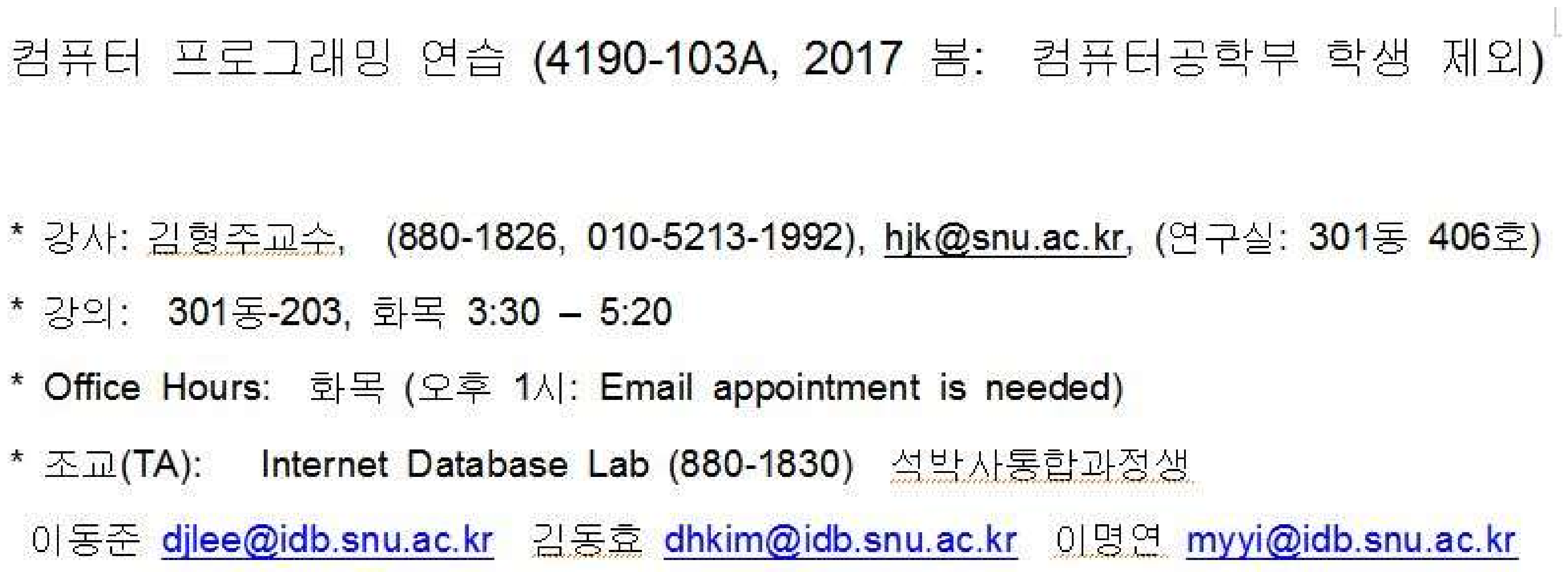
프로그래밍 연습

- Course Syllabus
- What is Programming?
- Learning Programming Languages
- Python Programming Environments



컴퓨터 프로그래밍 연습 (4190-103A, 2017 봄: 컴퓨터공학부 학생 제외)

* 강사: 김형주교수, (880-1826, 010-5213-1992), hjk@snu.ac.kr, (연구실: 301동 406호)

* 강의: 301동-203, 화목 3:30 – 5:20

* Office Hours: 화목 (오후 1시: Email appointment is needed)

* 조교(TA): Internet Database Lab (880-1830) 석박사통합과정생

이동준 djlee@idb.snu.ac.kr 김동효 dhkim@idb.snu.ac.kr 이명연 myyi@idb.snu.ac.kr

* Class Materials: Internet Database Lab Website: http://idb.snu.ac.kr

* Notebook PC를 가져와야 함

* 평가: 5 Programming (6% each), 2 midterms (20% each) and 1 final exam (30%)

(변경 가능함)

