Python

Programming Language

#1

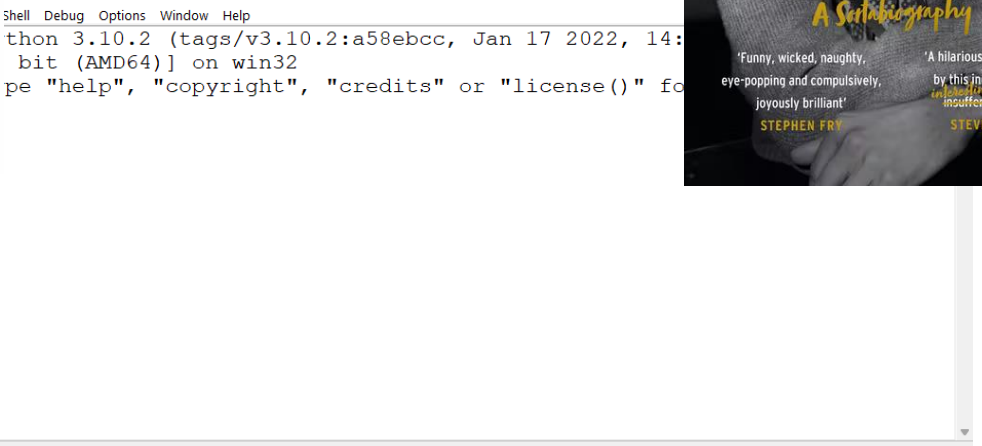
Serdar ARITAN

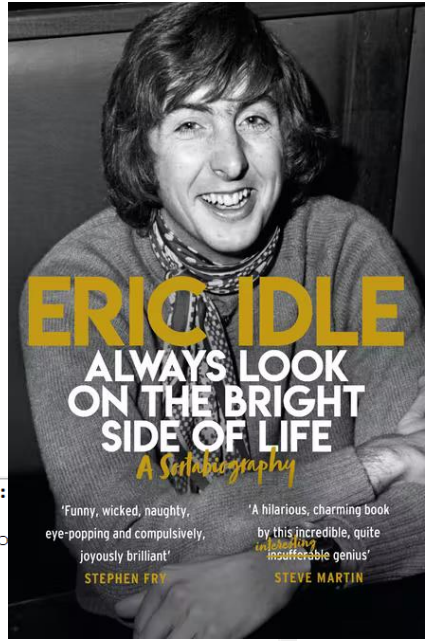
Biomechanics Research Group,
Faculty of Sports Sciences, and
Department of Computer Graphics
Hacettepe University, Ankara, Turkey

Snake logos and mascot not with standing, it's named after *Monty Python's Flying Circus*

Monty ***Python*** (sometimes known as The Pythons) was a British surreal comedy group that created Monty Python's Flying Circus, a British television comedy sketch show that first aired on the BBC on 5 October 1969.

PYTHON PROGRAMMING





Brief History of Python

- Invented in the Netherlands, early 90s by Guido van Rossum
- Named after Monty Python
- Open sourced from the beginning, man-aged by Python Software Foundation
- Considered a scripting language, but is much more
- Scalable, object oriented and functional from the beginning
- Used by Google from the beginning

Python's Benevolent Dictator For Life

"Python is an **experiment** in how much freedom program-mers need. Too much freedom and nobody can read another's code; too little and expressive-ness is endangered."

- Guido van Rossum

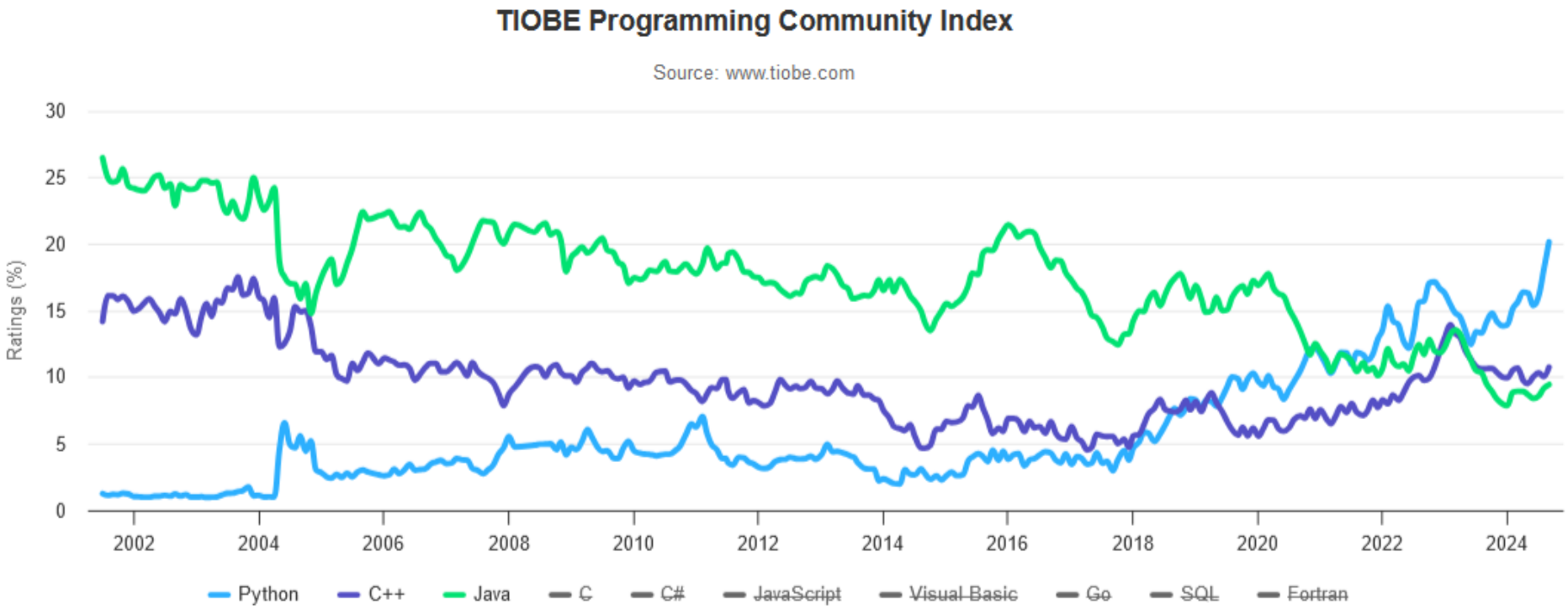
PYTHON PROGRAMMING

- TIOBE has been collecting data on programming language “popularity” for many years

