Add (Mul (Add (Num 8)
 (Num 7)) (Mul (Num 10) (Add (Mul (Num 3) (Num 9)) (Num 6)))) (Mul (Add (Add (Ad
d (Num 9) (Num 7)) (Mul (Num 6) (Num 6))) (Num 6)) (Add (Add (Num 1) (Num 5)) (M
ul (Mul (Num 1) (Add (Add (Mul (Add (Add (Add (Add (Mul (Num 2) (Num 8)) (Num 5))
(Mul (Mul (Num 10) (Mul (Num 1) (Mul (Add (Num 6) (Add (Num 2) (Add (Num 7) (Mul
(Mul (Num 6) (Add (Num 3) (Num 2))) (Num 5))))) (Num 6)))) (Add (Add (Mul (Num 1)
(Mul (Add (Num 5) (Num 2)) (Num 8))) (Mul (Num 6) (Num 7))) (Add (Add (Num 4) (M
ul (Mul (Add (Add (Add (Add (Num 10) (Num 3)) (Num 4)) (Num 4)) (Num 10)) (Add (
Num 4) (Num 4))) (Num 4))) (Mul (Mul (Num 1) (Num 7)) (Num 8)))))) (Num 1)) (Add
 (Num 1) (Num 6))) (Mul (Num 2) (Add (Num 8) (Add (Num 4) (Mul (Num 10) (Mul (Mu
l (Add (Mul (Add (Add (Num 2) (Mul (Num 8) (Mul (Num 10) (Num 6)))) (Mul (Mul (N
um 3) (Add (Num 10) (Add (Add (Num 2) (Num 10)) (Num 8)))) (Num 8))) (Num 6)) (A
dd (Num 4) (Mul (Num 7) (Add (Add (Num 3) (Num 2)) (Num 3))))) (Mul (Add (Add (A
dd (Num 1) (Num 6)) (Num 5)) (Add (Num 9) (Num 6))) (Mul (Num 9) (Num 7)))) (Num
 6)))))) (Num 5))) (Add (Num 7) (Add (Num 9) (Num 6)))))))) (Num 1)))))) (Add (
Add (Num 10) (Mul (Num 2) (Add (Mul (Add (Num 10) (Num 9)) (Mul (Mul (Num 10) (N
um 1)) (Add (Num 7) (Num 10)))) (Num 8)))) (Add (Num 6) (Mul (Num 5) (Add (Num 2
) (Add (Num 6) (Mul (Add (Num 7) (Num 2)) (Mul (Num 4) (Add (Num 10) (Mul (Num 1
0) (Add (Add (Num 8) (Num 7)) (Num 6)))))))))))) (Num 4))) (Mul (Num 8) (Mul (A
dd (Mul (Add (Mul (Num 8) (Mul (Num 4) (Num 8))) (Mul (Add (Add (Mul (Num 9) (Nu
m 7)) (Mul (Add (Mul (Num 4) (Num 5)) (Add (Add (Num 3) (Num 10)) (Num 7))) (Num
 1))) (Mul (Mul (Num 8) (Mul (Num 5) (Mul (Mul (Num 8) (Num 5)) (Num 9)))) (Mul
(Num 4) (Num 10)))) (Mul (Add (Num 10) (Add (Num 10) (Mul (Mul (Num 7) (Mul (Num
 7) (Mul (Mul (Mul (Mul (Add (Add (Add (Add (Add (Num 8) (Mul (Mul (Add (Num 1)
(Num 1)) (Mul (Add (Mul (Num 7) (Num 6)) (Num 8)) (Mul (Num 5) (Num 4)))) (Num 2
))) (Num 1)) (Add (Mul (Num 1) (Mul (Mul (Add (Num 7) (Num 5)) (Mul (Num 1) (Add
 (Num 6) (Mul (Add (Num 10) (Num 3)) (Add (Num 1) (Num 2)))))) (Num 9))) (Num 8)
)) (Mul (Add (Num 10) (Mul (Mul (Add (Num 8) (Add (Num 1) (Add (Num 6) (Num 1)))
) (Num 2)) (Num 9))) (Num 3))) (Mul (Num 8) (Mul (Add (Mul (Num 9) (Num 9)) (Num
 5)) (Add (Num 2) (Num 3))))) (Num 5)) (Add (Num 2) (Num 5))) (Mul (Add (Add (Nu
m 3) (Num 1)) (Num 10)) (Num 5))) (Mul (Num 10) (Num 6))))) (Num 8)))) (Num 9)))
) (Mul (Mul (Add (Add (Num 4) (Num 7)) (Num 2)) (Num 10)) (Mul (Num 2) (Mul (Num
 5) (Add (Mul (Num 4) (Num 1)) (Mul (Add (Num 5) (Add (Num 7) (Mul (Num 2) (Mul
(Add (Mul (Num 10) (Num 9)) (Mul (Num 7) (Num 4))) (Add (Mul (Num 1) (Mul (Mul (
Num 3) (Num 3)) (Num 4))) (Num 5)))))) (Mul (Num 6) (Num 5)))))))) (Add (Mul (Mu
l (Num 8) (Num 8)) (Num 3)) (Num 9))) (Add (Num 9) (Mul (Num 4) (Num 1)))))) (Nu
m 6)) (Add (Num 5) (Num 8)))) (Num 4))) (Num 7)) (Num 7)) (Add (Num 2) (Num 10))
) (Add (Add (Num 8) (Add (Num 7) (Num 8))) (Add (Num 7) (Num 9))))) (Mul (Mul (A
dd (Mul (Mul (Add (Add (Mul (Num 7) (Num 2)) (Num 4)) (Mul (Num 4) (Num 10))) (A
dd (Num 10) (Add (Num 5) (Num 6)))) (Num 7)) (Num 5)) (Num 1)) (Mul (Mul (Num 9)

(Mul (Add (Mul (Num 5) (Num 5)) (Mul (Num 10) (Num 2))) (Num 10))) (Num 9)))) (Add (Num 4) (Add (Add (Mul (Num 5) (Mul (Mul (Num 1) (Num 6)) (Num 2))) (Add (Num 7) (Num 1))) (Num 5)))) (Num 3)) (Add (Add (Num 6) (Num 1)) (Num 8)))))) (Mul (Num 6) (Num 1))) (Num 6)) (Num 9)))) (Num 1)))) (Num 7)) (Num 5)) (Mul (Num 3) (Add (Num 10) (Add (Num 8) (Num 8)))))) (Num 10)))) (Mul (Add (Num 10) (Add (Mul (Add (Num 5) (Num 3)) (Add (Add (Add (Num 10) (Num 3)) (Num 1)) (Mul (Num 5) (Add (Num 4) (Num 8))))) (Add (Mul (Mul (Add (Num 3) (Num 9)) (Mul (Mul (Num 5) (Num 4)) (Add (Mul (Num 6) (Add (Num 6) (Mul (Mul (Num 3) (Num 10)) (Add (Num 5) (Num 4))))) (Mul (Num 6) (Num 7))))) (Mul (Mul (Add (Mul (Mul (Num 3) (Num 7)) (Num 4)) (Num 7)) (Add (Mul (Num 4) (Add (Num 8) (Mul (Num 4) (Mul (Num 7) (Num 9))))) (Num 1))) (Add (Add (Num 8) (Num 10)) (Add (Num 9) (Num 4))))) (Num 8)))) (Mul (Add (Num 5) (Mul (Mul (Num 2) (Num 8)) (Mul (Mul (Add (Num 3) (Num 9)) (Mul (Add (Mul (Add (Num 7) (Num 6)) (Num 10)) (Num 4)) (Num 7))) (Mul (Num 9) (Mul (Add (Mul (Mul (Num 