LAB # 02

VARIABLES AND OPERATORS

OBJECTIVE

Implement different type of data types, variables and operators used in Python.

THEORY

Variable:

- Variables are nothing but reserved memory locations to store values.
- This means that when you create a variable you reserve some space in memory.
- variables are used to store data that can be referenced and manipulated during program execution.
- A variable is essentially a name that is assigned to a value.

Rules for Constructing Variable Names:

- A variable can have a short name (like x and y) or a more descriptive name (age, carname, total_volume).

Rules for Python variables:

- A variable name must start with a letter or the underscore character.
- A variable name cannot start with a number.
- A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _).
- Variable names are case-sensitive (age, Age and AGE are three different variables).
- Avoid using Python keywords (e.g., if, else, for) as variable names.
- Variables do not need to be declared with any particular type and can even change type after they have been set.

Example

```
# Variable 'x' stores the integer value 5
x= 5

# Variable 'name' stores the string "Touqeer"
y= "Touqeer"

print(x)
print (y)
```

LAB # 02

Output:

```
>>> %Run lab_02.py
5
Touqeer
```

Example

```
# Variable 'x' stores the integer value 4
x= 4

# Variable 'name' stores the string "Rafay"
x= "Rafay"

print(x)
```

Output:

```
>>> %Run lab_02.py

Rafay
```

Assigning Values to Multiple Variables

- Variables in Python are assigned values using the = operator.
- Python allows you to assign values to multiple variables in one line.
- Python variables are dynamically typed, meaning the same variable can hold different types of values during execution.
- We can assign different values to multiple variables simultaneously, making the code concise and easier to read.

Example:

```
x = 10
y = 5.10
z = "Hello!"

print(x,'\n', y,'\n', z)
```

Output:

```
>>> %Run lab_02.py
10
5.1
Hello!
```

Example:

```
x, y, z = "Orange", "Banana", "Cherry"
print(x)
print(y)
print(z)
```

Output:

```
>>> %Run lab_02.py
Orange
Banana
```

```
Cherry
```

Example

```
x = 9
x = "Now a string not number"
```

Output:

```
>>> %Run lab_02.py
Now a string not number
```

To combine both text and a variable, Python uses the + character

Example:

```
x= "awesome"
print("Python is " , x)
```

Output:

```
>>> %Run lab_02.py
Python is awesome
```

Python Keywords

- Keywords are the words whose meaning have already been explained to the Python compiler.
- The keywords cannot be used as variable names,function name or any identifier because if we do so we are trying to assign a new meaning to the keyword, which is not allowed by the computer.
- Keywords are also called '**Reserved words**'.

Some keywords are as follows:

false	class	finally	is	return	none	continue	for	try	Break
true	def	For	from	while	and	del	not	with	As
elif	if	Or	except	in	raise	yield			

Valid Example

```
age = 26
_flower = "Lilly"
total_number = 82
```

Invalid Example

```
1error = "Error404"  # Starts with a digit
class = 09       # 'class' is a reserved keyword
user-name = "Aeel"  # Contains a hyphen
```

LAB # 02

Data Types

- Data types specify how we enter data into our programs.
- What type of data we enter.
- Python Data Types are used to define the type of a variable.
- You can get the data type of any object by using the **"type( )"** function.

Python has five standard data types −

1. Numbers (int, float)
2. String – " "
3. List – [ ]
4. Tuple – ()
5. Dictionary - – { }

Example:

```
# Define variables with different data types
n = 42                    # Integer
f = 3.14                  # Float
s = "Hello, World!"       # String
li = [1, 2, 3]            # List
t = (10, 20, 30)          # Tuple
d = {'key': 'value'}      # Dictionary
b = True                  # Boolean

# Print all variables and their data types
print(type(n))
print(type(f))
print(type(s))
print(type(li))
print(type(t))
print(type(d))
print(type(b))
```

Output:

```
>>> %Run lab_02.py
<class 'int'>
<class 'float'>
<class 'str'>
<class 'list'>
<class 'tuple'>
<class 'dict'>
<class 'bool'>
```

LAB # 02

Type Casting a Variable

Type casting refers to the process of converting the value of one data type into another. Python provides several built-in functions to facilitate casting, including int(), float() and str() among others.

