Python Programming

Dec 1, 2020

Dec 2020

Why Python?

- Simple, yet powerful
- Easy to learn
- East to read
- Rich set of features
- Widely used
- Wide variety of applications

Applications

- Internet of Things
- Data analytics
- Web programming
- Server programming
- Scientific computing
- Productivity tools
- Desktop applications
- Gaming applications

Users

- Google
- NASA
- IBM
- Digital Illusions – Battle Field 2
- Yahoo Maps
- Walt Disney
- Honeywell

Installations

Installations

- You must install *Python3* from https://www.python.org/downloads/
- https://www.python.org/ftp/python/3.9.0/python-3.9.0.exe
- You can also use the Python Interpreter online
  - https://www.onlinegdb.com/online_python_interpreter

Your first Python Program

Hello World

- Open *IDLE*
- Click File -> New-> Python File
- Type *print "Hello World!"*
- File-Save-As *HelloWorld.py*
- Click Run -> Run Module or Click F5

Hello World – Further Practice

- In PyCharm open file D:\Exercises\HelloWorld.py
- Type the following text
  - print("Hello World!")
  - print("Hello Again")
  - print("I like typing this.")
- Click File->Save
- Click F5
- What do you see as output?

Identifiers, Variables and Assignments

Identifiers

- Name used to identify a variable, function, class, module or other object.
- Starts with an alphabet (A-Z, a-z) or underscore (_)
- Followed by 0 or more alphabets, underscores and digits (0 to 9)
- Examples : a, ab, A, Ab, myVar, my_var, var1, _myvar, my_var2
- No characters such as @, $, and % within identifiers.
- Not variables : 1, 1a, $one, %âbc
- They can be of any length.
- Case sensitive. ***Manpower*** and ***manpower*** are two different identifiers

Practice - Identify the variables

- A1
- a1a_b123
- _xyzABC
- $varName_123
- myVarName$xyz
- myVarName
- i

Keywords

- Reserved words
- Cannot be used as identifiers
- All the Python keywords contain lowercase letters only.

Keywords

-

and	exec	not
assert	finally	or
break	for	pass
class	from	print
continue	global	raise
def	if	return
del	import	try
elif	in	while
else	is	with
except	lambda	if

Keywords - Practice

- Open IDLE
- Open file D:\Exercises\keywords.py
- Click F5
- What do you see as output?

Invalid syntax

Use other keywords and observe the output

Lines and Indentation

- Blocks of code are denoted by line indentation
- Rigidly enforced.
- No specific number of spaces in the indentation
- All statements within the block must be indented the same amount.

```
if True:
        print "True"
else:
        print "False"
```

- Python provides no braces to indicate blocks of code for class and function definitions or flow control.

Variables

- Variables are locations in memory with a particular name
- Variable values change
- Consider it as a box where you can put values
- Create a variable, name it myVar and put 10 in it
- myVar = 10
- Print the value in myVar
- print myVar

Variable Creation

- Many variables are created in a program
- myVar1 = 100
- myVar2 = 11.3
- myVar3 = “Global”

- Variables can be of different types
- No need to declare variables before using. They are created automatically
- Print the variables
- Print myVar1, myVar2, myVar3

Variable Assignment

- Open PyCharm
- Open file D:\Exercises\Variable_Assignment.py
- Click Run->Run ->*Variable_Assignment*
- What do you see as output?

10 11.3 Global

Variables - changing

- Create variables and assign
- myVar1 = 2018

- myVar3 = “Global”

- Print the variables // Prints 2018 Global
- Print myVar1, myVar3
- Change the Variable myVar3
- myVar3 = “Academy”

- Print the variables // Prints 2018 Academy
- Print myVar1, myVar3

Variable Changing

- Open PyCharm
- Open file D:\Exercises\Variable_Changing.py
- Click Run->Run ->*Variable_Changing*
- What do you see as output?

2018 Global

2018 Academy

Variable Assignment – Further Practice

- Open file D:\Exercises\Variable_Assignment.py
- *Add variable pi and assign 3.14 to it*
- *Add variable empname and assign "python is great" to it*
- *Assign 100 to a, b and c variables*
- *Print pi, empname a, b and c*
- Click Run->Run->*Variable_Assignment*
- What do you see as output?

Variables Assignment

```
x = 100                                  # x is integer
pi = 3.14                                # pi is float
empname = "python is great"   # empname is string

a = b = c = 100 # this statement assign 100 to c, b and a.

print ("x:", x)
print ("pi:", pi
print "empname:", empname
print "a:", a
print "b:", b
print "c:", c
```

Objects and References

- Everything is object in Python
  - Includes basic data types like int, float, string
  - Variables store reference to an object
- x = 100
  - 100 is an int object (int)
  - x stores reference to 100
  - x doesn't store 100 itself

Objects and References

- **pi = 3.14**
  - 3.14 is an int object (float)
  - pi stores reference to 3.14
  - pi doesn't store 3.14 itself
- comment = "python is great"
  - comment stores reference "python is great" (string)
  - comment doesn't store the string "python is great"

Simultaneous Assignment

- var1, var2, ..., varn = exp1, exp2, ..., expn
- Example
  - myVar1, myVar2, myVar3 = 100, 11.3, “ABC”
  - myVar1 = 100, myVar2 = 11.3, myVar3 = “ABC”
- Evaluate expressions on the right and assign them to the corresponding variables on the left.
- Simultaneous Assignments is helpful to swap values of two variables.

