

# CIS 6 :: Lab 05 - Defining and Using Functions

Student Name: Naveed Yeganegi

## Task 1: Definitions & Concepts

**Instructions:** Answer the questions below.

1. What is a function in programming?  
=> A subprogram, or smaller program within a program
2. How does a function return a value?  
=> Name a block of code and execute this block by calling its name
3. Parameters are an important concept in defining functions.
  - (a) What is the purpose of parameters? => It allows you to supply the function with data when you call it. Functions can return completely different values when given different parameters.
  - (b) What is the difference between a formal parameter and an actual parameter(argument)? => Formal parameters are identifiers used when defining the function and actual parameters are the variables used in parentheses when calling the function
  - (c) In what ways are parameters similar to and different from ordinary variables? => A variable is an identifier that refers to a value. Similarly, parameters refer to a value, but are arguments used as input to a function.

4 Discussion: In your own words, describe the two (or three) motivations for using functions in your programs.

=> Having functions whenever possible makes larger programs much more organized. Not only do functions optimize your workload by making you repeat yourself less, but also make the program much easier to understand, change, and maintain.

## Task 2: Understanding Functions

1. Consider this very simple function:

```
def cube(x):
```

```
    answer = x * x * x
```

```
return answer
```

(a) What does this function do? => **Gives the user the cube of x**

(b) Show how a program could use this function to print the value of  $y^3$ , assuming y is already defined. => **cube(y)**

(c) Here is a fragment of a program that uses this function:

```
answer = 3
```

```
result = cube(answer)
```

```
print(answer, result)
```

The output from this fragment is 3 27. Explain why the output is not 27 27, even though the cube seems to change the value of "answer" to 27. => **Variables used in a function are local to that function, even with the same name. Functions can only see variables passed as parameters, but do not have access to the variable itself.**

### Task 3: Programming Exercises

Instructions: Use Python IDLE to execute below codes. Attach Snipping photos of your code and execution.

