Suryanarayan.B

CB.EN.U4CSE19056

1) li :: ([Int]) -> Int

li (a) = do

 if a /= [] then

  head a\*li(drop 1 a)

 else 1

main = do

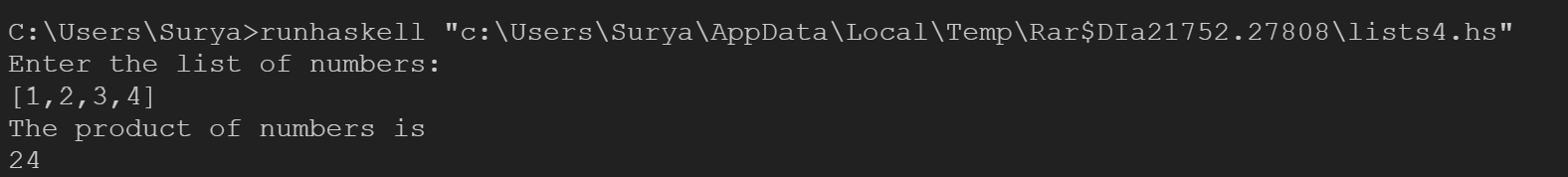
 putStrLn("Enter the list of numbers:")

 num <- getLine

 let a = read num :: [Int]

 putStrLn("The product of numbers is ")

 print(li a)



2) prime1 :: (Int,Int) -> Int

prime1 (a,1) = 1

prime1 (a,n) = (rem a n)\* (prime1(a ,n-1))

main = do

    putStrLn("Enter the number:")

    num <- getLine

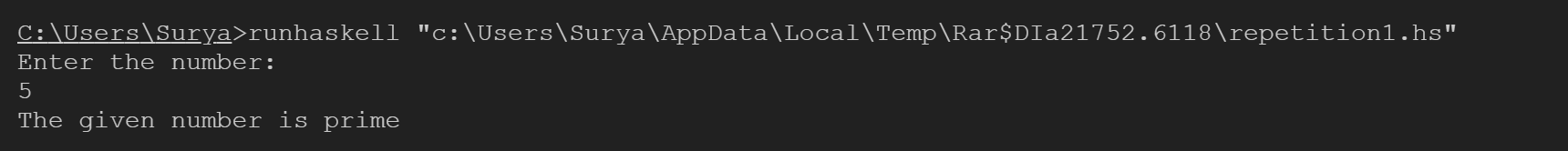
    let n = read num :: Int

    if (prime1(n,n-1) /= 0)

        then putStrLn("The given number is prime")

    else

        putStrLn("The given number is not a prime")



3) pal a = do

 let b = reverse a

 if b == a then putStrLn("The given number is palindrome")

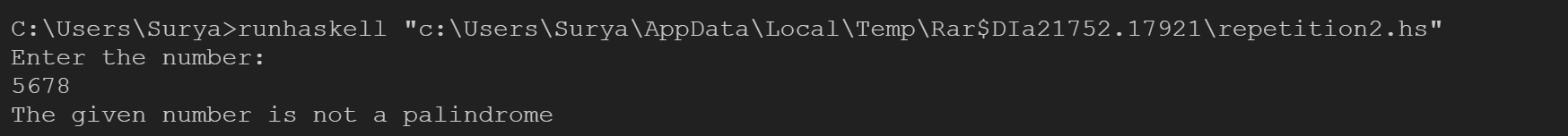
 else putStrLn("The given number is not a palindrome")

main=do

 putStrLn("Enter the number:")

 a <- getLine

 pal a



4) digit :: Int -> [Int]

digit = map (read . (:[])) . show

listsum:: [Int] -> Int

listsum a = do

    let b = head a

    let c = tail a

    if null c

    then b\*b\*b

    else

        b\*b\*b + listsum c

main=do

    putStrLn("Enter the number:")