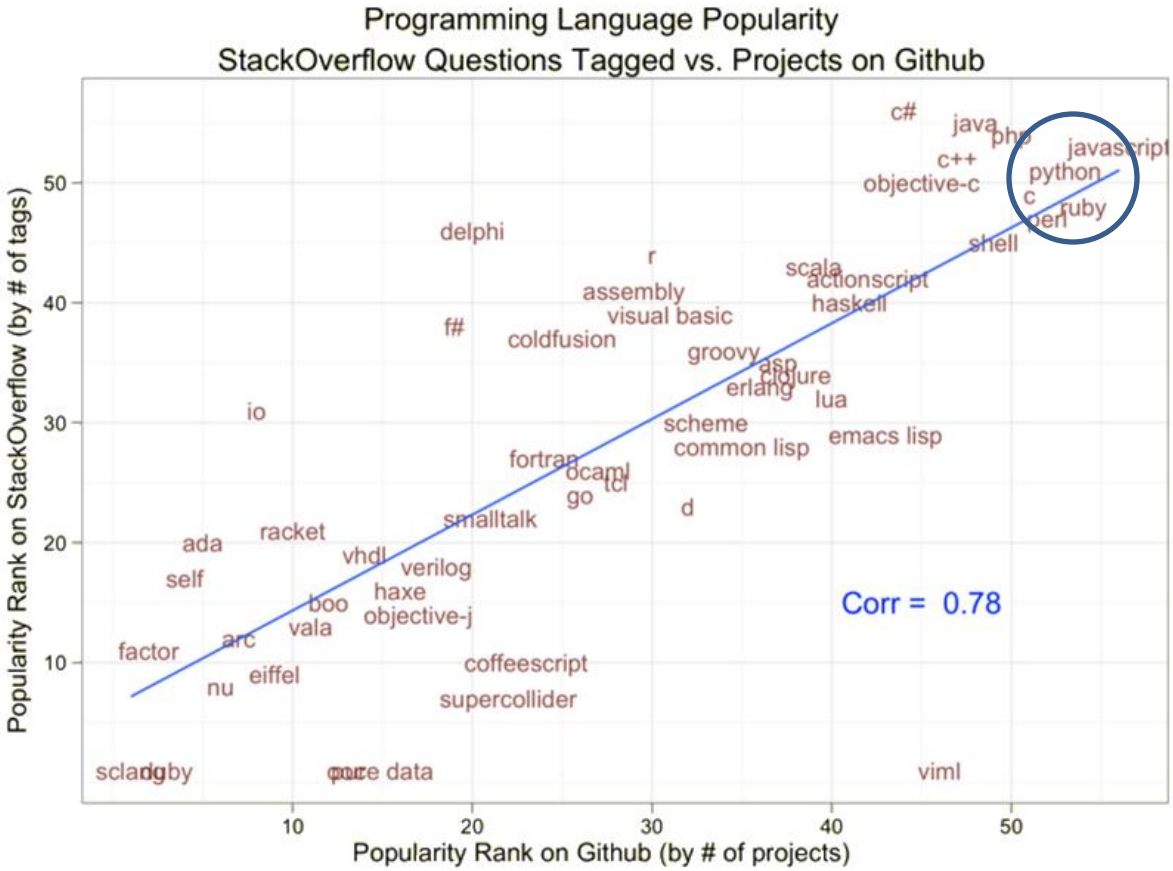
Year	Winner
2023	C#
2022	C++
2021	Python
2020	Python
2019	C
2018	Python
2017	C
2016	Go
2015	Java
2014	JavaScript
2013	Transact-SQL
2012	Objective-C
2011	Objective-C
2010	Python
2009	Go
2008	C
2007	Python
2006	Ruby

Sep 2024	Sep 2023	Change	Programming Language
1	1		Python
2	3	^	C++
3	4	^	Java
4	2	v	C
5	5		C#
6	6		JavaScript
7	7		Visual Basic
8	12	^^	Go
9	10	^	SQL
10	11	^	Fortran

Python's place in the Market



Python's place in the Market



Python's place in the Market

Distinct Features of Python

- Extensible (packages)
- Embeddable into applications
- Functional programming
- Object-Oriented programming
- Rapid Prototyping
- Great for readability and presentation
- White space is significant
- Low maintenance costs
- Exception handling
- Free (open source)

The core philosophy of the language is summarized by the document "PEP 20 (The Zen of Python)",

- **Beautiful** is better than **ugly**.
- **Explicit** is better than **implicit**.
- **Simple** is better than **complex**.
- **Complex** is better than **complicated**.
- **Readability** counts.

```
Try;
>>> import this
```

In 1994, Mike Gancarz, a member of Digital Equipment Corporation's Unix Engineering Group (UEG), published The UNIX Philosophy",

The nine basic "tenets" he claims to be important are

- Small is beautiful.
- Make each program do one thing well.
- Build a prototype as soon as possible.
- Choose portability over efficiency.
- Store data in flat text files.
- Use software leverage to your advantage.
- Use shell scripts to increase leverage and portability.
- Avoid captive user interfaces.
- Make every program a filter.

http://www.python.org/

Python | PSF | Docs | PyPI | Jobs | Community

python™

Search GO Socialize Sign In

About | Downloads | Documentation | Community | Success Stories | News | Events

```
# Python 3: For loop on a list
>>> list = [2, 4, 6, 8]
>>> sum = 0
>>> for num in list:
>>>     sum = sum + num
>>> print("The sum is:", sum)
The sum is: 20
```

All the Flow You'd Expect

Python knows the usual control flow statements that other languages speak — `if`, `for`, `while` and `range` — with some of its own twists, of course. More control flow tools in Python 3

1 2 3 4 5

Python is a programming language that lets you work quickly and integrate systems more effectively. >>> Learn More

https://docs.python.org/3/

3.12.6 | Quick search | Go

Python 3.12.6 documentation

Welcome! This is the official documentation for Python 3.12.6.

Documentation sections:

What's new in Python 3.12?
Or all "What's new" documents since Python 2.0

Tutorial
Start here: a tour of Python's syntax and features

Library reference
Standard library and builtins

Language reference
Syntax and language elements

Python setup and usage
How to install, configure, and use Python

Python HOWTOs
In-depth topic manuals

Installing Python modules
Third-party modules and PyPI.org

Distributing Python modules
Publishing modules for use by other people

Extending and embedding
For C/C++ programmers

Python's C API
C API reference

FAQs
Frequently asked questions (with answers!)