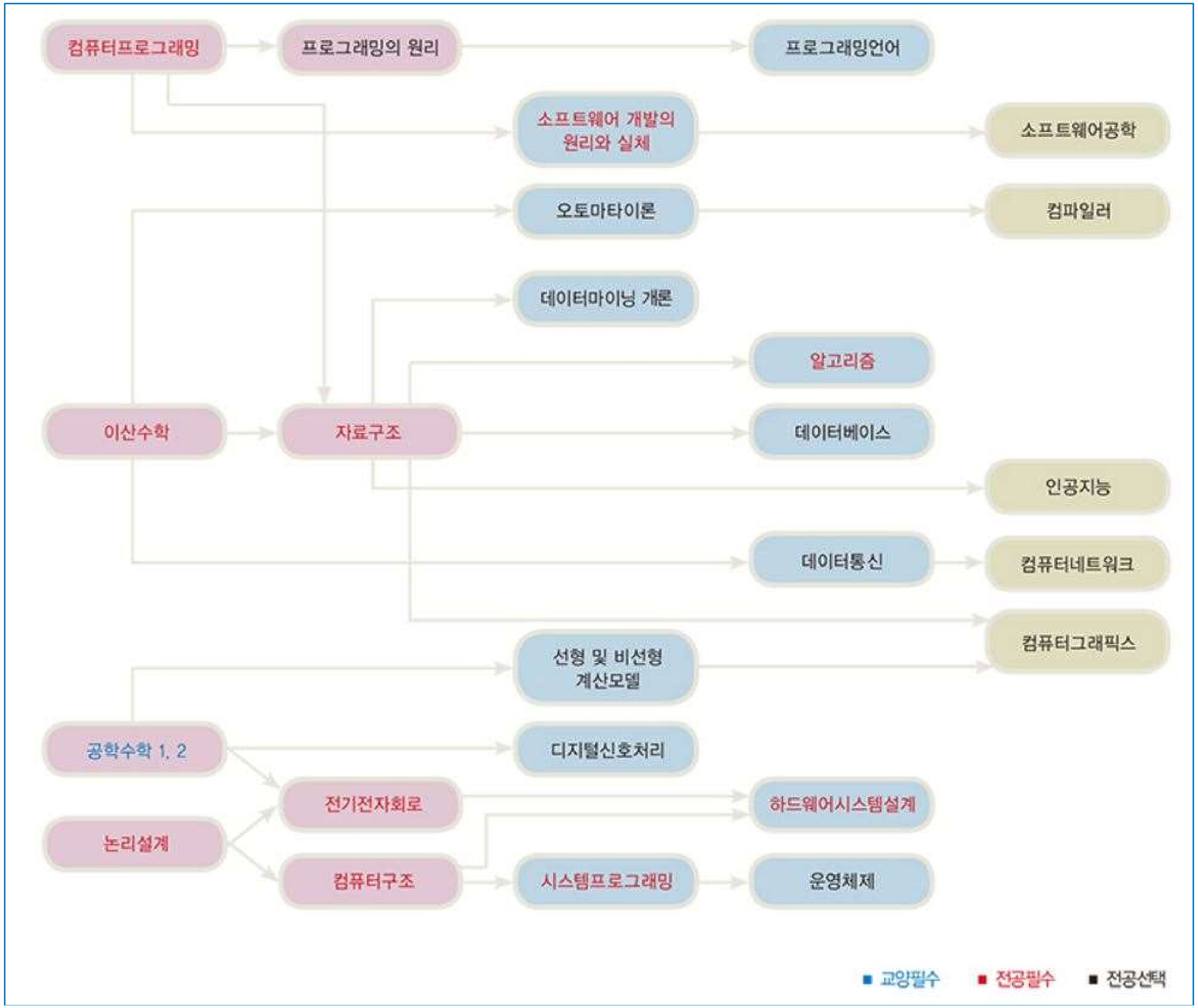
* 수업 카카오톡 방을 만들어서 운영할 예정!

Table of Contents

컴퓨터공학부 Curriculum Structure

컴퓨터공학의 개론및 실습

프로그래밍 연습



프로그래밍 연습

- Course Syllabus
- What is Programming?
- Learning Programming Languages
- Python Programming Environments

What is Programming?

- The Real World Problem: P
- Transform P into AP (Abstract Problem) through Abstraction
- Represent the AP using the given Programming Language
  - Using Basic Data Types, Advanced Data Types, User-defined Data Types
- Solve the AP with Algorithm based on Computational Thinking
  - Defining functions

Python Data Types과 연산

- Basic Data Types
  - Integer
  - Floating Number
  - Boolean
  - Character
  - String
  - List

우리가 익숙한 mathematical notation으로 연산

Ex: 3 + 4

- Advanced Data Types
  - Tuple
  - Dictionary
  - Set

특정 data type에 정의된 function들을 call해서 연산

```
Ex:  mySet = {3, 5, 9}
     myString.remove(5)
```

