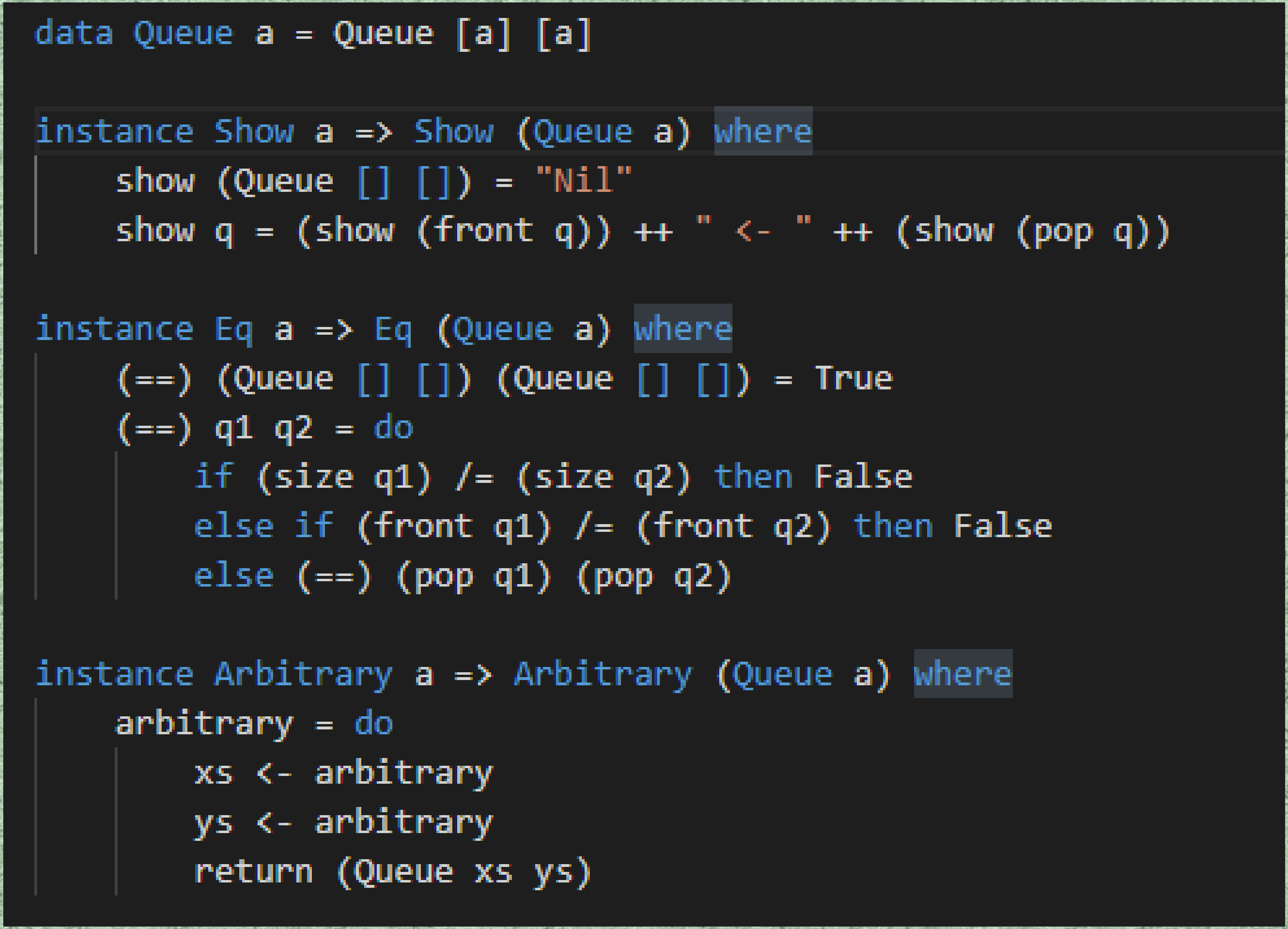
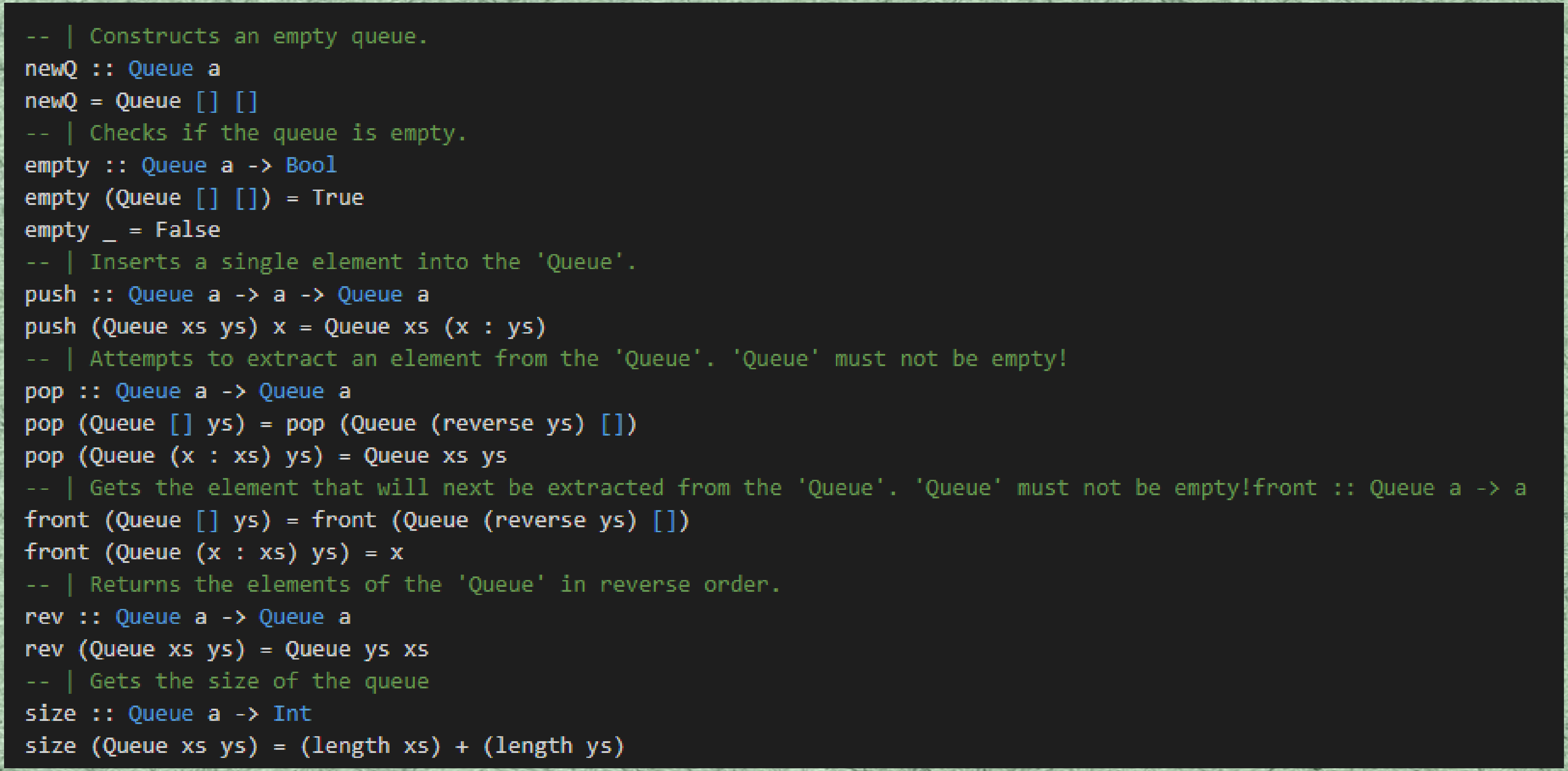
Real-Time Queue

Definire tip date



```haskell
data Queue a = Queue [a] [a]

instance Show a => Show (Queue a) where
    show (Queue [] []) = "Nil"
    show q = (show (front q)) ++ " <- " ++ (show (pop q))

instance Eq a => Eq (Queue a) where
    (==) (Queue [] []) (Queue [] []) = True
    (==) q1 q2 = do
        if (size q1) /= (size q2) then False
        else if (front q1) /= (front q2) then False
        else (==) (pop q1) (pop q2)

instance Arbitrary a => Arbitrary (Queue a) where
    arbitrary = do
        xs <- arbitrary
        ys <- arbitrary
        return (Queue xs ys)
```

Functii



```haskell
-- | Constructs an empty queue.
newQ :: Queue a
newQ = Queue [] []
-- | Checks if the queue is empty.
empty :: Queue a -> Bool
empty (Queue [] []) = True
empty _ = False
-- | Inserts a single element into the 'Queue'.
push :: Queue a -> a -> Queue a
push (Queue xs ys) x = Queue xs (x : ys)
-- | Attempts to extract an element from the 'Queue'. 'Queue' must not be empty!
pop :: Queue a -> Queue a
pop (Queue [] ys) = pop (Queue (reverse ys) [])
pop (Queue (x : xs) ys) = Queue xs ys
-- | Gets the element that will next be extracted from the 'Queue'. 'Queue' must not be empty!front :: Queue a -> a
front (Queue [] ys) = front (Queue (reverse ys) [])
front (Queue (x : xs) ys) = x
-- | Returns the elements of the 'Queue' in reverse order.
rev :: Queue a -> Queue a
rev (Queue xs ys) = Queue ys xs
-- | Gets the size of the queue
size :: Queue a -> Int
size (Queue xs ys) = (length xs) + (length ys)
```

De ce trebuie sa instantiem clasa Eq?

