Developing Python Apps

Part 1: Basics A)

Kauko Kolehmainen

2020

Contents

[Introduction 2](#_Toc40162543)

[What is programming? 2](#_Toc40162544)

[Programming languages 2](#_Toc40162545)

[Install Python 3](#_Toc40162546)

[Variables 6](#_Toc40162547)

[Python Data types names 6](#_Toc40162548)

[Arithmetic operators 8](#_Toc40162549)

[Examples 8](#_Toc40162550)

[Decision making (branching) 9](#_Toc40162551)

[Relational operators (to create conditions) 9](#_Toc40162552)

[if statement 9](#_Toc40162553)

[Example of if else 10](#_Toc40162554)

[Several choices -> several if-statements 11](#_Toc40162555)

[Nested if-else statements 11](#_Toc40162556)

[Logical operators 12](#_Toc40162557)

[Loops 14](#_Toc40162558)

[for-loop 14](#_Toc40162559)

[About random numbers 15](#_Toc40162560)

[while loop 17](#_Toc40162561)

[Break and continue statements 17](#_Toc40162562)

[Some special math operators 18](#_Toc40162563)

Developing Python Apps

# Introduction

## What is programming?

We give instructions to the computer: set of instructions is a program.  
Computer is mainly the processor that can understand machine code.  
So our instructions are compiled to machine code so that it can be executed by  
the computer.

A program contains  
storages, data structures  
functions, activities, operations

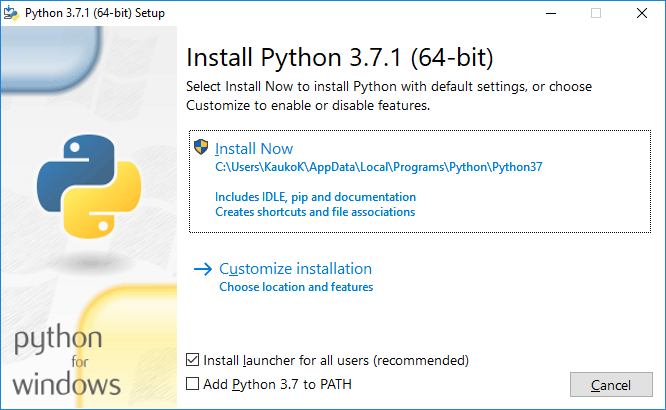
## Programming languages

Java: workstation and enterprise applications AND Android phones  
Object Oriented Programming language (OOP)  
C and C++  
C is used in Embedded programming, games and so on  
Procedural Programming language (not OOP!!)  
C++  
Object Oriented Programming language   
Games, operating systems   
We use C#: developed by Microsoft, used in ASP.NET, workstation software, games etc.  
OOP  
PHP: for web programming  
JavaScript: for web programming  
HTML: web page contents markup language  
Others: python, ruby, basic (first languages),  
SQL: for database queries  
ObjectiveC: for IOS   
Assembler: symbolic machine language  
Pascal: procedural language, easy to learn (Delphi?)

