# Python Forensics

tutorialspoint

SIMPLY EASY LEARNING

www.tutorialspoint.com

https://www.facebook.com/tutorialspointindia

https://twitter.com/tutorialspoint

## About the Tutorial

Python has built-in capabilities to support digital investigation and protect the integrity of evidence during an investigation. In this tutorial, we will explain the fundamental concepts of applying Python in computational (digital) forensics that includes extracting evidence, collecting basic data, and encryption of passwords as required.

## Audience

This tutorial is meant for all those readers who seek to increase their understanding in digital or computational forensics through the use of Python. It will help you understand how to integrate Python in computational forensics.

## Prerequisites

Before starting with this tutorial, it is important that you understand the basic concepts of computational forensics. And, it will definitely help if you have prior exposure to Python.

## Copyright & Disclaimer

© Copyright 2018 by Tutorials Point (I) Pvt. Ltd.

All the content and graphics published in this e-book are the property of Tutorials Point (I) Pvt. Ltd. The user of this e-book is prohibited to reuse, retain, copy, distribute or republish any contents or a part of contents of this e-book in any manner without written consent of the publisher.

We strive to update the contents of our website and tutorials as timely and as precisely as possible, however, the contents may contain inaccuracies or errors. Tutorials Point (I) Pvt. Ltd. provides no guarantee regarding the accuracy, timeliness or completeness of our website or its contents including this tutorial. If you discover any errors on our website or in this tutorial, please notify us at contact@tutorialspoint.com

# Table of Contents

# 1. Python Forensics – Introduction

Python is a general-purpose programming language with easy, readable code that can be easily understood by both professional developers as well as novice programmers. Python comprises of many useful libraries that can be used with any stack framework. Many laboratories rely on Python to build basic models for predictions and to run experiments. It also helps to control critical operational systems.

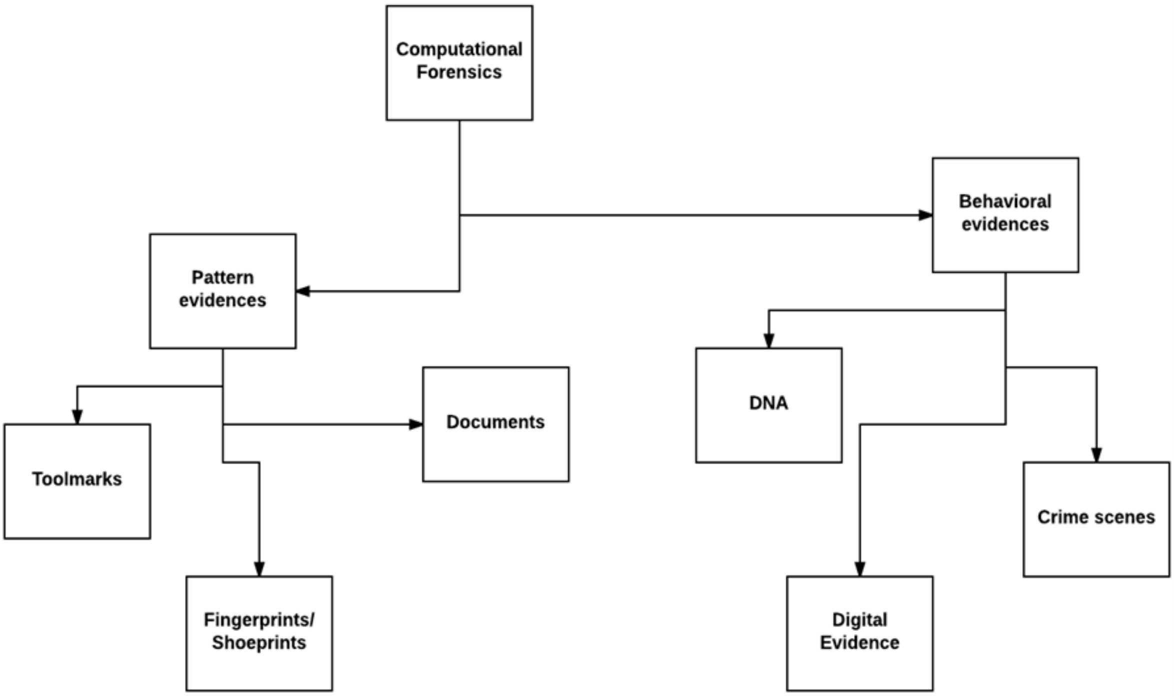
Python has built-in capabilities to support digital investigation and protect the integrity of evidence during an investigation. In this tutorial, we will explain the fundamental concepts of applying Python in digital or computation forensics.