De ce trebuie sa instantiem clasa Eq?

Pentru a raspunde la aceasta intrebare trebuie sa raspundem prima data la intrebarea: Cum arata \`coada\` 1 <- 2 <- 3 <- 4 <- Nil ?

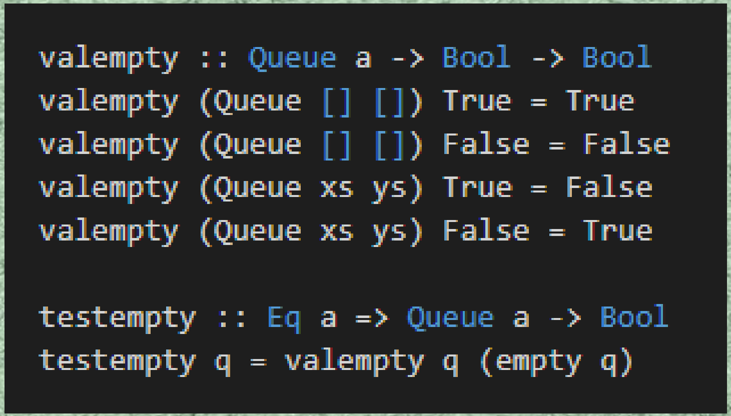
De ce trebuie sa instantiem clasa Eq?

Pentru a raspunde la aceasta intrebare trebuie sa raspundem prima data la intrebarea: Cum arata \`coada\` 1 <- 2 <- 3 <- 4 <- Nil ?

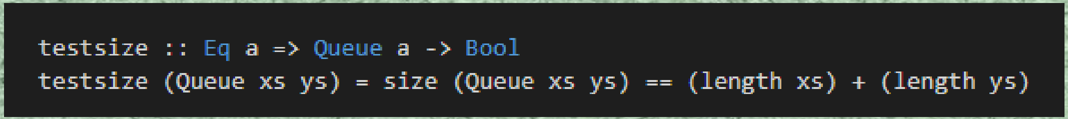
Queue [] [4, 3, 2, 1] | Queue [1] [4, 3, 2]

Queue [1, 2] [4, 3] | Queue [1, 2, 3] [4]

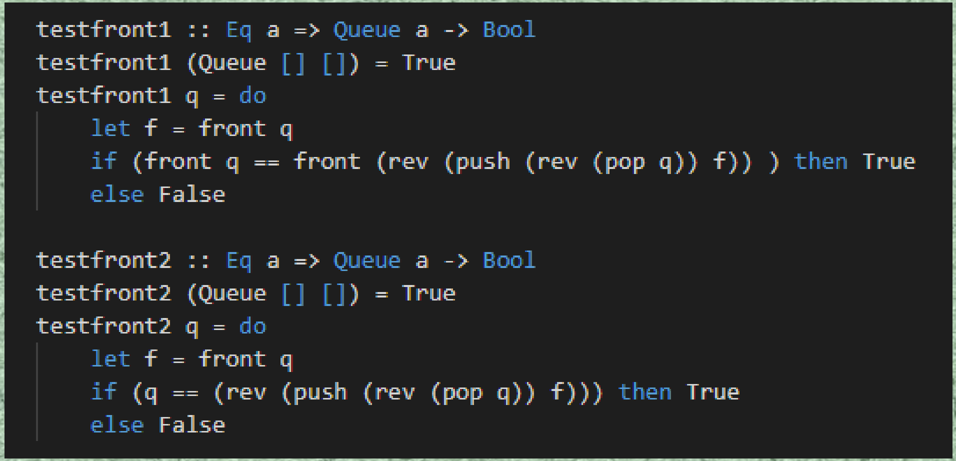
Property based testing



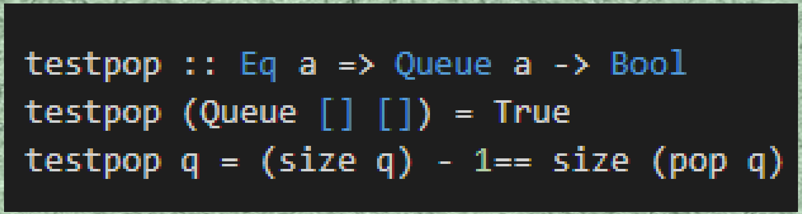
```haskell
valempty :: Queue a -> Bool -> Bool
valempty (Queue [] []) True = True
valempty (Queue [] []) False = False
valempty (Queue xs ys) True = False
valempty (Queue xs ys) False = True

testempty :: Eq a => Queue a -> Bool
testempty q = valempty q (empty q)
```



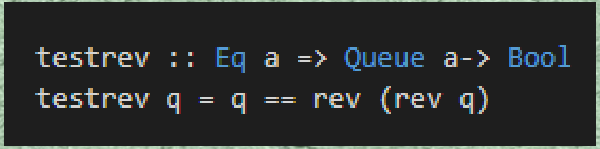
```haskell
testsize :: Eq a => Queue a -> Bool
testsize (Queue xs ys) = size (Queue xs ys) == (length xs) + (length ys)
```



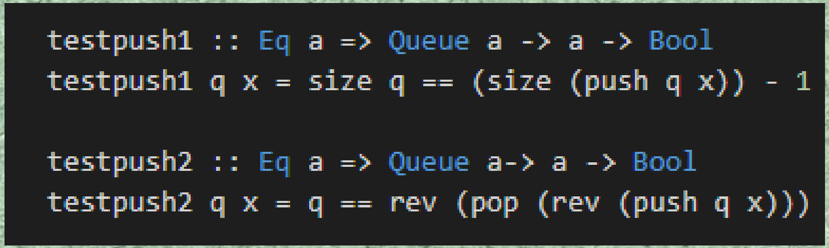
```haskell
testfront1 :: Eq a => Queue a -> Bool
testfront1 (Queue [] []) = True
testfront1 q = do
    let f = front q
    if (front q == front (rev (push (rev (pop q)) f)) ) then True
    else False

testfront2 :: Eq a => Queue a -> Bool
testfront2 (Queue [] []) = True
testfront2 q = do
    let f = front q
    if (q == (rev (push (rev (pop q)) f))) then True
    else False
```



```haskell
testpop :: Eq a => Queue a -> Bool
testpop (Queue [] []) = True
testpop q = (size q) - 1== size (pop q)
```

