Keywords

Keywords are reserved words In python.

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

Types of Operators

An operator is a symbol that performs a certain operation between operands.

Arithmetic Operators (+ , - ,*, / , % ,**)
Relational / Comparison Operators (== , != , > , < , >= , <=)
Assignment Operators (= , +=, -= ,*= , /= , %= ,**=)
Logical Operators (not , and , or)
Membership Operators (in, not in)
Identity operators (is, is not)
Bitwise Operator (&, |, ^)

Expressions in Python

Combination of operators and operands.

x =10

x + 3

Operands

operator

Operands

Constant Expressions

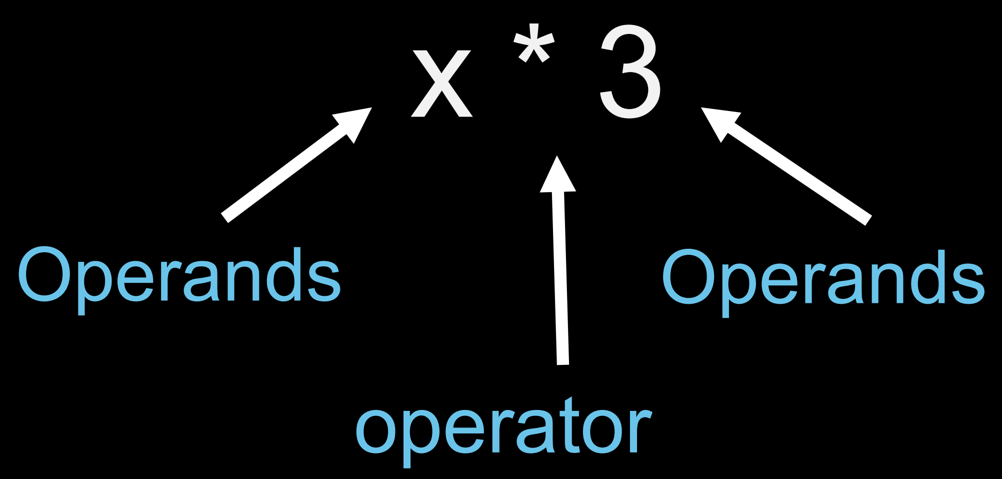
A constant expression has only constants as operands

$$\underset{\text{Operands}}{10} \; \underset{\text{operator}}{+} \; \underset{\text{Operands}}{30}$$

Arithmetic Expressions

Contains numeric values por strings as operands,
must has at least one arithmetic operators and sometimes parentheses.

```
x ="Arif"
```



Integral Expressions

Results on integer value after performing the necessary type conversions

```
x = 5
y = 7.5
result =  x + int(y)
print(result)
```

```
x = 5
y = “5”
result =  x + int(y)
print(result)
```

Floating-point Expressions

Results on floating-point value after performing the necessary type conversions

```
x = 10
y = 17.5
result =  float(x) + y
print(result)
```

```
x = 10
y = 5
result =  x/y
print(result)
```

Relational Expressions

Also called Boolean expressions
Returns a Boolean value

$$\underset{\substack{\uparrow \\ \text{Arithmetic} \\ \text{Expressions}}}{(10 + 13)} \; \underset{\substack{\uparrow \\ \text{Relational} \\ \text{operator}}}{<=} \; \underset{\substack{\uparrow \\ \text{Arithmetic} \\ \text{Expressions}}}{(2 + 3)}$$

Logical Expressions

Consists of relational expressions connected using logical operators.
Returns a Boolean value

$$\underbrace{(10 < 13)}_{\text{Relational Expressions}}\ \underbrace{\text{and}}_{\text{logical operator}}\ \underbrace{(1 == 1)}_{\text{Relational Expressions}}$$

Bitwise Expressions

Contains Bitwise operators
Computations are performed at bit-level.

10 << 2

Relational Expressions

Bitwise operator

Relational Expressions

Combinational Expressions

Combination of Different expressions

```
x = 10
y = 20
z = y + (x<<1) – x*3
```

Comments in Python

```
# Single Line Comment

"""

Multi Line

Comment

"""
```

Input in Python

Input() statement is used to accept values (using keyword) from user

- String input

  ```
  name = input("name :  ")
  ```

- int input

  ```
  age = int(input("age :  "))
  ```

- float input

  ```
  price = float(input("price:  "))
  ```

Practice Time

Write a program to input 2 numbers & print their sum

Practice Time

Write a Program to input side of square & print its area

Practice Time

Write a Program to input 2 floating point & print their average

Practice Time

Write a Program to input 2 int numbers , a and b.
Print True if a is greater than or equal to b. if not print False.

Practice Time

State True or Flase

1. /* is a symbol used in python for single line comment
2. 2ndName is an invalid identifier in python
3. ** is a valid arithmetic operator in python
4. In is a logical operator in Python
5. Variable declaration is implicit in Python

Practice Time

6. Consider the given expression: not True and False or Ture
Which of the following will be correct output if the given expression is evaluated?

a) True
b) False
c) NONE
d) NULL

Strings

Strings is a data type that stores a sequence of characters.

Basic Operations

- concatenation
  “hello” + “world” → “helloworld”
- length of str
  len(str)

Indexing

Ariful islam

0 1 2 3 4 5 6 7 8 9 10 11

str = “Ariful islam”

str[0] is ‘A’, str[1] is ‘r’

str[0] = ‘B’ #not allowed

Slicing

Accessing parts of a string

str[starting_index : ending_index] #ending index is not included

str = “Ariful islam”

str[1:4] is “rif”

str[:4] is same as str[0:4]

str[1:] is same as str[1:len(str)]

Slicing

Negative Index

Ariful

-6 -5 -4-3 -2 -1

Str = “Ariful”

str[-3:-1] is “fu”

String Functions

```
str = “I am a coder.”
str.endsWith(“er.“) #returns true if string ends with substr
str.capitalize( ) #capitalizes 1st char
str.replace( “o”, “n” ) #replaces all occurrences of old with new
str.find( word ) #returns 1st index of 1st occurrence
str.count(“am“) #counts the occurrence of substr in string
```

Let’s Practice

Write a program that asks the user to enter their full name and prints the number of characters in their name (excluding spaces).

Let’s Practice

Write a program to count how many times the character #
appears in a given string.