```
# Variables
# A variable is a container to hold the data.
# A variable is a name given to a memory location.
# They are used for reuse of the same data.
# variables are case-sensitive.
# variables are created the moment a value is assigned to them
# Syntax:
# <variable_name> = <value>
# <variable_name> = <expression>

# PEP guidelines for declaring variables:
# 1. variable name should either start with an alphabet or an underscore.
# 2. variable name can contain alphabets, numbers and an underscore.
# 3. Special characters are not allowed inside variable names except underscore.
# 4. Constant variables are declared in Capital letters.
# 5. variable names with multiple words are separated by an underscore.
# 6. Variable names cannot contain spaces.
# 7. We cannot use reserved words (if, else, for, while, etc..) as variable names.



# value = 3.14
# print(value + 1)
# print("Hello World!!!")
# print("Hello World!!!")
# print("Hello World!!!")
# print("Hello World!!!")
# print(value + 2)
# print(value * 5)


a = 10
print(a + 1)
print(a + 20)
print(a + 100)

abc = 40
print(abc)
_value = 50
print(_value)

value1 = 100
print(value1)
value2 = 200
print(value2)

students_count = 100
studentscount = 100
noofstudents = 100

# var$ = 100
# va@riable = 100
# var-able = 10
var = 100
VAR = 200
Var = 300
vAr = 400
VaR = 500
```

```python
print(var) # 100
print(VAR) # 200
print(VaR) # 500
print(vAr) # 400
print(Var) # 300
# print(vAR) # Error

CONSTANT_VAR = 1000

# type = 10
# print(type)

# print(type(10)) # 10(10)


a = 100
print(a) # 100
print("a")

print("kzsfhbbsekrvubn seli slreifuserliu")
# print(var_name)



# Literals
# Numeric Literals
# Integer - 19, -475, 0, 343, -84, -837, 654
# Floating Numbers - 34.65, 3943.45, -28372.434, 3.14
# String Literals - "Text", 'String', 'Python', "Program", "123", '456'
# Complex Literals
# a + bj => 3 + 4j,  34.45 + 45.65j
# Boolean Values
# True, False

print(type(19))
print(type(-34))
print(type(0))

print(type(8.545))
print(type(-74.54))

print(type("Text"))
print(type('String'))
print(type('123'))

print(type(10 + 20j))
print(type(1.4 + 40.64j))

print(type(True))
print(type(False))

# Data Types:
# int - +ve, -ve, 0
# Ex: +2, -65, 0, 89
#
# float - value with decimal point
# Ex: 0.1, 3.14, 0.9, -54.6
#
# boolean: has only 2 states
# True, False
#
# string: Any value enclosed in '' or ""
# 'A', "Python", 'Program'
#
# complex - any value of the form a + bj
# Ex:
# 4 + 30j
#
# list
# tuple
# set
# dictionary

a = 5
print(type(a)) # <class 'int'>
a = 6.8
print(type(a)) # <class 'float'>


# Type Casting:
"""
Changing the variable from one data type to another.
There are 2 types of type casting.
1. Implicit Type Casting
   In Implicity Type Casting, Python Interpreter
   converts the poor / low data type to a rich / high level data type
   without the intervention of the programmer.

   Reason:
   Python Interpreter will convert to avoid data loss.

Ex:
"""
a = 10
b = 5.5
c = a + b
print(c)
print(type(c))

a = 15
b = 4
c = a / b
print(c) # 3.75
print(type(c)) # <class 'float'>


# 2. Explicit Type Casting
"""
Programmer will manually convert the data type.
Functions for converting to data types:
Integer => int()
Float => float()
String => str()
Boolean => bool()
List => list()
Tuple => tuple()
Set => set()
Dictionary => dict()
```

```
"""
"""
a = int(input("Enter a Number: "))
# a = int(a)
print(a)
print(type(a))


a = float(input("Enter a Number: "))
# a = float(a)
print(a)
print(type(a))
"""


# Printing with variables:
a = 10
print(a) #
print('a') # a
# print(name)

"""
a = int(input("Enter a Number: "))
b = int(input("Enter a Number: "))
# Output: a value is <value of a> and b value is <value of b>
print("a value is ", a, "and b value is ", b)
print("a value is " + str(a) + "and b value is " + str(b))
print("a value is %d and b value is %d" % (a, b))
print("a value is {} and b value is {}".format(a, b))
print("a value is {value1} and b value is {value2}".format(value1=a, value2=b))
print(f"a value is {a} and b value is {b}")


a = int(input("Enter a Number: "))
b = int(input("Enter a Number: "))
# Output: a value is <value of a> and b value is <value of b> and c value is <value
of c> and d value is <value of d>
print("a value is ", a, "and b value is ", b, ", a value is ", a, ", b value is ",
a)
print("a value is " + str(a) + "and b value is " + str(b) + " and a value is " +
str(a) + " and b value is " + str(b))
print("a value is %d and b value is %d, c value is %d and d value is %d" % (a, b,
a, b))
print("a value is {} and b value is {} and a value is {} and a value is
{}".format(a, b, a, b))
print("a value is {value1} and b value is {value2} and a value is {value1} and b
value is {value2}".format(value1=a, value2=b))
print(f"a value is {a} and b value is {b} and a value is {a} and b value is {b}")
"""


# Output: "Hello World!!!"
print("Hello World!!!")
print('"Hello World!!!"')
# I am Learning "Python" Program
# print("I am Learning "Python" Program")
print('I am Learning "Python" Program')

# Output: I am Learning 'Python' Program
# print('I am Learning 'Python' Program')
print("I am Learning 'Python' Program")

# Output: I'm Learning "Python" Program
# print('I'm Learning "Python" Program')
# print("I'm Learning "Python" Program")

# Escape Sequences
"""
\n => New Line
\t => Tab Space
\b => Backspace
\f => form feed
\r => carriage return
\' => '
\" => "
\\ => \
"""

print("I'm Learning \"Python\" Program")
print('I\'m Learning "Python" Program')
print("Hello\nWorld!!!")
print("Hello\tWorld!!!")
print("Hello\bWorld!!!")
# print("Hello \World")
# Output: Hello \n World!!!
print("Hello \\n World!!!")
print(r"Hello \n \t World!!!")
```