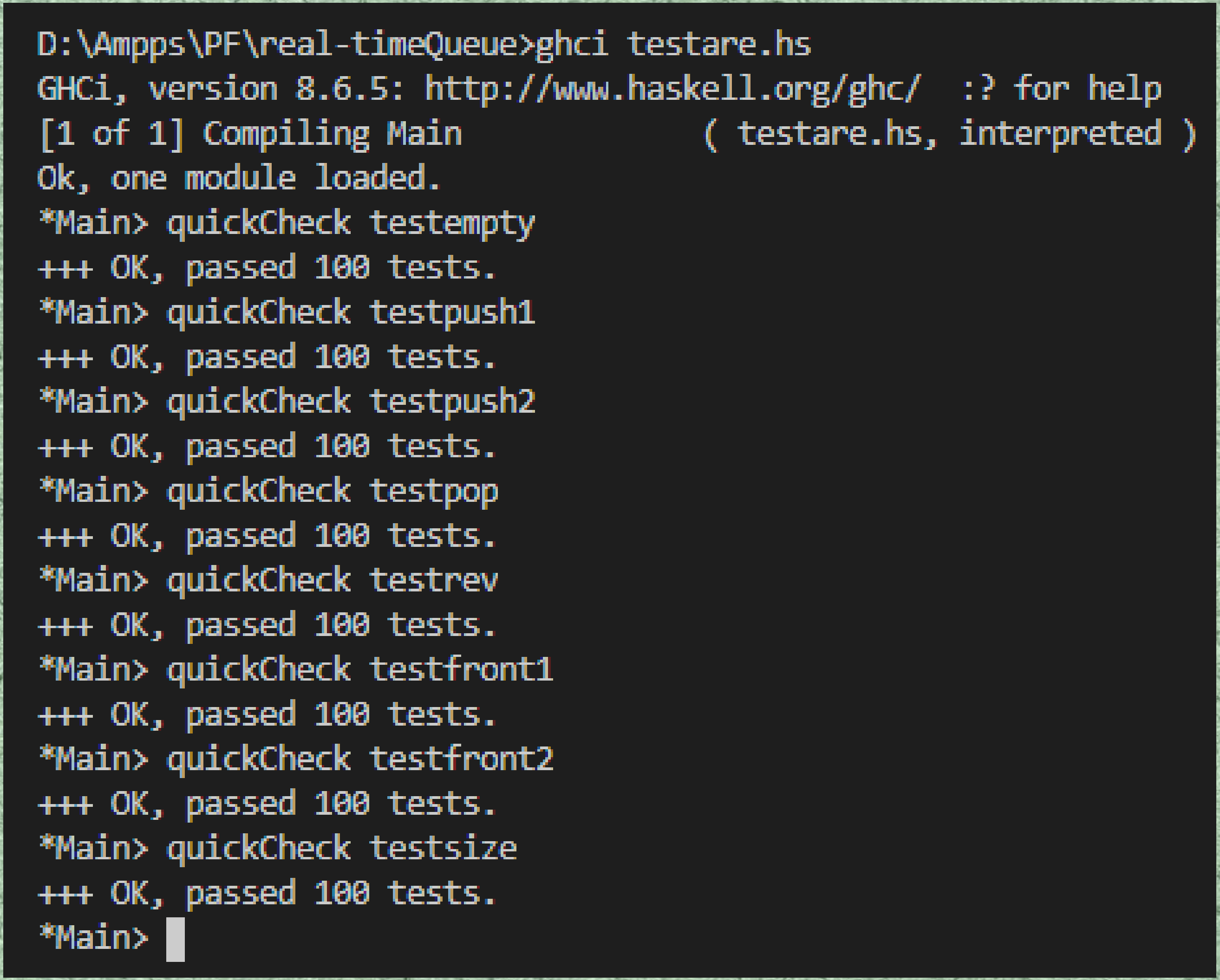


```haskell
testrev :: Eq a => Queue a-> Bool
testrev q = q == rev (rev q)
```



```haskell
testpush1 :: Eq a => Queue a -> a -> Bool
testpush1 q x = size q == (size (push q x)) - 1

testpush2 :: Eq a => Queue a-> a -> Bool
testpush2 q x = q == rev (pop (rev (push q x)))
```

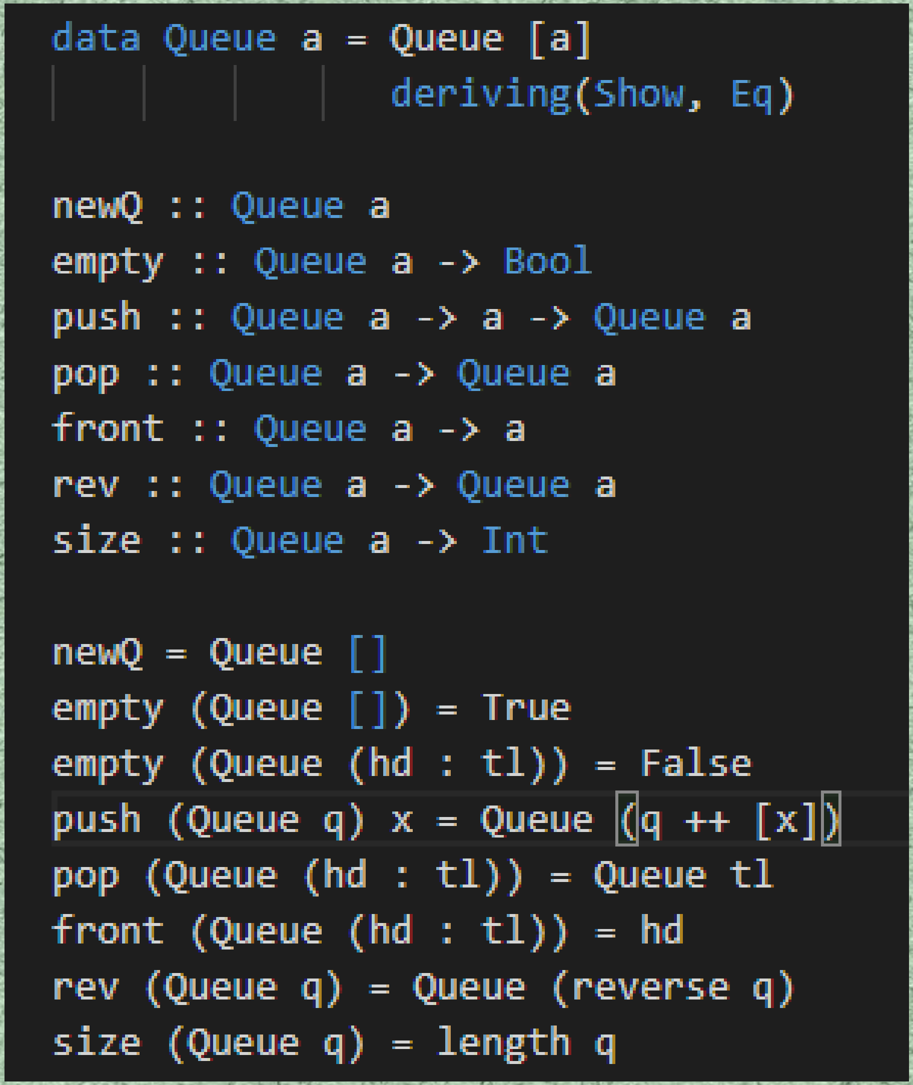
Property based testing



```
D:\Ampps\PF\real-timeQueue>ghci testare.hs
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
[1 of 1] Compiling Main             ( testare.hs, interpreted )
Ok, one module loaded.
*Main> quickCheck testempty
+++ OK, passed 100 tests.
*Main> quickCheck testpush1
+++ OK, passed 100 tests.
*Main> quickCheck testpush2
+++ OK, passed 100 tests.
*Main> quickCheck testpop
+++ OK, passed 100 tests.
*Main> quickCheck testrev
+++ OK, passed 100 tests.
*Main> quickCheck testfront1
+++ OK, passed 100 tests.
*Main> quickCheck testfront2
+++ OK, passed 100 tests.
*Main> quickCheck testsize
+++ OK, passed 100 tests.
*Main>
```

