Principles of Programming Language

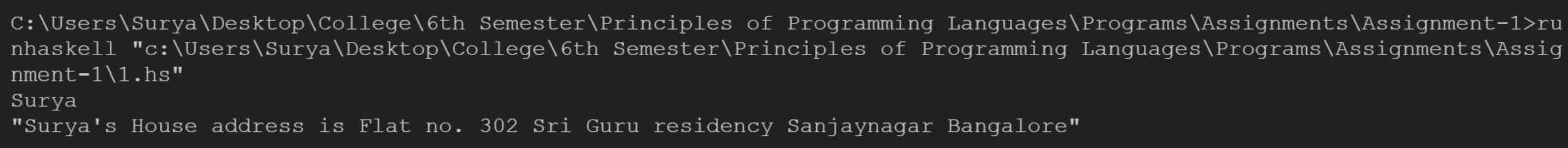
CB.EN.U4CSE19056

Suryanarayan.B

1) main =do

    name <- getLine

    print(name++ "'s House address is Flat no. 302 Sri Guru residency Sanjaynagar Bangalore")



2) mygcd :: Num a => (a, a) -> a

mygcd (a, b) = a + b

mygcd1 :: Num a => (a, a) -> a

mygcd1 (a, b) = a - b

mygcd2 :: Num a => (a, a) -> a

mygcd2 (a, b) = a \* b

divi :: Int -> Int -> Int

divi x y = x `div` y

modu :: Int -> Int -> Int

modu x y = x `mod` y

main = do

  a <- readLn

  b <- readLn

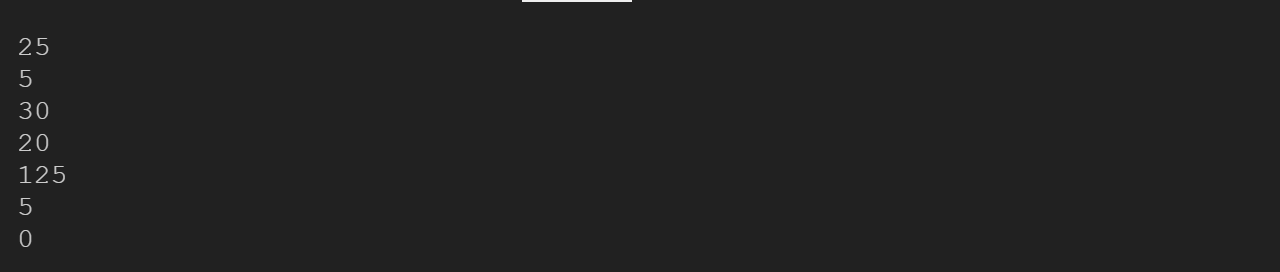
  print $ mygcd (a, b)

  print $ mygcd1 (a, b)

  print $ mygcd2 (a, b)

  print (divi a b)

  print( modu a b)

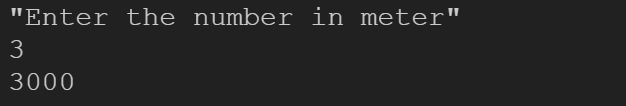


3) main = do

    print("Enter the number in meter")

    x <- readLn

    print(x\*1000)

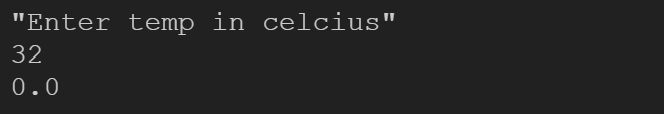


4) main = do

    print("Enter temp in celcius")

    x <- readLn

    print((x - 32) \* (5/9))

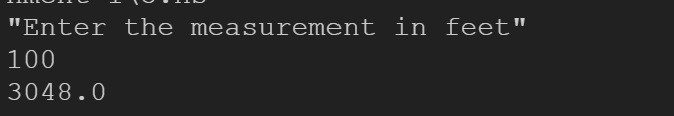


5) main = do

    print("Enter the measurement in feet")

    x <- readLn

    print(x \* 30.48)