    a<-getLine

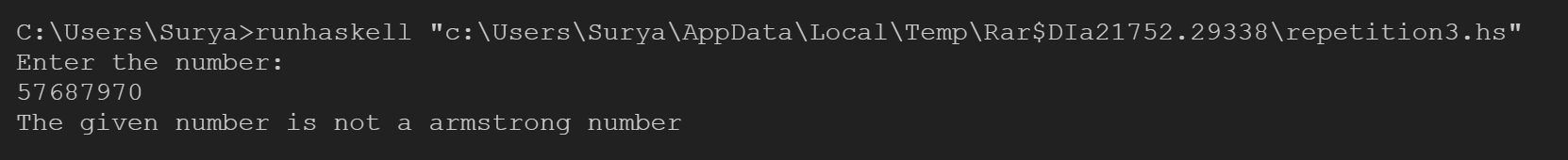
    let x = read a::Int

    let l= digit x

    if listsum l == x

    then putStrLn("The given number is armstrong number")

    else putStrLn("The given number is not a armstrong number")



5) evens a =[x | x <- [1..a], even x]

rsum a = sum (evens a)

main=do

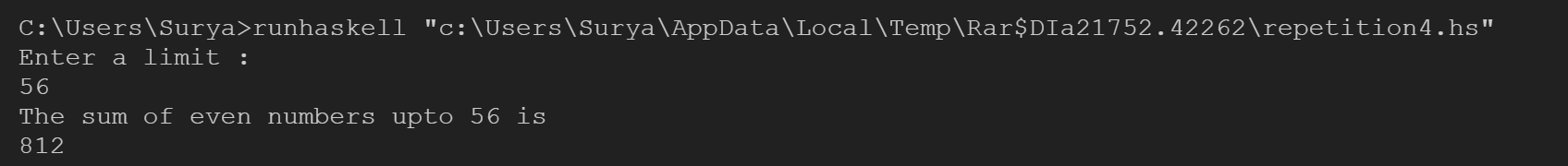
 putStrLn("Enter a limit :")

 num <- getLine

 let a = read num

 putStrLn("The sum of even numbers upto "++show(a)++" is " )

 print(rsum a)



6) odds a =[x | x <- [1..a], odd x]

rsum a = sum (odds a)

main=do

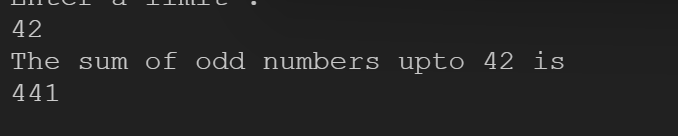
 putStrLn("Enter a limit :")

 num <- getLine

 let a = read num

 putStrLn("The sum of odd numbers upto "++show(a)++" is " )

 print(rsum a)



7) dig :: Int -> [Int]

dig = map (read . (:[])) . show

dec::[Int]->Int->Int

dec l n = do

    let bit = head l

    let rest = tail l

    if n==0

    then bit

    else do

        let nn=n-1

        ((2^n)\*bit )+ dec rest nn

main=do

 putStrLn ("Enter the number in binary:")

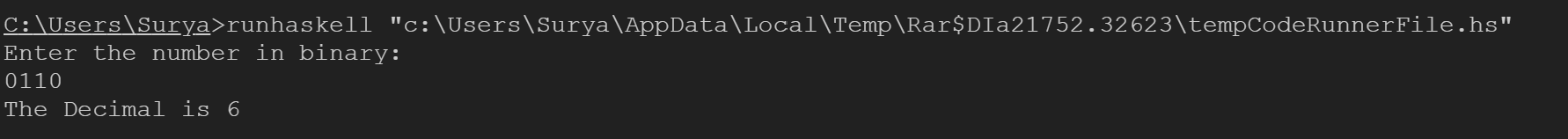
 a<-getLine

 let x=read a::Int

 let l= dig x

 let len=((length l) - 1)

 putStrLn ("The Decimal is "++show(dec l len))