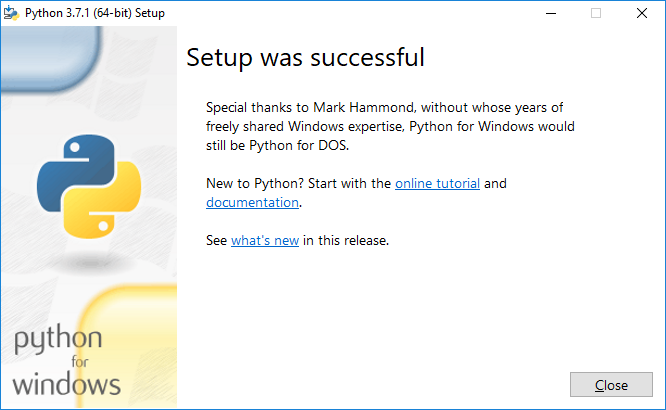
Tool: Python

# Install Python

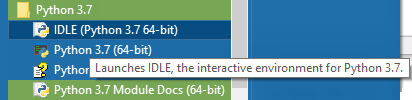
<https://www.python.org/downloads/>

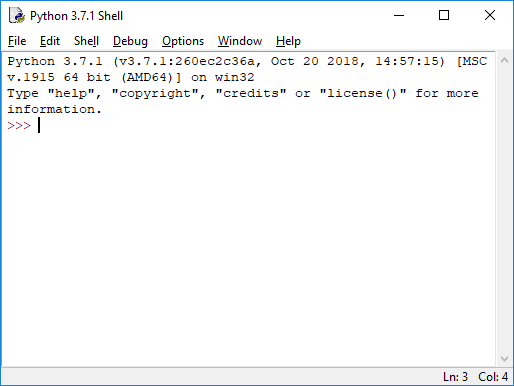


AND

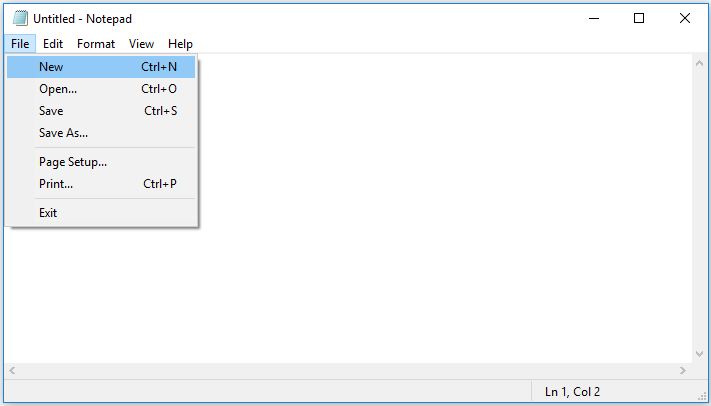


Start python





Start a new program

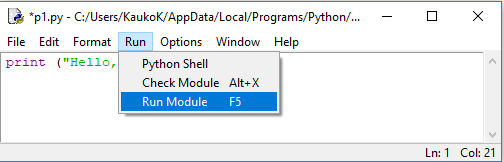


IDE: Integrated Development Environment  
RAD: Rapid Application Development

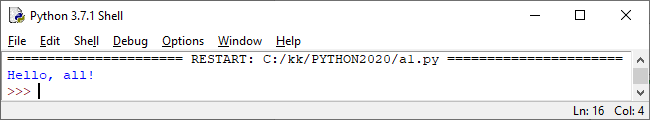
Let’s get to know a bit about Python tool!

Try first a pure console code:

print(”Hello, all!”)

Then choose Run  


You get



# Variables

Storages used by the program.   
Variables have to be defined before they can be used.

In definition we need **normally (C, C++, Java,…)** to tell:  
data type and name of the variable.

Data type tells what kinds of values I can store to a variable:  
Integer? 1, 20, 10000  
Floating point value (decimal) value? 2.35, 100.5555  
Character? ‘a’, ‘4’  
Boolean value? true, false  
Text (string)? “Kokkola”, “USA”

Concerning data types of variables Python differs from e.g. C, C++, C# and Java:  
Python sets the variable type based on the value that is assigned to it.

## Python Data types names

Booleans

Numbers

Strings

Bytes

Lists

Tuples

Sets

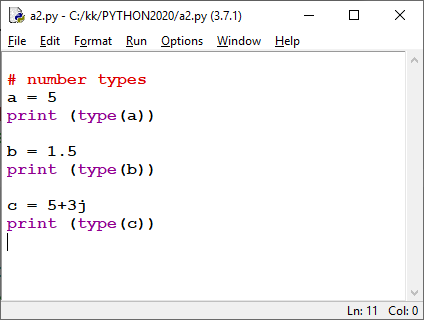
Dictionaries

Number types are

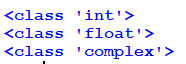
int, float, and complex

We concentrate on types numbers, bytes and strings and booleans.

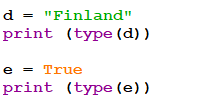
Examples of datatypes (function type tells the datatype)



We get



AND



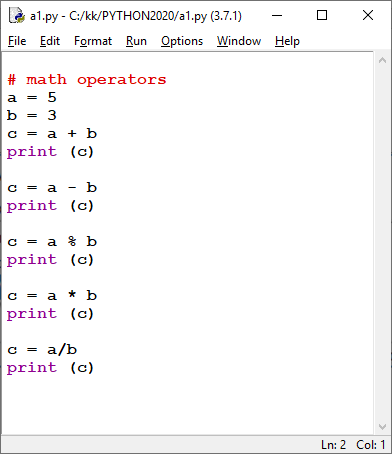
Gives



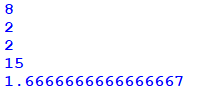
# Arithmetic operators

|  |  |
| --- | --- |
| Operator | Explanation |
| + | Addition |
| - | Subtraction |
| \* | Multiplication |
| % | Remainder |
| / | Division |
|  |  |

## Examples



We get



# Decision making (branching)

Program flow is decided depending on the condition

# Relational operators (to create conditions)

|  |  |
| --- | --- |
| Operator | Explanation |
| < | Smaller than |
| <= | Smaller than or equal to |
| > | Bigger than |
| >= | Bigger than or equal to |
| == | Equal to |
| != | Not equal to |
|  |  |

# if statement

Syntax:

if this is true:  
 this code is executed

Example

a = 5

if a != 5:

print ("a is NOT 5!")