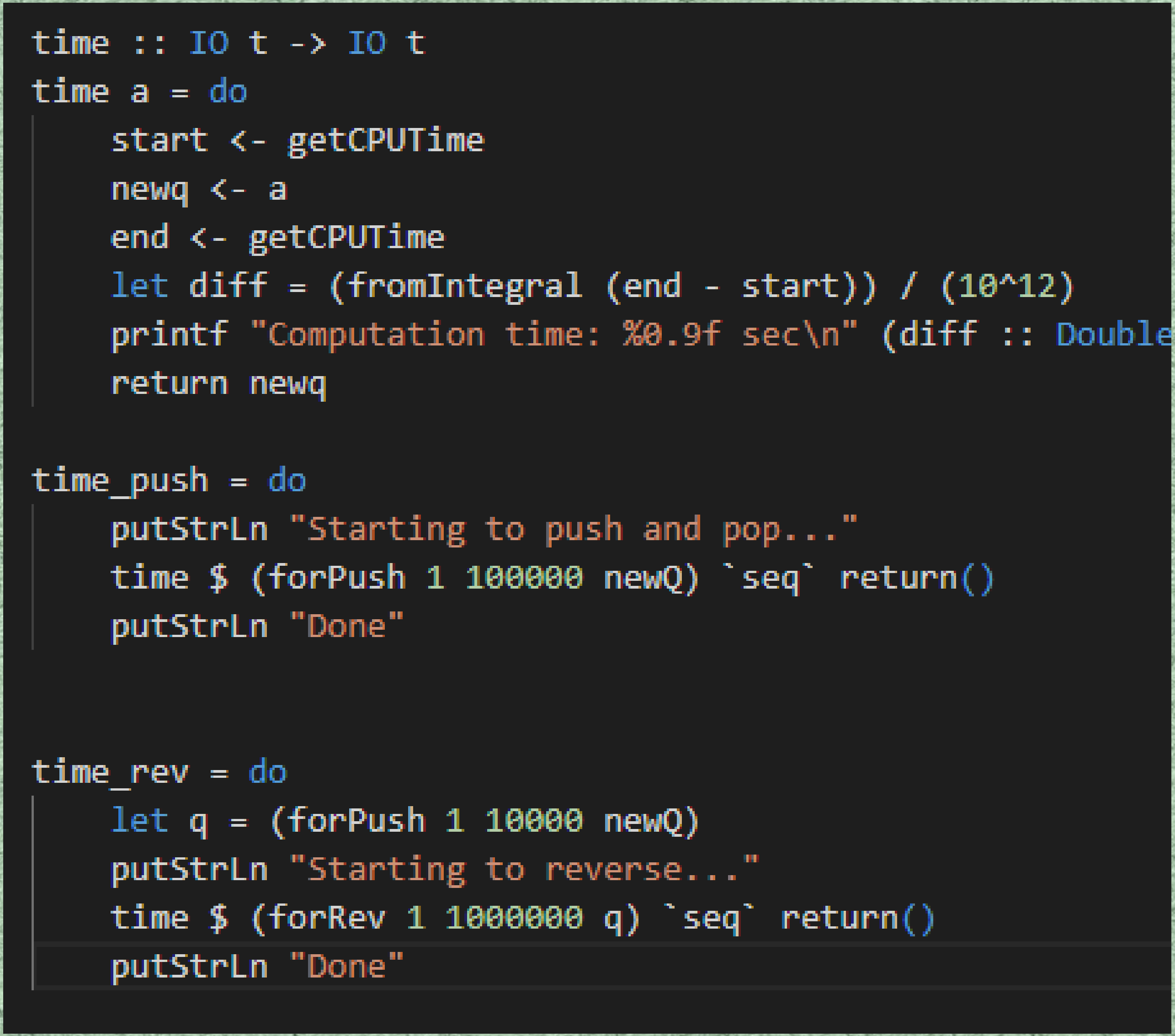
Diferenta timpului de executie

Diferenta timpului de executie



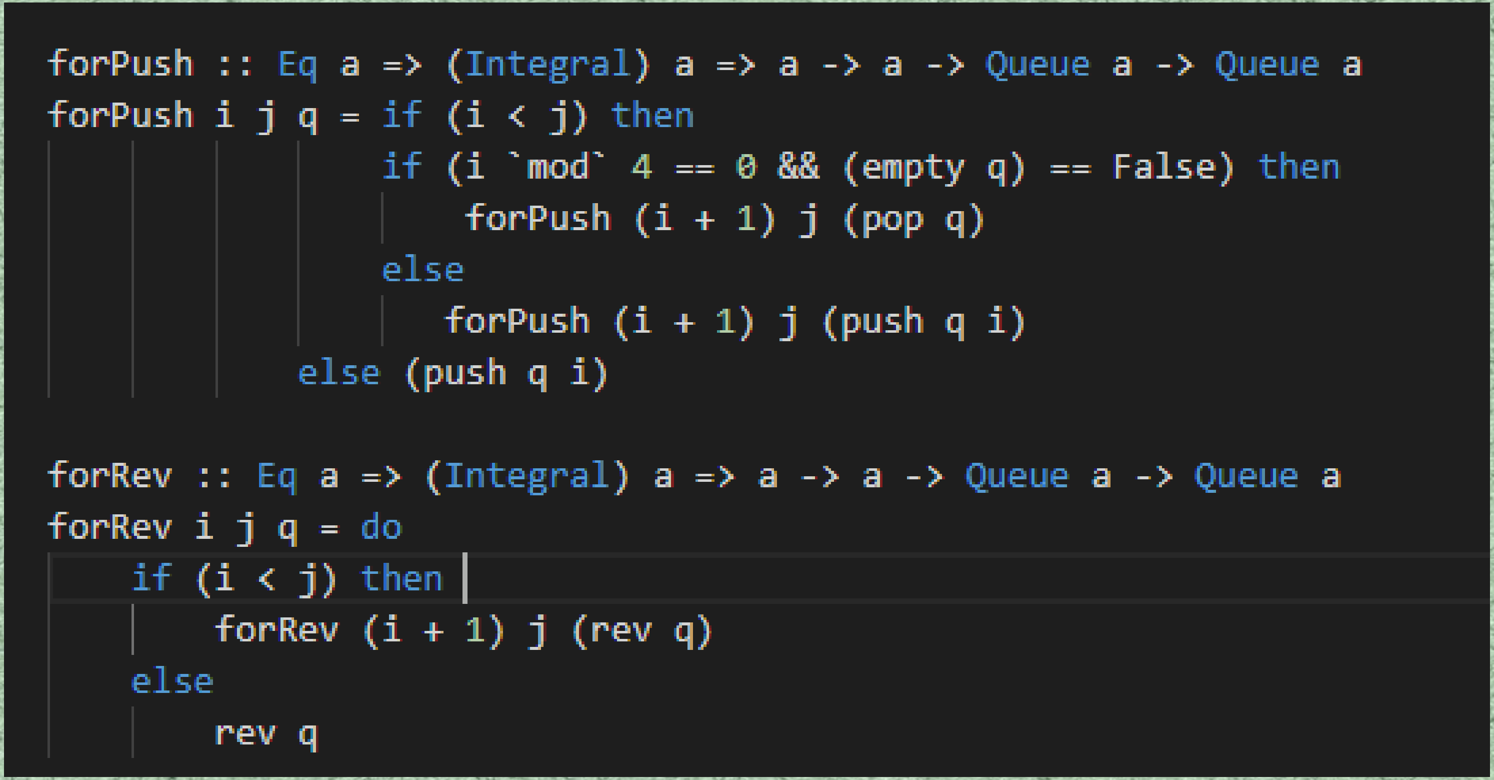
```haskell
data Queue a = Queue [a]
               deriving(Show, Eq)

newQ :: Queue a
empty :: Queue a -> Bool
push :: Queue a -> a -> Queue a
pop :: Queue a -> Queue a
front :: Queue a -> a
rev :: Queue a -> Queue a
size :: Queue a -> Int

newQ = Queue []
empty (Queue []) = True
empty (Queue (hd : tl)) = False
push (Queue q) x = Queue (q ++ [x])
pop (Queue (hd : tl)) = Queue tl
front (Queue (hd : tl)) = hd
rev (Queue q) = Queue (reverse q)
size (Queue q) = length q
```

Diferenta timpului de executie



```
time :: IO t -> IO t
time a = do
    start <- getCPUTime
    newq <- a
    end <- getCPUTime
    let diff = (fromIntegral (end - start)) / (10^12)
    printf "Computation time: %0.9f sec\n" (diff :: Double
    return newq

time_push = do
    putStrLn "Starting to push and pop..."
    time $ (forPush 1 100000 newQ) `seq` return()
    putStrLn "Done"


time_rev = do
    let q = (forPush 1 10000 newQ)
    putStrLn "Starting to reverse..."
    time $ (forRev 1 1000000 q) `seq` return()
    putStrLn "Done"
```