8) covert::Int->String

fun::Int->String

fun a=do

    if mod a 2 ==1 then "1" else "0"

covert 0 = ""

covert a = fun a ++ covert (div a 2)

main =do

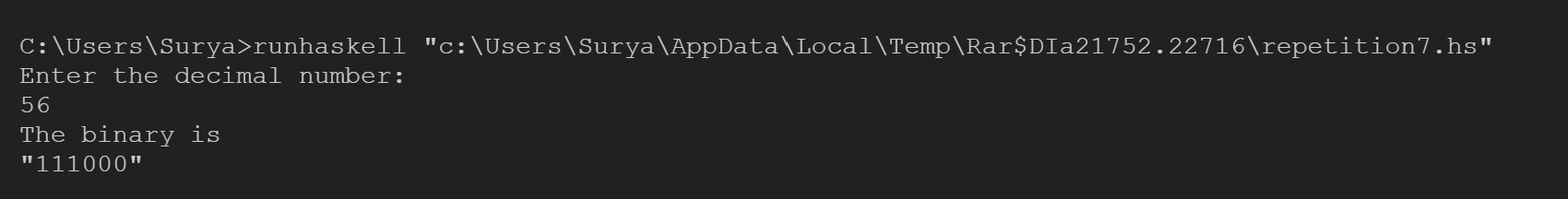
    putStrLn("Enter the decimal number:")

    x<-getLine

    let a=read x::Int

    putStrLn("The binary is")

    print(reverse (covert a))



9) fun::Char->String

fun '0'=" Zero "

fun '1'=" One "

fun '2'=" Two "

fun '3'=" Three "

fun '4'=" Four "

fun '5'=" Five "

fun '6'=" Six "

fun '7'=" Seven "

fun '8'=" Eight "

fun '9'=" Nine "

function::String->String

function "" = ""

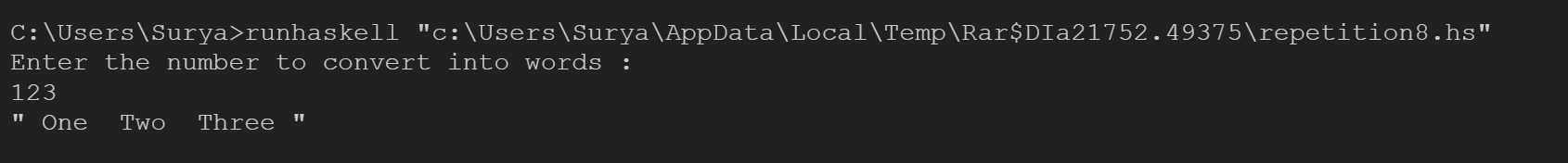
function (x:xs)=fun x ++ function xs

main =do

    putStrLn("Enter the number to convert into words :")

    x<-getLine

    print(function x)



10) primecheck::Int->Int->Bool

primecheck x n = do

    if x==1

    then True

    else if mod n x == 0

        then False

    else do

        let c=x-1

        primecheck c n

main=do

    putStrLn("Enter the first number :")

    a<-getLine

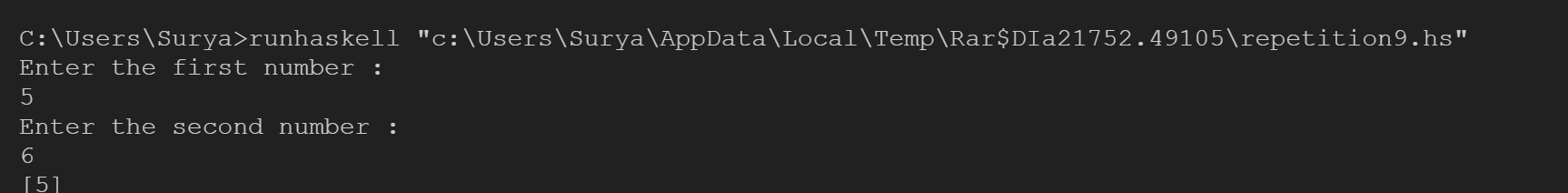
    putStrLn("Enter the second number :")

    b<-getLine

    let n = read a::Int

    let m = read b::Int

    print([x|x<-[n..m],primecheck (x-1) x])