10) (Num 7)) (Num 7)) (Num 4)) (Mul (Mul (Add (Add (Add (Num 9) (Num 10)) (Num 10)) (Num 3)) (Add (Num 7) (Num 9))) (Add (Num 6) (Num 10))))))) (Num 6)))) (Num 7))))) (Num 6)))) (Num 7)) (Add (Add (Num 3) (Num 6)) (Add (Mul (Add (Num 3) (Num 2)) (Mul (Mul (Num 7) (Num 1)) (Add (Num 3) (Add (Mul (Num 6) (Mul (Add (Mul (Add (Mul (Mul (Mul (Mul (Mul (Mul (Mul (Num 9) (Num 5)) (Num 2)) (Num 4)) (Num 6)) (Num 8)) (Add (Add (Num 6) (Num 2)) (Mul (Num 5) (Add (Num 9) (Num 2))))) (Num 7)) (Num 1)) (Num 3)) (Num 3)) (Num 2))) (Add (Add (Mul (Add (Num 2) (Num 4)) (Mul (Add (Mul (Add (Mul (Num 2) (Num 4)) (Num 6)) (Add (Num 3) (Num 5))) (Num 8)) (Num 9))) (Mul (Num 10) (Add (Num 2) (Add (Mul (Num 2) (Num 5)) (Num 3))))) (Mul (Num 3) (Num 7))))))) (Add (Mul (Num 3) (Mul (Add (Num 3) (Num 7)) (Mul (Add (Num 4) (Num 4)) (Mul (Add (Num 3) (Mul (Num 7) (Mul (Add (Num 6) (Num 9)) (Add (Num 2) (Num 9))))) (Num 6))))) (Num 3)))))))) (Mul (Num 5) (Num 2))) (Add (Add (Num 5) (Num 3)) (Mul (Add (Num 4) (Add (Mul (Mul (Num 1) (Add (Num 2) (Num 7))) (Mul (Num 10) (Num 8))) (Num 8))) (Num 9)))) (Num 4)) (Mul (Num 7) (Num 3)))) (Mul (Num 4) (Mul (Num 4) (Num 6)))))) (Mul (Add (Add (Mul (Mul (Num 7) (Mul (Num 6) (Num 5))) (Mul (Num 3) (Add (Add (Mul (Add (Add (Add (Num 7) (Num 6)) (Num 7)) (Num 4)) (Mul (Num 4) (Add (Num 2) (Mul (Mul (Add (Mul (Mul (Num 9) (Add (Num 9) (Num 4))) (Num 10)) (Num 5)) (Num 1)) (Add (Add (Add (Num 8) (Num 4)) (Num 7)) (Mul (Num 3) (Num 4))))))) (Num 5)))) (Num 7)) (Add (Mul (Add (Num 2) (Add (Num 7) (Mul (Add (Mul (Add (Num 9) (Mul (Num 9) (Num 2))) (Num 4)) (Mul (Num 3) (Num 5))) (Mul (Mul (Mul (Num 4) (Add (Num 2) (Mul (Num 3) (Add (Mul (Mul (Num 9) (Mul (Num 1) (Num 5))) (Add (Mul (Mul (Num 9) (Num 5)) (Add (Num 10) (Mul (Num 3) (Add (Add (Num 10) (Num 6)) (Add (Mul (Mul (Add (Add (Num 4) (Mul (Num 8) (Num 9))) (Num 2)) (Mul (Num 10) (Mul (Num 9) (Num 2)))) (Num 3)) (Add (Num 9) (Mul (Num 1) (Add (Num 4) (Num 5))))))))) (Num 7))) (Add (Num 10) (Add (Num 9) (Add (Add (Num 4) (Num 3)) (Add (Add (Mul (Mul (Num 2) (Num 3)) (Add (Add (Add (Num 1) (Num 3)) (Num 8)) (Mul (Mul (Mul (Num 10) (Num 5)) (Mul (Num 7) (Add (Num 10) (Add (Mul (Num 6) (Num 10)) (Add (Num 5) (Num 1)))))) (Add (Num 6) (Num 9))))) (Mul (Mul (Num 4) (Mul (Num 8) (Add (Mul (Num 4) (Add (Mul (Add (Mul (Add (Num 5) (Num 3)) (Num 1)) (Num 4)) (Num 7)) (Num 4))) (Num 6)))) (Add (Num 9) (Num 10)))) (Num 10))))))))) (Mul (Mul (Mul (Add (Add (Mul (Num 8) (Mul (Num 2) (Add (Mul (Num 1) (Num 10)) (Num 3)))) (Add (Add (Num 2) (Num 1)) (Mul (Num 3) (Add (Mul (Num 3) (Num 2)) (Num 1))))) (Add (Mul (Num 7) (Mul (Mul (Num 7) (Num 6)) (Num 8))) (Mul (Num 8) (Num 5)))) (Num 4)) (Mul (Num 1) (Num 1))) (Num 7))) (Num 6))))) (Num 1)) (Add (Num 5) (Num 10)))) (Num 7))) (Add (Add (Num 5) (Num 5)) (Num 7)))) (Mul (Add (Num 7) (Add (Num 4) (Mul (Num 10) (Num 3)))) (Num 6))))))) (Mul (Add (Add (Add (Add (Num 8) (Add (Num 1) (Num 9))) (Add (Num 9) (Num 9))) (Num 5)) (Add (Add (Num 9) (Num 8)) (Num 3))) (Num 8))))) (Add (Mul (Add (Mul (Mul (Num 3) (Add (Num 4) (Num 9))) (Add (Mul (Num 9) (Add (Add (Add (Num 7) (Num 3)) (Add (Num 1) (Mul (Add (Mul (Add (Num 7) (Num 6)) (Num 8)) (Num 10)) (Mul (Mul (Mul (Add (Add (Mul (Add (Num 3) (Num 7)) (Add (Mul (Num 3) (Num 8

)) (Num 8))) (Mul (Mul (Mul (Mul (Add (Add (Mul (Num 10) (Num 7)) (Add (Add (Mul
 (Num 9) (Num 2)) (Add (Num 8) (Num 10))) (Num 9))) (Add (Mul (Num 9) (Num 6)) (
Num 2))) (Num 8)) (Add (Mul (Mul (Mul (Mul (Num 1) (Num 1)) (Num 10)) (Num 9)) (
Num 3)) (Num 5))) (Num 8)) (Num 6))) (Add (Add (Mul (Num 7) (Add (Num 5) (Num 6)
)) (Num 1)) (Num 5))) (Num 7)) (Num 2)) (Mul (Num 5) (Num 10)))))) (Add (Num 10)
 (Add (Add (Mul (Num 6) (Num 3)) (Add (Num 4) (Num 7))) (Add (Num 3) (Num 2)))))
) (Num 7))) (Num 4)) (Mul (Num 2) (Num 4))) (Num 8)))))))) (Add (Mul (Num 8) (Nu
m 4)) (Num 8)))) (Num 3)) (Num 2))) (Mul (Add (Num 3) (Mul (Add (Num 3) (Num 3))
 (Mul (Num 6) (Num 2)))) (Add (Num 9) (Mul (Add (Add (Num 8) (Num 5)) (Add (Num
10) (Num 6))) (Mul (Num 9) (Add (Num 2) (Mul (Num 4) (Num 5)))))))))))) (Num 7)
)) (Num 8)) (Num 10)) (Add (Num 8) (Add (Add (Num 7) (Mul (Num 2) (Add (Add (Mul
 (Num 6) (Num 10)) (Num 4)) (Add (Mul (Mul (Mul (Num 9) (Num 10)) (Num 5)) (Num