6) main=do

    x <- readLn

    putStrLn "Square"

    print(x^2)

    putStrLn "Cube"

    print(x^3)

    putStrLn "SquareRoot"

    print(sqrt(x))



7) main = do

    putStrLn "Enter value of P"

    p <- getLine

    putStrLn "Enter value of T"

    t <- getLine

    putStrLn "Enter value of R"

    r <- getLine

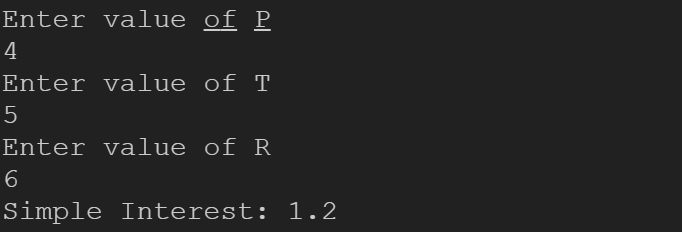
    let p1 = read p :: Double

    let r1 = read r :: Double

    let t1 = read t :: Double

    let s1 = (p1\*r1\*t1)/100

    putStrLn ("Simple Interest: " ++ show(s1))



8) main = do

    putStrLn "Enter the value of p"

    p <- getLine

    putStrLn "Enter the value of t"

    t <- getLine

    putStrLn "Enter the value of r"

    r <- getLine

    let px = read p :: Double

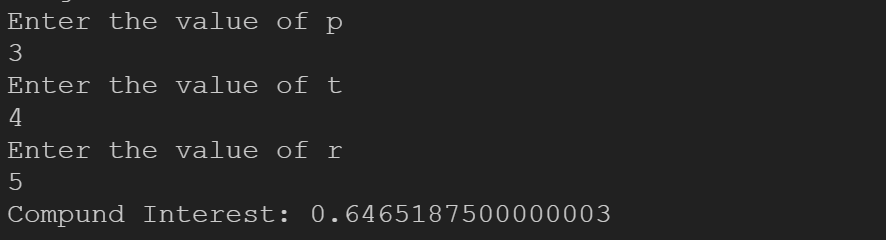
    let rx = read r :: Double

    let tx = read t :: Int

    let a = px\*(((1+(rx/100)))^tx)

    let c1 = a - px

    putStrLn ("Compund Interest: " ++ show(c1))



9) vol :: Float -> Float

vol r1 = (4/3) \* pi \* r1^3

ar :: Float -> Float

ar r1 = 4 \* pi \* r1^2

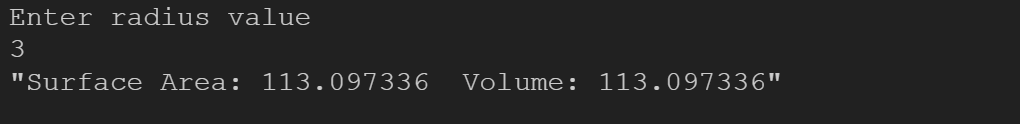
main = do

    putStrLn "Enter radius value"

    ra <- getLine

    let r1 = read ra :: Float

    print("Surface Area: " ++ show(ar r1) ++ "  Volume: " ++ show(vol r1))



10) hi :: Float -> Float -> Float -> Float

hi a b c = let s = (a + b + c)/2 in sqrt(s \* (s-a) \* (s-b) \* (s-c))

main = do

    putStrLn "Enter side values"

    a <- getLine

    let a1 = read a :: Float

    b <- getLine

    let b1 = read b :: Float

    c <- getLine

    let c1 = read c :: Float

    print("Area of the triangle " ++ show(hi a1 b1 c1))



