```python
# print("Hello World")


# The value of a literal constant can be used directly in programs
# Literal derived form term literally
# For example - 7, 3.9, 'X' , "Hello" these are all literal constanst these values cannot be changed
#
#Numbers
# We can use 4 types of numbers in python programs
# 1. Integers
# 2. Long Integers
# 3. Floating point
# 4. Complex Numbers
#  To represent a long Integers we need to put suffix L or l
# 53248932872L - Long Integers
# Numbers like 3.23 or 89.5E-2 are termed as floating point numbers
# Capital E notation indicates power of 10
# so 91.5E10 means 91.5*10^10 remember
# that comma is never used for numeric literals
# For Example - 1,234,567 is Not valid..! in python
# Note that there is no limit in the size of Integers though floating point numbers has limit
# Floating point range 10^-308 to 10^308
# so 5.02348*106 means 5.02348E6/5.02348e6
# Floating point Numbers can be efficiently used in python with following limitations
# 1. Arithmetic Overflow Problem - When a large number of floating point numbers are multiplied together arithmetic overflow occurs & the result is represented as 'inf'

#    Example - 2.34e200 * 4.3e200 so it will result 'inf'


# 2. Arithmetic Underflow Problem - If 2 floating point numbers are divided then the result will become very small in magnitude and results in '-inf'


# 3. Loss of Precision Problem - When we divide 1/3 the result is 0.333333333 , Since any floating point number has limited precision and range the number will be

#    rounded off after 2 decimal number so result is '0.34' that is the actual value is approximated

#

# Example - We use built-in format() function

print(float (16/float(3)))

#>>> 5.333333333333333

# Using format() function

print(format(float (16/float(3)),'.2f'))

# >>> 5.33 (viva)

# for very Large or Small numbers we can use small 'e' as format specifier

# Example  -

print(format(3**50,'.5e'))

# >>> 7.17898e+23

## ** = To the power in python

# using format we can also use comma with-in the digits

print(format(1234567,','))

# >>> 1,234,567
# Different Numerical Operations
# >>> 10 + 7
# >>> 17




#15/0
#Traceback (most recent call last):
#  File "<pyshell#34>", line 1, in <module>
#    15/0
#ZeroDivisionError: division by zero
#Exception has been handled by python Runtime-enviroment, division by
zero occurs.

# 15 / 3.0
# 5.0
# python automatically converts int to float

# Quotient and Remainders

# Quotient
# 78 // 5
# 15
# Remainders
# 78 % 5
# 3

# 152.78 // 3.0
# 50.0

# 152.78 % 3.0
# 2.780000000000001

# Strings:- '',""
# Example - A string is a group of characters
# we can use either of the 3 forms
#
# 'Hello' , "Hello" both are exactly same and valid..!
# Using triple quote ''' : we can use multi-Line statements
# using triple quotes so ,
# '''Good Morning Everyone
#Welcome to the World of Python
#ThankYou'''
#'Good Morning Everyone\nWelcome to the World of Python\nThankYou'

# We should keep-it in mind that strings are immutable
# (Cannot be Changed)
#
# When more than 1 strings are placed one of the another
# they are automatically concatenated

print('Michael'' Gomes')
>>> Michael Gomes

# UNICODE Character Codes
# unicode character are represented using a prefix 'u' or 'U'
# For Example - u 'sample of unicode string'
```