8)) (Add (Num 5) (Mul (Num 4) (Mul (Add (Mul (Num 6) (Num 10)) (Mul (Mul (Add (M
ul (Mul (Num 5) (Add (Mul (Num 8) (Num 2)) (Add (Num 9) (Add (Add (Add (Num 10)
(Mul (Add (Add (Num 1) (Add (Mul (Mul (Num 10) (Num 1)) (Num 8)) (Add (Mul (Mul
(Add (Mul (Add (Mul (Num 9) (Num 1)) (Mul (Num 8) (Mul (Add (Num 5) (Mul (Add (A
dd (Num 9) (Add (Add (Num 7) (Num 4)) (Num 2))) (Add (Mul (Add (Add (Num 7) (Num
 3)) (Num 7)) (Add (Add (Num 8) (Mul (Num 3) (Add (Num 5) (Mul (Num 6) (Num 9)))
)) (Add (Add (Num 8) (Mul (Add (Num 7) (Num 1)) (Num 6))) (Add (Num 8) (Mul (Num
 9) (Num 4)))))) (Num 8))) (Num 8))) (Mul (Mul (Num 1) (Num 3)) (Num 6))))) (Num
 1)) (Num 1)) (Mul (Mul (Num 4) (Num 3)) (Mul (Num 8) (Num 1)))) (Num 3)) (Add (
Num 7) (Num 5))))) (Add (Num 8) (Num 3))) (Num 8))) (Num 8)) (Num 3))))) (Add (A
dd (Num 3) (Add (Num 3) (Add (Num 2) (Add (Num 10) (Num 5))))) (Num 10))) (Num 6
)) (Num 7)) (Num 2))) (Mul (Add (Num 7) (Add (Mul (Num 7) (Mul (Mul (Num 7) (Add
 (Num 2) (Num 1))) (Add (Mul (Add (Num 8) (Mul (Add (Num 2) (Add (Num 9) (Num 4)
)) (Num 9))) (Add (Mul (Num 5) (Mul (Add (Num 6) (Add (Num 5) (Num 3))) (Add (Mu
l (Num 2) (Num 1)) (Num 8)))) (Num 8))) (Add (Num 5) (Num 5))))) (Num 6))) (Num
5)))))))) (Num 2)))))) (Num 2))) (Mul (Mul (Add (Num 7) (Mul (Mul (Mul (Num 1)
(Num 4)) (Num 6)) (Num 1))) (Add (Mul (Add (Num 3) (Num 1)) (Add (Mul (Num 9) (A
dd (Num 9) (Add (Num 7) (Num 8)))) (Mul (Num 3) (Num 8)))) (Num 4))) (Num 3))) (
Mul (Add (Num 3) (Mul (Add (Mul (Num 7) (Add (Add (Num 3) (Num 6)) (Add (Num 3)
(Num 1)))) (Num 7)) (Add (Mul (Num 5) (Add (Num 1) (Mul (Mul (Num 10) (Num 4)) (
Mul (Num 3) (Add (Num 10) (Add (Add (Add (Num 4) (Mul (Num 8) (Mul (Add (Add (Mu
l (Add (Mul (Num 9) (Num 3)) (Mul (Num 5) (Mul (Add (Mul (Num 1) (Num 8)) (Mul (
Mul (Num 4) (Num 3)) (Add (Mul (Mul (Add (Add (Add (Num 2) (Num 2)) (Num 5)) (Nu
m 4)) (Num 1)) (Mul (Add (Mul (Num 5) (Num 5)) (Add (Num 8) (Num 9))) (Num 9)))
(Num 10)))) (Mul (Mul (Mul (Mul (Add (Add (Num 5) (Add (Add (Num 2) (Add (Num 5)
 (Num 3))) (Mul (Num 2) (Mul (Num 3) (Mul (Add (Num 2) (Num 4)) (Num 1)))))) (Nu
m 10)) (Add (Num 10) (Add (Num 7) (Add (Num 9) (Num 10))))) (Add (Num 10) (Mul (
Num 9) (Num 7)))) (Add (Num 8) (Num 10))) (Add (Num 3) (Num 10)))))) (Mul (Num 7
) (Num 7))) (Num 2)) (Num 3)) (Num 5)))) (Num 1)) (Num 7))))))) (Num 10)))) (Num
 8))) (Mul (Num 8) (Num 2))) (Num 5))))))) (Mul (Add (Num 3) (Num 5)) (Num 7))))
 (Add (Add (Add (Num 4) (Num 2)) (Mul (Num 8) (Num 5))) (Num 8)))) (Num 1))) (Nu
m 9)) (Add (Num 4) (Num 8))) (Mul (Mul (Mul (Add (Num 10) (Num 7)) (Add (Num 5)
(Num 3))) (Add (Num 4) (Add (Add (Add (Mul (Num 1) (Mul (Add (Num 9) (Mul (Num 1) (Nu
m 9))) (Mul (Num 8) (Num 3)))) (Add (Mul (Num 5) (Mul (Mul (Mul (Mul (Mul (Add (
Add (Add (Mul (Mul (Num 1) (Mul (Add (Add (Num 6) (Num 9)) (Mul (Add (Num 7) (Ad
d (Add (Num 10) (Num 3)) (Num 4))) (Add (Mul (Add (Add (Num 5) (Mul (Add (Num 2)
 (Add (Num 8) (Num 8))) (Num 9))) (Num 5)) (Num 5)) (Mul (Add (Add (Num 7) (Num
7)) (Num 8)) (Num 6))))) (Mul (Num 8) (Num 4)))) (Mul (Add (Num 8) (Num 4)) (Mul
 (Add (Add (Num 3) (Num 8)) (Mul (Mul (Mul (Add (Mul (Mul (Add (Num 4) (Num 10))
 (Num 6)) (Num 5)) (Mul (Num 8) (Num 8))) (Num 10)) (Add (Num 1) (Mul (Add (Num
10) (Add (Mul (Add (Num 2) (Mul (Mul (Num 5) (Num 8)) (Num 10))) (Mul (Add (Mul
(Add (Mul (Num 1) (Num 9)) (Num 3)) (Mul (Add (Num 10) (Add (Add (Mul (Add (Add

(Mul (Mul (Num 3) (Add (Mul (Num 8) (Num 6)) (Add (Mul (Mul (Mul (Num 3) (Num 7)
) (Mul (Num 1) (Add (Mul (Num 2) (Add (Add (Num 7) (Num 8)) (Mul (Num 1) (Num 7)
))) (Add (Add (Num 9) (Num 3)) (Mul (Mul (Add (Mul (Num 5) (Num 8)) (Num 4)) (Mu
l (Num 1) (Num 1))) (Mul (Num 1) (Mul (Mul (Mul (Mul (Num 5) (Add (Mul (Num 2) (
Num 1)) (Mul (Num 6) (Add (Mul (Add (Add (Add (Mul (Add (Num 1) (Mul (Add (Num 5) (Nu
m 6)) (Num 2))) (Num 1)) (Num 1)) (Mul (Add (Mul (Num 10) (Num 2)) (Num 7)) (Add
 (Mul (Num 8) (Num 4)) (Add (Add (Add (Add (Add (Num 5) (Mul (Num 9) (Num 6))) (
Add (Num 3) (Add (Num 10) (Add (Add (Mul (Num 5) (Add (Add (Add (Num 4) (Num 8))
 (Mul (Num 1) (Add (Num 5) (Num 8)))) (Num 9))) (Num 4)) (Num 6))))) (Num 9)) (M
ul (Num 6) (Num 4))) (Add (Num 6) (Num 1)))))) (Num 1)) (Num 10))))) (Num 9)) (N