## Chapter 6 - Programming Exercises (Page 206-208).

### Exercise 1: Old MacDonald Song

(Lyrics: <http://www.dltk-teach.com/rhymes/macdonald/mlyrics.htm>):

```
oldmacdonald.py - C:/Users/Admin/AppData/Local/Programs/Python/Python39/oldmacdonald.py (3.9.1)
File Edit Format Run Options Window Help

def chorus():
    print("Old MacDonald had a farm, E-I-E-I-O!")

def animal(kind, sound):
    chorus()
    print("And on that farm he had a ", kind, " E-I-E-I-O!")
    print("With a ", sound, ", ", sound, " here and a ", sound, ", ", sound, " there.")
    print("Here a ", sound, ", there a ", sound, ", everywhere a ", sound, ", ", sound, ".")
    chorus()

def main():
    animal("cow", "moo")
    print()
    animal("pig", "oink")
    print()
    animal("horse", "neigh")
    print()
    animal("chicken", "cluck")
    print()
    animal("duck", "quack")

main()

IDLE Shell 3.9.1
File Edit Shell Debug Options Window Help

= RESTART: C:/Users/Admin/AppData/Local/Programs/Python/Python39/oldmacdonald.py
Old MacDonald had a farm, E-I-E-I-O!
And on that farm he had a cow E-I-E-I-O!
With a moo , moo here and a moo , moo there.
Here a moo , there a moo , everywhere a moo , moo .
Old MacDonald had a farm, E-I-E-I-O!

Old MacDonald had a farm, E-I-E-I-O!
And on that farm he had a pig E-I-E-I-O!
With a oink , oink here and a oink , oink there.
Here a oink , there a oink , everywhere a oink , oink .
Old MacDonald had a farm, E-I-E-I-O!

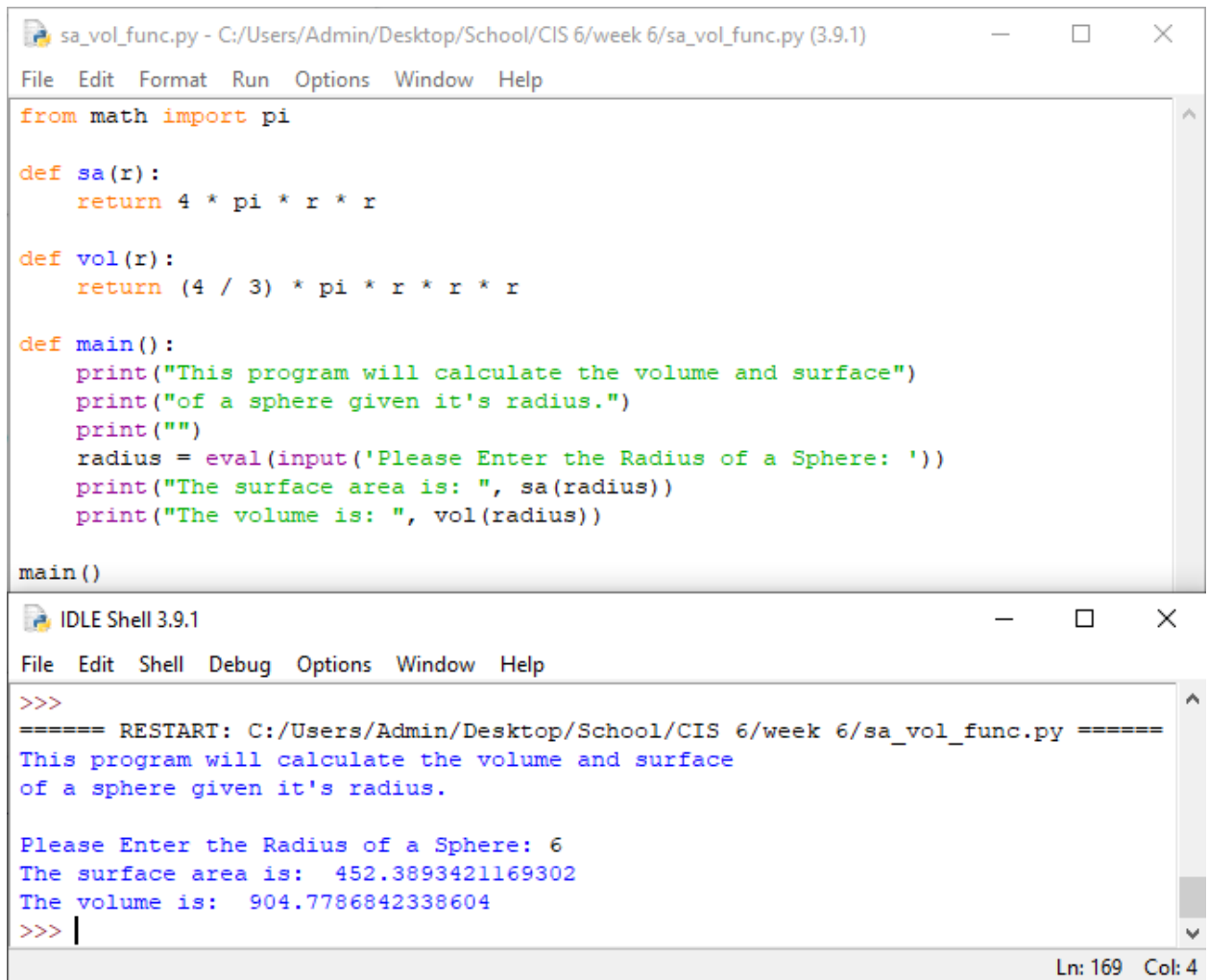
Old MacDonald had a farm, E-I-E-I-O!
And on that farm he had a horse E-I-E-I-O!
With a neigh , neigh here and a neigh , neigh there.
Here a neigh , there a neigh , everywhere a neigh , neigh .
Old MacDonald had a farm, E-I-E-I-O!

Old MacDonald had a farm, E-I-E-I-O!
And on that farm he had a chicken E-I-E-I-O!
With a cluck , cluck here and a cluck , cluck there.
Here a cluck , there a cluck , everywhere a cluck , cluck .
Old MacDonald had a farm, E-I-E-I-O!

Old MacDonald had a farm, E-I-E-I-O!
And on that farm he had a duck E-I-E-I-O!
With a quack , quack here and a quack , quack there.
Here a quack , there a quack , everywhere a quack , quack .
Old MacDonald had a farm, E-I-E-I-O!
>>>

Ln: 129 Col: 4
```

### Exercise 3: Surface area and volume of a sphere function



The image shows two windows from the Python IDLE 3.9.1 environment. The top window, titled 'sa\_vol\_func.py - C:/Users/Admin/Desktop/School/CIS 6/week 6/sa\_vol\_func.py (3.9.1)', contains the following Python code:

```
from math import pi

def sa(r):
    return 4 * pi * r * r

def vol(r):
    return (4 / 3) * pi * r * r * r

def main():
    print("This program will calculate the volume and surface")
    print("of a sphere given it's radius.")
    print("")
    radius = eval(input('Please Enter the Radius of a Sphere: '))
    print("The surface area is: ", sa(radius))
    print("The volume is: ", vol(radius))

main()
```