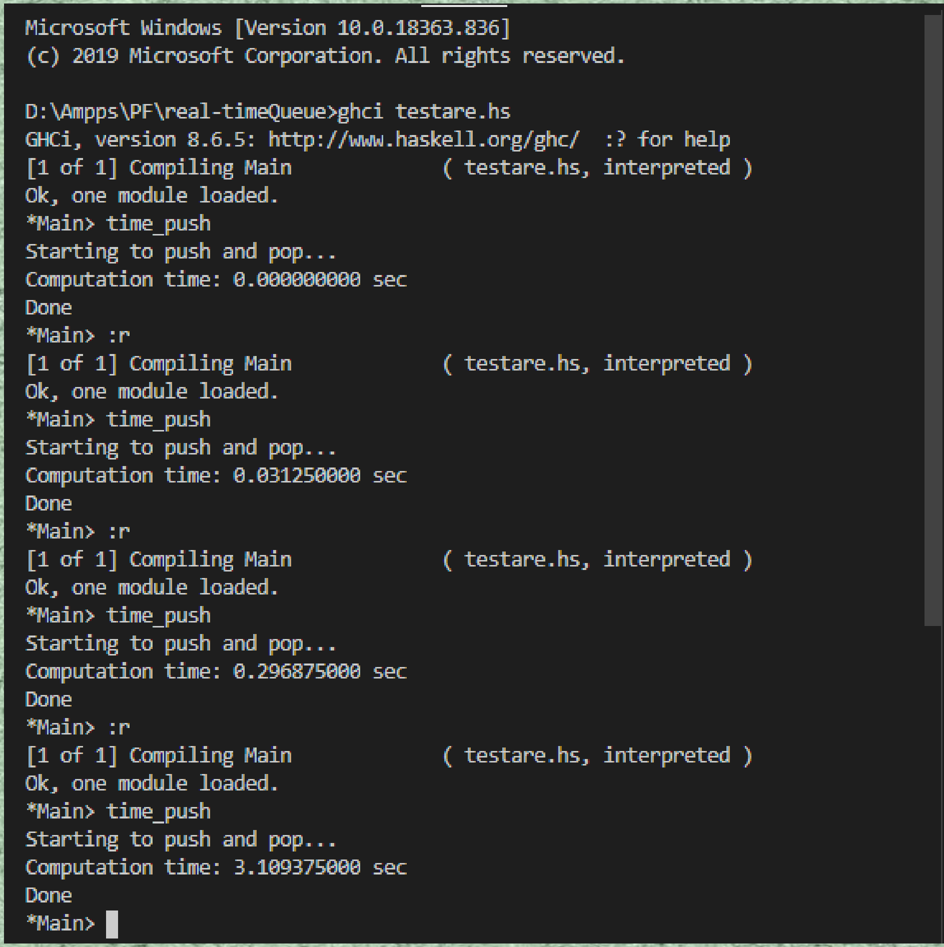
Diferenta timpului de executie



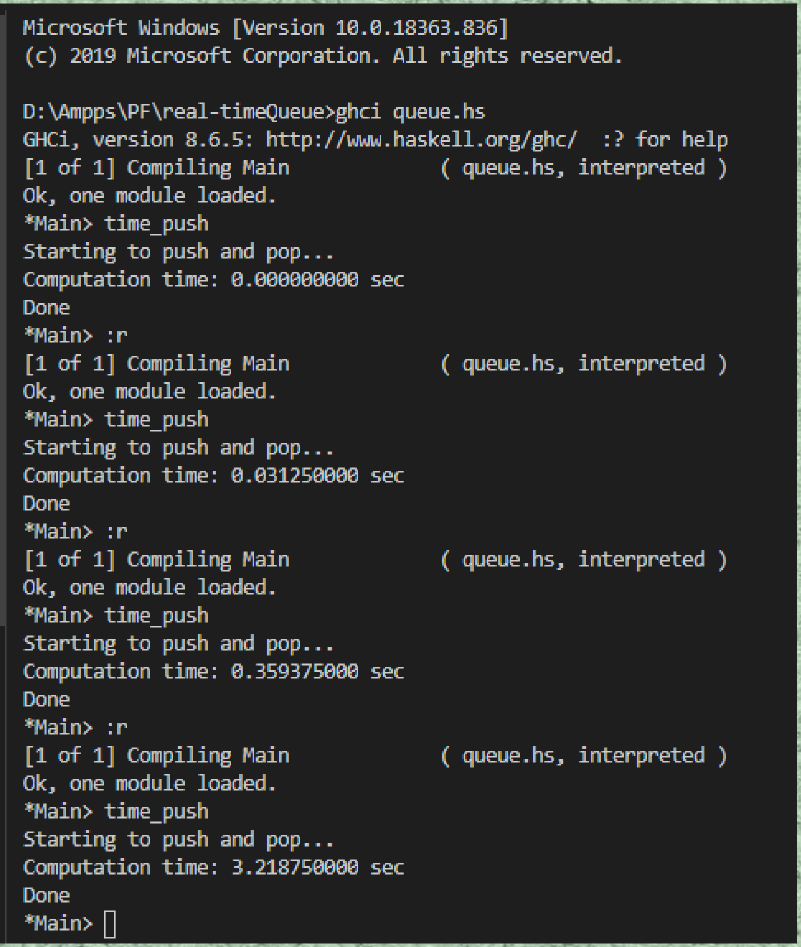
```haskell
forPush :: Eq a => (Integral) a => a -> a -> Queue a -> Queue a
forPush i j q = if (i < j) then
                    if (i `mod` 4 == 0 && (empty q) == False) then
                        forPush (i + 1) j (pop q)
                    else
                       forPush (i + 1) j (push q i)
                else (push q i)

forRev :: Eq a => (Integral) a => a -> a -> Queue a -> Queue a
forRev i j q = do
    if (i < j) then
        forRev (i + 1) j (rev q)
    else
        rev q
```

Rezultate

Push de 1.000 -> 1.000.000 de elemente fara a face pop

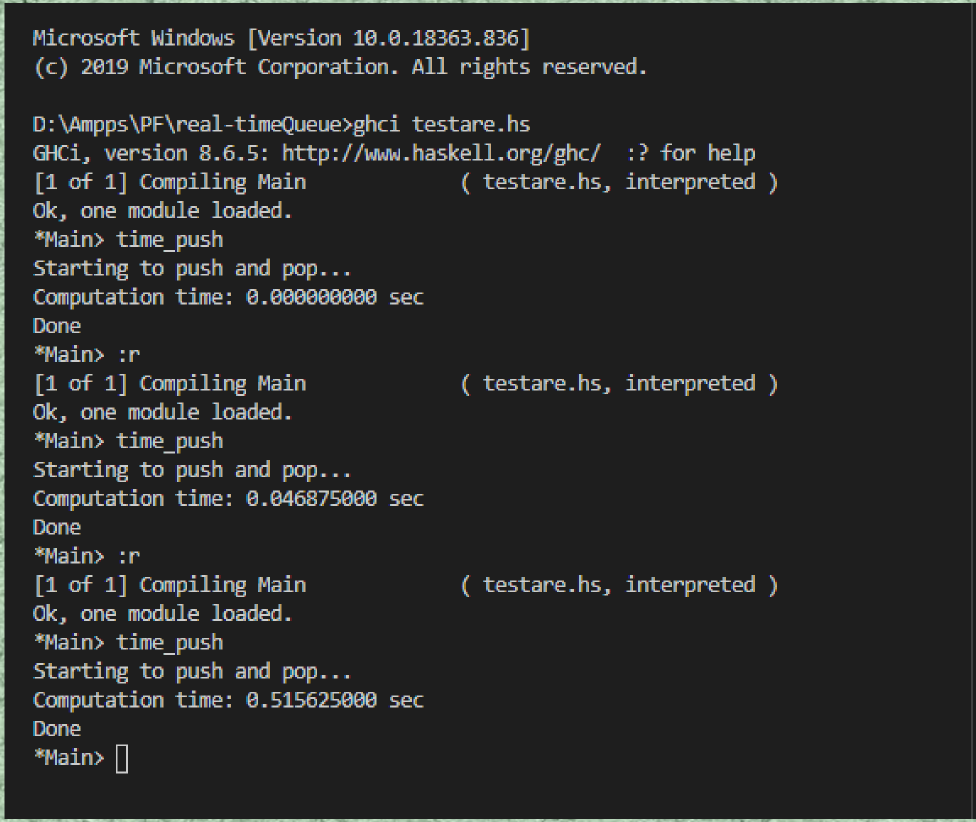


```
Microsoft Windows [Version 10.0.18363.836]
(c) 2019 Microsoft Corporation. All rights reserved.

D:\Ampps\PF\real-timeQueue>ghci testare.hs
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
[1 of 1] Compiling Main             ( testare.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 0.000000000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( testare.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 0.031250000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( testare.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 0.296875000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( testare.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 3.109375000 sec
Done
*Main>
```