Simultaneous Assignment

- Open file D:\Exercises\Variable_Assignment_2.py
- Click Run->Run->*Variable_Assignment_2*
  - myVar1, myVar2, myVar3 = 100, 11.3, "ABC"
  - print myVar1, myVar2, myVar3

- What do you see as output?
  - 100 11.3 ABC

Simultaneous Assignment - Swap

- Open file D:\Exercises\Variable_Assignment_2.py
- Click Run->Run->*Variable_Assignment_3*
- What do you see as output?

Simultaneous Assignment

```
x = 1
y = 2

print "x: ", x
print "y: ", y

y, x = x, y # assign y value to x and x value to y

print "After Swapping"

print "x: ", x
print "y: ", y
```

Simultaneous Assignment – Further Practice

- Assign into three variables 22, “myString”, 4
- Use simultaneous assignment
- Print the variables
- Program should have only 2 lines

Receiving Inputs

Receiving Data from Console

- *raw_input()* function is used to receive input from the console
- Syntax
  - varName = raw_input();
- Example
  - myVar = raw_input()

Receiving Data from Console

- *raw_input()* function is used to receive input from the console
- Syntax
  - varName = raw_input();
- Example
  - myVar = raw_input()

Receiving Data - 1

- Open PyCharm
- Open file D:\Exercises\Receive_Data.py
- Click Run->Run->*Receive_Data_1*
- Enter any string in the bottom window (Global)
- Global
- Received input is : Global

Receiving Data from Console

- Prompt can be used receive the input
- Syntax : raw_input([prompt])
- Example
- myVar = raw_input("Enter Your Name : ")
- Print "Name entered is : "", myVar

Receiving Data - 2

- Open file D:\Exercises\Receive_Data2.py
- Click Run->Run->*Receive_Data_2*
- Enter name and age as prompted
- Enter Your Name : Global
- Received input is Global
- Enter Your Age : 21
- Your age is 21
- Type of age is <type 'str'>

Receiving Data from Console - Integer

- *raw_input()* returns string even if you enter a number
- To convert it to an integer you can use int()
- myVar = int(raw_input("Enter Your Age : ")

Receiving Data - 3

- Open PyCharm
- Open file D:\Exercises\Receive_Data_3.py
- Click Run->Run->*Receive_Data_3*
- Enter your name : Global
- Your Name is Global
- Enter your age : 21
- Your age is 21
- Type of age is : <type 'int'>

Receiving Data – Further Practice

- Open file D:\Exercises\Receive_Data2.py
- *Modify the program*
  - *Receive Year, branch name and course into yr branchName and courseName*
  - *Print the year, branch name and course*
-

Receiving Data – Further Practice

- branchName = raw_input(“Enter Branch Name : ”)
- courseName = raw_input(“Enter Course Name : ”)
- print “Branch Name : ”, branchName
- print “Course Name : ”, courseName

Input() function

- *input()* function is also used to receive input from the console
- Syntax: input([prompt])
- Example
- myVar = input("Enter Value");
- *input([prompt])*
  - Assumes the input is a valid Python expression
  - Returns the evaluated result to you.

Input() function - Practice

- Open *PyCharm*
- Open file D:\Exercises\Receive_Data_4.py
- Click Run->Run->*Receive_Data_4*
- Enter the input at console as *1+2+3*
- *Expression 1+2+3 gets evaluated as 6*
- *Output*
- *6*

Input() function - Practice

- Open *PyCharm*
- Open file D:\Exercises\Receive_Data_4.py
- Click Run->Run->*Receive_Data_4*
- Enter the input from console as *[x*5 for x in range(2,10,2)]*
- *Range (2,10,2) indicates 2, 4, 6, 8*
- *x takes the value 2,4,6, 8, gets multiplied by 5 and gets printed*
- *Output : 10, 20, 30, 40*

Input() function - Practice

- Open *PyCharm*
- Open file D:\Exercises\Receive_Data_4.py
- Click Run->Run->*Receive_Data_4*
- Enter the input from console any other expression
- *2+3*5 + 7*
- *x*2 for x in (1, 10, 1)*

Input() function - Practice

- Open file D:\Exercises\Receive_Data_5.py
- Click Run->Run->*Receive_Data_5*
- *Enter a : 3*
- *Enter b : 4*
- *Enter expression : a * b*
- *12*
- Try other expressions
- Define one more variable **c** and add it to expression

Importing Modules

Importing Modules

- Python comes with many in built modules ready to use
- *math* module for mathematics related functions
- Import them before using them with the following:
  - *import module_name*
- Import multiple modules using the following syntax:
  - *import module_name_1, module_name_2*

Importing Module - Practice

- Open *PyCharm*
- Open file D:\Exercises\import_module.py
- Click Run->Run->*import_module*
- What do you see as output?

Importing *math* Modules

```
import math

print "Value of Pi is:", math.pi
print "Value of 2 to the power 3 is:", math.pow(2, 3)
print "90 degree angle in radians is: ", math.radians(90)
print "sin(45) is:", math.sin(math.radians(45))
```

Importing Module – Computing n power m

- Open *PyCharm*
- Create new Python file using File->New->Python File -> power.py
- Read n and m using input() function
- print n power m using math.pow()
- Save the file

Importing Module – Computing n power m

- Open *PyCharm*
- Create new Python file using File->New->Python File -> misc.py
- Program should print the following
- cos(45)
- exp(10)
- factorial(5)

Importing Module – Further Practice

```
import math
print "Value of Pi is:", math.pi
print "Value of 2 to the power 3 is:", math.pow(2, 3)
print "90 degree angle in radians is: ", math.radians(90)
print "sin(45) is:", math.sin(math.radians(45))

print "exp(10) is:", math.exp(10)
print "factorial 5 is: ", math.factorial(5)
print "cos(45) is:", math.cos(math.radians(45))
```

Thank You