## What is Computational Forensics?

Computational Forensics is an emerging research domain. It deals with solving forensic problems using digital methods. It uses computational science to study digital evidence.

Computation Forensics includes a broad range of subjects which has objects, substances, and processes investigated, mainly based on pattern evidence, such as toolmarks, fingerprints, shoeprints, documents etc., and also includes physiological and behavioral patterns, DNA, and digital evidence at crime scenes.

The following diagram shows the broad range of subjects covered under Computational Forensics.



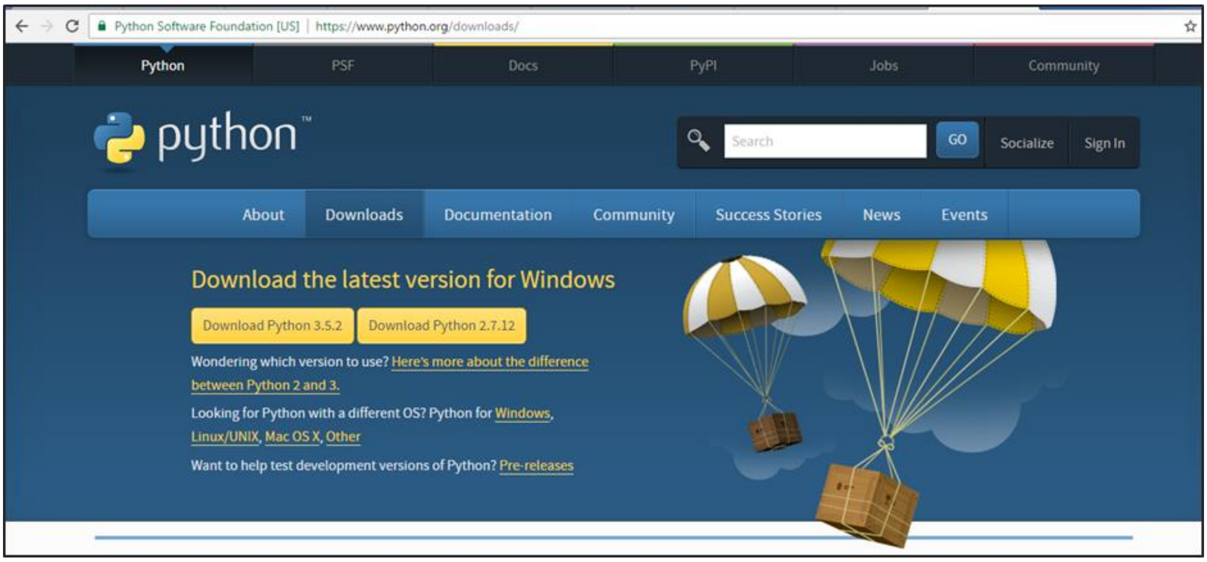
Computational forensics is implemented with the help of some algorithms. These algorithms are used for signal and image processing, computer vision and graphics. It also includes data mining, machine learning, and robotics.

Computational forensics involves diverse digital methods. The best solution to ease all digital methods in forensics is to use a general-purpose programming language like Python.

# 2. Python Forensics – Installation of Python

As we need Python for all the activities of computational forensics, let us move step by step and understand how to install it.

**Step 1**: Go to https://www.python.org/downloads/ and download the installation files of Python according to the Operating System you have on your system.



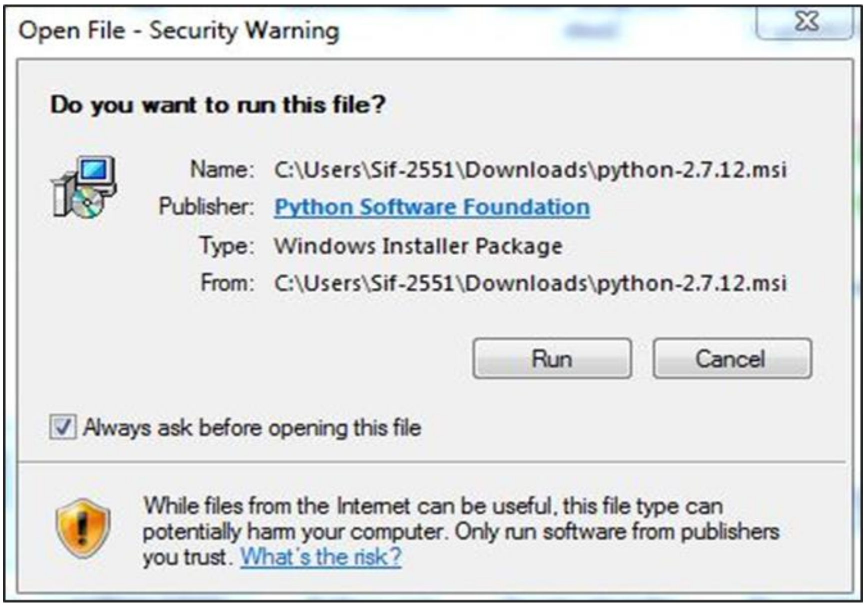
Download the latest version for Windows

Wondering which version to use? Here's more about the difference between Python 2 and 3.