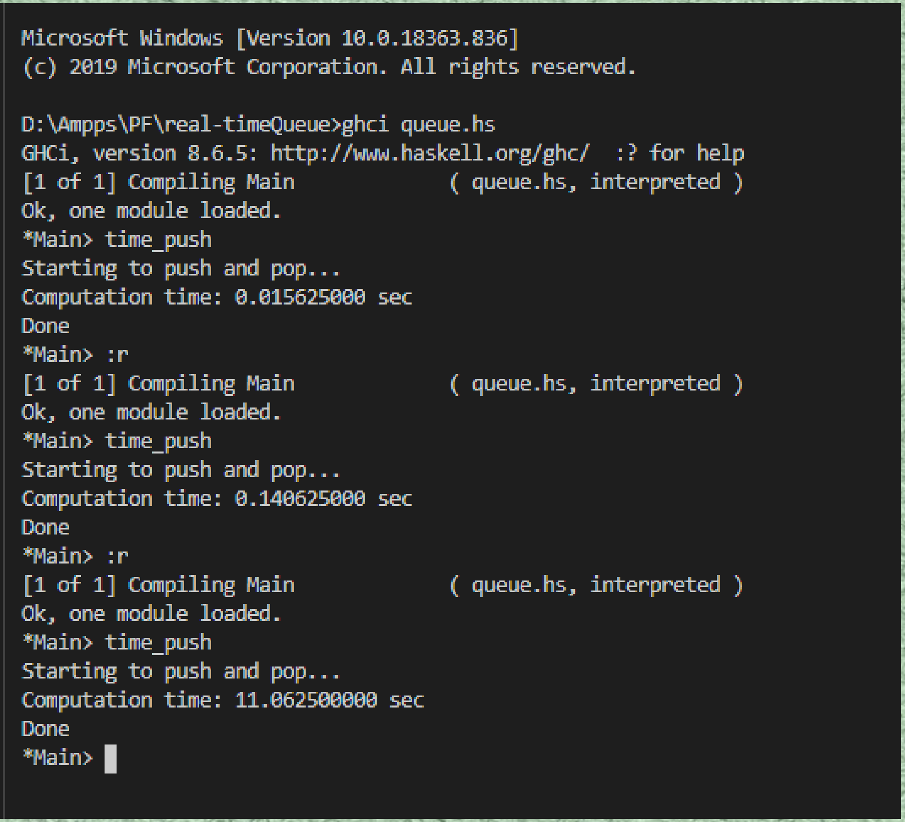


```
Microsoft Windows [Version 10.0.18363.836]
(c) 2019 Microsoft Corporation. All rights reserved.

D:\Ampps\PF\real-timeQueue>ghci queue.hs
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
[1 of 1] Compiling Main             ( queue.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 0.000000000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( queue.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 0.031250000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( queue.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 0.359375000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( queue.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 3.218750000 sec
Done
*Main>
```

Push de 1.000 -> 1.000.000 de elemente facand pop din 100 in 100

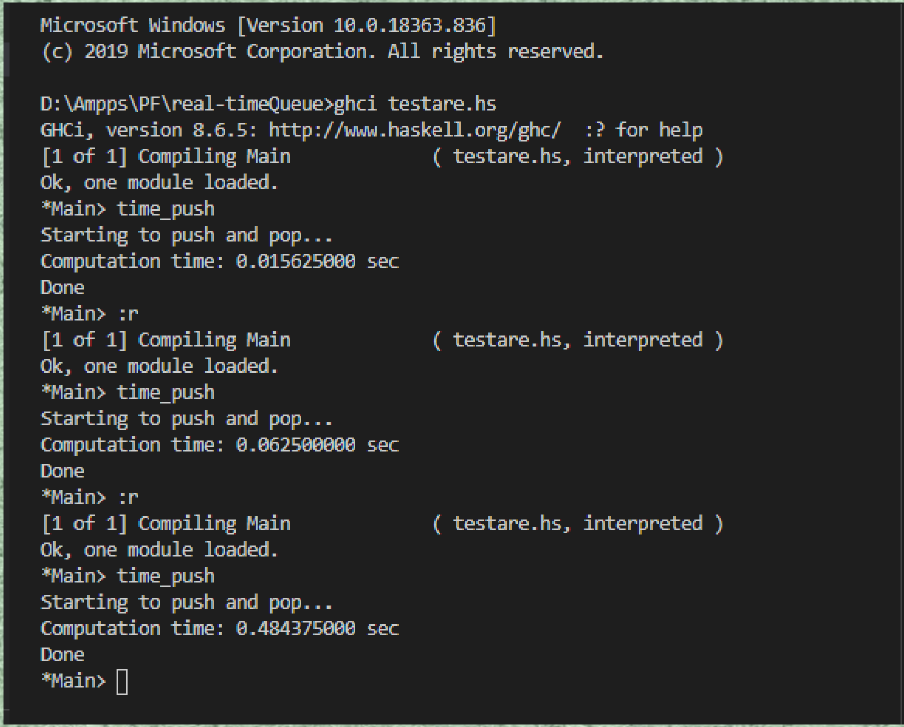


```
Microsoft Windows [Version 10.0.18363.836]
(c) 2019 Microsoft Corporation. All rights reserved.

D:\Ampps\PF\real-timeQueue>ghci testare.hs
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
[1 of 1] Compiling Main             ( testare.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 0.000000000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( testare.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 0.046875000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( testare.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 0.515625000 sec
Done
*Main>
```

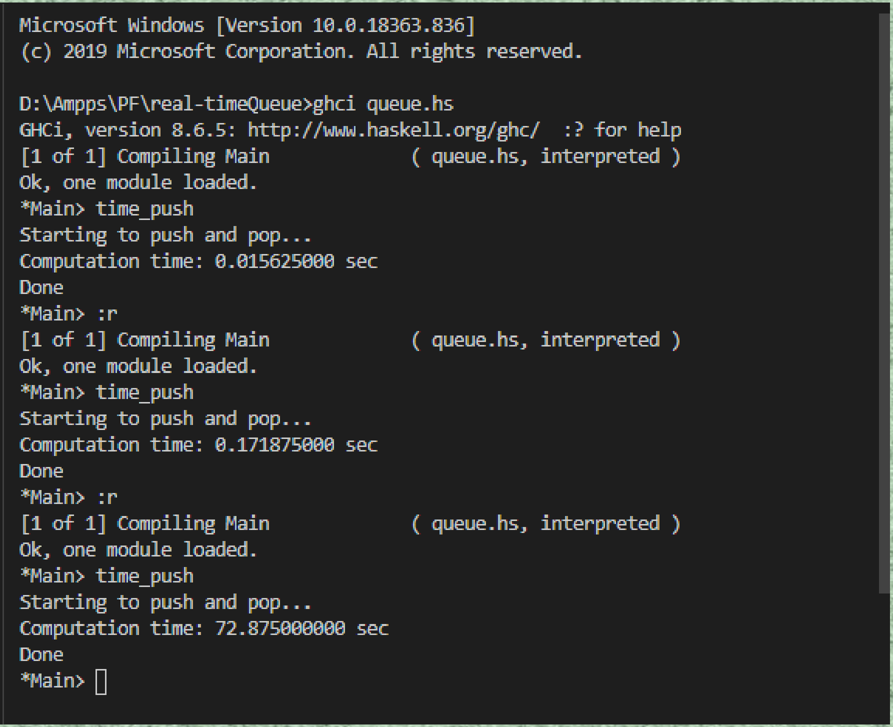


```
Microsoft Windows [Version 10.0.18363.836]
(c) 2019 Microsoft Corporation. All rights reserved.

D:\Ampps\PF\real-timeQueue>ghci queue.hs
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
[1 of 1] Compiling Main             ( queue.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 0.015625000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( queue.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 0.140625000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( queue.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 11.062500000 sec
Done
*Main>
```