If several statements are executed after if, we use a program block that is created automatically  
with indents:

if this is true :  
 this code is executed  
 and this code is executed  
 and …

We can have else part, too:  
if this is true:  
 this code is executed  
else:  
 this code is executed

## Example of if else

a = 5

if a != 5:

print ("a is NOT 5!")

else:

print ("a is 5!")

Example: program tells if given number is positive or not.

n = int(input("Give a whole number "))

if n >= 0:

print ("is positive")

else:

print("is negative")

We get



## Several choices -> several if-statements

n = int(input("Give a whole number "))

if n == 0:

print ("zero")

elif n == 1:

print ("one")

elif n == 2:

print ("two")

else:

print("other value")

We get



## Nested if-else statements

Example: is given number between 1 and?

x = 5

if x >= 1:

if x <= 5:

print("x is between 1 and 5")

else:

print("x is NOT between 1 and 5")

else:

print("x is NOT between 1 and 5")

We get



# Logical operators

|  |  |  |
| --- | --- | --- |
| Operator | Explanation | Example |
| && | AND | int a = 5;  (a >= 0 && a <=10)  true |
| || | OR | (a < 0 || a > 10)  false |

Example: is given value between 0 and 10?

// way 1

x = -3

if x >= 0:

if x <= 10:

print("x is between 0 and 10")

else:

print("x is NOT between 0 and 10")

else:

print("x is NOT between 0 and 10")

// way 2

x = -3

if x >= 0 and x <= 10:

print("x is between 0 and 10")

else:

print("x is NOT between 0 and 10")

// way 3

x = -3

if x < 0 or x > 10:

print("x is NOT between 0 and 10")

else:

print("x is between 0 and 10")

# Loops

We repeat the code until some solution is found

Used for example:  
when searching for a value from an array  
when generating and printing hundreds of random numbers  
in iterations

## for-loop

syntax  
for definition:

body of the loop

Program flow:  
go straight forward (step by step)  
decision making (branching)  
loops

**Examples of using for loop**

Example: print out values 1 to 5  
#print values 0 to 5  
for x in range(6):  
 print(x)

Example: print out 4, 8, 12, … 24  
#print values 4, 8, ... 24  
for x in range(4, 24, 4):  
 print(x)

Example: sum of values 1 - 5

#Program calculates the sum of values 1 - 5

sum = 0

for x in range(6):

sum = sum + x

print(sum)

More examples

#Program calculates the sum of even numbers between 2 - 40

sum = 0

for x in range(2,42, 2):

sum = sum + x

print(sum)

#Program calculates sum: 5, 10, 15, .. 100.

sum = 0

for x in range(5, 105, 5):

sum = sum + x

print(sum)

## About random numbers

How to get random numbers?

Random object is needed: we have to import random module.

Import random

Then we can e.g. method   
random.randint(lower limit, upper limit)  
to get random values.

Getting values between 1 and 10:

x = random.randint(1,10)

Example: random numbers

import random

#Program generates 50 random numbers (between 1 to 10)

for x in range(50):

y = random.randint(1,10)

print (y)

#Counting amounts of different numbers  
n1 = 0

n2 = 0

n3 = 0

n4 = 0

n5 = 0

for x in range(50):

y = random.randint(1,11)

if y == 1:

n1 += 1

elif y == 2:

n2 += 1

elif y == 3:

n3 += 1

elif y == 4:

n4 += 1

elif y == 5:

n5 += 1

print("Amounts:")

print(n1)

print(n2)

print(n3)

print(n4)

print(n5)

We get

Amounts:

5

7

3

3

4

Conditional loops

## while loop

Syntax:

while (condition is true):  
 code

Examples of while loop

#while

print ("while loop example")

k = 1

while k < 6:

print(k)

k = k + 1

## Break and continue statements

Used with loops  
Note: break was used even with switch-case

With break  
you can terminate the loop when some condition is true

E.g.  
When searching for a value from an array by using a loop:  
when the value is found, there no use to go on searching,  
just terminate the loop with break statement!

With continue  
you can start a new round without executing the code that exists below continue statement

Example of break:

#equation: 3x^3 - 2x^2 + 4x -7 = 0

x = -5.0

y = 0.0

while True:

y = 3\*x\*\*3 - 2\*x\*\*2 + 4\*x -7

if y > -0.001 and y < 0.001:

break

x += 0.0001

print(x)

print(y)

We get

1.191400000001571

7.293985086676003e-05

# Some special math operators

Assignment and math operators combined:  
+=  
-=  
\*=  
/=  
%=  
  
Example:  
x = 10;  
x += 5 # same as x = x + 5