Looking for Python with a different OS? Python for Windows, Linux/UNIX, Mac OS X, Other

Want to help test development versions of Python? Pre-releases

**Step 2**: After downloading the package/installer, click on the **exe** file to start the installation process.



Open File - Security Warning

Do you want to run this file?

Name: C:\Users\Sif-2551\Downloads\python-2.7.12.msi

Publisher: Python Software Foundation

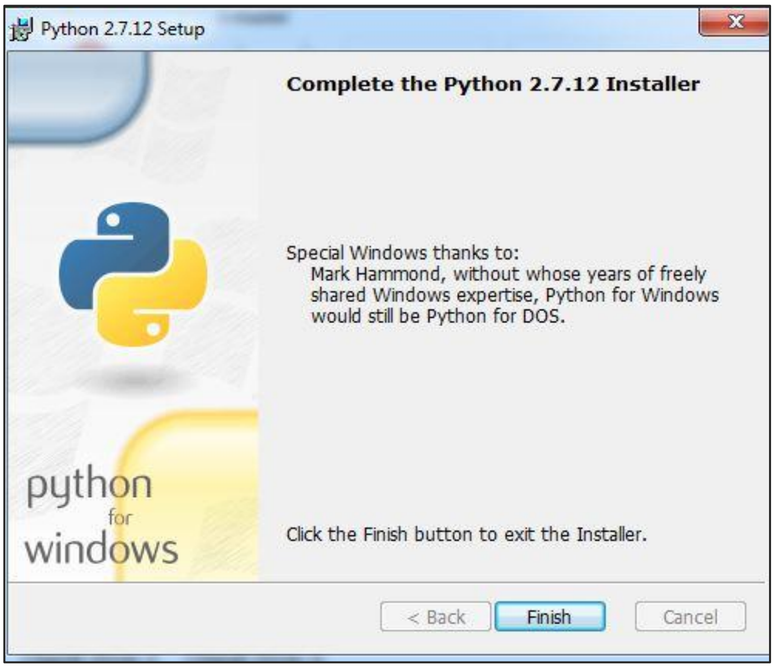
Type: Windows Installer Package

From: C:\Users\Sif-2551\Downloads\python-2.7.12.msi

☑ Always ask before opening this file

While files from the Internet can be useful, this file type can potentially harm your computer. Only run software from publishers you trust. What's the risk?

You will get to see the following screen after the installation is complete.