Push de 1.000 -> 100.000 de elemente facand pop din 10 in 10

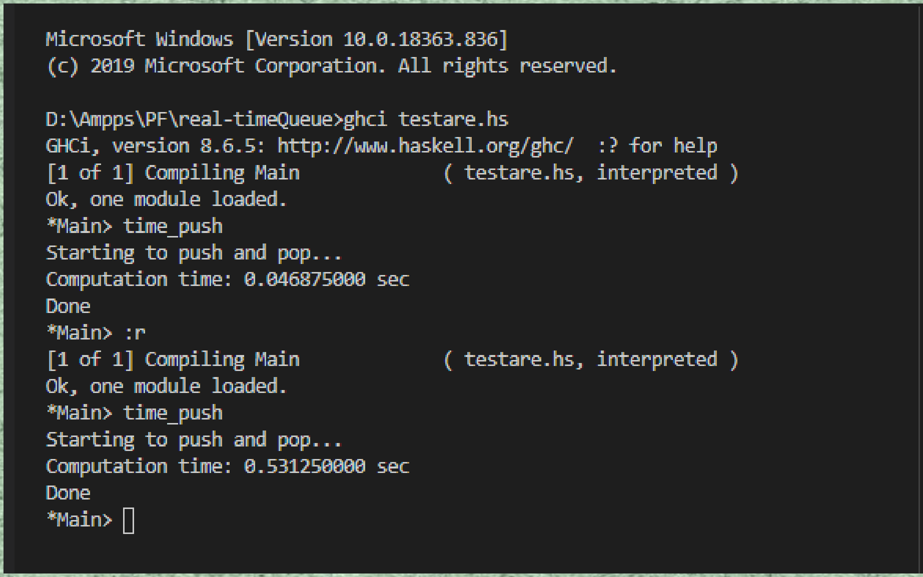


```
Microsoft Windows [Version 10.0.18363.836]
(c) 2019 Microsoft Corporation. All rights reserved.

D:\Ampps\PF\real-timeQueue>ghci testare.hs
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
[1 of 1] Compiling Main             ( testare.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 0.015625000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( testare.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 0.062500000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( testare.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 0.484375000 sec
Done
*Main> 
```

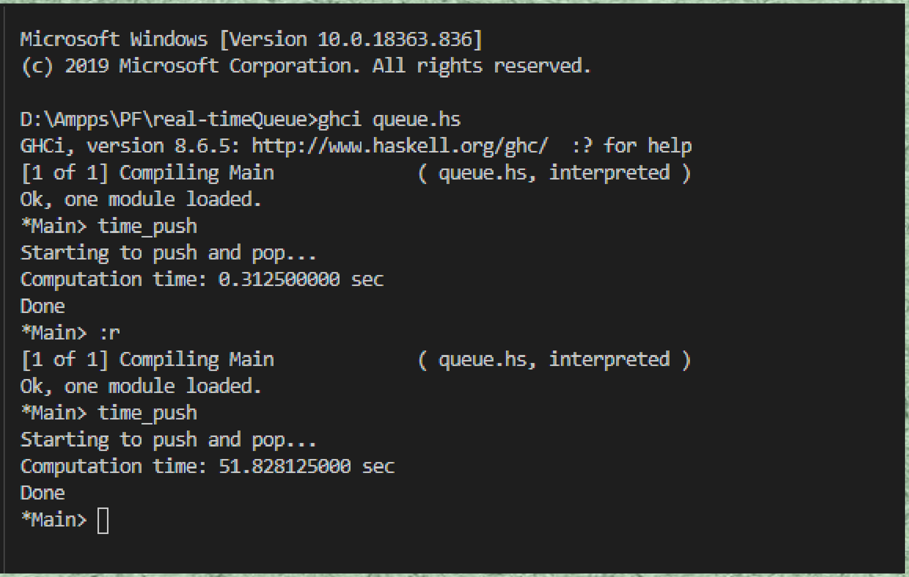


```
Microsoft Windows [Version 10.0.18363.836]
(c) 2019 Microsoft Corporation. All rights reserved.

D:\Ampps\PF\real-timeQueue>ghci queue.hs
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
[1 of 1] Compiling Main             ( queue.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 0.015625000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( queue.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 0.171875000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( queue.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 72.875000000 sec
Done
*Main> 
```

Push de 1.000 -> 10.000 de elemente facand pop din 3 in 3

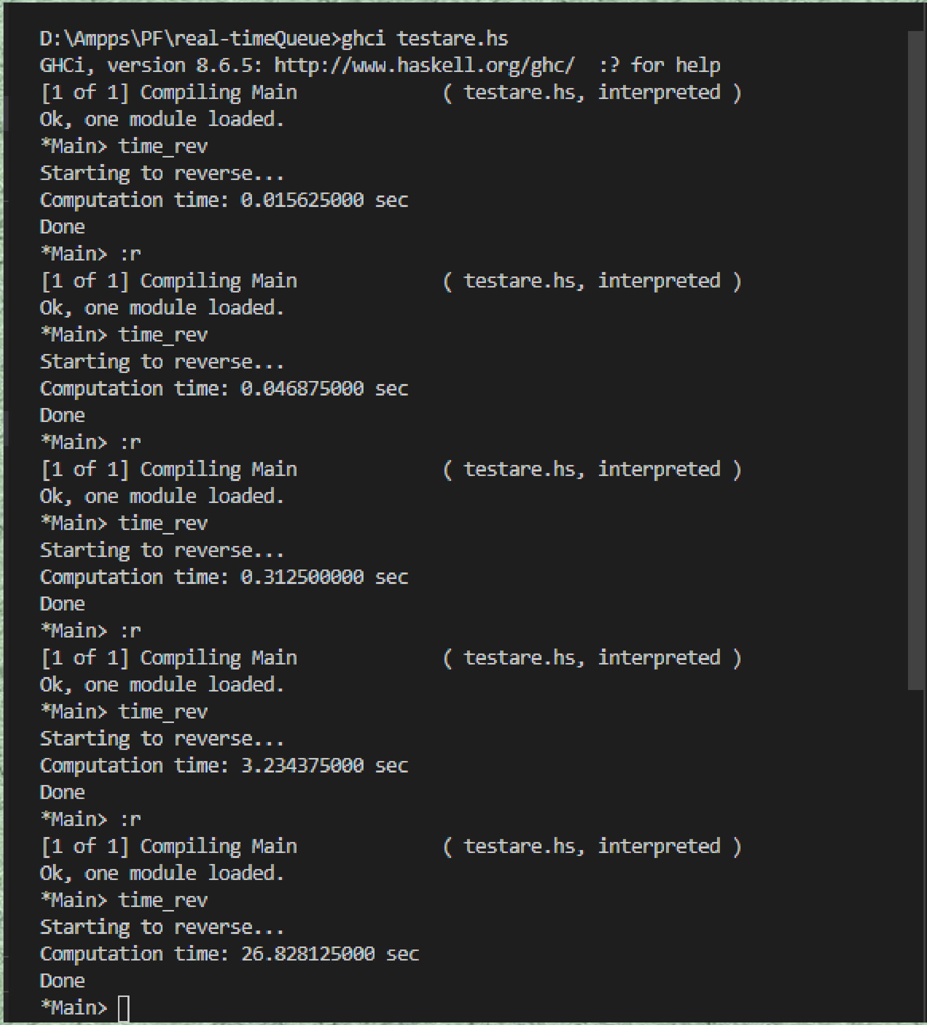


```
Microsoft Windows [Version 10.0.18363.836]
(c) 2019 Microsoft Corporation. All rights reserved.

D:\Ampps\PF\real-timeQueue>ghci testare.hs
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
[1 of 1] Compiling Main             ( testare.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 0.046875000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( testare.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 0.531250000 sec
Done
*Main>
```

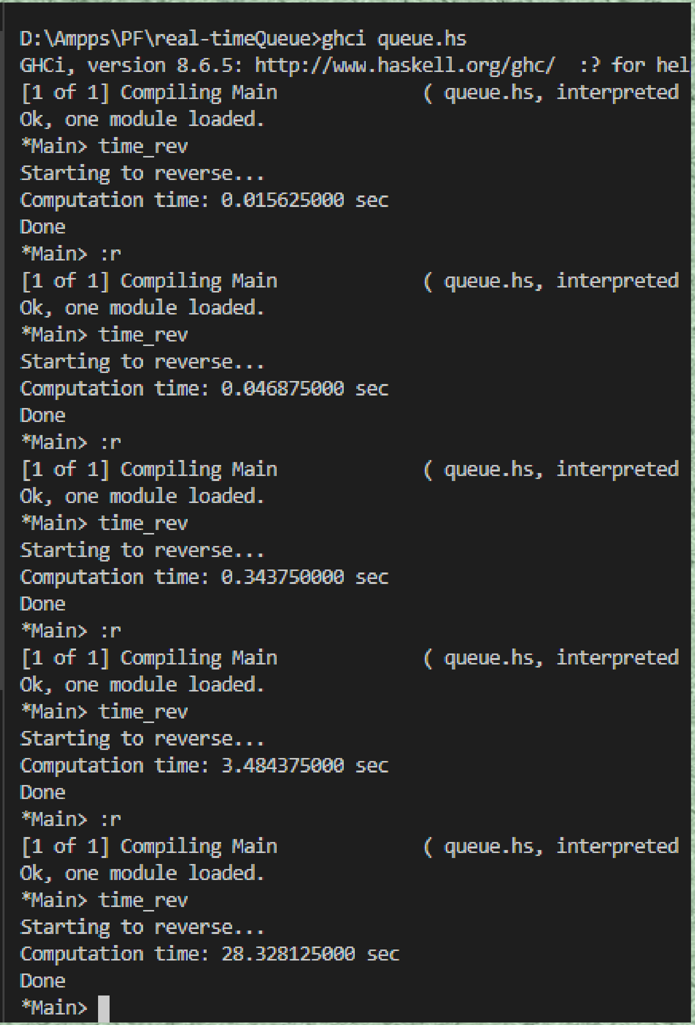


```
Microsoft Windows [Version 10.0.18363.836]
(c) 2019 Microsoft Corporation. All rights reserved.

D:\Ampps\PF\real-timeQueue>ghci queue.hs
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
[1 of 1] Compiling Main             ( queue.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 0.312500000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( queue.hs, interpreted )
Ok, one module loaded.
*Main> time_push
Starting to push and pop...
Computation time: 51.828125000 sec
Done
*Main>
```