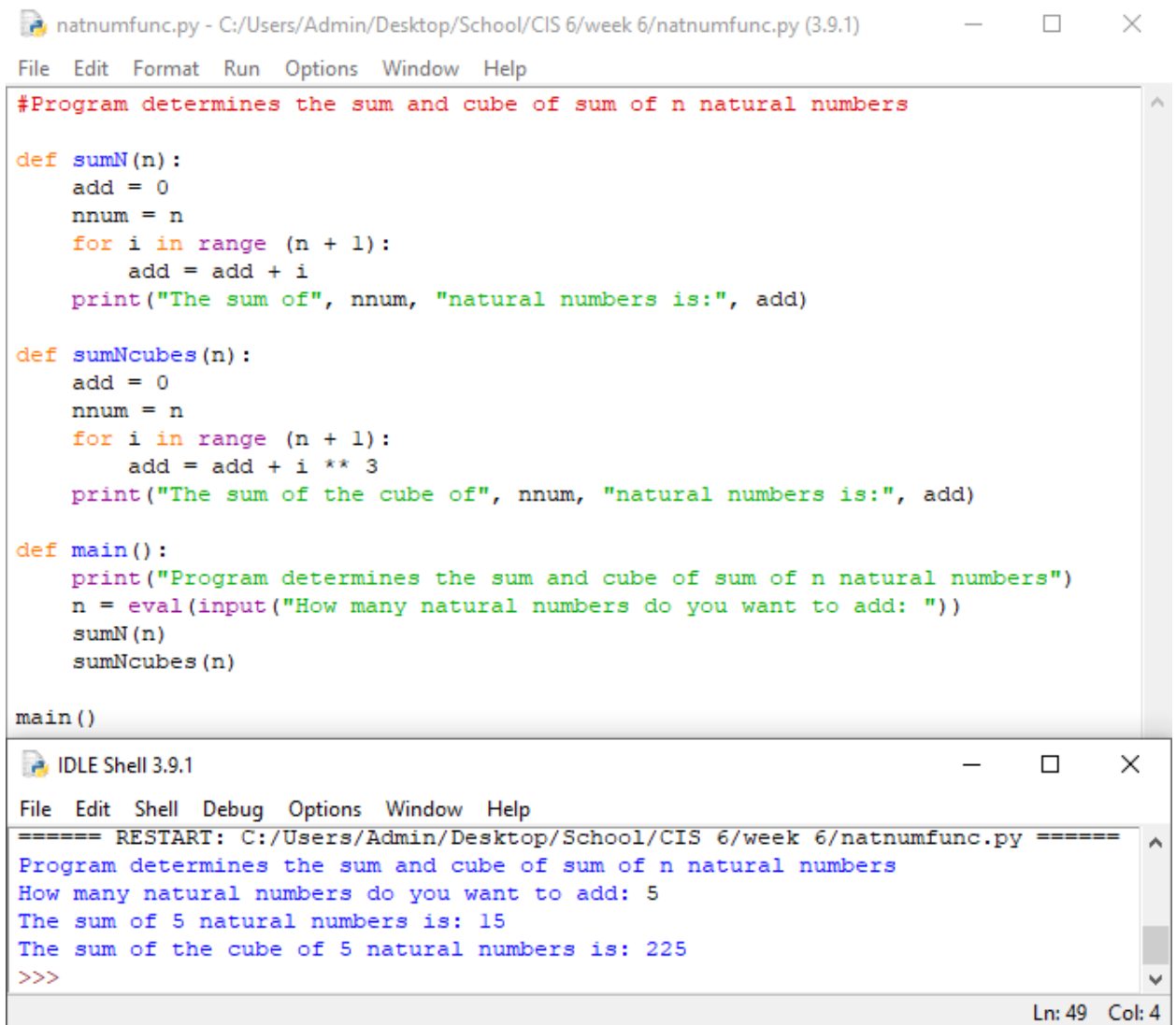
The bottom window, titled 'IDLE Shell 3.9.1', shows the output of running the script. It starts with a restart message, followed by the program's introductory text. It then prompts for the radius, which is entered as 6. The program calculates and displays the surface area as 452.3893421169302 and the volume as 904.7786842338604.

```
>>>
===== RESTART: C:/Users/Admin/Desktop/School/CIS 6/week 6/sa_vol_func.py =====
This program will calculate the volume and surface
of a sphere given it's radius.

Please Enter the Radius of a Sphere: 6
The surface area is:  452.3893421169302
The volume is:  904.7786842338604
>>> |
```