um 7)) (Add (Mul (Add (Num 7) (Mul (Add (Num 2) (Num 8)) (Add (Add (Mul (Mul (Ad
d (Num 2) (Add (Num 1) (Add (Num 3) (Mul (Num 5) (Num 10))))) (Num 10)) (Num 2))
 (Num 9)) (Num 3)))) (Mul (Num 8) (Mul (Add (Num 5) (Num 5)) (Num 7)))) (Num 6))
)))))))) (Num 9)) (Num 5)))) (Add (Num 6) (Num 9))) (Num 10)) (Num 3)) (Add (Add
 (Num 3) (Num 4)) (Mul (Num 9) (Mul (Num 1) (Mul (Mul (Num 8) (Num 6)) (Mul (Mul
 (Add (Mul (Num 4) (Num 5)) (Mul (Add (Num 3) (Add (Num 10) (Mul (Add (Add (Add (
Num 10) (Add (Num 1) (Num 10))) (Num 4)) (Mul (Add (Num 10) (Num 4)) (Num 5))) (
Num 8)))) (Mul (Mul (Num 6) (Num 10)) (Mul (Mul (Num 10) (Num 3)) (Num 2))))) (N
um 9)) (Num 9))))))) (Num 5)) (Num 10))) (Add (Num 10) (Num 4)))) (Num 2)) (Add
 (Num 6) (Mul (Num 4) (Num 4))))) (Num 2))) (Num 6)))) (Add (Add (Num 9) (Add (Nu
m 10) (Mul (Num 1) (Num 1)))) (Add (Num 8) (Mul (Num 2) (Num 9)))))) (Num 4))))
(Num 5)) (Mul (Add (Add (Mul (Mul (Num 3) (Num 5)) (Num 2)) (Num 7)) (Add (Add (
Num 7) (Add (Num 6) (Add (Add (Num 6) (Mul (Num 4) (Mul (Mul (Add (Num 5) (Num 9
)) (Add (Mul (Add (Num 6) (Add (Num 6) (Num 7))) (Num 1)) (Add (Num 7) (Num 2)))
) (Mul (Num 3) (Num 1))))) (Num 3)))) (Mul (Mul (Num 4) (Add (Add (Mul (Add (Mul
 (Num 9) (Num 2)) (Mul (Num 2) (Num 4))) (Mul (Num 3) (Mul (Add (Mul (Add (Mul (
Add (Mul (Mul (Num 1) (Num 7)) (Mul (Mul (Num 7) (Num 8)) (Add (Num 4) (Mul (Mul
 (Num 5) (Num 9)) (Num 4))))) (Num 8)) (Num 4)) (Num 6)) (Num 9)) (Num 3)) (Num
7)))) (Num 3)) (Mul (Mul (Num 1) (Num 3)) (Add (Num 8) (Num 6))))) (Num 6)))) (M
ul (Add (Num 5) (Num 8)) (Num 7)))) (Num 10)) (Add (Mul (Mul (Mul (Num 8) (Num 2
)) (Num 10)) (Add (Num 1) (Add (Mul (Add (Mul (Num 4) (Num 8)) (Mul (Add (Mul (N
um 10) (Add (Num 5) (Num 1))) (Num 8)) (Mul (Num 1) (Add (Num 6) (Mul (Mul (Add
 (Num 9) (Add (Add (Num 4) (Add (Mul (Mul (Add (Num 2) (Mul (Num 4) (Num 6))) (Ad
d (Num 3) (Mul (Num 6) (Add (Mul (Num 10) (Num 3)) (Num 8))))) (Mul (Mul (Num 1)
 (Mul (Num 8) (Num 3))) (Num 6))) (Mul (Add (Num 7) (Mul (Num 8) (Mul (Mul (Num
5) (Num 9)) (Mul (Num 6) (Mul (Mul (Num 3) (Mul (Num 6) (Mul (Mul (Num 10) (Mul
 (Num 4) (Add (Add (Num 10) (Num 4)) (Num 2)))) (Num 10)))) (Num 2)))))) (Mul (Nu
m 10) (Mul (Mul (Mul (Num 10) (Add (Num 5) (Num 9))) (Mul (Mul (Num 10) (Add (Mu
l (Num 7) (Mul (Num 7) (Num 2))) (Num 3))) (Mul (Mul (Add (Mul (Mul (Add (Num 10
) (Mul (Num 10) (Mul (Mul (Add (Mul (Add (Num 8) (Add (Add (Num 2) (Num 3)) (Mul
 (Mul (Num 6) (Num 10)) (Num 7)))) (Num 4)) (Num 8)) (Mul (Mul (Num 8) (Add (Add
 (Num 5) (Num 2)) (Num 5))) (Num 3))) (Add (Num 5) (Num 8))))) (Num 4)) (Num 3))
 (Mul (Num 8) (Num 9))) (Num 2)) (Add (Num 1) (Add (Add (Add (Num 3) (Num 5)) (N
um 4)) (Mul (Num 4) (Num 4))))))) (Add (Mul (Num 8) (Num 6)) (Num 4))))))) (Num
3))) (Mul (Mul (Add (Num 9) (Num 4)) (Add (Num 1) (Mul (Mul (Add (Num 4) (Add (A
dd (Num 2) (Num 6)) (Num 1))) (Mul (Num 1) (Add (Mul (Add (Num 5) (Num 2)) (Num
7)) (Mul (Num 5) (Num 9))))) (Mul (Add (Num 1) (Num 9)) (Mul (Num 2) (Num 3)))))
) (Num 4))) (Add (Mul (Num 2) (Num 8)) (Num 4))))))) (Add (Num 1) (Num 8))) (Add
 (Add (Mul (Num 6) (Num 9)) (Num 7)) (Add (Mul (Mul (Num 8) (Num 8)) (Add (Add (
Add (Num 3) (Mul (Num 5) (Num 8))) (Num 8)) (Num 4))) (Add (Num 10) (Add (Mul (N
um 5) (Num 10)) (Add (Mul (Num 7) (Num 10)) (Add (Mul (Mul (Mul (Num 6) (Mul (Nu
m 2) (Num 7))) (Add (Add (Mul (Num 6) (Mul (Num 9) (Num 1))) (Mul (Num 10) (Mul
 (Add (Mul (Num 3) (Mul (Num 9) (Add (Mul (Num 7) (Num 6)) (Num 7)))) (Num 8)) (A
dd (Num 10) (Add (Num 6) (Num 5)))))) (Add (Num 8) (Num 1)))) (Num 2)) (Add (Mul

(Mul (Mul (Add (Add (Num 9) (Add (Mul (Add (Num 4) (Mul (Num 2) (Num 8))) (Mul (Num 3) (Mul (Add (Add (Num 10) (Num 1)) (Add (Num 7) (Mul (Add (Num 5) (Add (Num 1) (Num 8))) (Num 8)))) (Num 8)))) (Num 6))) (Num 1)) (Num 3)) (Add (Mul (Add (Num 4) (Mul (Add (Add (Num 1) (Num 2)) (Num 3)) (Add (Add (Num 6) (Num 10)) (Num 1)))) (Num 8)) (Num 10))) (Num 10)) (Num 10))))))))))) (Num 2))) (Add (Num 4) (Num 8))) (Add (Add (Num 7) (Mul (Num 5) (Num 7))) (Num 10))) (Add (Mul (Add (Num 10) (Num 2)) (Num 7)) (Num 5))) (Add (Mul (Num 8) (Num 2)) (Num 6)))) (Num 6))) (Mul (Mul (Mul (Num 1) (Num 10)) (Num 10)) (Mul (Mul (Num 4) (Add (Add (Mul (Num 3) (Num 9)) (Mul (Add (Num 6) (Num 9)) (Add (Add (Mul (Add (Mul (Num 9) (Num 2)) (Add (Add (Num 7) (Mul (Mul (Mul (Mul (Mul (Num 4) (Mul (Num 1) (Add (Mul (Mul (Mul (Add (Num 9) (Add (Mul (Num 7) (Add (Mul (Num 1) (Mul (Num 9) (Num 3))) (Num 8))) (Mul (Num 6) (Add (Num 4) (Num 1))))) (Mul (Num 8) (Mul (Mul (Mul (Num 4) (Num 2)) (Mul (Add (Mul (Add (Num 5) (Num 5)) (Add (Num 2) (Num 7))) (Num 2)) (Add (Num 5) (Add (Add (Num 2) (Add (Num 9) (Num 9))) (Num 10))))) (Add (Add (Num 7) (Num 6)) (Num 8))))) (Add (Add (Add (Add (Mul (Mul (Num 6) (Add (Mul (Num 9) (Add (Mul (Num 9) (Mul (Mul (Add (Num 9) (Mul (Mul (Num 7) (Add (Mul (Num 2) (Num 4)) (Add (Num 6) (Num 9)))) (Num 8))) (Num 4)) (Add (Mul (Add (Num 10) (Num 2)) (Num 8)) (Mul (Mul (Num 4) (Num 6)) (Add (Num 2) (Mul (Add (Num 7) (Num 9)) (Mul (Num 8) (Num 5)))))))) (Num 10))) (Num 7))) (Mul (Add (Add (Num 7) (Add (Mul (Num 1) (Add (Num 7) (Mul (Add (Num 2) (Add (Add (Mul (Mul (Mul (Add (Mul (Mul (Mul (Add (Add (Num 2) (Add (Num 2) (Num 4))) (Num 5)) (Mul (Num 3) (Num 10))) (Num 10)) (Add (Add (Num 8) (Mul (Num 10) (Num 1))) (Num 3))) (Add (Mul (Mul (Num 1) (Num 1)) (Num 10)) (Num 5))) (Add (Num 10) (Mul (Num 9) (Num 8)))) (Num 10)) (Mul (Num 10) (Num 7))) (Mul (Add (Num 7) (Num 3)) (Num 3))) (Num 5))) (Num 9)))) (Num 2))) (Add (Num 9) (Mul (Add (Num 10) (Num 9)) (Mul (Add (Add (Num 3) (Num 2)) (Mul (Mul (Num 3) (Num 7)) (Mul (Num 10) (Mul (Add (Mul (Num 3) (Num 7)) (Num 5)) (Num 7))))) (Add (Num 5) (Num 10)))))) (Num 1))) (Num 2)) (Num 7)) (Add (Num 10) (Num 9))) (Num 8))) (Add (Num 5) (Add (Num 9) (Num 6)))) (Add (Add (Num 5) (Add (Add (Mul (Add (Add (Num 10) (Mul (Mul (Num 3) (Num 1)) (Num 4))) (Num 8)) (Add (Num 7) (Mul (Num 1) (Mul (Add (Num 1) (Num 6)) (Num 6))))) (Num 8)) (Num 4))) (Num 8))))) (Num 7)) (Mul (Mul (Mul (Num 4) (Num 1)) (Add (Add (Num 4) (Num 6)) (Add (Mul (Num 10) (Num 9)) (Add (Num 10) (Num 4))))) (Num 10))) (Add (Num 6) (Mul (Mul (Num 4) (Add (Num 10) (Add (Num 2) (Add (Add (Num 8) (Num 3)) (Num 5))))) (Num 10)))) (Mul (Add (Num 10) (Num 6)) (Num 9)))) (Num 9))) (Num 1)) (Num 3)) (Num 4)))) (Add (Num 7) (Mul (Num 3) (Mul (Num 9) (Num 3))))))) (Add (Num 9) (Mul (Num 3) (Add (Add (Add (Num 4) (Mul (Mul (Num 7) (Mul (Add (Num 6) (Mul (Num 2) (Num 5))) (Num 5))) (Num 3))) (Num 9)) (Num 8))))))))) (Mul (Mul (Num 3) (Add (Add (Mul (Num 6) (Mul (Num 4) (Num 9))) (Add (Num 4) (Num 2))) (Add (Mul (Add (Num 7) (Mul (Add (Num 5) (Num 7)) (Add (Num 9) (Num 3)))) (Mul (Mul (Num 10) (Num 8)) (Num 1))) (Mul (Num 2) (Mul (Num 6) (Num 10))))))) (Num 1))))) (Num 10))))))))) (Num 1)) (Mul (Mul (Num 9) (Add (Num 8) (Num 7))) (Num 5)))) (Add (Add (Num 9) (Num 8)) (Num 3))) (Add (Num 9) (Num 3))) (Num 3))) (Mul (Mul (Add (Num 10) (Num 3)) (Mul (Mul (Add (Mul (Num 2) (Num 9)) (Num 2)) (Num 4)) (Add (Num 5) (Add (Add (Num 8) (Mul (Add (Mul (Mul (Num 4) (Add (Mul (Num 1) (Add (Num 7) (Mul (Num 6) (Mul (Add (Num 5) (Num 1)) (Num 10))))))) (Mul (Num 5) (Num 3)))) (Mul (Mul (Num 7) (Mul (Mul (Add (Mul (Mul (Num 6) (Num 1)) (Num 5)) (Num 1)) (Add (Mul (Mul (Num 1) (Num 5)) (Num 4)) (Add (Mul (Num 10) (Num 4)) (Num 6)))) (Num 9))) (Mul (Num 8) (Add (Num 3) (Num 9))))) (Num 10)) (Add (Num 2) (Num 10)))) (Num 1))))) (Num 1))) (Mul (Mul (Num 4) (Mul (Add (Num 8) (Mul (Add (Num 8) (Mul (Add (Add (Num 10) (Mul (Add (Num 3) (Mul (Mul (Add (Mul (Add (Add (Mul (Num 5) (Num 7)) (Mul (Add (Num 5) (Num 2)) (Add (Add (Num 6) (Num 7)) (Num 5)))) (Mul (Add (Add (Num 7) (Mul (Num 10) (Mul (Add (Add (Num 2) (Add (Mul (Num 6) (Add (Num 4) (Mul (Num 5) (Num 9)))) (Add (Num 6) (Num 6)))) (Add (Add (Add (Mul (Num 4) (Num 1)) (Num 1)) (Num 9)) (Add (Num 10) (Num 5)))) (Num 3)))) (Mul (Add (N

um 7) (Num 4)) (Num 10))) (Mul (Num 2) (Num 1)))) (Mul (Add (Mul (Add (Num 1) (N
um 10)) (Num 2)) (Add (Num 1) (Add (Mul (Num 4) (Num 8)) (Num 10)))) (Add (Num 9
) (Mul (Add (Add (Mul (Mul (Num 9) (Num 1)) (Mul (Num 7) (Num 7))) (Mul (Num 2)
(Mul (Mul (Add (Mul (Mul (Add (Mul (Num 1) (Mul (Num 4) (Num 3))) (Num 7)) (Num
5)) (Add (Mul (Add (Num 6) (Num 9)) (Add (Mul (Num 4) (Mul (Num 2) (Num 1))) (Nu
m 1))) (Add (Mul (Num 7) (Num 7)) (Num 10)))) (Add (Add (Mul (Num 6) (Add (Add (
Mul (Num 4) (Num 9)) (Num 1)) (Num 10))) (Add (Mul (Add (Num 8) (Add (Mul (Add (
Mul (Add (Mul (Num 4) (Num 4)) (Mul (Num 8) (Add (Mul (Add (Num 10) (Mul (Num 5)
