**…………….0Academic Year 2024-25 Even**

**19CSE313 – Principles of Programming Language**

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

**Practice Set 2 - Writing simple functions**

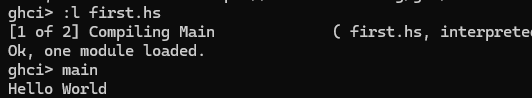
1. Printing a string

**printString :: String -> IO()**

**printString a = putStrLn a**

**main=do**

**printString "Hello World"**

****

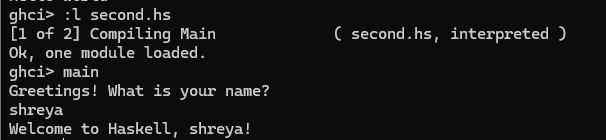
1. Reading a string and printing it

**main = do**

**putStrLn "Greetings! What is your name?"**

**inpStr <- getLine**

**putStrLn $ "Welcome to Haskell, " ++ inpStr ++ "!"**



1. Length of the string

**name2reply :: String -> String**

**name2reply name =**

**"Pleased to meet you, " ++ name ++ ".\n" ++**

**"Your name contains " ++ show (length name) ++ " characters."**

**main :: IO ()**

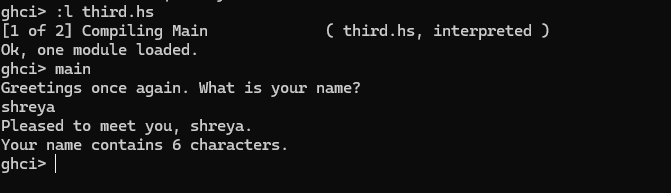
**main = do**

**putStrLn "Greetings once again. What is your name?"**

**inpStr <- getLine**

**let outStr = name2reply inpStr**

**putStrLn outStr**

****

1. **Write a method that returns the length of a given string**

**name2reply :: String -> Int**

**name2reply name =(length name)**

**main :: IO ()**

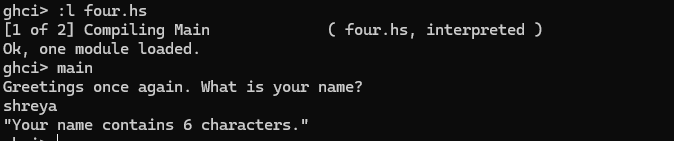
**main = do**

**putStrLn "Greetings once again. What is your name?"**

**inpStr <- getLine**

**let l=name2reply inpStr**

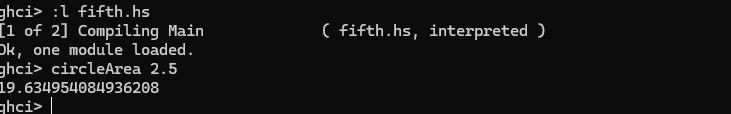
**print("Your name contains " ++ show (l) ++ " characters.")**

****

1. Calculating the area of a circle – only module

**circleArea :: Double -> Double**

**circleArea r = (r ^ 2) \* pi**

****

1. Calculating the area of a circle – with main

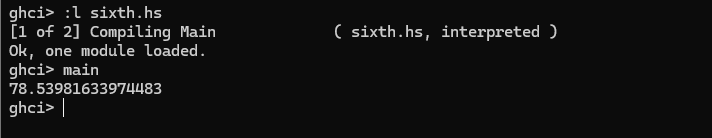
**circleArea :: Double -> Double**

**circleArea r = (r ^ 2) \* pi**

**main=do**

**let a = circleArea 5**

**print(a)**

****

1. Reading the radius from the user for area calculation

**circleArea :: Double -> Double**

**circleArea r = (r ^ 2) \* pi**

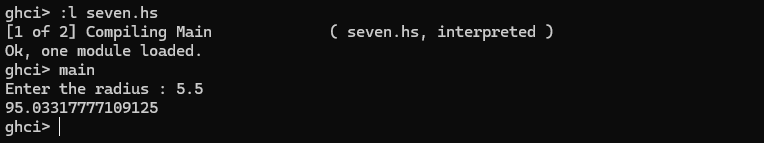
**main=do**

**putStr "Enter the radius : "**

**inp1 <- getLine**

**let a = (read inp1 :: Double)**

**print(circleArea a)**

****

1. **Calculate the area of a rectangle with user-input Length and Breadth values**

**arearect :: Double->Double->Double**

**arearect l b= l\*b**

**main=do**

**putStr "enter the length: "**

**inp1 <- getLine**

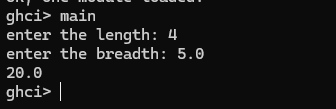
**putStr "enter the breadth: "**

**inp2 <-getLine**

**let a=(read inp1 :: Double)**

**let b=(read inp2 ::Double)**

**print(arearect a b)**

****

1. Some built-in functions – only main

**main = do**

**print(odd 3)**

**print(compare 2 3)**

**print(head [1,2,3,4])**

**print(tail [1,2,3,4])**

**print(tail "list")**

**print(take 2 [1,2,3,4,5])**

**print(drop 3 [1,2,3,4,5])**

****

1. Arithmetic operations using functions

**add :: Int->Int->Int**

**add a b=a+b**

**sub :: Int->Int->Int**

**sub c d=c-d**

**multiply :: Int->Int->Int**

**multiply c d=c\*d**

**divide :: Float->Float->Float**

**divide c d=c/d**

**exponentiate :: Int->Int->Int**

**exponentiate c d =c^d**

**main=do**

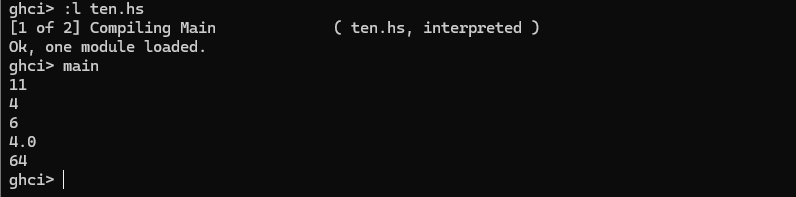
**print(add 5 6)**

**print(sub 8 4)**

**print(multiply 2 3)**

**print(divide 8 2)**

**print(exponentiate 4 3)**

****

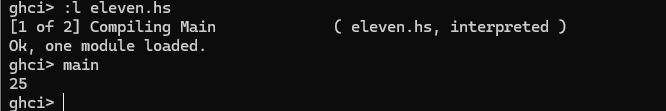
1. Squaring a number

**square :: Int->Int**

**square a = a^2**

**main=do**

**print(square 5)**

****

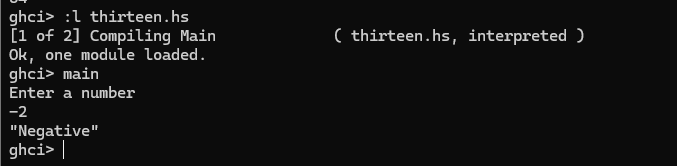
1. Get a number. Double it, if even. Cube it otherwise (if…then…else)

**doubleEvenCubeOdd :: Int -> Int**

**doubleEvenCubeOdd x = if even x then (2\*x) else (x^3)**

****

1. **Check whether a number is Negative, Zero or Positive and print the result**

****

**numb ::Int->String**

**numb x**

**|x>0 ="Positive"**

**|x==0= "Zero"**

**|otherwise="Negative"**

**main =do**

**putStrLn "Enter a number"**

**a<- getLine**

**let num=(read a ::Int)**

**print(numb num)**