- User-Defined Data Types (Classes)
  - Student
  - Automobile
  - ……

특정 data type에 정의된 function들을 call해서 연산

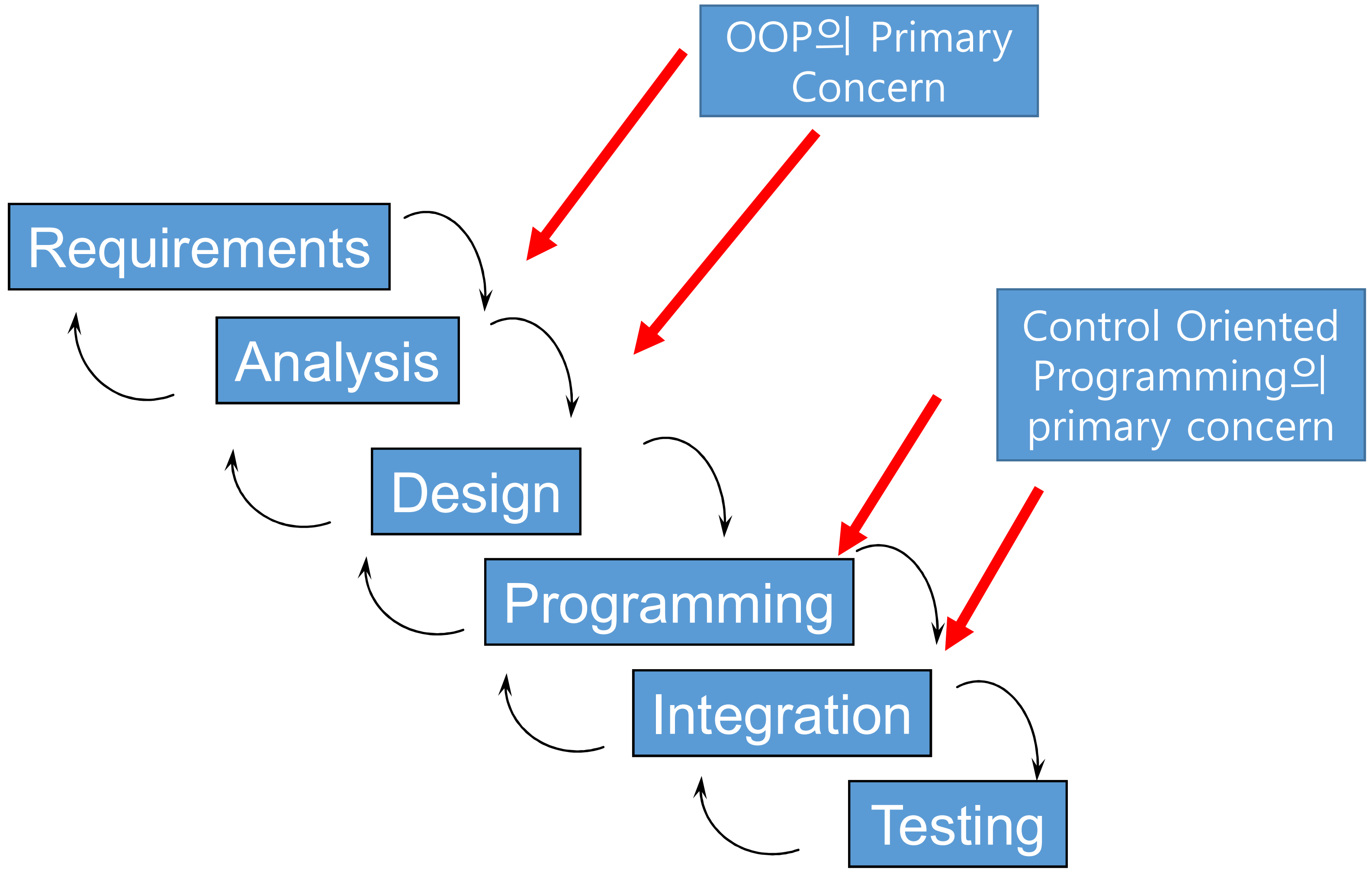
```
Ex:  myAuto = Automobile("GM", "2016", "5Door")
     myAuto.print()
```

- Library
  - Math
  - Random
  - ….

특정 library에 정의된 function들을 call해서 연산

```
Ex:  import math
     math.sqrt(4)
```

Waterfall SW Development Model



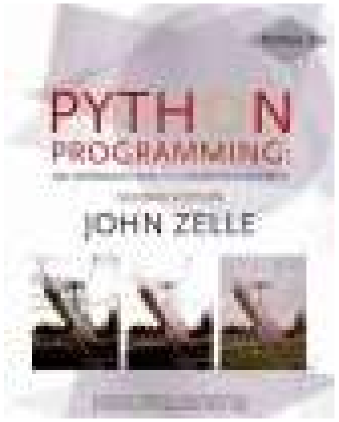
The Software Development Process: The WaterFall Model

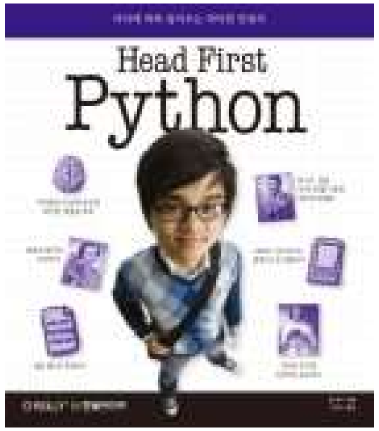
- Analyze the Problem
  - Figure out exactly the problem to be solved.
- Determine Specifications
  - Describe exactly what your program will do. (not **How**, but **What**)
  - Includes describing the inputs, outputs, and how they relate to one another.
- Create a Design
  - Formulate the overall structure of the program. (*how* of the program gets worked out)
  - You choose or develop your own algorithm that meets the specifications.
- Implement the Design (coding!)
  - Translate the design into a computer language.
- Test/Debug the Program
  - Try out your program to see if it worked.
  - Errors (Bugs) need to be located and fixed. This process is called debugging.
  - Your goal is to find errors, so try everything that might "break" your program!
- Maintain the Program
  - Continue developing the program in response to the needs of your users.
  - In the real world, most programs are never completely finished – they evolve over time.

