**Academic Year 2024-25 Even**

**19CSE313 – Principles of Programming Language**

**B.Tech CSE 2022-26 F Section**

**Practice Set 1 – Working with Haskell Shell**

1. Command Prompt 🡪ghci
2. To exit, :quit
3. Arithmetic
   1. 2+2



* 1. 2\*3



* 1. + 2 2



* 1. (+) 2 2



* 1. 9/4



* 1. 9.0/4.0



* 1. 2^3



* 1. 8+9-2\*6/3^2



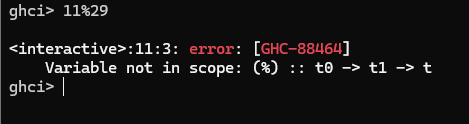
* 1. 5



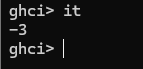
* 1. -3



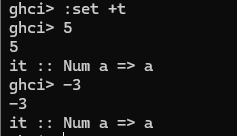
* 1. 11%29



1. it, :set, :unset, :info
   1. it



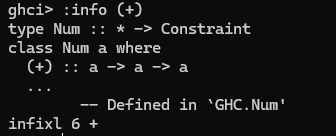
* 1. :set +t
     1. 5
     2. -3



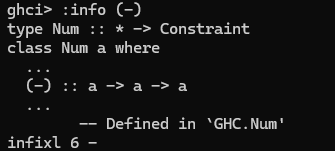
* 1. :unset +t



* 1. :info (operator)
     1. :info (+)



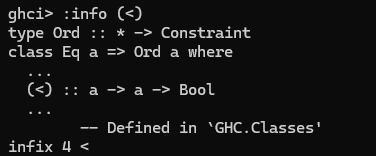
* + 1. :info (-)



* + 1. :info (^)



* + 1. :info (<)



1. Relative and Logical operators
   1. 5>7



* 1. 7>=5



* 1. 7==5



* 1. 7/=5



* 1. 5<=6



* 1. 5<6



* 1. 5-3\*4



* 1. 5-4\*3 < 13



* 1. 5-4\*3 < 13 && 6>3



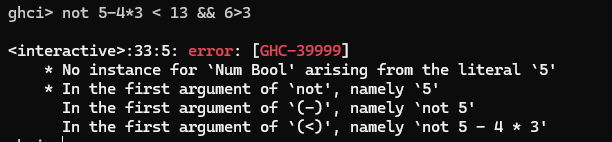
* 1. 5-4\*3 < 13 || 6>3



* 1. 5-4\*3 > 13 || 6<3



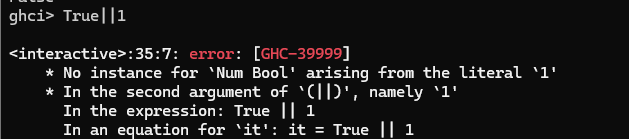
* 1. not 5-4\*3 < 13 && 6>3



* 1. not (5-4\*3 < 13 && 6>3)



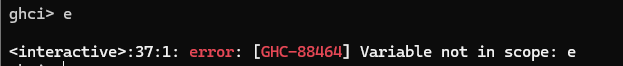
* 1. True || 1



1. Constants, variables and some predefined functions
   1. Pi



* 1. E



* 1. exp 1



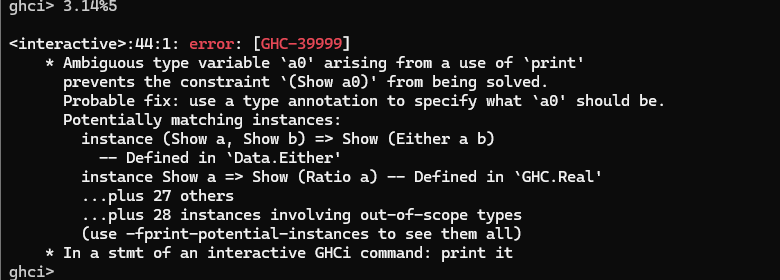
* 1. let e = exp 1



* 1. :m +Data.Ratio (for creating rationals using %)
  2. 11%29



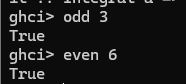
* 1. 3.14 % 5



* 1. :type it



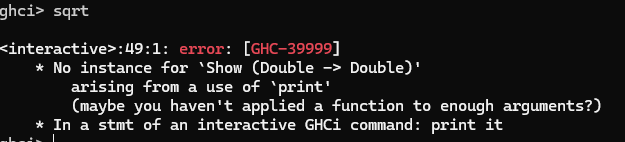
* 1. odd 3
  2. even 6



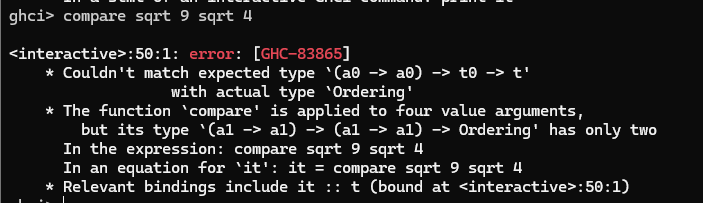
* 1. compare 5 6



* 1. sqrt



* 1. compare sqrt 9 sqrt 4



* 1. compare (sqrt 9) (sqrt 4)