(Num 7))) (Num 3)) (Mul (Num 7) (Add (Num 4) (Mul (Add (Mul (Add (Mul (Add (Num
8) (Add (Num 7) (Num 9))) (Add (Add (Add (Mul (Mul (Mul (Mul (Mul (Num 9) (Mul
(Num 4) (Mul (Num 10) (Mul (Num 8) (Num 1))))) (Num 6)) (Add (Add (Mul (Add (Num
7) (Num 10)) (Mul (Num 7) (Num 7))) (Num 8)) (Num 2))) (Add (Mul (Add (Mul (Add
(Num 4) (Num 3)) (Num 10)) (Mul (Add (Mul (Num 2) (Mul (Num 4) (Num 7))) (Num 7
)) (Num 7))) (Num 6)) (Add (Num 9) (Mul (Add (Num 8) (Num 2)) (Num 5))))) (Num 7
)) (Mul (Num 4) (Mul (Num 5) (Num 8)))) (Mul (Num 1) (Num 10))) (Num 2))) (Num 9
)) (Num 9)) (Num 2)) (Num 10))))))) (Num 10)) (Num 6)) (Add (Num 9) (Mul (Num 4)
(Num 3)))) (Add (Num 9) (Mul (Add (Add (Num 7) (Mul (Add (Num 1) (Num 9)) (Mul
(Num 4) (Num 4)))) (Num 4)) (Add (Num 9) (Mul (Mul (Add (Num 2) (Num 4)) (Num 10
)) (Mul (Add (Mul (Num 9) (Num 3)) (Num 8)) (Num 2)))))))) (Add (Num 1) (Num 6))
) (Num 4))) (Num 1))) (Num 9)) (Num 2)))) (Add (Num 5) (Num 9))) (Add (Num 7) (N
um 3)))))) (Num 4)) (Add (Add (Num 9) (Num 3)) (Mul (Add (Num 4) (Num 8)) (Num 3
)))) (Mul (Num 2) (Add (Num 6) (Num 5))))) (Add (Mul (Num 9) (Add (Add (Num 2) (
Num 3)) (Num 8))) (Mul (Num 8) (Num 8))))) (Mul (Num 10) (Mul (Add (Add (Num 2)
(Add (Add (Mul (Add (Num 8) (Mul (Num 3) (Mul (Num 2) (Add (Add (Num 7) (Num 8))
(Num 8))))) (Mul (Add (Add (Add (Mul (Add (Num 6) (Mul (Num 4) (Num 6))) (Num 7
)) (Num 5)) (Num 3)) (Mul (Mul (Mul (Mul (Mul (Num 2) (Num 10)) (Num 4)) (Add (N
um 3) (Add (Add (Add (Num 2) (Num 9)) (Add (Num 3) (Num 10))) (Num 8)))) (Num 3)
) (Add (Num 6) (Add (Num 5) (Num 4))))) (Add (Num 3) (Add (Num 8) (Num 6))))) (A
dd (Num 5) (Num 5))) (Num 4))) (Num 1)) (Num 1)))) (Num 2))) (Num 7))) (Add (Add
(Num 5) (Num 3)) (Add (Add (Num 2) (Add (Add (Num 2) (Num 3)) (Mul (Mul (Num 9)
(Num 4)) (Add (Num 6) (Mul (Mul (Mul (Num 6) (Num 3)) (Num 4)) (Num 6)))))) (Ad
d (Num 9) (Add (Num 1) (Mul (Mul (Num 3) (Add (Mul (Num 2) (Num 9)) (Num 7))) (M
ul (Add (Num 8) (Mul (Num 8) (Num 9))) (Add (Add (Num 6) (Mul (Num 5) (Add (Add
(Add (Mul (Num 1) (Add (Num 3) (Num 8))) (Num 8)) (Num 4)) (Mul (Mul (Mul (Num 2
) (Add (Num 9) (Add (Mul (Num 5) (Num 8)) (Num 3)))) (Num 5)) (Num 4))))) (Mul (
Num 7) (Num 1))))))))))) (Num 7)))) (Num 8)) (Add (Num 3) (Num 4))))))) (Num 3))
)) (Add (Num 6) (Num 10))))) (Add (Mul (Num 5) (Num 5)) (Num 8)))) (Mul (Num 1)
(Add (Num 2) (Num 7)))) (Num 3)) (Num 6)) (Num 6)))) (Add (Num 7) (Num 1))))) (N
um 2)) (Num 1))))))))) (Num 9))) (Mul (Num 3) (Num 6))) (Num 10)) (Num 1)) (Mul (
Num 9) (Add (Num 5) (Mul (Num 10) (Num 7))))) (Num 7)) (Num 6)) (Num 4)) (Mul (N
um 7) (Add (Add (Mul (Num 5) (Num 7)) (Num 6)) (Num 7))))) (Mul (Add (Num 10) (N
um 2)) (Mul (Num 9) (Num 4)))))) (Mul (Num 5) (Mul (Num 5) (Mul (Mul (Num 9) (Nu
m 4)) (Mul (Mul (Num 7) (Mul (Num 2) (Num 2))) (Num 4)))))) (Num 7)) (Add (Num 2
) (Num 2)))) (Add (Num 6) (Num 4))) (Num 6))))))) (Mul (Add (Num 2) (Num 7)) (Mu
l (Add (Add (Mul (Num 6) (Num 5)) (Mul (Num 9) (Add (Mul (Num 9) (Num 5)) (Mul (
Mul (Add (Add (Num 1) (Num 6)) (Mul (Num 2) (Num 10))) (Num 6)) (Add (Num 10) (A
dd (Add (Num 2) (Add (Add (Num 7) (Mul (Add (Add (Add (Num 5) (Num 5)) (Num 1))
(Num 5)) (Num 7))) (Add (Num 7) (Num 6)))) (Add (Num 7) (Num 2)))))))) (Add (Add
(Mul (Add (Add (Num 5) (Mul (Add (Mul (Add (Num 7) (Num 5)) (Add (Mul (Num 8) (
Mul (Add (Num 7) (Mul (Add (Num 8) (Add (Num 4) (Add (Add (Num 8) (Mul (Mul (Mul
(Mul (Num 9) (Num 3)) (Num 5)) (Num 9)) (Num 8))) (Num 9)))) (Mul (Num 10) (Num
8)))) (Num 6))) (Mul (Num 1) (Add (Num 8) (Add (Num 1) (Num 7)))))) (Num 1)) (M
ul (Num 8) (Num 2)))) (Num 1)) (Mul (Num 10) (Add (Num 7) (Mul (Num 4) (Num 2)))
)) (Num 1)) (Num 5))) (Num 5)))) (Add (Num 6) (Mul (Mul (Num 8) (Num 3)) (Num 1)

```
))) (Num 6)))))) (Add (Num 8) (Num 7))
Num 3
Add (Num 1) (Mul (Num 5) (Num 9))
Num 4
Mul (Num 4) (Num 6)
Num 10
Mul (Add (Add (Num 6) (Num 8)) (Mul (Num 10) (Add (Num 9) (Num 2)))) (Num 9)
Mul (Num 4) (Num 5)
Mul (Num 8) (Num 6)
Add (Num 6) (Num 10)
*Main> :r
[1 of 1] Compiling Main             ( ArithmeticQuiz.hs, interpreted )

ArithmeticQuiz.hs:60:19: error: parse error on input 'return'
   |
60 |                   return (Num n),
   |                   ^^^^^^
Failed, no modules loaded.