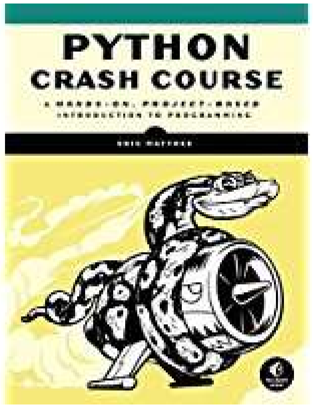
프로그래밍 연습

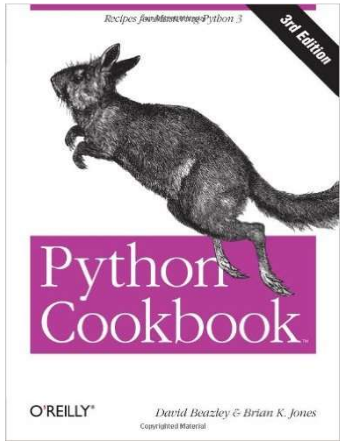
- Course Syllabus
- What is Programming?
- Learning Programming Languages
- Python Programming Environments

· Python Books









· Online Tutorials

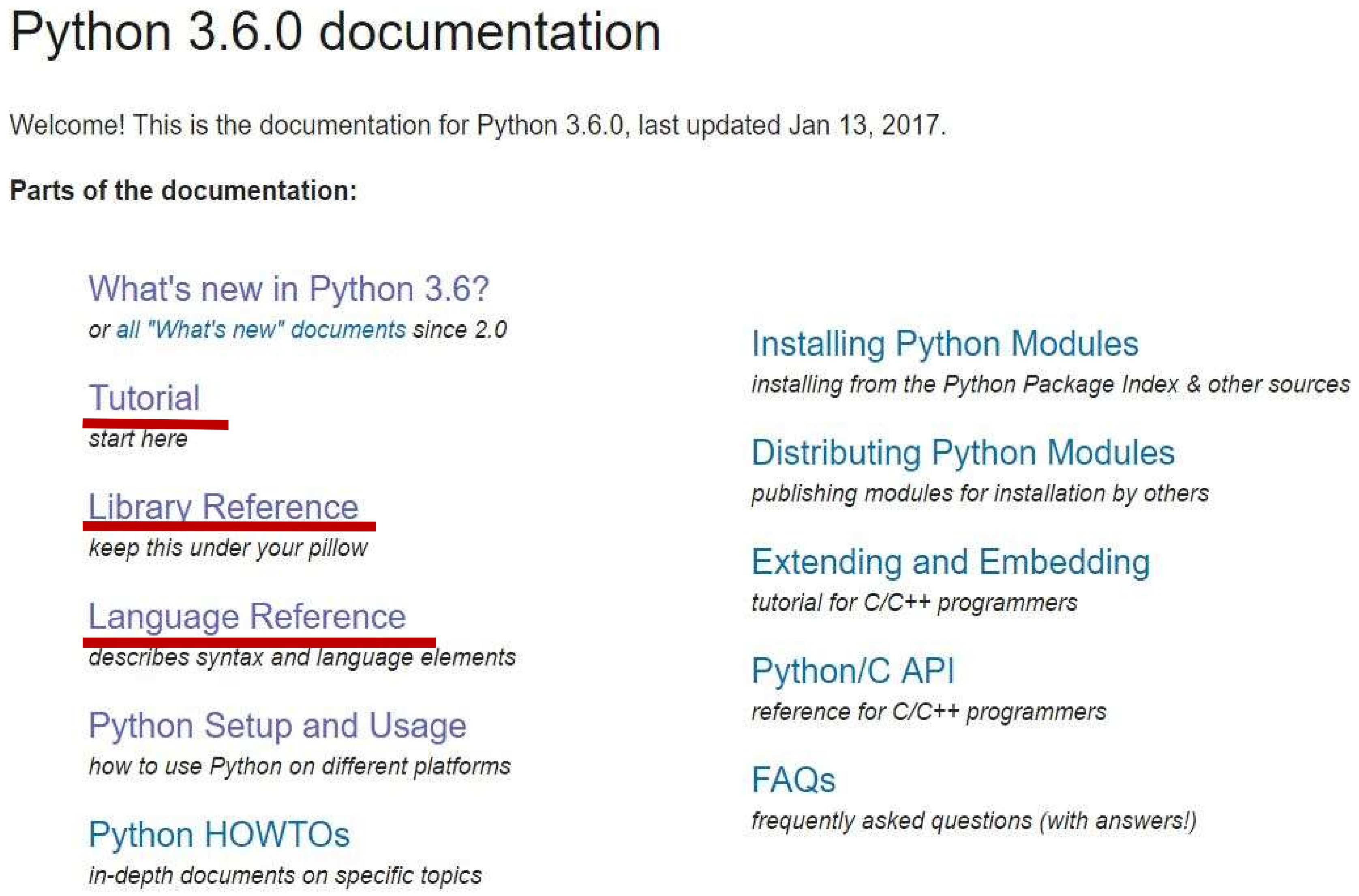
https://docs.python.org/3/tutorial/

http://www.python-course.eu/index.php

http://interactivepython.org/courselib/static/thinkcspy/index.html

· Just "class notes + Googling" is Enough!

