ITA6017
Python Programming

Dr. Arun Pandian J

Module 2: Python Operators, Expressions and Flow controls

All Operations and simple expressions, Conditional blocks using if, else and elif, Simple for loops in python, For loop using ranges, Use of while and do while-loop in python, Loop manipulation using pass, continue, break and else.

Operators

- In computer programming languages operators are special symbols which represent computations, conditional matching etc.
- The value of an operator used is called operands.
    - **Arithmetic operators**
    - **Relational or Comparative operators**
    - **Logical operators**
    - **Assignment operators**
    - **Conditional operator**

Arithmetic operators

- An arithmetic operator is a mathematical operator that takes two operands and performs a calculation on them.

Operator - Operation	Examples	Result
Assume a=100 and b=10. Evaluate the following expressions		
+ (Addition)	>>> a + b	110
- (Subtraction)	>>>a – b	90
* (Multiplication)	>>> a*b	1000
/ (Divisioin)	>>> a / b	10.0
% (Modulus)	>>> a % 30	10
** (Exponent)	>>> a ** 2	10000
// (Floor Division)	>>> a//30 (Integer Division)	3

Relational or Comparative operators

- A Relational operator is also called as Comparative operator which checks the relationship between two operands.

Operator - Operation	Examples	Result
Assume the value of a=100 and b=35. Evaluate the following expressions.		
== (is Equal)	>>> a==b	False
> (Greater than)	>>> a > b	True
< (Less than)	>>> a < b	False
>= (Greater than or Equal to)	>>> a >= b	True
<= (Less than or Equal to)	>>> a <= b	False
!= (Not equal to)	>>> a != b	True

Logical operators

- Logical operators are used to perform logical operations on the given relational expressions.

Operator	Example	Result
Assume a = 97 and b = 35, Evaluate the following Logical expressions		
or	>>> a>b or a==b	True
and	>>> a>b and a==b	False
not	>>> not a>b	False i.e. Not True

Assignment operators

- = is a simple assignment operator to assign values to variable.

Operator	Description	Example
Assume x=10		
=	Assigns right side operands to left variable	>>> x=10 >>> b="Computer"
+=	Added and assign back the result to left operand i.e. x=30	>>> x+=20 # x=x+20
-=	Subtracted and assign back the result to left operand i.e. x=25	>>> x-=5 # x=x-5
*=	Multiplied and assign back the result to left operand i.e. x=125	>>> x*=5 # x=x*5
/=	Divided and assign back the result to left operand i.e. x=62.5	>>> x/=2 # x=x/2

Conditional Operator

- Ternary operator is also known as conditional operator that evaluate something based on a condition being true or false.
- It simply allows testing a condition in a single line

Variable Name = [on_true] if [Test expression] else [on_false]

```
a=10
b=20
min = a if a < b else b
```

Task:

Reads two integers from user, a and b. Add code to print three lines where:

- The first line contains the sum of the two numbers.
- The second line contains the difference of the two numbers (first - second).
- The third line contains the product of the two numbers.

Task:

Reads two integers, a and b, from input. Add logic to print two lines. The first line should contain the result of integer division, a//b. The second line should contain the result of float division, a/b.

No rounding or formatting is necessary.

Simple Expression

- An expression is a combination of operators and operands that is interpreted to produce some other value.
- In any programming language, an expression is evaluated as per the precedence of its operators.
- So that if there is more than one operator in an expression, their precedence decides which operation will be performed first.

Example:

```
x = 15 + 1.3
add = x + y
sub = x - y
pro = x * y
div = x / y
c = a + int(b)
p = (a + b) >= (c - d)
```

Variables and Expressions

Constants

- Fixed values such as numbers, letters, and strings are called "constants" because their value does not change

  Numeric constants are as you expect

  String constants use single quotes (') or double quotes (")

```
>>> print(123)
123
>>> print(98.6)
98.6
>>> print('Hello world')
Hello world
```