Rev de 100 -> 1.000.000 la 1.000 de elemente

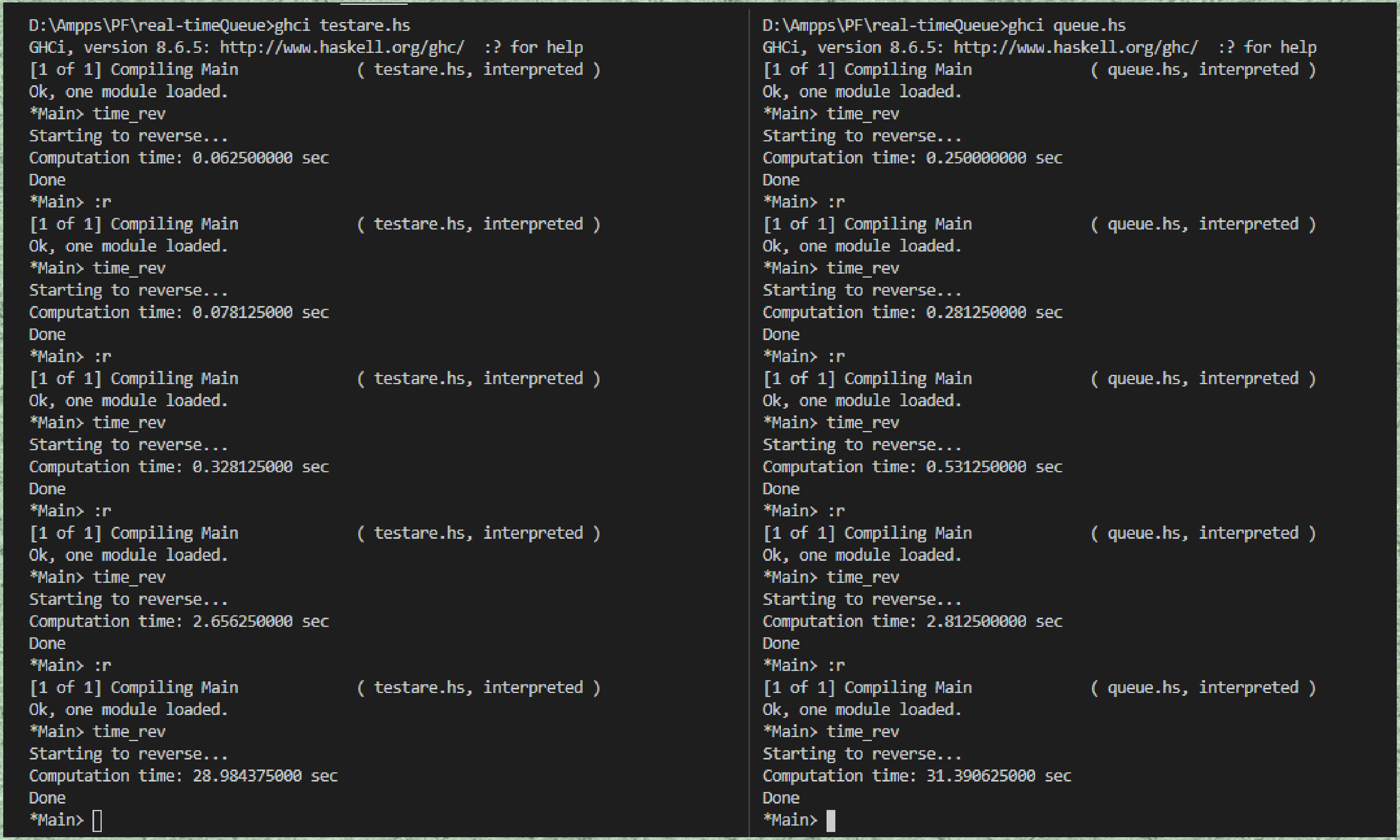


```
D:\Ampps\PF\real-timeQueue>ghci testare.hs
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
[1 of 1] Compiling Main             ( testare.hs, interpreted )
Ok, one module loaded.
*Main> time_rev
Starting to reverse...
Computation time: 0.015625000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( testare.hs, interpreted )
Ok, one module loaded.
*Main> time_rev
Starting to reverse...
Computation time: 0.046875000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( testare.hs, interpreted )
Ok, one module loaded.
*Main> time_rev
Starting to reverse...
Computation time: 0.312500000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( testare.hs, interpreted )
Ok, one module loaded.
*Main> time_rev
Starting to reverse...
Computation time: 3.234375000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( testare.hs, interpreted )
Ok, one module loaded.
*Main> time_rev
Starting to reverse...
Computation time: 26.828125000 sec
Done
*Main>
```



```
D:\Ampps\PF\real-timeQueue>ghci queue.hs
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for hel
[1 of 1] Compiling Main             ( queue.hs, interpreted
Ok, one module loaded.
*Main> time_rev
Starting to reverse...
Computation time: 0.015625000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( queue.hs, interpreted
Ok, one module loaded.
*Main> time_rev
Starting to reverse...
Computation time: 0.046875000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( queue.hs, interpreted
Ok, one module loaded.
*Main> time_rev
Starting to reverse...
Computation time: 0.343750000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( queue.hs, interpreted
Ok, one module loaded.
*Main> time_rev
Starting to reverse...
Computation time: 3.484375000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( queue.hs, interpreted
Ok, one module loaded.
*Main> time_rev
Starting to reverse...
Computation time: 28.328125000 sec
Done
*Main>
```

Rev de 100 -> 1.000.000 la 10.000 de elemente



```
D:\Ampps\PF\real-timeQueue>ghci testare.hs
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
[1 of 1] Compiling Main             ( testare.hs, interpreted )
Ok, one module loaded.
*Main> time_rev
Starting to reverse...
Computation time: 0.062500000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( testare.hs, interpreted )
Ok, one module loaded.
*Main> time_rev
Starting to reverse...
Computation time: 0.078125000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( testare.hs, interpreted )
Ok, one module loaded.
*Main> time_rev
Starting to reverse...
Computation time: 0.328125000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( testare.hs, interpreted )
Ok, one module loaded.
*Main> time_rev
Starting to reverse...
Computation time: 2.656250000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( testare.hs, interpreted )
Ok, one module loaded.
*Main> time_rev
Starting to reverse...
Computation time: 28.984375000 sec
Done
*Main>
```

```
D:\Ampps\PF\real-timeQueue>ghci queue.hs
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
[1 of 1] Compiling Main             ( queue.hs, interpreted )
Ok, one module loaded.
*Main> time_rev
Starting to reverse...
Computation time: 0.250000000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( queue.hs, interpreted )
Ok, one module loaded.
*Main> time_rev
Starting to reverse...
Computation time: 0.281250000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( queue.hs, interpreted )
Ok, one module loaded.
*Main> time_rev
Starting to reverse...
Computation time: 0.531250000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( queue.hs, interpreted )
Ok, one module loaded.
*Main> time_rev
Starting to reverse...
Computation time: 2.812500000 sec
Done
*Main> :r
[1 of 1] Compiling Main             ( queue.hs, interpreted )
Ok, one module loaded.
*Main> time_rev
Starting to reverse...
Computation time: 31.390625000 sec
Done
*Main>
```