11) import Data.Char

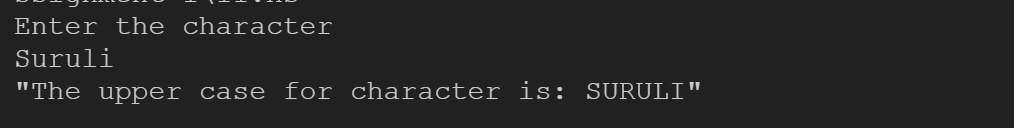
main = do

    putStrLn "Enter the character"

    x <- getLine

    let c1 = map toUpper x

    print("The upper case for character is: " ++ c1)



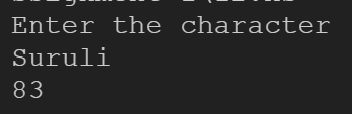
12) import Data.Char(ord)

main = do

    putStrLn "Enter the character"

    x <- getChar

    print(ord x)



13) listdigits :: Int -> [Int]

listdigits n = if n < 10 then [n]

               else (listdigits (n `div` 10)) ++ (listdigits (n `mod` 10))

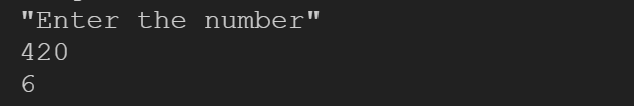
main = do

    print("Enter the number")

    x <- getLine

    let a1 = read x :: Int

    print(sum(listdigits a1))



14) getDouble :: IO Double

getDouble = readLn

main = do

    putStrLn "Enter The First Number "

    a <- getDouble

    putStrLn "Enter the Second Number"

    b <- getDouble

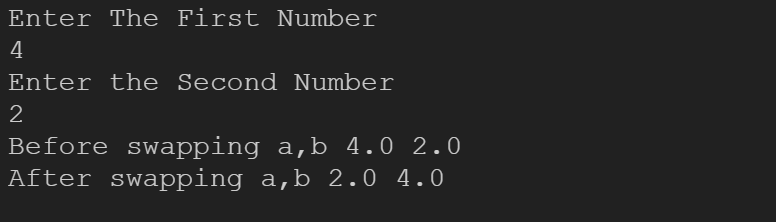
    putStrLn $ "Before swapping a,b " ++ show a ++" "++  show b

    let tem = a

    let a = b

    let b = tem

    putStrLn $ "After swapping a,b " ++ show a ++" "++  show b



15) main = do

    putStrLn "Enter the 1st number"

    a <- getLine

    putStrLn "Enter the 2nd number"

    b <- getLine

    let a1 = read a :: Int

    let b1 = read b :: Int

    putStrLn("The 1st variable is " )

    print a1

    putStrLn(" and the second variable is " )

    print b1

    putStrLn("After Swapping : ")

    let a1 = a1 + b1

    let b1 = a1 - b1

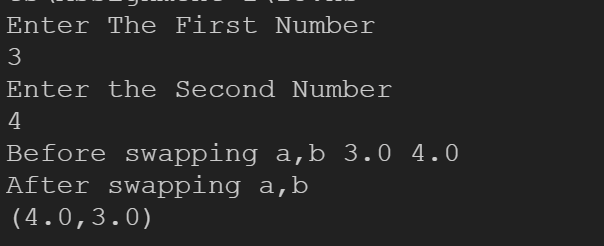
    let a1 = a1 - b1

    putStrLn $ "Before swapping a,b " ++ show a ++" "++  show b

    putStrLn $ "After swapping a,b "

print (a1)

print (b1)



16) disp :: Float -> Float -> Float -> Float

disp u t a = (u\*t) + 0.5\*(a\*(t^2))

main = do

    print("Enter the u t and a values: ")

    u <- getLine

    t <- getLine

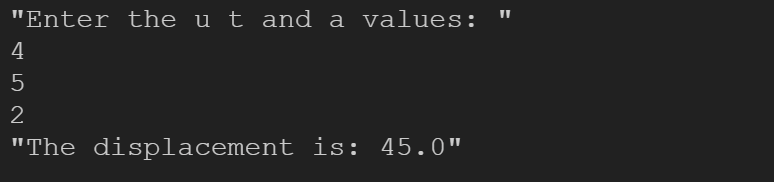
    a <- getLine

    let u1 = read u :: Float

    let t1 = read t :: Float

    let a1 = read a :: Float

    print("The displacement is: " ++ show(disp u1 t1 a1))