Deprecations
Deprecated functionality

Indices, glossary, and search:

Global module index
All modules and libraries

General index
All functions, classes, and terms

Glossary
Terms explained

Search page
Search this documentation

Complete table of contents
Lists all sections and subsections

3.12.6 | Hızlı Arama | Git

Python 3.12.6 belgelendirmesi

Hoş geldin! Bu sayfada, Python 3.12.6 için resmi dokümantasyonu bulabilirsin.

Documentation sections:

Python 3.12 sürümündeki yenilikler nelerdir?
Or all "What's new" documents since Python 2.0

Öğretici
Start here: a tour of Python's syntax and features

Library reference
Standard library and builtins

Language reference
Syntax and language elements

Python setup and usage
How to install, configure, and use Python

Python NASIL'ları
In-depth topic manuals

Installing Python modules
Third-party modules and PyPI.org

Distributing Python modules
Publishing modules for use by other people

Extending and embedding
For C/C++ programmers

Python's C API
C API reference

SSS (Sıkça Sorulan Sorular)
Frequently asked questions (with answers!)

Deprecations
Deprecated functionality

Indices, glossary, and search:

Global module index
All modules and libraries

General index
All functions, classes, and terms

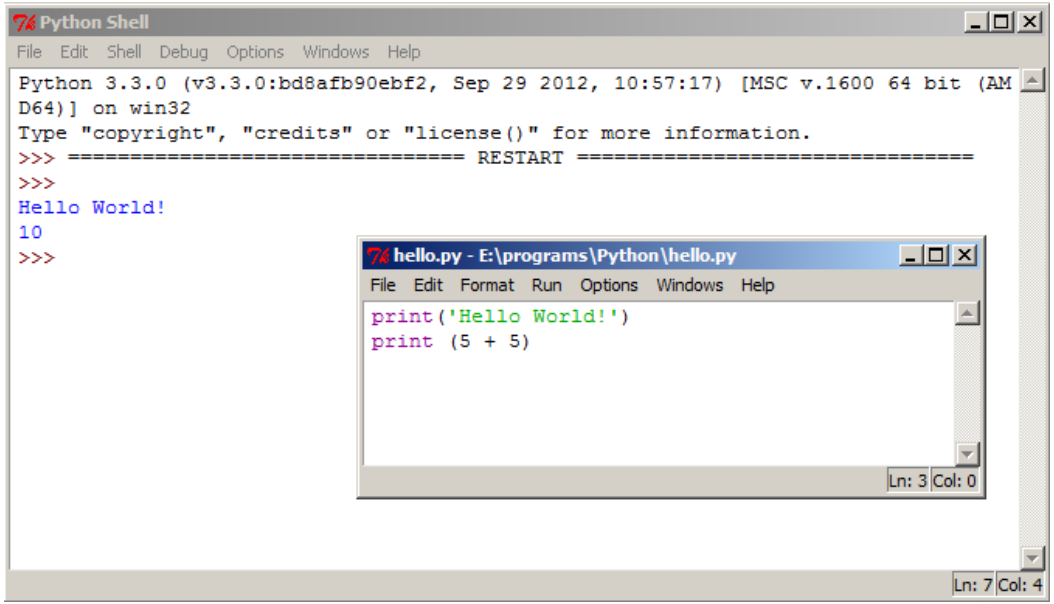
Sözlük
Terms explained

Arama sayfası
Search this documentation

Complete table of contents
Lists all sections and subsections

Programming Basics

- **code** or **source code**: The sequence of instructions in a program.
- **syntax**: The set of legal structures and commands that can be used in a particular programming language.
- **output**: The messages printed to the user by a program.
- **console**: The text box onto which output is printed.
  - Some source code editors pop up the console as an external window, and others contain their own console window.



Compiling and Interpreting

Both types of languages have their strengths and weaknesses. Usually, the decision to use an interpreted language is based on time restrictions on development or for ease of future changes to the program.

Compiled languages are all translated by running the source code through a compiler. This results in very efficient code that can be executed any number of times. The overhead for the translation is incurred just once, when the source is compiled; thereafter, it need only be loaded and executed. Interpreted languages, in contrast, must be parsed, interpreted, and executed each time the program is run, thereby greatly adding to the cost of running the program. For this reason, interpreted programs are usually less efficient than compiled programs.

Compiling and Interpreting

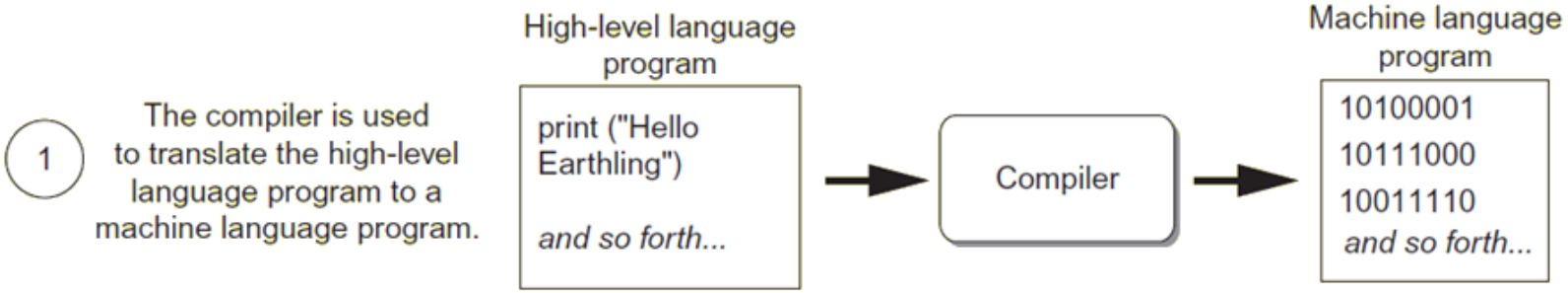
Python is an interpreted language, as opposed to a compiled one, though the distinction can be blurry because of the presence of the **bytecode** compiler. This means that source files can be run directly without explicitly creating an executable which is then run.

Python is not interpreted. The standard implementation compiles to **bytecode**, and then executes in a virtual machine.

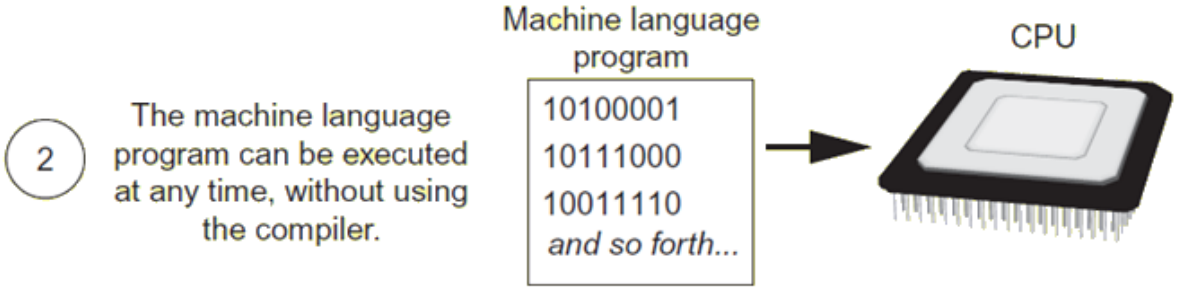
This is the approach taken by languages like **Java** and **C#**. The code is transformed into instructions for a "**virtual machine**". These instructions are then interpreted.