Prelude> :r
[1 of 1] Compiling Main             ( ArithmeticQuiz.hs, interpreted )

ArithmeticQuiz.hs:68:1: error:
    Duplicate type signatures for 'rExpr'
    at ArithmeticQuiz.hs:58:1-5
       ArithmeticQuiz.hs:68:1-5
   |
68 | rExpr :: Int -> Gen Expr
   | ^^^^^
Failed, no modules loaded.
Prelude> :r
[1 of 1] Compiling Main             ( ArithmeticQuiz.hs, interpreted )
Ok, one module loaded.
*Main> sample (rExpr 0)
Num 4
Num 3
Num 10
Num 10
Num 7
Num 7
Num 2
Num 2
Num 3
Num 1
Num 8
*Main> sample (rExpr 1)
Mul (Num 8) (Num 9)
Mul (Num 10) (Num 2)
Mul (Num 1) (Num 4)
Add (Num 3) (Num 3)
Add (Num 7) (Num 5)
Mul (Num 8) (Num 9)
Add (Num 5) (Num 3)
Mul (Num 1) (Num 6)
Add (Num 7) (Num 6)
```

```
Add (Num 9) (Num 1)
Mul (Num 10) (Num 9)
*Main> sample (rExpr 2)
Add (Num 5) (Mul (Num 5) (Num 6))
Mul (Num 7) (Add (Num 1) (Num 9))
Mul (Num 5) (Add (Num 10) (Num 7))
Mul (Num 10) (Mul (Num 5) (Num 5))
Add (Mul (Num 6) (Num 4)) (Num 6)
Mul (Num 10) (Mul (Num 2) (Num 9))
Add (Num 10) (Mul (Num 6) (Num 9))
Mul (Num 6) (Add (Num 3) (Num 8))
Add (Mul (Num 6) (Num 9)) (Num 3)
Add (Add (Num 1) (Num 10)) (Num 1)
Add (Num 4) (Add (Num 4) (Num 7))
*Main> sample (rExpr 3)
Mul (Num 2) (Add (Add (Num 2) (Num 3)) (Num 2))
Mul (Num 10) (Add (Num 10) (Mul (Num 2) (Num 4)))
Mul (Add (Num 3) (Mul (Num 4) (Num 9))) (Num 5)
Mul (Add (Num 10) (Mul (Num 5) (Num 10))) (Num 3)
Mul (Add (Add (Num 8) (Num 1)) (Num 1)) (Num 10)
Add (Num 9) (Add (Num 4) (Mul (Num 2) (Num 1)))
Mul (Mul (Num 8) (Num 9)) (Mul (Num 7) (Num 4))
Add (Mul (Mul (Num 3) (Num 8)) (Num 2)) (Num 2)
Add (Mul (Num 4) (Num 5)) (Add (Num 1) (Num 6))
Mul (Add (Num 7) (Add (Num 2) (Num 2))) (Num 10)
Mul (Num 6) (Mul (Num 6) (Add (Num 6) (Num 9)))
*Main> sample (rExpr 4)
Add (Add (Mul (Num 7) (Num 7)) (Num 4)) (Add (Num 9) (Num 5))
Mul (Mul (Mul (Num 4) (Num 7)) (Num 8)) (Mul (Num 9) (Num 3))
Add (Add (Num 1) (Mul (Num 8) (Num 10))) (Add (Num 4) (Num 6))
Add (Mul (Num 5) (Mul (Num 9) (Num 2))) (Mul (Num 5) (Num 2))
Add (Num 10) (Mul (Num 9) (Mul (Mul (Num 10) (Num 5)) (Num 7)))
Add (Add (Num 2) (Num 3)) (Mul (Num 4) (Add (Num 5) (Num 5)))
Add (Num 4) (Mul (Num 3) (Mul (Num 2) (Add (Num 4) (Num 7))))
Add (Num 5) (Mul (Mul (Num 8) (Num 1)) (Mul (Num 7) (Num 3)))
Mul (Mul (Add (Num 2) (Num 8)) (Add (Num 5) (Num 5))) (Num 7)
Add (Num 6) (Add (Num 8) (Add (Num 9) (Add (Num 8) (Num 3))))
Mul (Mul (Num 9) (Mul (Num 3) (Num 3))) (Add (Num 7) (Num 3))
*Main> :r
[1 of 1] Compiling Main             ( ArithmeticQuiz.hs, interpreted )

ArithmeticQuiz.hs:18:11: error:
    • Couldn't match type ‘Gen’ with ‘IO’
      Expected type: IO b
        Actual type: Gen b
    • In a stmt of a 'do' block: forever quiz
      In the expression:
        do putStrLn "Welcome to the arithmetic quiz!"
           forever quiz
      In an equation for ‘main’:
          main
            = do putStrLn "Welcome to the arithmetic quiz!"
                 forever quiz
   |
```

```
18 |           forever quiz
   |           ^^^^^^^^^^^^

ArithmeticQuiz.hs:21:11: error:
    • Couldn't match type ‘IO’ with ‘Gen’
      Expected type: Gen ()
        Actual type: IO ()
    • In a stmt of a 'do' block:
        putStrLn ("What is " ++ showExpr e ++ "?")
      In the expression:
        do e <- arbitrary
           putStrLn ("What is " ++ showExpr e ++ "?")
           answer <- readLn
           let correct = eval e
           ....
      In an equation for ‘quiz’:
          quiz
            = do e <- arbitrary
                 putStrLn ("What is " ++ showExpr e ++ "?")
                 answer <- readLn
                 ....
   |
21 |           putStrLn ("What is "++showExpr e++"?")
   |           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

ArithmeticQuiz.hs:22:21: error:
    • Couldn't match type ‘IO’ with ‘Gen’
      Expected type: Gen Integer
        Actual type: IO Integer
    • In a stmt of a 'do' block: answer <- readLn
      In the expression:
        do e <- arbitrary
           putStrLn ("What is " ++ showExpr e ++ "?")
           answer <- readLn
           let correct = eval e
           ....
      In an equation for ‘quiz’:
          quiz
            = do e <- arbitrary
                 putStrLn ("What is " ++ showExpr e ++ "?")
                 answer <- readLn
                 ....
   |
22 |           answer <- readLn
   |                     ^^^^^^

ArithmeticQuiz.hs:25:18: error:
    • Couldn't match type ‘IO’ with ‘Gen’
      Expected type: Gen ()
        Actual type: IO ()
    • In the expression: putStrLn "Yes, that is correct!"
      In a stmt of a 'do' block:
        if answer == correct then
            putStrLn "Yes, that is correct!"
```

```
        else
            putStrLn ("Sorry, the correct answer is: " ++ show correct)
      In the expression:
        do e <- arbitrary
           putStrLn ("What is " ++ showExpr e ++ "?")
           answer <- readLn
           let correct = eval e
           ....
   |
25 |             then putStrLn "Yes, that is correct!"
   |                  ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^

ArithmeticQuiz.hs:26:18: error:
    • Couldn't match type ‘IO’ with ‘Gen’
      Expected type: Gen ()
        Actual type: IO ()
    • In the expression:
        putStrLn ("Sorry, the correct answer is: " ++ show correct)
      In a stmt of a 'do' block:
        if answer == correct then
            putStrLn "Yes, that is correct!"
        else
            putStrLn ("Sorry, the correct answer is: " ++ show correct)
      In the expression:
        do e <- arbitrary
           putStrLn ("What is " ++ showExpr e ++ "?")
           answer <- readLn
           let correct = eval e
           ....
   |
26 |             else putStrLn ("Sorry, the correct answer is: "++show correct)
   |                  ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
Failed, no modules loaded.