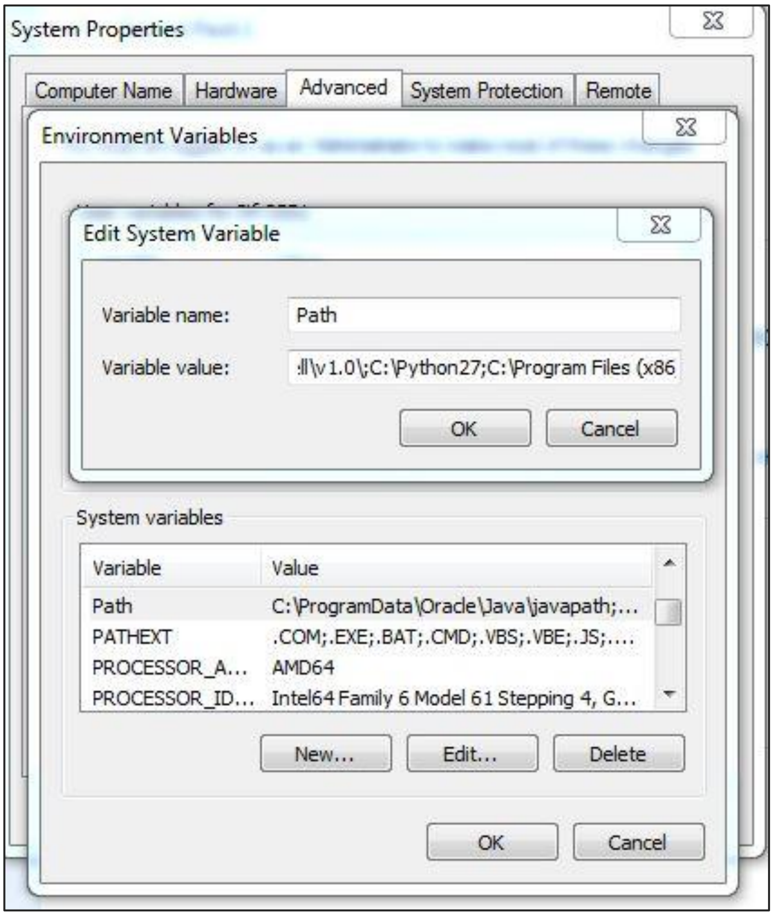


Complete the Python 2.7.12 Installer

Special Windows thanks to:
Mark Hammond, without whose years of freely shared Windows expertise, Python for Windows would still be Python for DOS.

Click the Finish button to exit the Installer.

**Step 3**: The next step is to set the environment variables of Python in your system.



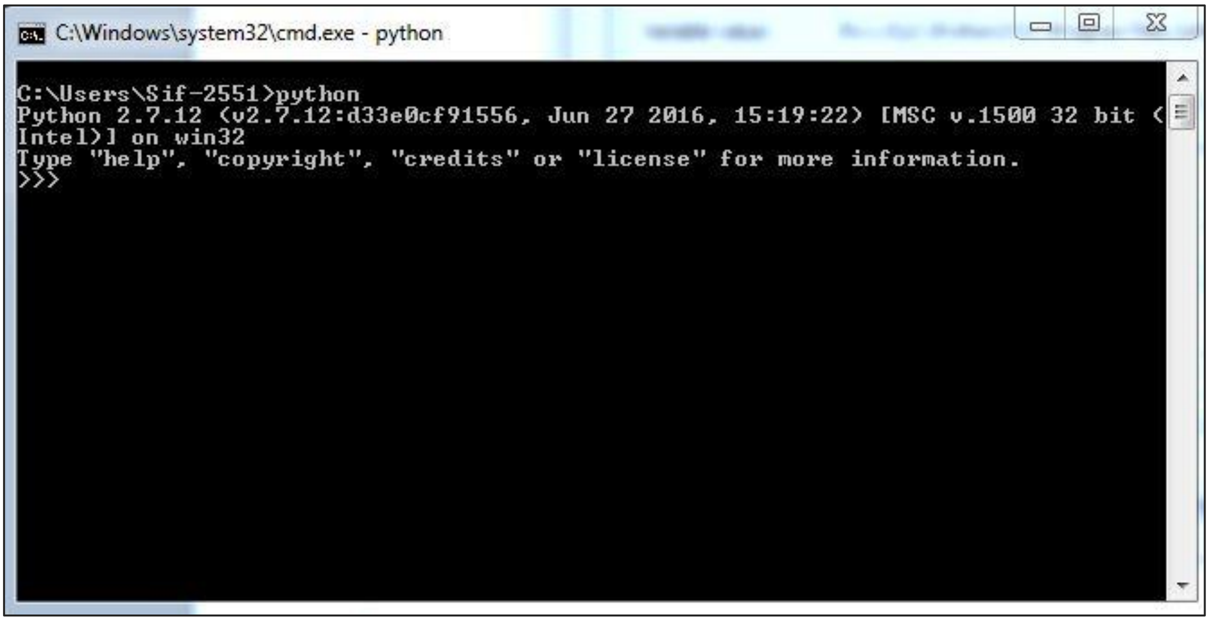
Variable name: Path

Variable value: :ll\v1.0\;C:\Python27;C:\Program Files (x86

| Variable | Value |
|---|---|
| Path | C:\ProgramData\Oracle\Java\javapath;... |
| PATHEXT | .COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.... |
| PROCESSOR_A... | AMD64 |
| PROCESSOR_ID... | Intel64 Family 6 Model 61 Stepping 4, G... |

**Step 4**: Once the environment variables are set, type the command "python" on the command prompt to verify whether the installation was successful or not.

If the installation was successful, then you will get the following output on the console.



```
C:\Users\Sif-2551>python
Python 2.7.12 (v2.7.12:d33e0cf91556, Jun 27 2016, 15:19:22) [MSC v.1500 32 bit (
Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

# 3. Python Forensics – Overview of Python

The codes written in Python look quite similar to the codes written in other conventional programming languages such as C or Pascal. It is also said that the syntax of Python is heavily borrowed from C. This includes many of the Python keywords which are similar to C language.

Python includes conditional and looping statements, which can be used to extract the data accurately for forensics. For flow control, it provides **if/else**, **while**, and a high-level **for** statement that loops over any "iterable" object.