Bytecode, also known as p-code (portable code), is a form of instruction set designed for efficient execution by a software interpreter.

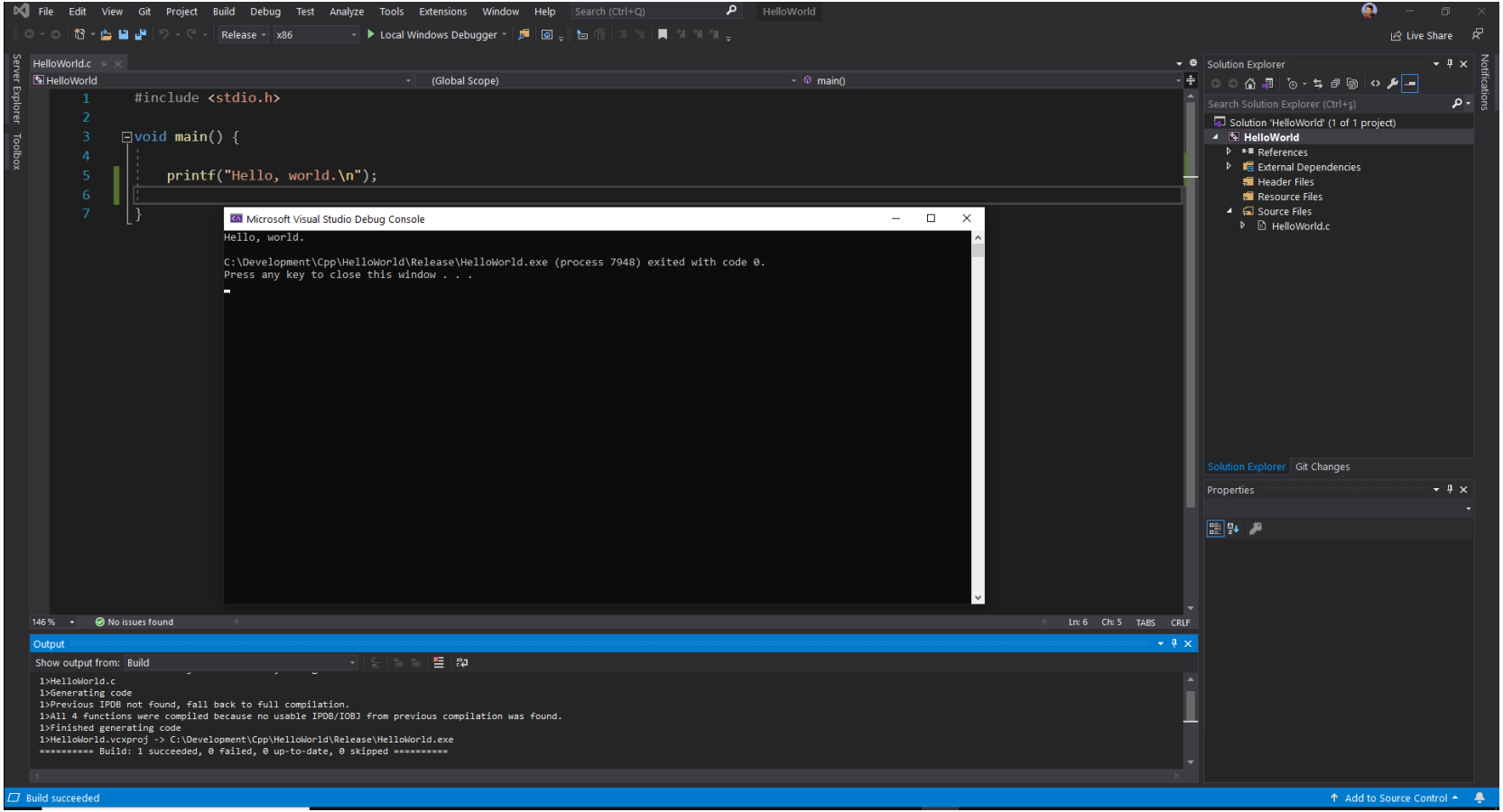
Compiling and Interpreting



- **Many languages require you to compile (translate) your program into a form that the machine understands.**