Ln: 169 Col: 4

#### Exercise 4: Sum of n numbers function



The image shows a screenshot of a Python IDE with two windows. The top window, titled 'natnumfunc.py - C:/Users/Admin/Desktop/School/CIS 6/week 6/natnumfunc.py (3.9.1)', contains the following Python code:

```
#Program determines the sum and cube of sum of n natural numbers

def sumN(n):
    add = 0
    nnum = n
    for i in range (n + 1):
        add = add + i
    print("The sum of", nnum, "natural numbers is:", add)

def sumNcubes(n):
    add = 0
    nnum = n
    for i in range (n + 1):
        add = add + i ** 3
    print("The sum of the cube of", nnum, "natural numbers is:", add)

def main():
    print("Program determines the sum and cube of sum of n natural numbers")
    n = eval(input("How many natural numbers do you want to add: "))
    sumN(n)
    sumNcubes(n)

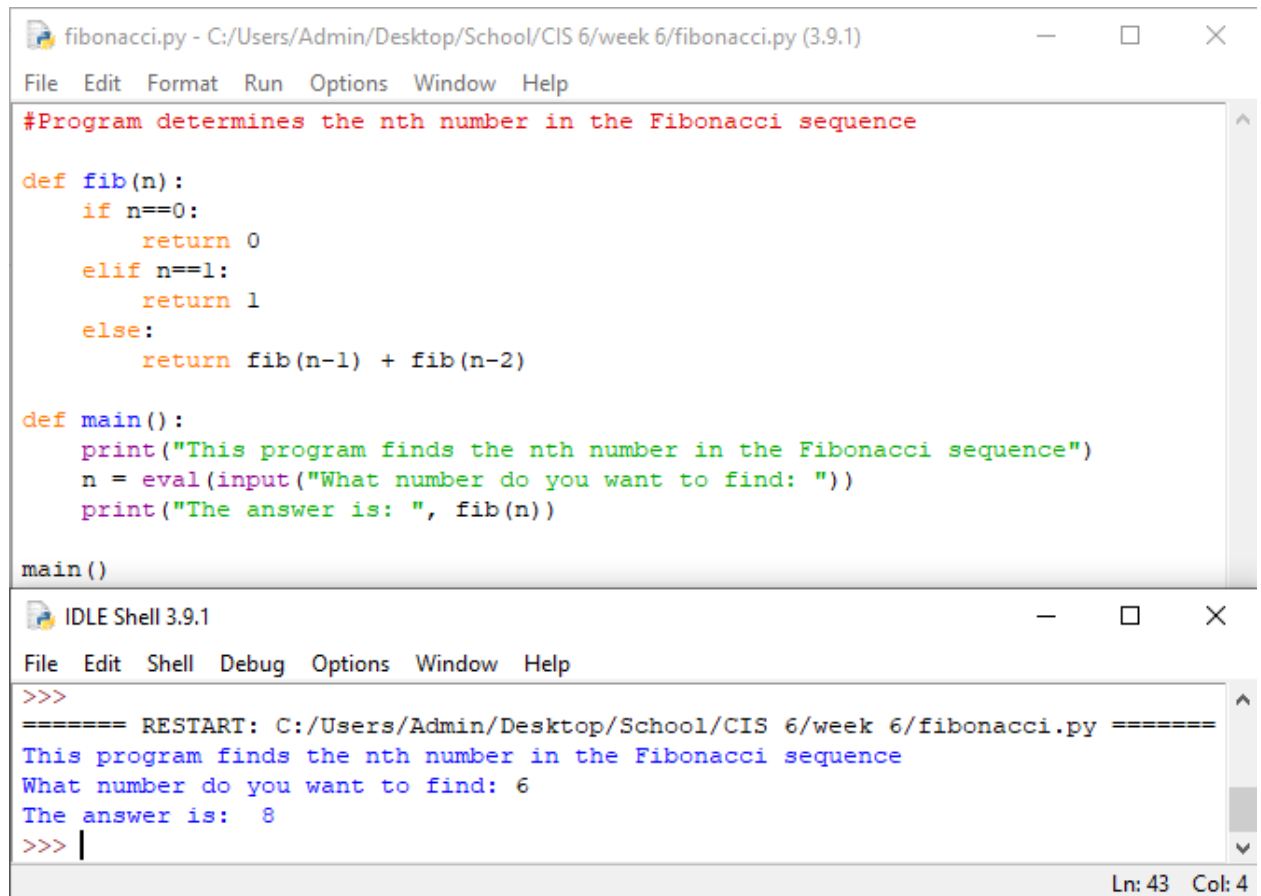
main()
```