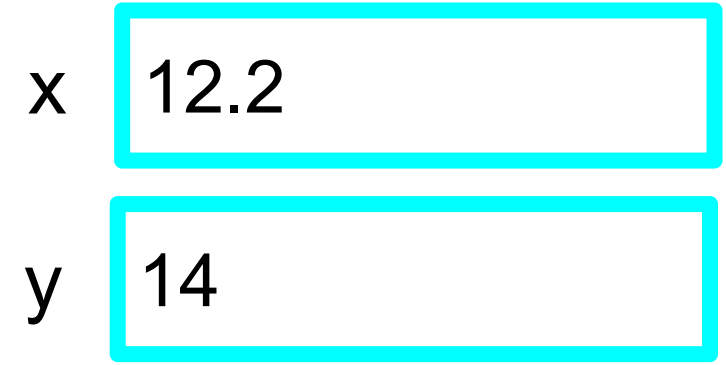
Variables

A variable is a named place in the memory where a programmer can store data and later retrieve the data using the variable “name”

Programmers get to choose the names of the variables

You can change the contents of a variable in a later statement

```
x = 12.2
y = 14
```



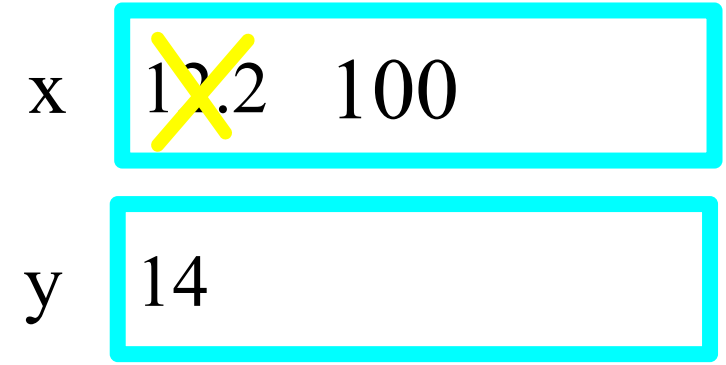
Variables

A variable is a named place in the memory where a programmer can store data and later retrieve the data using the variable “name”

Programmers get to choose the names of the variables

You can change the contents of a variable in a later statement

```
x = 12.2
y = 14
x = 100
```



Python Variable Name Rules

- Must start with a letter or underscore _
- Must consist of letters and numbers and underscores
- Case Sensitive

Good: spam eggs spam23 _speed
Bad: 23spam #sign var.12
Different: spam Spam SPAM

Reserved Words

You cannot use reserved words as variable names / identifiers

False	**class**	**return**	**is**		**finally**
None	**if**		**for**	**lambda**	**continue**
True	**def**	**from**	**while**	**nonlocal**	
and	**del**	**global**	**not**	**with**	
as	**elif**	**try**		**or**	
	yield				
assert	**else**	**import**	**pass**		
break	**except**	**in**		**raise**	

Sentences or Lines

x = 2	Assignment statement
x = x + 2	Assignment with expression
print(x)	Print statement

Variable Operator Constant Reserved Word

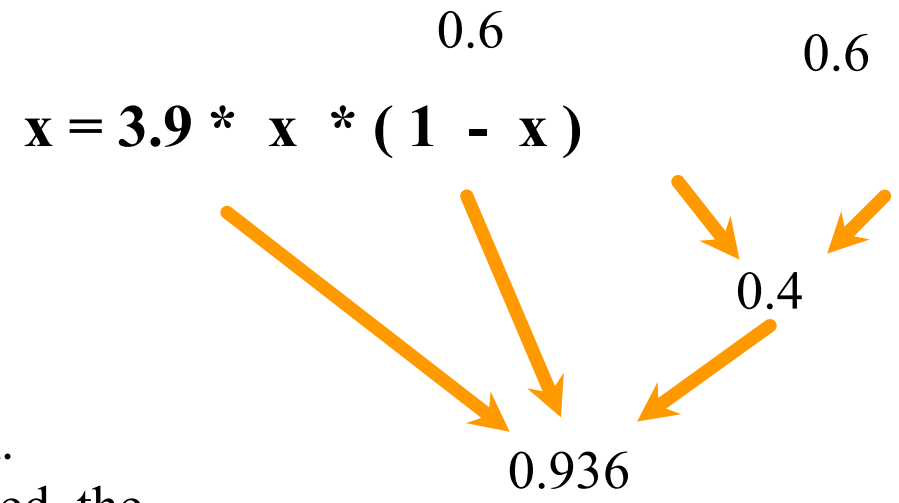
Assignment Statements

- We assign a value to a variable using the assignment statement (=)

- An assignment statement consists of an expression on the right-hand side and a variable to store the result