11) summ::[Int]->Int

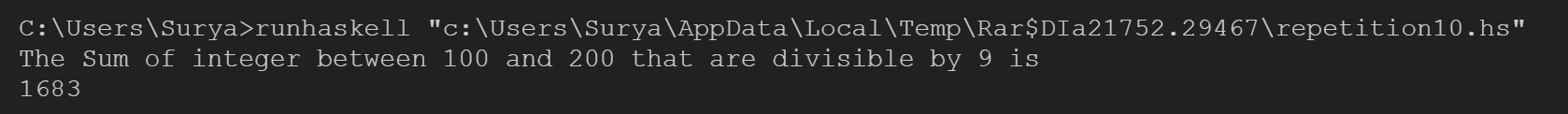
summ []=0

summ (x:xs)=x + summ xs

main =do

    putStrLn("The Sum of integer between 100 and 200 that are divisible by 9 is ")

    print(summ [x|x<-[100..200],mod x 9 ==0])



12)

main=do

 putStrLn ("Enter the Fahrenheit temperature : ")

 tem <- getLine

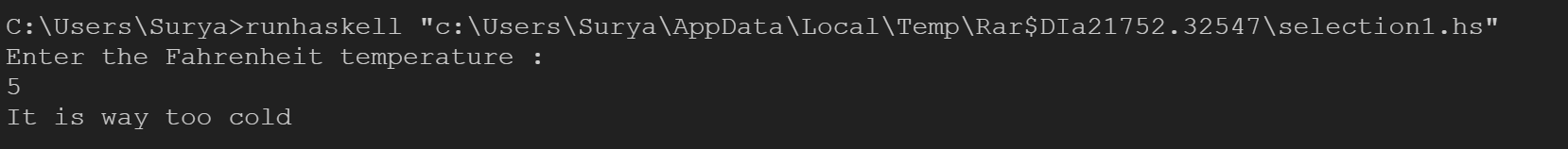
 let a = (read tem)

 if a >= 80 then putStrLn("Go play golf")

 else if a >= 70 && a <= 79

 then putStrLn("Put on a jacket")

 else putStrLn("It is way too cold")



13) main = do

 putStrLn("Enter your name :")

 name <- getLine

 putStrLn("Enter your Age : ")

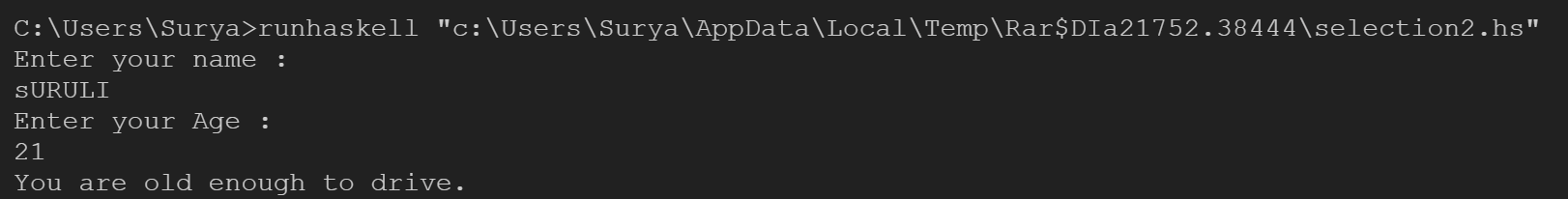
 age <- getLine

 let a = (read age)

 let b = 16 - a

 if a >= 16 then putStrLn("You are old enough to drive.")

 else putStrLn("You have to wait for "++ show (b)++" years.")