The bottom window, titled 'IDLE Shell 3.9.1', shows the output of the program after a restart:

```
===== RESTART: C:/Users/Admin/Desktop/School/CIS 6/week 6/natnumfunc.py =====
Program determines the sum and cube of sum of n natural numbers
How many natural numbers do you want to add: 5
The sum of 5 natural numbers is: 15
The sum of the cube of 5 natural numbers is: 225
>>>
```

The status bar at the bottom right of the shell window indicates 'Ln: 49 Col: 4'.

Exercise 7: Function for the nth Fibonacci number



The image shows a screenshot of a Python IDE with two windows. The top window, titled 'fibonacci.py - C:/Users/Admin/Desktop/School/CIS 6/week 6/fibonacci.py (3.9.1)', contains the following Python code:

```
#Program determines the nth number in the Fibonacci sequence

def fib(n):
    if n==0:
        return 0
    elif n==1:
        return 1
    else:
        return fib(n-1) + fib(n-2)

def main():
    print("This program finds the nth number in the Fibonacci sequence")
    n = eval(input("What number do you want to find: "))
    print("The answer is: ", fib(n))

main()
```

The bottom window, titled 'IDLE Shell 3.9.1', shows the execution of the program. It displays the following output:

```
>>>
===== RESTART: C:/Users/Admin/Desktop/School/CIS 6/week 6/fibonacci.py =====
This program finds the nth number in the Fibonacci sequence
What number do you want to find: 6
The answer is:  8
>>> |
```

The status bar at the bottom right of the IDE shows 'Ln: 43 Col: 4'.

## Exercise 9: Function to return letter grade

```
leterscorefunc.py - C:/Users/Admin/Desktop/School/CIS 6/week 6/leterscorefunc.py (3.9.1)
File Edit Format Run Options Window Help

#This program converts an exam grade from a number to the letter grade"

def letter(score):
    if score >=90:
        return 'A'
    elif score >=80:
        return 'B'
    elif score >=70:
        return 'C'
    elif score >=60:
        return 'D'
    elif score <60:
        return 'F'

def main():
    print("This program converts an exam grade from a number to the letter grade")
    n = eval(input("Enter the exam score on a 0-100 scale: "))
    print("Your letter grade is:", letter(n))

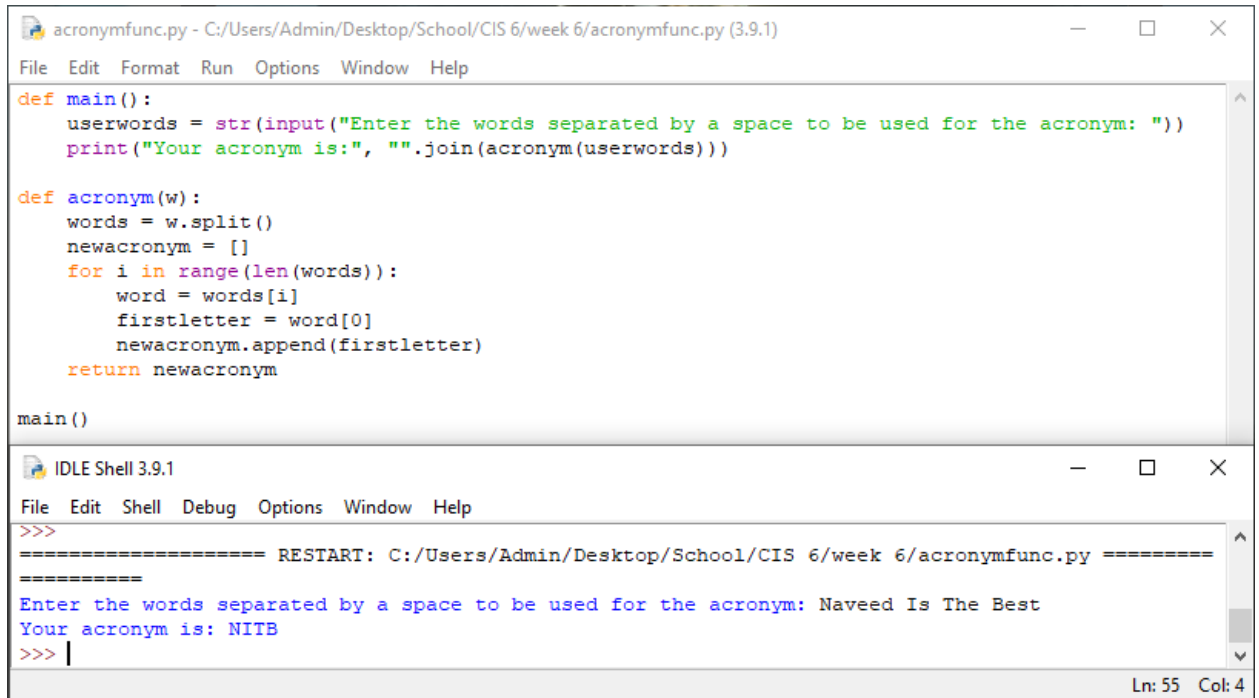
main()

IDLE Shell 3.9.1
File Edit Shell Debug Options Window Help

==== RESTART: C:/Users/Admin/Desktop/School/CIS 6/week 6/leterscorefunc.py ====
This program converts an exam grade from a number to the letter grade
Enter the exam score on a 0-100 scale: 69
Your letter grade is: D
>>>
===== RESTART: C:/Users/Admin/Desktop/School/CIS 6/week 6/leterscorefunc.py =====
This program converts an exam grade from a number to the letter grade
Enter the exam score on a 0-100 scale: 95
Your letter grade is: A
>>> |

Ln: 61 Col: 4
```

