Python - Variables, Expressions & Statements

Variables

The major purpose of variable is to store a value from one part of program so that it can be used in other part of program. It can be used to store the value of computation that is carried out during the flow of the program. A variable is a name that referes to a value.

Assignment statement

Assignment statement creates a new variable and store a value in it. For example,

In [ ]:

```
message = "this is first string variable"  # a string type variable named message is create
print(message)
```

In [ ]:

```
num = 5 # an int type variable named num is created and 5 is stored as value in int
print("num contains ----> ", num)
print("type of num -----> ", type(num))
```

In [ ]:

```
float_num = 3.5 # a variable of type float is created and value 3.5 stored in it
print("float_num contains ----->  ", float_num)

float_num = 4.5 # value of variable can be changed
print("float_num now contains ----->  ", float_num)
```

Variable Naming

Choose the variable name that is meaningful and improves the readability of the program.
Rules to be kept in mind while naming the variables :
- Letters, numbers and underscore are allowed - Can not contain spaces - Can not start with number - It's case sensitive - It should not be from the reserved keyword list

In [ ]:

```
name = "valid variable name" #valid variable name with letters
```

In [ ]:

```
number1 = 12 # valid varibale name of type int
```

In [ ]:

```
_name = "valid variable"  #variable name with underscore
```

In [ ]:

```
# var name = "some value" #invalid variable name as it contains space in it
```

In [ ]:

```
#1_name = "invalid variable"  # throws error as it starts with number
```

Variable names are case sensitive. Following three variable has same spelling but still contains different values in it.

In [ ]:

```
name1 = "xyz"
```

In [ ]:

```
Name1 = "por"
```

In [ ]:

```
NaMe1 = "abc"
```

In [ ]:

```
print("name1--->", name1)
```

In [ ]:

```
print("Name1--->", Name1)
```

In [ ]:

```
print("NaMe1--->", NaMe1)
```

Variable name can not be a keyword.

In [ ]:

```
#class = "my class value"    # throws erros as class is reserved word
```

Statements

It is unit of code that Python interpreter executes. It can be a single statement or group of statement like statements within loops, functions etc. When interpreter hits statement, it executes the statement and displays

the result (if there is any).

In [ ]:

```
#Function call is statement
print('this is statement')
```

In [ ]:

```
#Assignment is statement but does not produce any output
x = 5
```

In [ ]:

```
#Compound statment - for loop
for i in range(5):
    print("Coming inside the loop ----> " , i + 1 , " times")
    print("i is ---> ", i)
```

Expressions

Expressions in Python can include operations among compatible types (e.g., integers and floats). For example, basic arithmetic operations like adding multiple numbers:

In [ ]:

```
35 + 45 + 23
```

Python follows well accepted mathematical conventions when evaluating mathematical expressions. In the following example, Python adds 3 to the result of the multiplication (i.e., 12).

In [ ]:

```
4 * 3 + 3
```

Operators and Operands

These are special symbols that represents computations like addition, multiplication etc. Operand are values on which operators are applied.

In [ ]:

```
x = 5 + 6  # + is operator applied on operands 5 & 6
```

In [ ]:

```
y = x * 9.5 # * is operator applied on operands x & 9.5
```

In [ ]:

```
# z = y + 'abcs'  #throws error as operands should be compatible with the operator
```

In [ ]:

```
string = 'BITS' + 'Pilani' #for string + is concatenation
string
```

In [ ]:

```
string = "BITS Pilani " * 3 #for string * is repetation
string
```

Mathematical Operations / Expressions

In [ ]:

```
5 + 3 # addition
```

In [ ]:

```
5 - 3 # subtraction
```

In [ ]:

```
5 * 3 #multiplication
```

In [ ]:

```
5 / 3 # regular division
```

In [ ]:

```
5 // 3 # modulo division, returns reminder after division
```

In [ ]:

```
5 % 3 #integer division
```

In [ ]:

```
5 ** 3 # exponent
```

In [ ]:

```
num1 = 2
num2 = 3

print(num1 + num2)
print(num1 - num2)
print(num1 * num2)
print(num1 / num2)
print(num1 // num2)
print(num1 % num2)
print(num1 ** num2)
```

In [ ]:

```
num1 = 8
num1 += 5
print(num1)
num1 -= 1
print(num1)
```

Comparison Operations

In [ ]:

```
num1 = 4
num2 = 12
print(num1 > num2)
print(num1 < num2)
print(num1 >= num2)
print(num1 <= num2)
print(num1 == num2)
print(num1 != num2)
```

Exercise

Q1. Compute the simple interest using the formula interest = p * n * r / 100 where p is principal, n is tenure of loan, and r is rate of interest. Accept p, n, r from users and then perform the interest calculation.

In [ ]:

```
#Try it here
```

Q2. Write a program that asks user weight in kilogram and converts into pounds. There are 2.2 pound in one kilogram.

In [ ]:

```
#Try it here
```

Q3. If a four digit number is entered through keyboard by user. Write a program to calculate the sum of its digits.

In [ ]:

```
#Try it here
```

In [ ]:

```

```