Compiling and Interpreting

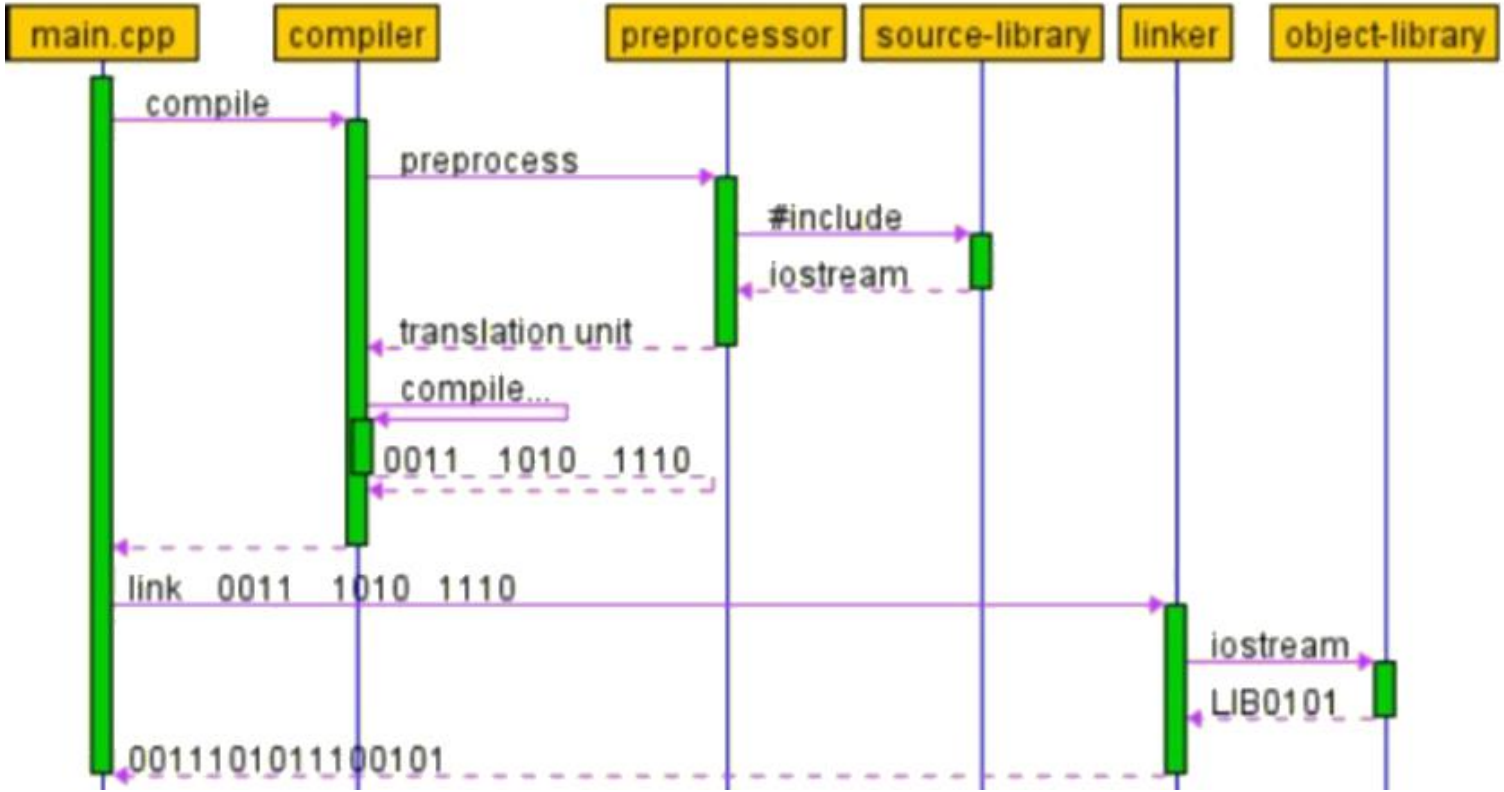


Compiling and Interpreting

Developer Command Prompt for VS 2019

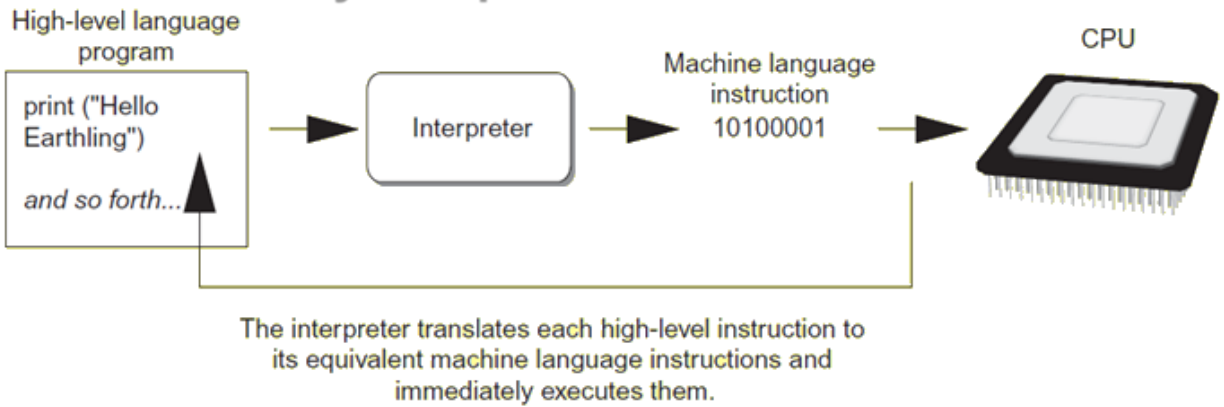
```
**********************************************************************
** Visual Studio 2019 Developer Command Prompt v16.8.4
** Copyright (c) 2020 Microsoft Corporation
**********************************************************************

C:\Program Files (x86)\Microsoft Visual Studio\2019\Enterprise>cl
Microsoft (R) C/C++ Optimizing Compiler Version 19.28.29336 for x86
Copyright (C) Microsoft Corporation.  All rights reserved.

usage: cl [ option... ] filename... [ /link linkoption... ]

C:\Program Files (x86)\Microsoft Visual Studio\2019\Enterprise>
```