14) main = do

 putStrLn ("Enter the  hourly pay rate: ")

 inp <- getLine

 putStrLn ("Enter the number of hours worked for the week: ")

 inp <-getLine

 let a = (read inp) :: Float

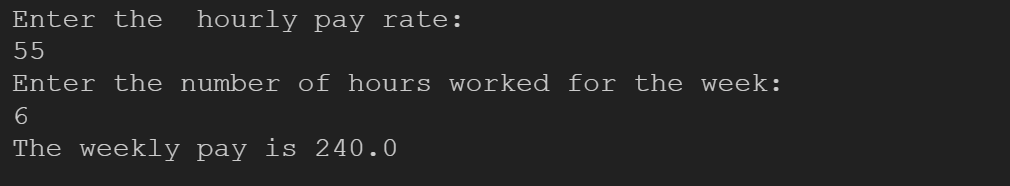
 let b = (read inp) :: Float

 let c = (40\*a)+((b-40)\*1.5\*a)

 let d = (40\*a)

 if a >= 40 then putStrLn ("The weekly pay is " ++ show (c))

 else putStrLn ("The weekly pay is " ++ show (d))



15)

main=do

 putStrLn("Enter the length of sides of traingle\nSide1:")

 num <- getLine

 let a= read num::Int

 putStrLn("Side2:")

 num <-getLine

 let b= read num::Int

 putStrLn("Side3:")

 num <-getLine

 let c= read num::Int

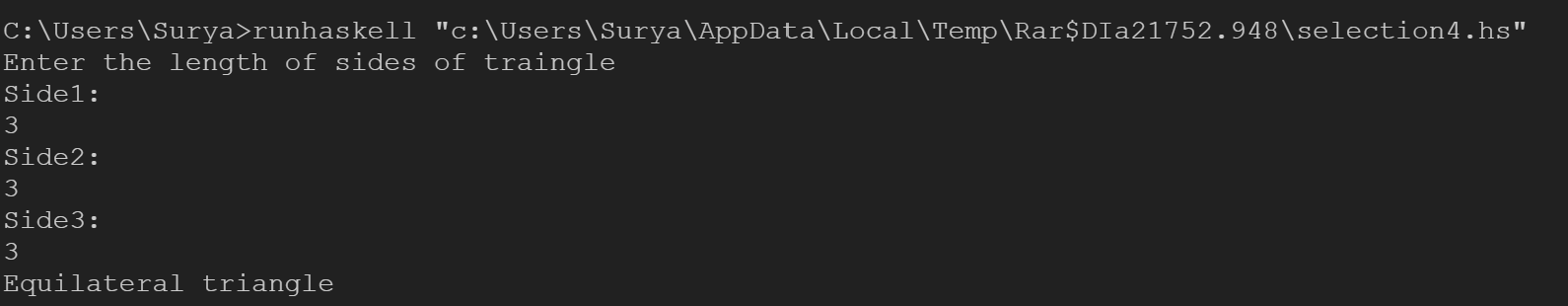
 if a==b && b==c

      then putStrLn("Equilateral triangle")

   else if a==b || b==c ||a==c

      then putStrLn("Isosceles triangle")

   else putStrLn("Scalene tringle")



16)

conv a=do

    let year=div a 365

    let week=div (mod a 365) 7

    let day=mod (mod a 365) 7

    putStrLn ("Years :" ++ show (year) ++"\nweeks: "++show(week)++"\nDays: "++show(day))

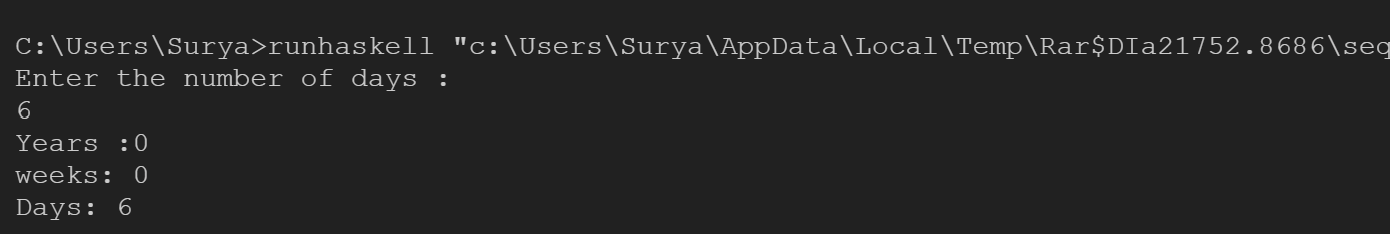
main = do

 putStrLn ("Enter the number of days : ")

 num <- getLine

 let a = (read num) :: Int

 conv(a)