## Exercise 10: Function to return acronym



The screenshot displays two windows from the IDLE 3.9.1 Python IDE. The top window, titled 'acronymfunc.py - C:/Users/Admin/Desktop/School/CIS 6/week 6/acronymfunc.py (3.9.1)', contains the following Python code:

```
def main():
    userwords = str(input("Enter the words separated by a space to be used for the acronym: "))
    print("Your acronym is:", "".join(acronym(userwords)))

def acronym(w):
    words = w.split()
    newacronym = []
    for i in range(len(words)):
        word = words[i]
        firstletter = word[0]
        newacronym.append(firstletter)
    return newacronym

main()
```

The bottom window, titled 'IDLE Shell 3.9.1', shows the execution of the script. It displays a restart message, followed by the input 'Naveed Is The Best' and the output 'Your acronym is: NITB'.

```
>>>
===== RESTART: C:/Users/Admin/Desktop/School/CIS 6/week 6/acronymfunc.py =====
=====
Enter the words separated by a space to be used for the acronym: Naveed Is The Best
Your acronym is: NITB
>>> |
```

The status bar at the bottom right of the shell window indicates 'Ln: 55 Col: 4'.

#### Task 4 - Creative Program Development:

**Continue from Lab 2:** Modify your tip calculator program so that it uses a function that takes the bill amount, tax and tip, and outputs the final bill. Name the function **getBill()** and call the function in the main function (you may define more than one function; one for tip, one for tax and one for finalBill).



bill.py - C:/Users/Admin/Desktop/School/CIS 6/week 6/bill.py (3.9.1)

File Edit Format Run Options Window Help

# A program to calculate the total bill after taxes and tip  
  
def getBill(bill,tax,tip):  
 total = bill + (bill \* tax) + (bill \* tip)  
 print("Your final bill is:", total)  
  
def getTax(bill,tax):  
 taxes = bill \* tax  
 print("Taxes paid:", taxes)  
  
def getTip(bill,tip):  
 tips = bill \* tip  
 print("Amount tipped:", tips)  
  
def main() :  
 print("This program calculates the total bill after taxes")  
 bill = eval(input("Enter the initial bill amount: "))  
 tax = eval(input("Enter the sales tax % as a decimal: "))  
 tip = eval(input("Enter the tip % as a decimal: "))  
 print()  
 getBill(bill,tax,tip)  
 getTax(bill,tax)  
 getTip(bill,tip)  
  
main()

IDLE Shell 3.9.1

File Edit Shell Debug Options Window Help

>>>  
===== RESTART: C:\Users\Admin\Desktop\School\CIS 6\week 3\bill.py =====  
This program calculates the total bill after taxes  
Enter the initial bill amount: 50  
Enter the sales tax % as a decimal: .10  
Enter the tip % as a decimal: .15  
  
Your final bill is: 62.5  
Taxes paid: 5.0  
Amount tipped: 7.5  
>>>

Ln: 21 Col: 18