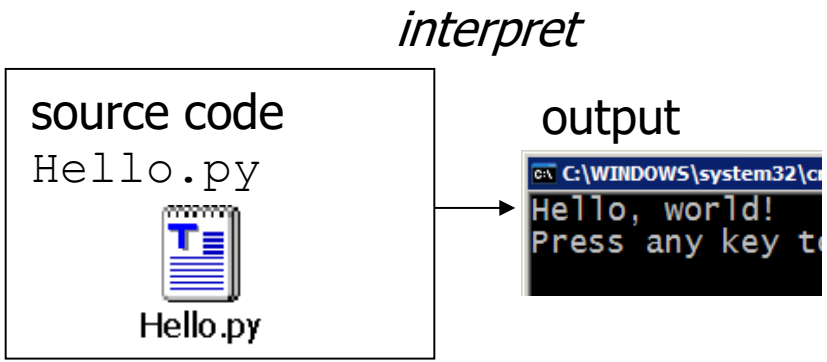
Compiling and Interpreting



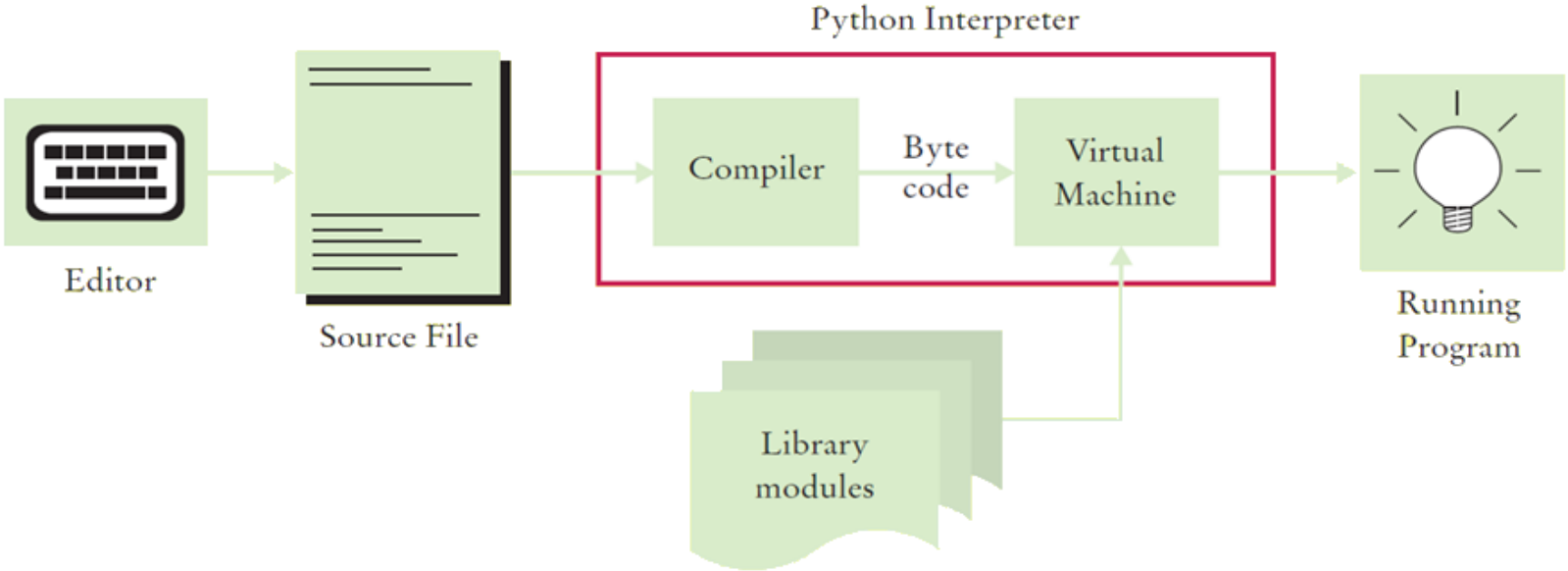
Compiling and Interpreting

- **Python is instead directly interpreted into machine instructions.**





Compiling and Interpreting



Your source code doesn't contain all the information that the virtual machine needs. For example, it does not contain the implementation of the print function. The virtual machine locates functions such as print in library modules. Generally, you need not be concerned with library modules.

Compiling and Interpreting

An interpreter is a translating program that translates and executes the statements in sequence. Unlike an assembler or compiler that produces machine code as output, which is then executed in a separate step, an interpreter translates a statement and then immediately executes the statement.

By definition, machine code differs from machine to machine. That is, each type of CPU has its own machine language that it understands. So how can we give each of you the experience of using machine language when you may be working on different machines? We solve that problem by using a virtual computer. A virtual computer is a hypothetical machine, in this case one that is designed to contain the important features of real computers that we want to illustrate.

Hello World

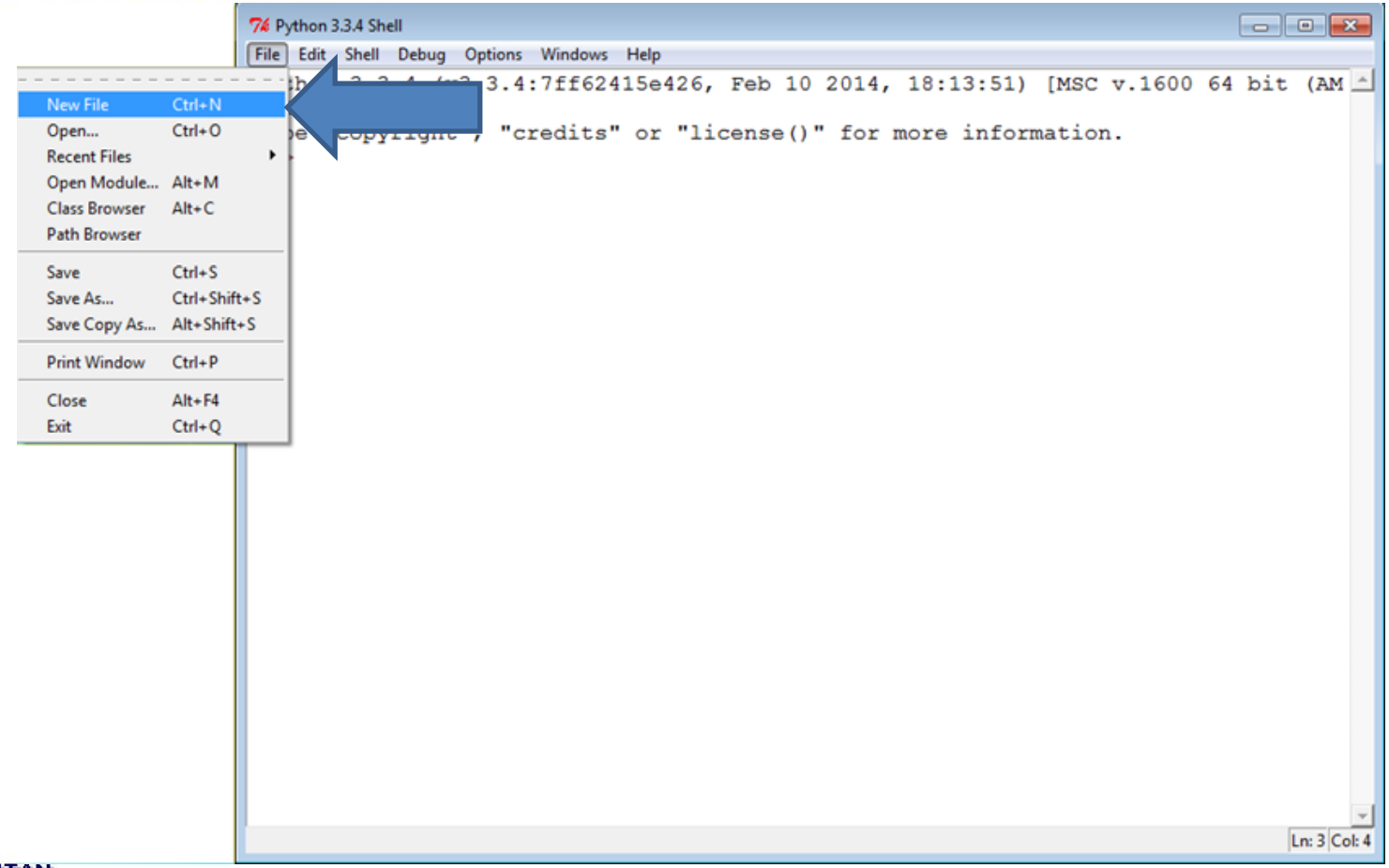
Python 3.3.4 Shell

File Edit Shell Debug Options Windows Help

```
Python 3.3.4 (v3.3.4:7ff62415e426, Feb 10 2014, 18:13:51) [MSC v.1600 64 bit (AM
D64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
```