17) conv a = do

    let b = ((mod a 10)\*10)+(div a 10)

    let c = abs (a-b)

    putStrLn ("Reverse is :" ++ show (b)++"\nDifference is :" ++ show (c))

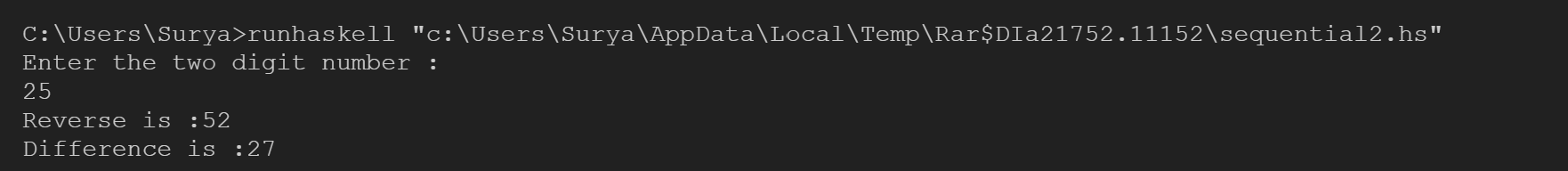
main = do

 putStrLn ("Enter the two digit number : ")

 num <- getLine

 let a = (read num)

 conv(a)



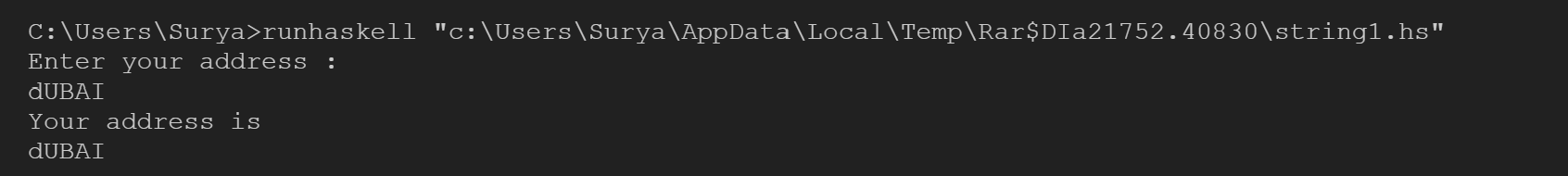
18) main = do

 putStrLn("Enter your address : ")

 a <- getLine

 putStrLn("Your address is")

 putStrLn(a)



19) palindrome a = do

 let b = reverse a

 if b == a then putStrLn("The given word is palindrome")

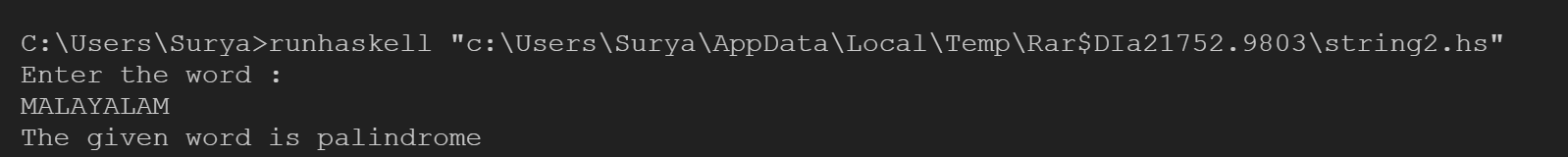
 else putStrLn("The given word is not a palindrome")

main=do

 putStrLn("Enter the word :")

 a <- getLine

 palindrome a



20) countv::[Char]->(Int, Int)

countv txt = countc txt 0 0

countc::[Char]->Int->Int->(Int, Int)

countc [] con vow = (con, vow)

countc (c:r) con vow

    |c=='a' || c=='e' || c=='i' || c=='o' || c=='u' = countc r con (vow + 1)

    |otherwise = countc r (con + 1) vow

main = do

 putStrLn("Enter the text :")

 str <- getLine

 print(countv(str))