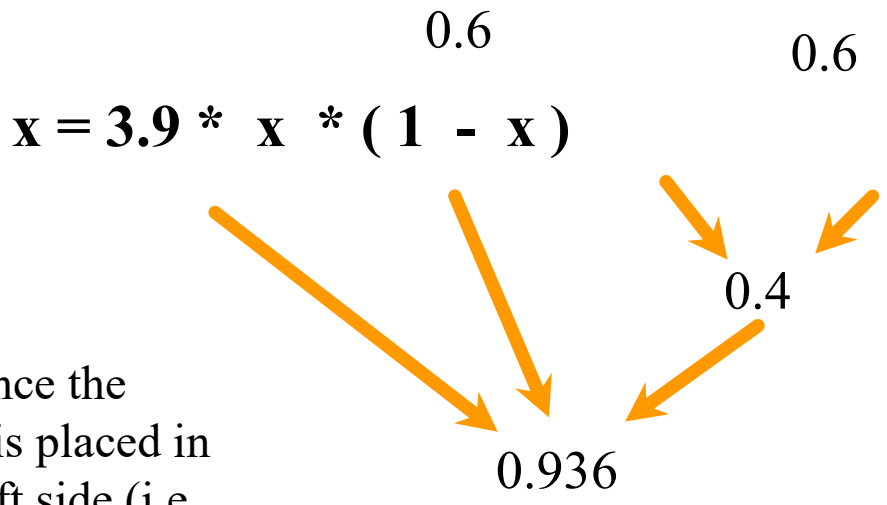
x = 3.9 * x * (1 - x)

A variable is a memory location used to store a value (0.6)



The right side is an expression.
Once the expression is evaluated, the result is placed in (assigned to) x.

A variable is a memory location used to store a value. The value stored in a variable can be updated by replacing the old value (0.6) with a new value (0.93).



The right side is an expression. Once the expression is evaluated, the result is placed in (assigned to) the variable on the left side (i.e., x).

Numeric Expressions

Because of the lack of mathematical symbols on computer keyboards - we use “computer-speak” to express the classic math operations

Asterisk is multiplication

Exponentiation (raise to a power) looks different from in math.

Operator	Operation
+	Addition
-	Subtraction
*	Multiplication
/	Division
**	Power
%	Remainder

Order of Evaluation

When we string operators together - Python must know which one to do first

This is called “operator precedence”

Which operator “takes precedence” over the others?

x = 1 + 2 * 3 - 4 / 5 ** 6

Operator Precedence Rules

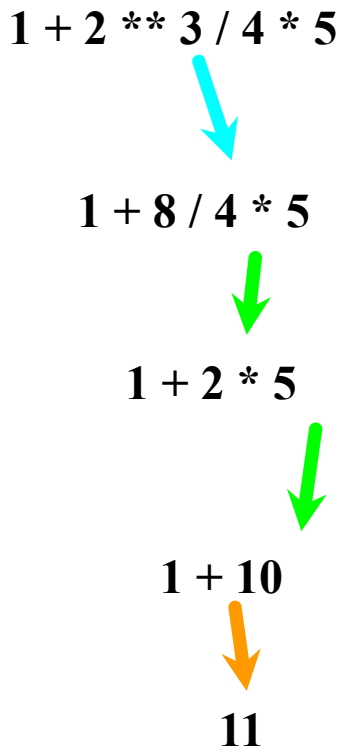
Highest precedence rule to lowest precedence rule:

- Parenthesis are always respected
- Exponentiation (raise to a power)
- Multiplication, Division, and Remainder
- Addition and Subtraction
- Left to right

Parenthesis
Power
Multiplication
Addition
Left to Right

```
>>> x = 1 + 2 ** 3 / 4 * 5
>>> print(x)
11.0
>>>
```

Parenthesis
Power
Multiplication
Addition
Left to Right



Operator Precedence

Parenthesis
Power
Multiplication
Addition
Left to Right

Remember the rules top to bottom

When writing code - use parenthesis

When writing code - keep mathematical expressions simple enough that they are easy to understand

Break long series of mathematical operations up to make them more clear

Task: x = 1 + 2 * 3 - 4 / 5

Task:

Read the inputs as dictionary containing key/value pairs of name:[marks] for a list of students. Print the average of the marks array for the student name provided, showing 2 places after the decimal.

Example

marks key:value pairs are

'alpha': [20, 30, 40]

'beta': [30, 50, 70]

query_name = 'beta'

The **query_name** is 'beta'. beta's average score is $(30 + 50 + 70)/3 = 50.0$.