Ln: 3 Col: 4

Hello World


Python 3.3.4 Shell
File Edit Shell Debug Options Windows Help
3.4:7ff62415e426, Feb 10 2014, 18:13:51) [MSC v.1600 64 bit (AM
"credits" or "license()" for more information.
New File Ctrl+N
Open... Ctrl+O
Recent Files
Open Module... Alt+M
Class Browser Alt+C
Path Browser
Save Ctrl+S
Save As... Ctrl+Shift+S
Save Copy As... Alt+Shift+S
Print Window Ctrl+P
Close Alt+F4
Exit Ctrl+Q
Ln: 3 Col: 4

Hello World

Python 3.3.4 Shell

File Edit Shell Debug Options Windows Help

```
Python 3.3.4 (v3.3.4:7ff62415e426, Feb 10 2014, 18:13:51) [MSC v.1600 64 bit (AM
D64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
```

Python 3.3.4: Untitled

File Edit Format Run Options Windows Help

```
print('Hello World')
```

Ln: 1 Col: 20

Hello World

Python 3.3.4 Shell

File Edit Shell Debug Options Windows Help

```
Python 3.3.4 (v3.3.4:7ff62415e426, Feb 10 2014, 18:13:51) [MSC v.1600 64 bit (AM
D64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
```

Python 3.3.4: Untitled

File Edit Format Run Options Windows Help

```
print('Hello World')
```

Save As

Save in: BCO 601 Python Programming

Recent Places, Desktop, Libraries, Computer, Network

Name	Date modified	Type
__pycache__	21-Oct-13 9:53 PM	File fol
2013Final	19-Jan-14 2:24 PM	File fol
2013MidTerm	18-Jan-14 7:19 PM	File fol
Blender	25-Nov-13 10:41 P...	File fol
Class	12-Nov-13 5:00 PM	File fol
cxfreeze	22-Oct-13 4:21 PM	File fol
GUI	01-Jan-14 2:57 PM	File fol
Loops and Function	17-Oct-13 11:47 AM	File fol
Network and Web Programming	17-Oct-13 5:27 PM	File fol
OpenGL	29-Dec-13 7:22 PM	File fol
Programming in Python 3 - A Complete Intr...	01-Jan-14 5:44 PM	File fol
Python in Practice Create Better Programs U...	29-Dec-13 9:08 PM	File fol
PythonFinal	21-Dec-13 2:40 PM	File fol

File name: helloWorld

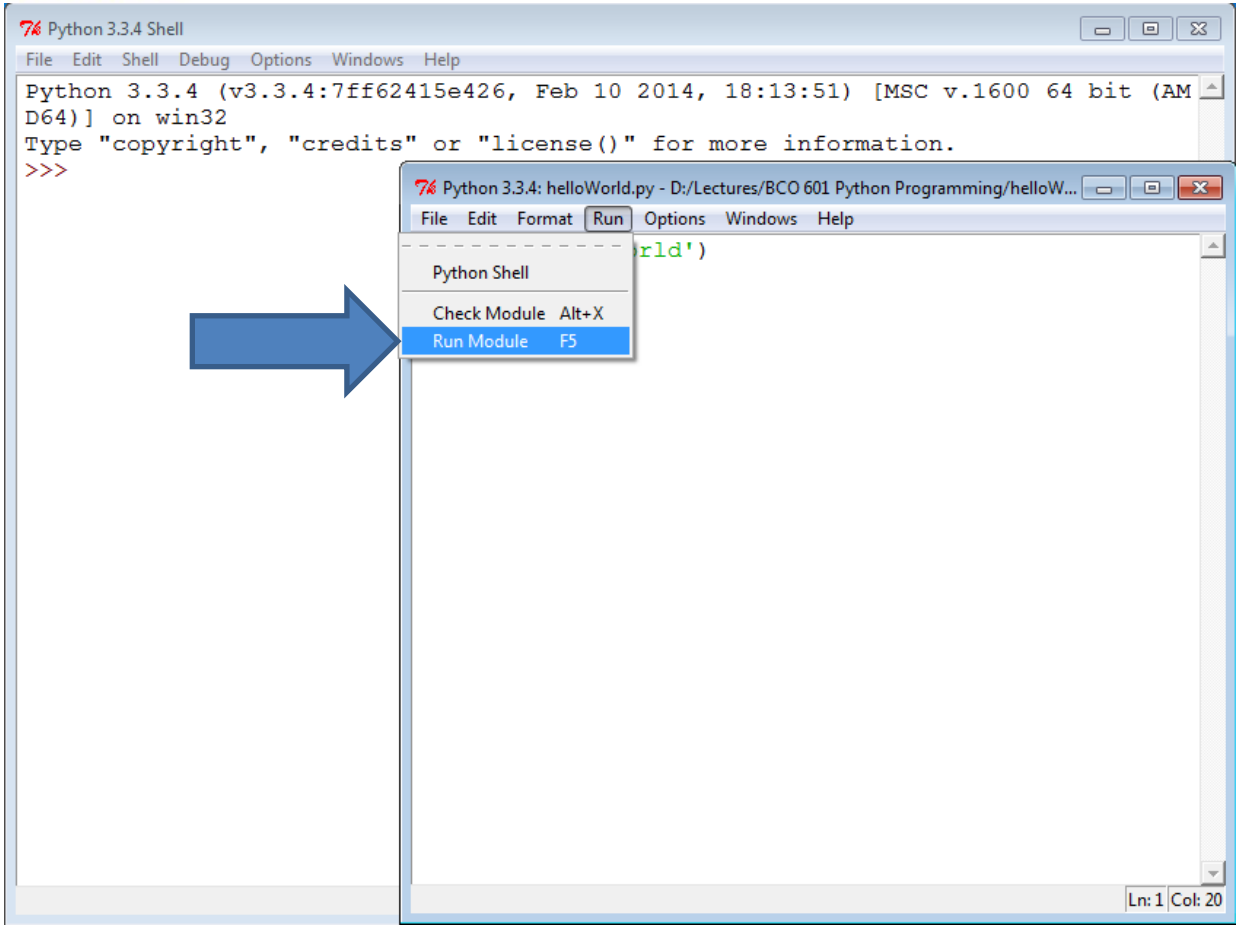
Save as type: Python files (*.py,*.pyw)

Save

Cancel

Ln: 1 Col: 20

Hello World


Python 3.3.4 Shell
File Edit Shell Debug Options Windows Help
Python 3.3.4 (v3.3.4:7ff62415e426, Feb 10 2014, 18:13:51) [MSC v.1600 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
Python 3.3.4: helloWorld.py - D:/Lectures/BCO 601 Python Programming/helloW...
File Edit Format Run Options Windows Help
rld')
Python Shell
Check Module Alt+X
Run Module F5
Ln: 1 Col: 20