Conditional statements

1) grt :: Int -> Int -> Int

grt a 0 = 0

grt a b

    |a>b = a

    |otherwise = b

main = do

    putStrLn "Enter the 1st number"

    a <- getLine

    putStrLn "Enter the 2nd number"

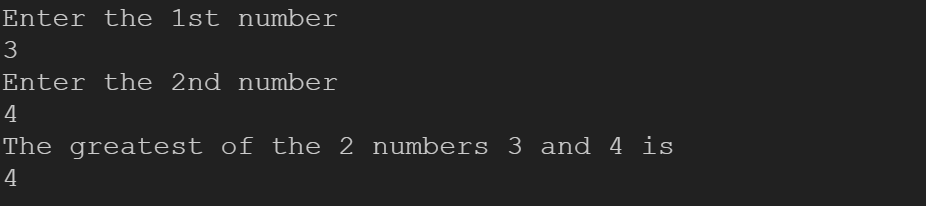
    b <- getLine

    let ax = read a :: Int

    let bx = read b :: Int

    putStrLn( "The greatest of the 2 numbers " ++ a ++ " and " ++ b ++ " is ")

    print(grt ax bx)



2) lrt :: Int -> Int -> Int

lrt a 0 = 0

lrt a b

    |a<b = a

    |otherwise = b

main = do

    putStrLn "Enter the 1st number"

    a <- getLine

    putStrLn "Enter the 2nd number"

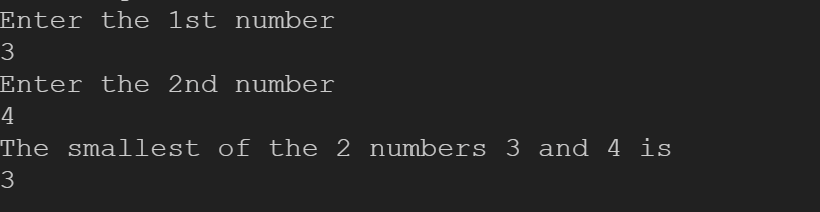
    b <- getLine

    let ax = read a :: Int

    let bx = read b :: Int

    putStrLn( "The smallest of the 2 numbers " ++ a ++ " and " ++ b ++ " is ")

    print(lrt ax bx)



3) max1 :: Int -> Int ->Int -> Int

max1 a b c

    |(a>=b && a>=c)=a

    |(b>=a && b>=c) = b

    |otherwise = c

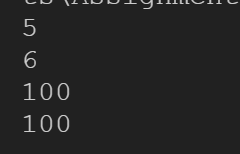
main = do

    a<- readLn

    b<- readLn

    c<- readLn

    print(max1 a b c)



4) oe::Int->[Char]

oe n

    | mod n 2==0 ="Even"

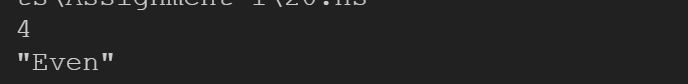
    | otherwise = "Odd"

main::IO()

main=do

    a <- readLn

    print(oe a)



5) pno :: Int -> [Char]

pno a

    |(a>0)="Positive"

    |(a<0) = "Negative"

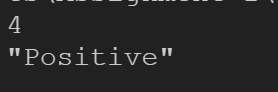
    |otherwise = "Zero"

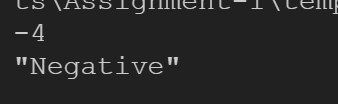
main = do

    a <- readLn

    print(pno a)







6) di::Int->[Char]

di n

    | mod n 7==0 ="Yes Divisible by 7"

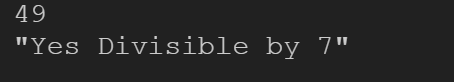
    | otherwise = "No"

main::IO()

main=do

    a <- readLn

    print(di a)