Basic Casting Functions

- **int()** - Converts compatible values to an integer.
- **float()** - Transforms values into floating-point numbers.
- **str()** - Converts any data type into a string

Example:

```
# Casting variables
s = "10"          # Initially a string
n = int(s)        # Cast string to integer
cnt = 5           # Initially an integer
f = float(cnt)    # Cast integer to float
age = 25          # Initially an integer
s2 = str(age)     # Cast integer to string

# Display results
print(n)
print(f)
print(s2)
```

Output:

```
>>> %Run lab_02.py
10
5.0
25
```

Delete a Variable Using del Keyword

We can remove a variable from the namespace using the del keyword. This effectively deletes the variable and frees up the memory it was using.

Example:

```
# Assigning value to variable
z = 10

# Removing the variable using del
del z
print(z)

# Trying to print z after deletion will raise an error
# print(z)  # Uncommenting this line will raise NameError: name 'z' is not defined.
```

Output:

```
Traceback (most recent call last):
  File "C:\Users\Dell\Documents\Python Labs\lab 02.py", line 90, in <module>
    print(z)
NameError: name 'z' is not defined
```

Swapping Two Variables

Using multiple assignments, we can swap the values of two variables without needing a temporary variable.

Example:

```
x, y = 5, 10
print("Before Swapping:","x = ", x, "y = ", y)
x, y = y, x
print("After Swapping:","x = ", x, "y = ", y)
```

Output:

```
>>> %Run lab_02.py
Before Swapping: x =  5 y =  10
After Swapping: x =  10 y =  5
```

Operators

Operators are special symbols in Python that carry out arithmetic or logical computation. Python divides the operators in the following groups:

- Arithmetic operators
- Assignment operators
- Comparison operators
- Logical operators
- Identity operators
- Membership operators
- Bitwise operators

Python Arithmetic Operators

Arithmetic operators are used with numeric values to perform common mathematical operations:

Operator	Name	Example
+	Addition	x + y
-	Subtraction	x - y
*	Multiplication	x * y
/	Division	x / y
%	Modulus	x % y
**	Exponentiation	x ** y
//	Floor division	x // y

LAB # 02

Python Relational Operators

Comparison operators are used to compare two values:

Operator	Name	Example
= =	Equal	x = = y
!=	Not equal	x != y
>	Greater than	x > y
<	Less than	x < y
>=	Greater than or equal to	x >= y
<=	Less than or equal to	x <= y

Python Logical Operators

Logical operators are used to combine conditional statements:

Operator	Name	Example
and	Return True if both statements are true	x < 5 and x < 10
Or	Return True if one of the statements is true	x < 5 or x < 4
not	Reverse the result, returns False if the result is true	not (x<5 and x< 10)

EXERCISE

A. Point out the errors, if any, in the following Python statements.

1. x=5:
print(x)

2. 1TEXT = "SSUET"
NUMBER = 1
print(NUMBER+ TEXT)

3. a = b = 3 = 4

LAB # 02

B. Evaluate the operation in each of the following statements, and show the resultant value after each statement is executed.

1. a = 2 % 2 + 2 * 2 - 2 / 2;

2. b = 3 / 2 + 5 * 4 / 3 ;

3. c = b = a = 3 + 4 ;

C. Write the following Python programs:

1. Write a program that calculates area of a circle $A = \pi r^2$. (Consider $r = 50$).

2. Write a program that performs the following four operations and prints their result on the screen.
a. 50 + 4
b. 50 – 4
c. 50 * 4
d. 50 / 4

3. Write a Python program to convert height (in feet and inches) to centimeters. Convert height of 5 feet 2 inches to centimeters.

- First, convert 5 feet to inches: 5 feet × 12 inches/foot = 60 inches
- Add up our inches: 60 + 2 = 62 inches
- Convert inches to cm: 62 inches × 2.54 cm/inch = 157.48 cm

4. Write a program to compute distance between two points by creating variables (Pythagorean Theorem)

Distance =((*x*2−*x*1)^2+(*y*2−*y*1)^2)^1/2