```
if a < b:

    max = b

else:

    max = a
```

The major area where Python differs from other programming languages is in its use of **dynamic typing**. It uses variable names that refer to objects. These variables need not be declared.

## Data Types

Python includes a set of built-in data types such as strings, Boolean, numbers, etc. There are also immutable types, which means the values which cannot be changed during the execution.

Python also has compound built-in data types that includes **tuples** which are immutable arrays, **lists**, and **dictionaries** which are hash tables. All of them are used in digital forensics to store values while gathering evidence.

## Third-party Modules and Packages

Python supports groups of modules and/or packages which are also called **third-party modules** (related code grouped together in a single source file) used for organizing programs.

Python includes an extensive standard library, which is one of the main reasons for its popularity in computational forensics.

## Life Cycle of Python Code

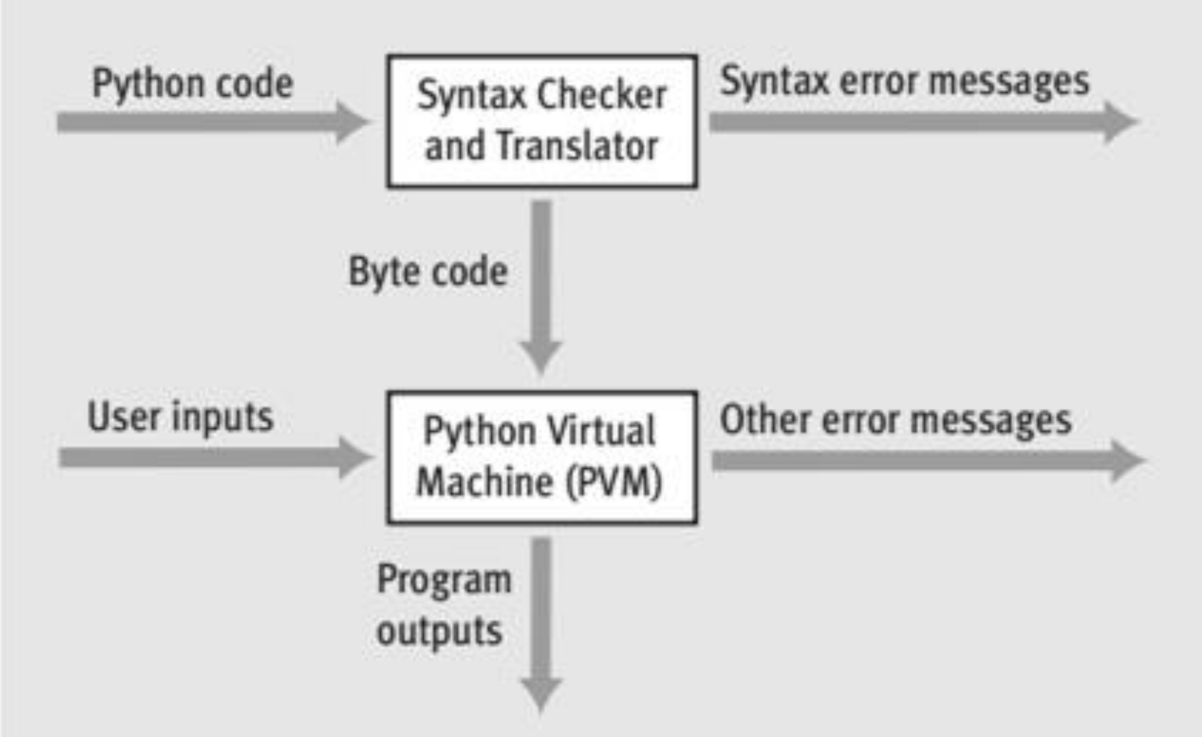
- At first, when you execute a Python code, the interpreter checks the code for syntax errors. If the interpreter discovers any syntax errors, then they are displayed immediately as error messages.

- If there are no syntax errors, then the code is compiled to produce a **bytecode** and sent to PVM (Python Virtual Machine).

- The PVM checks the bytecode for any runtime or logical errors. In case the PVM finds any runtime errors, then they are reported immediately as error messages.

- If the bytecode is error-free, then the code gets processed and you get its output.

The following illustration shows in a graphical manner how the Python code is first interpreted to produce a bytecode and how the bytecode gets processed by the PVM to produce the output.



End of ebook preview

If you liked what you saw…

Buy it from our store @ **https://store.tutorialspoint.com**