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



Python 3.6.0 documentation

Welcome! This is the documentation for Python 3.6.0, last updated Jan 13, 2017.

Parts of the documentation:

What's new in Python 3.6?
or all "What's new" documents since 2.0

Tutorial
start here

Library Reference
keep this under your pillow

Language Reference
describes syntax and language elements

Python Setup and Usage
how to use Python on different platforms

Python HOWTOs
in-depth documents on specific topics

Installing Python Modules
installing from the Python Package Index & other sources

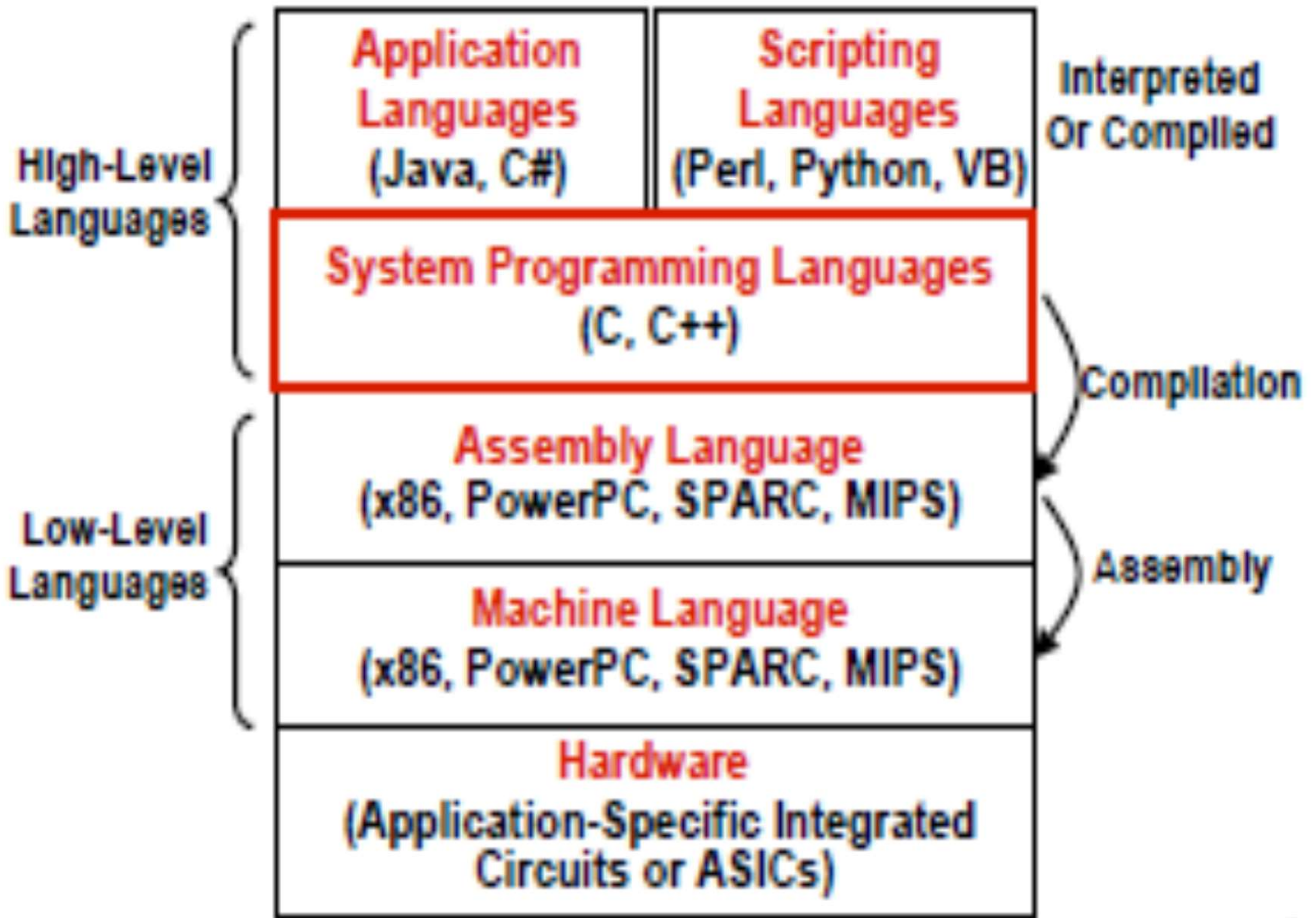
Distributing Python Modules
publishing modules for installation by others

Extending and Embedding
tutorial for C/C++ programmers

Python/C API
reference for C/C++ programmers

FAQs
frequently asked questions (with answers!)