Prelude> :r
[1 of 1] Compiling Main             ( ArithmeticQuiz.hs, interpreted )
Ok, one module loaded.
*Main> :t generate
generate :: Gen a -> IO a
*Main> main
Welcome to the arithmetic quiz!
What is 6 * (10 + 7)?
102
Yes, that is correct!
What is 6 * (4 + 10)?
240
Sorry, the correct answer is: 84
What is 8 + 7 + 6 + 7?
^CInterrupted.
*Main> :l SymbolicExpressions.hs
[1 of 1] Compiling SymbolicExpressions ( SymbolicExpressions.hs, interpreted )

SymbolicExpressions.hs:26:24: error:
    • Couldn't match expected type ‘Expr’ with actual type ‘[Char]’
    • In the second argument of ‘Mul’, namely ‘x’
```

```
        In the first argument of ‘Add’, namely ‘(Mul (Num 2) x)’
        In the expression: Add (Mul (Num 2) x) (Mul (Num 3) y)
   |
26 | ex5 = Add (Mul (Num 2) x) (Mul (Num 3) y)
   |                        ^

SymbolicExpressions.hs:26:40: error:
    • Couldn't match expected type ‘Expr’ with actual type ‘[Char]’
    • In the second argument of ‘Mul’, namely ‘y’
      In the second argument of ‘Add’, namely ‘(Mul (Num 3) y)’
      In the expression: Add (Mul (Num 2) x) (Mul (Num 3) y)
   |
26 | ex5 = Add (Mul (Num 2) x) (Mul (Num 3) y)
   |                                        ^

SymbolicExpressions.hs:27:29: error:
    • Couldn't match expected type ‘Expr’ with actual type ‘[Char]’
    • In the first argument of ‘Mul’, namely ‘x’
      In the second argument of ‘Mul’, namely ‘(Mul x x)’
      In the first argument of ‘Add’, namely ‘(Mul (Num 2) (Mul x x))’
   |
27 | ex6 = Add (Mul (Num 2) (Mul x x)) (Mul (Num 3) y)
   |                             ^

SymbolicExpressions.hs:27:31: error:
    • Couldn't match expected type ‘Expr’ with actual type ‘[Char]’
    • In the second argument of ‘Mul’, namely ‘x’
      In the second argument of ‘Mul’, namely ‘(Mul x x)’
      In the first argument of ‘Add’, namely ‘(Mul (Num 2) (Mul x x))’
   |
27 | ex6 = Add (Mul (Num 2) (Mul x x)) (Mul (Num 3) y)
   |                               ^

SymbolicExpressions.hs:27:48: error:
    • Couldn't match expected type ‘Expr’ with actual type ‘[Char]’
    • In the second argument of ‘Mul’, namely ‘y’
      In the second argument of ‘Add’, namely ‘(Mul (Num 3) y)’
      In the expression: Add (Mul (Num 2) (Mul x x)) (Mul (Num 3) y)
   |
27 | ex6 = Add (Mul (Num 2) (Mul x x)) (Mul (Num 3) y)
   |                                                ^
Failed, no modules loaded.
Prelude> :r
[1 of 1] Compiling SymbolicExpressions ( SymbolicExpressions.hs, interpreted )
Ok, one module loaded.
*SymbolicExpressions> ex6
2*(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
```

((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
((
(^C(((
((
((
((

```
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
(((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((((
((((((((((((((((((((((((((((((((((((^C(((((((((((((((((((((((Interrupted.
*SymbolicExpressions>
*SymbolicExpressions> :r
[1 of 1] Compiling SymbolicExpressions ( SymbolicExpressions.hs, interpreted )
Ok, one module loaded.
*SymbolicExpressions> ex6
2*x*x+3*y
*SymbolicExpressions> ex5
2*x+3*y
*SymbolicExpressions> ex4
1+2*3
*SymbolicExpressions> ex3
(1+2)*3
*SymbolicExpressions> ex2
2+2
*SymbolicExpressions> ex1
2
*SymbolicExpressions> :r
[1 of 1] Compiling SymbolicExpressions ( SymbolicExpressions.hs, interpreted )
Ok, one module loaded.
*SymbolicExpressions> ex5
2*x+3*y
*SymbolicExpressions> vars ex5
["x","y"]
*SymbolicExpressions> ex6
2*x*x+3*y
*SymbolicExpressions> vars ex6
["x","x","y"]
*SymbolicExpressions> :r
```

```
[1 of 1] Compiling SymbolicExpressions ( SymbolicExpressions.hs, interpreted )
Ok, one module loaded.
*SymbolicExpressions> vars ex6
["x","y"]
*SymbolicExpressions> :t lookup
lookup :: Eq a => a -> [(a, b)] -> Maybe b
*SymbolicExpressions> :i Maybe
data Maybe a = Nothing | Just a         -- Defined in 'GHC.Base'
instance Applicative Maybe -- Defined in 'GHC.Base'
instance Eq a => Eq (Maybe a) -- Defined in 'GHC.Base'
instance Functor Maybe -- Defined in 'GHC.Base'
instance Monad Maybe -- Defined in 'GHC.Base'
instance Monoid a => Monoid (Maybe a) -- Defined in 'GHC.Base'
instance Ord a => Ord (Maybe a) -- Defined in 'GHC.Base'
instance Show a => Show (Maybe a) -- Defined in 'GHC.Show'
instance Read a => Read (Maybe a) -- Defined in 'GHC.Read'
instance Foldable Maybe -- Defined in 'Data.Foldable'
instance Traversable Maybe -- Defined in 'Data.Traversable'
*SymbolicExpressions> :r
[1 of 1] Compiling SymbolicExpressions ( SymbolicExpressions.hs, interpreted )
Ok, one module loaded.
*SymbolicExpressions> ex6
2*x*x+3*y
*SymbolicExpressions> sub
substitute  subtract
*SymbolicExpressions> substitute [("x",Num 7),("y",Num 3)] ex6
2*7*7+3*3
*SymbolicExpressions> substitute [("x",Num 7),("y",Num 3)] ex5
2*7+3*3
*SymbolicExpressions> ex5
2*x+3*y
*SymbolicExpressions> substitute [("x",Num 7)] ex5
2*7+3**** Exception: Variable not defined: y
CallStack (from HasCallStack):
  error, called at SymbolicExpressions.hs:64:39 in main:SymbolicExpressions
*SymbolicExpressions> :r
[1 of 1] Compiling SymbolicExpressions ( SymbolicExpressions.hs, interpreted )

SymbolicExpressions.hs:76:32: error: parse error on input '->'
   |
76 |                       Nothing -> error ("Varible not defined: "++x)
   |                                ^^
Failed, no modules loaded.
Prelude> :r
[1 of 1] Compiling SymbolicExpressions ( SymbolicExpressions.hs, interpreted )
Ok, one module loaded.
*SymbolicExpressions> ex6
2*x*x+3*y
*SymbolicExpressions> eval [("x",2),("y",1)] ex6
11
*SymbolicExpressions> :r
[1 of 1] Compiling SymbolicExpressions ( SymbolicExpressions.hs, interpreted )
Ok, one module loaded.
*SymbolicExpressions> ex5
```