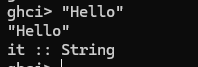
* 1. compare (sqrt 9) (sqrt 4) == LT



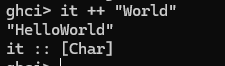
1. Strings and Lists
   1. :set +t



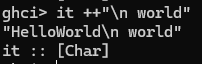
* 1. “Hello”



* 1. it ++ “ World”



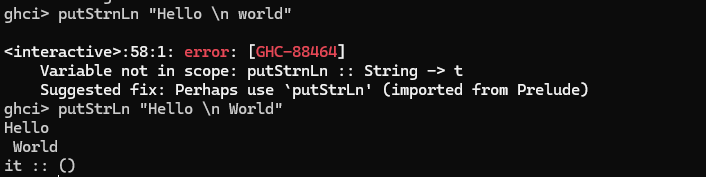
* 1. it ++ “\n World”



* 1. “Helllo \n World”



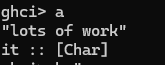
* 1. putStrLn "Hello \n World"



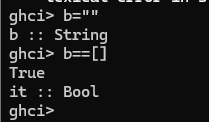
* 1. let a = ['l', 'o', 't', 's', ' ', 'o', 'f', ' ', 'w', 'o', 'r', 'k']



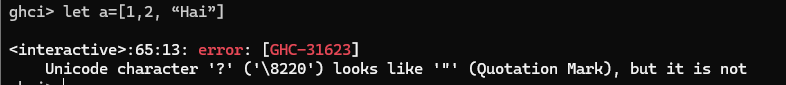
* 1. a



* 1. b = “”
  2. b==[]



* 1. let a=[1,2, “Hai”]



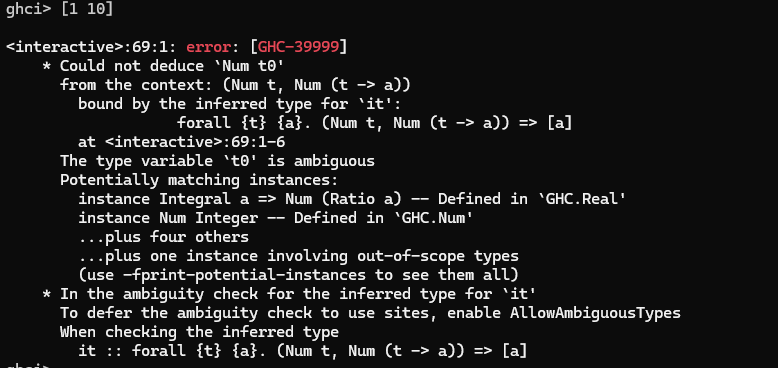
* 1. let a=[1,2,3]



* 1. :unset +t
  2. A



* 1. [1..10]



* 1. [1,1.5..10] , [1,4..15], [1,4..16]







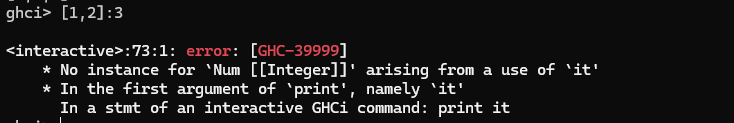
* 1. [1,2,3]++[4,5]



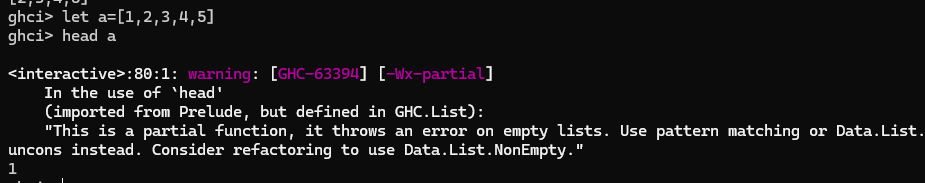
* 1. 1:[2,3]

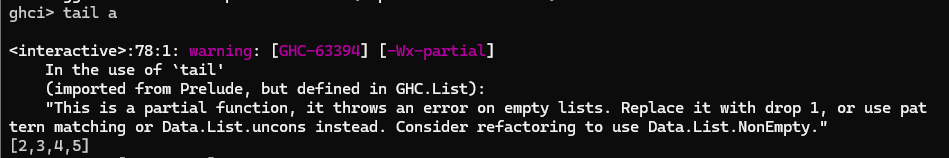


* 1. [1,2]:3



* 1. let a = [1,2,3,4,5]
  2. head a, tail a





* 1. head/tail on String? (head “World”/tail “World”)