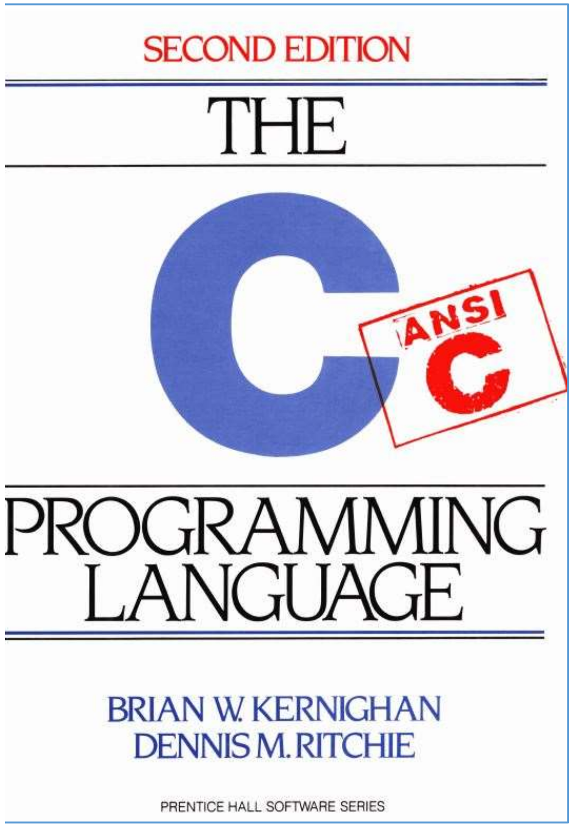
Programming Levels



High-Level Languages
Application Languages (Java, C#)
Scripting Languages (Perl, Python, VB)
Interpreted Or Compiled
System Programming Languages (C, C++)
Compilation
Low-Level Languages
Assembly Language (x86, PowerPC, SPARC, MIPS)
Assembly
Machine Language (x86, PowerPC, SPARC, MIPS)
Hardware (Application-Specific Integrated Circuits or ASICs)

1972
by Kerninghan and Richie

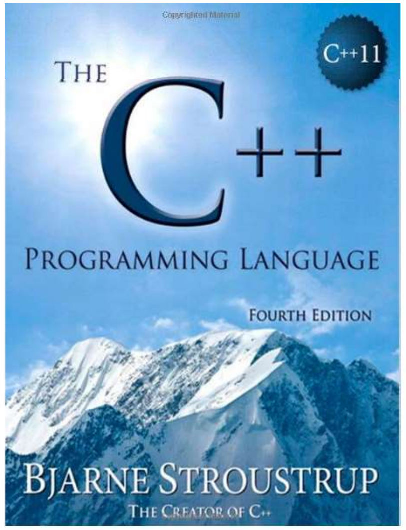
ISO/IEC:
C99 (1999), C11(2011)



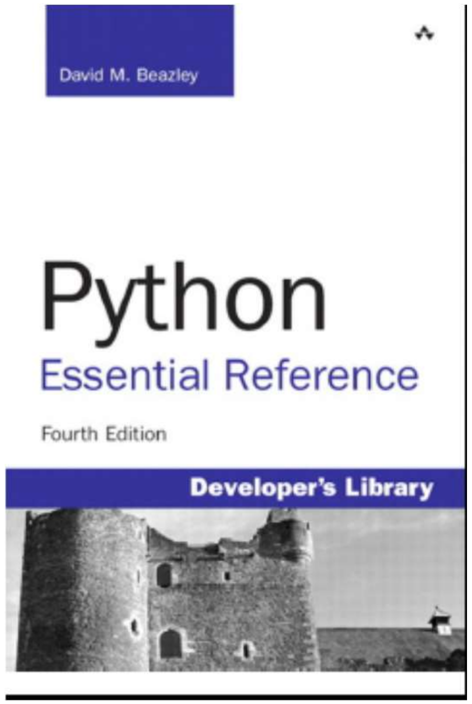
- Chapter 1. Tutorial Introduction
- Chapter 2. Types, Operators, and Expressions
- Chapter 3. Control Flow
- Chapter 4. Functions and Program Structure
- Chapter 5. Pointers and Arrays
- Chapter 6. Structures
- Chapter 7. Input and Output
- Chapter 8. The UNIX System Interface
- Appendix A. Reference Manual
- Appendix B. Standard Library
  - Input and Output: <stdio.h>
  - Character Class Tests: <ctype.h>
  - String Functions: <strings.h>
  - Mathematical Functions: <math.h>
  - Utility Functions: <stdlib.h>
  - Diagnosics: <assert.h>
  - Variable Argument Lists: <stdarg.h>
  - Non-local Jumps: <setjmp.h>
  - Signals: <signal.h>
  - Date and Time Functions: <time.h>
  - Implementation-defined Limits: <limits.h> and <float.h>