Hello World

Python 3.3.4 Shell

File Edit Shell Debug Options Windows Help

```
Python 3.3.4 (v3.3.4:7ff62415e426, Feb 10 2014, 18:13:51) [MSC v.1600 64 bit (AM
D64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
Hello World
>>> |
```

Ln: 6 Col: 4

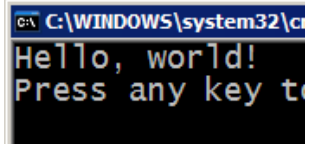
interpret

source code

HelloWorld.py

Hello.py

output



Google

serdar aritan

Sign in

All | Images | Videos | News | Maps | More | Tools | SafeSearch

About 23,800 results (0.36 seconds)

Serdar Aritan

Researcher

hacettepe.edu.tr

https://avesis.hacettepe.edu.tr › serd... · Translate this page

Dr.Öğr.Üyesi SERDAR ARITAN | AVESİS

Diğer E-posta: **serdar.aritan**@hacettepe.edu.tr; Web Sayfası: yunus.hacettepe.edu.tr/~**serdar.aritan**; İş Telefonu: +90 312 780 6893; İş Telefonu: +90 312...

Google Scholar

https://scholar.google.com › citations · Translate this page

Serdar Arıtan

Faculty Member, Hacettepe University - 598 tarafından alıntılandı - Biomechanics - Sports Biomechanics - Simulation - Computer Graphics

About

Education: Manchester Metropolitan University (1994–1998), MORE

Claim this knowledge panel

Feedback

Profiles

LinkedIn

CONTACT

Other Email	serdar.aritan@hacettepe.edu.tr
Web Page	yunus.hacettepe.edu.tr/~serdar.aritan
Office Phone	+90 312 780 6893
Office Phone	+90 312 297 6893
Fax Phone	+90 312 299 2167
Office	Spor Bilimleri Fakültesi Oda No:2-8
Address	Hacettepe Üniversitesi Spor Bilimleri Fakültesi Biyomekanik Araştırma Grubu 06810 Beytepe Ankara

File Edit View History Bookmarks Tools Help

Serdar Arıtan'nın Web Sayfası

https://yunus.hacettepe.edu.tr/~serdar.aritan/

Getting Started

Other Bookmarks

Ana Sayfa

Dersler

Araştırma

Serdar ARITAN

Biyomekanik Laboratuvarı Koordinatörü

1989 yılında Yıldız Teknik Üniversitesi'n den Endüstri Mühendisi olarak mezun oldum. 1994 yılında Hacettepe Üniversitesi'n den Spor Bilimleri Teknolojisi alanında Bilim Uzmanlığını derecesini aldıktan sonra 1998 yılında İngiltere'de the Manchester Metropolitan Üniversitesi'nde Biyomekanik Alanında Doktora öğrenimimi tamamladım.

1991 yılında göreve başladığım Hacettepe Üniversitesi'nde, 1999 yılından beri öğretim üyesi olarak çalışmaktayım. Ayrıca Biyomekanik Laboratuvarının araştırma yöneticiliği görevini yürütmekteyim. Spor Biyomekaniği, İnsan Hareket Analizi ve Benzeşim, Yumuşak Dokunun Biyomekanik Modellemesi, Görüntü İşleme ve Üç Boyutlu Yapılandırma Teknikleri ilgi alanlarımı kapsamaktadır. Yaklaşık 40 yıldır C/C++, MATLAB/Python ortamında yazılım geliştirmekteyim.

---

*ePosta : **serdar**{dot}**aritan**{at}**hacettepe**{dot}**edu**{dot}**tr***

Telefon : +90 312 297 6893

Belge Geçer : +90 312 299 2167

Güncelleme 26 Ağustos 2023

AVESİS

https://yunus.hacettepe.edu.tr/~serdar.aritan/courses.html

File Edit View History Bookmarks Tools Help

Serdar Aritan'nın Yürütülen Dersler

https://yunus.hacettepe.edu.tr/~serdar.aritan/courses.html

Getting Started

Other Bookmarks

Ana Sayfa

Dersler

Araştırma

Serdar ARITAN

2023-2024 Güz Yarıyılı (2 Ekim 2023 / 7 Ocak 2024)

- Bilişim Enstitüsü Lisansüstü Dersler

BCO 601 Python Programlama

BCO 602 Animasyon İçin Betik Diller

Sağlık Bilimleri Enstitüsü Lisansüstü Dersler

HAB 619 Spor Bilimlerinde Bilimsel Programlamaya Giriş

Genel sınavlar (8 Ocak 2024 / 21 Ocak 2024)

- Lisans Dersleri

ANR 305-02 Maç ve Yarışma Analizi **[Şube 2 Bireysel Sporlar için Hareket Analizi]**

ANR 413 Bilgisayar Programlama

Genel sınavlar (8 Ocak 2024 / 21 Ocak 2024)

Bütünleme sınavları (29 Ocak 2024 / 4 Şubat 2024)

Ders Takvimi

Salı, 29 Ağustos

File Edit View History Bookmarks Tools Help

BCO 601 Python Programlama

https://yunus.hacettepe.edu.tr/~serdar.aritan/bco601.html

Getting Started

Other Bookmarks

Ana Sayfa

Dersler

Araştırma

BCO 601 Python Programlama

Tartışılacak işlenecek konular

1. Hafta : Dersi tanıtım, Python dilini, sayıları ve işlemcileri tanıtma

2. Hafta : Python betimlemelerini test etme, değişkenler, veri türleri

3. Hafta : Karar verme ve Döngüler, TurtleGraphics

4. Hafta : Fonksiyonlar, TurtleGraphics

5. Hafta : Fonksiyonlar

6. Hafta : List, Dictionary, Tulip veri yapıları

7. Hafta : List ve Dictionary Comprehension, map, lambda and filter **< Ara Sınav >**

8. Hafta : Modüller, Hata Yakalama, Performans Ölçme

9. Hafta : Hata Bulucu Kullanımı, Kurallı İfadeler (RegEx)

10 Hafta : Nesneler ve Sınıflar I

11.Hafta : Nesneler ve Sınıflar II

12.Hafta : Python ile grafik arayüz uygulamaları I

13.Hafta : Python ile grafik arayüz uygulamaları II

14.Hafta : FTP, SMTP Kullanımı, HTTP sunucu uygulaması ve SQLite