7) import Text.Read

isVowel :: Char -> IO()

isVowel c

    | c == 'a'

      || c == 'e' || c == 'i' || c == 'o' || c == 'u' || c == 'y'

    = print "Vowel"

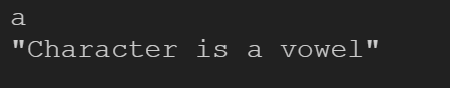
    | c == '1' || c == '2' || c == '3' || c == '4' || c == '5' || c == '6' || c == '7' || c == '8' || c == '9' || c == '0' = print "Number"

    | otherwise = print "Consonants"

main = do

    a <- getChar

    isVowel a



8) import Data.Char

uc :: String -> String

uc = map toUpper

lc :: String -> String

lc = map toLower

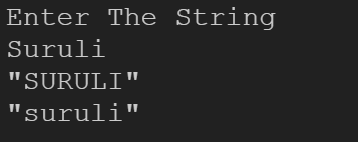
main = do

    putStrLn "Enter The String"

    a <- getLine

    print(uc(a))

    print(lc(a))



9) getDouble :: IO Int--user defined function

getDouble = readLn

sq :: Int -> IO()

sq s =

    if s>0 then print "Real Roots"

    else if s<0 then print "Imaginary Roots"

    else print "Equal Roots"

main = do

    putStrLn "Enter The Value of A"

    a <- readLn

    putStrLn "Enter The Value of B"

    b <- readLn

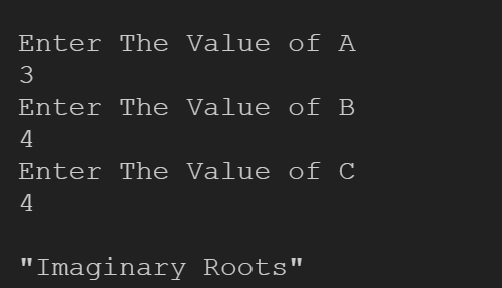
    putStrLn "Enter The Value of C"

    c <- readLn

    putStrLn ("")

    let s = b\*b - 4\*a\*c

    sq s



10) import Text.Read

isVowel :: Char -> IO()

isVowel c

    | c == '1' = print "One"

    | c == '2' = print "Two"

    | c == '3' = print "Three"

    | c == '4' = print "Four"

    | c == '5' = print "Five"

    | c == '6' = print "Six"

    | c == '7' = print "Seven"

    | c == '8' = print "Eight"

    | c == '9' = print "Nine"

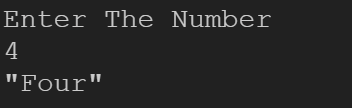
    | c == '0' = print "Zero"

main = do

    putStrLn "Enter The Number"

    num <- getChar

    isVowel num



11)

isVowel :: Double -> IO()

isVowel c

    | c >= 80 = print "A"

    | c >= 70 = print "B"

    | c >= 60 = print "C"

    | c >= 50 = print "D"

    | c < 40 = print "F"

main = do

    putStrLn "Enter Mark 1"

    n1 <- readLn

    putStrLn "Enter Mark 2"

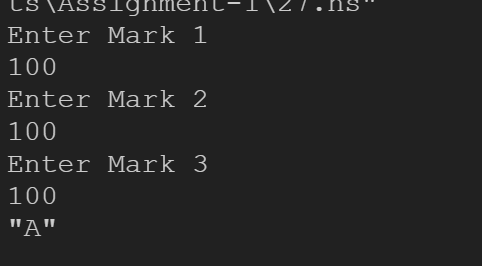
    n2 <- readLn

    putStrLn "Enter Mark 3"

    n3 <- readLn

    let n4 = (n1+n2+n3)/3

    isVowel n4



12) main = do

    putStrLn "enter number 1"

    num1 <- readLn

    putStrLn "enter number 2"

    num2 <- readLn

    print(num1+num2)

    print(num1-num2)

    print(num1\*num2)

    print(num1`div`num2)

    print(num1`rem`num2)

    print(num1`mod`num2)