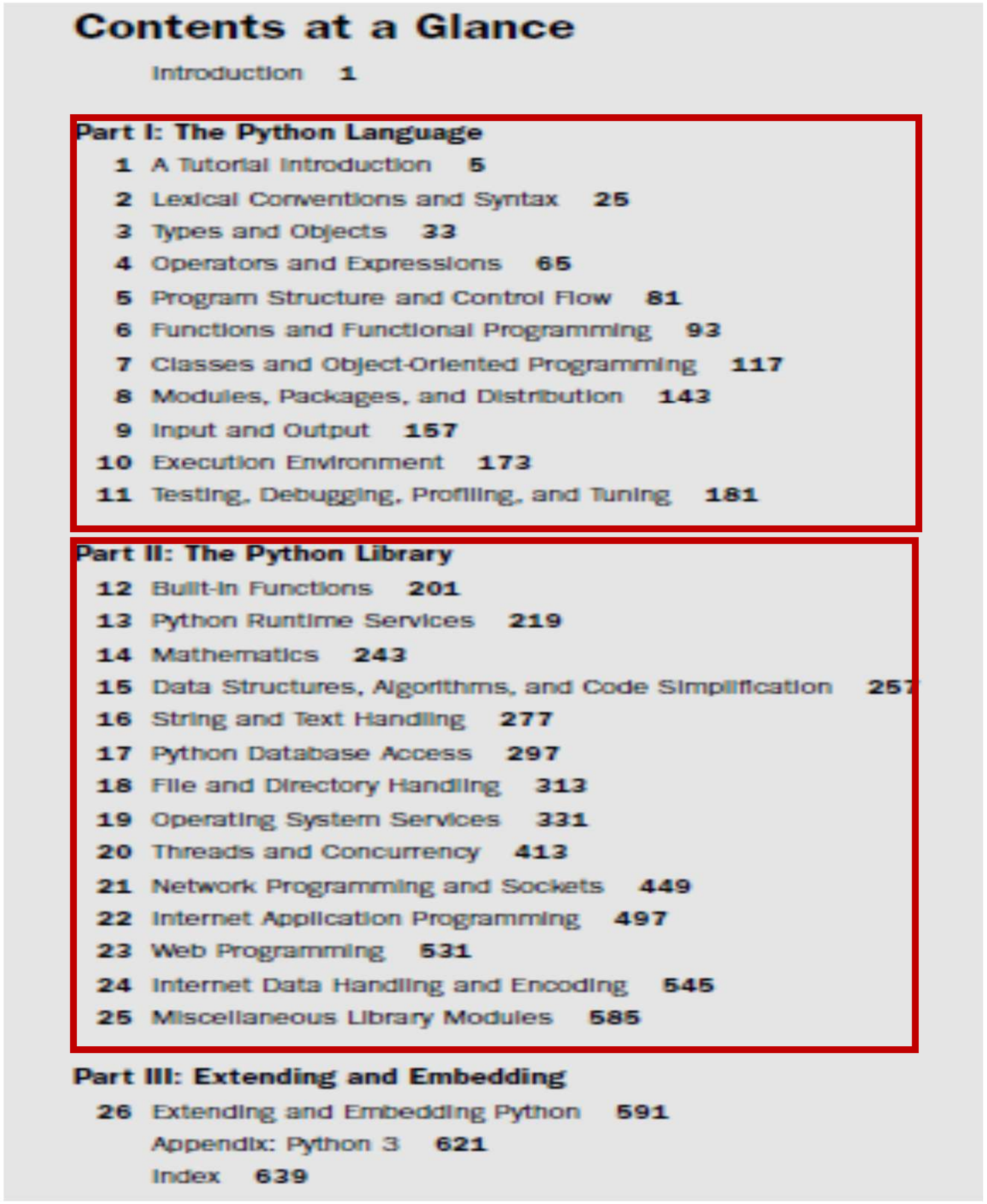
C++ 1983 by Bjarne Stroustrup

C++11 (2011)



At 1991
by Guido Rossum
Now owned by Python Org
Python 3.6 (2016)

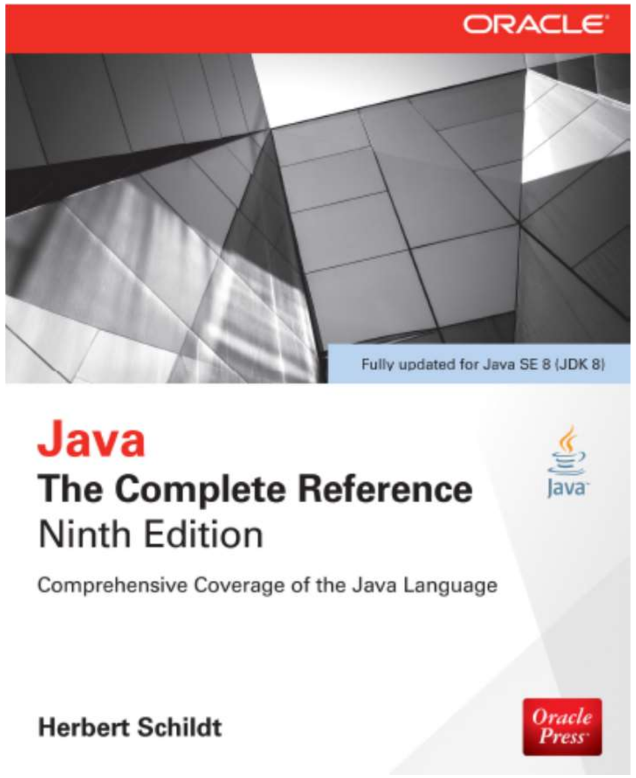


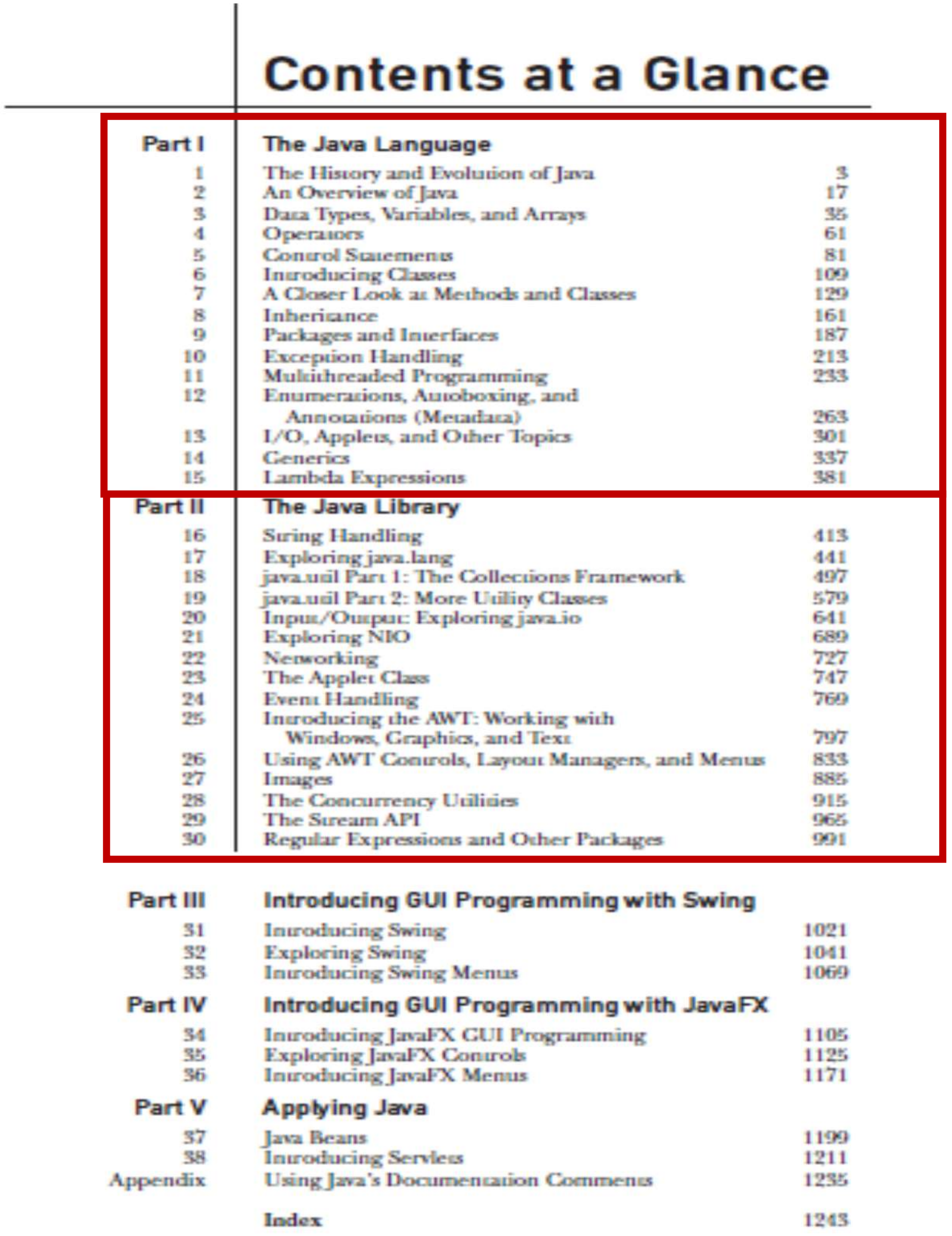


Contents at a Glance

At 1995 by James Gosling
Now owned by ORACLE

Java 8 (2014)


ORACLE
Fully updated for Java SE 8 (JDK 8)
Java
The Complete Reference
Ninth Edition
Comprehensive Coverage of the Java Language
Java
Herbert Schildt
Oracle Press



Contents at a Glance

Programming Languages: Compiler vs Interpreter

- Compiled programs generally run **faster** since the translation of the source code happens only once.
- Once program is compiled, it can be executed over and over without the source code or compiler.

- Interpreted programs are more **portable**, meaning the same program can run on a Intel PC and on a Mac as long as the interpreter is available
- Interpreted languages are part of a more **flexible** programming environment since they can be developed and run interactively

프로그래밍 연습

- Course Syllabus
- What is Programming?
- Learning Programming Languages
- Python Programming Environments

Why Python?

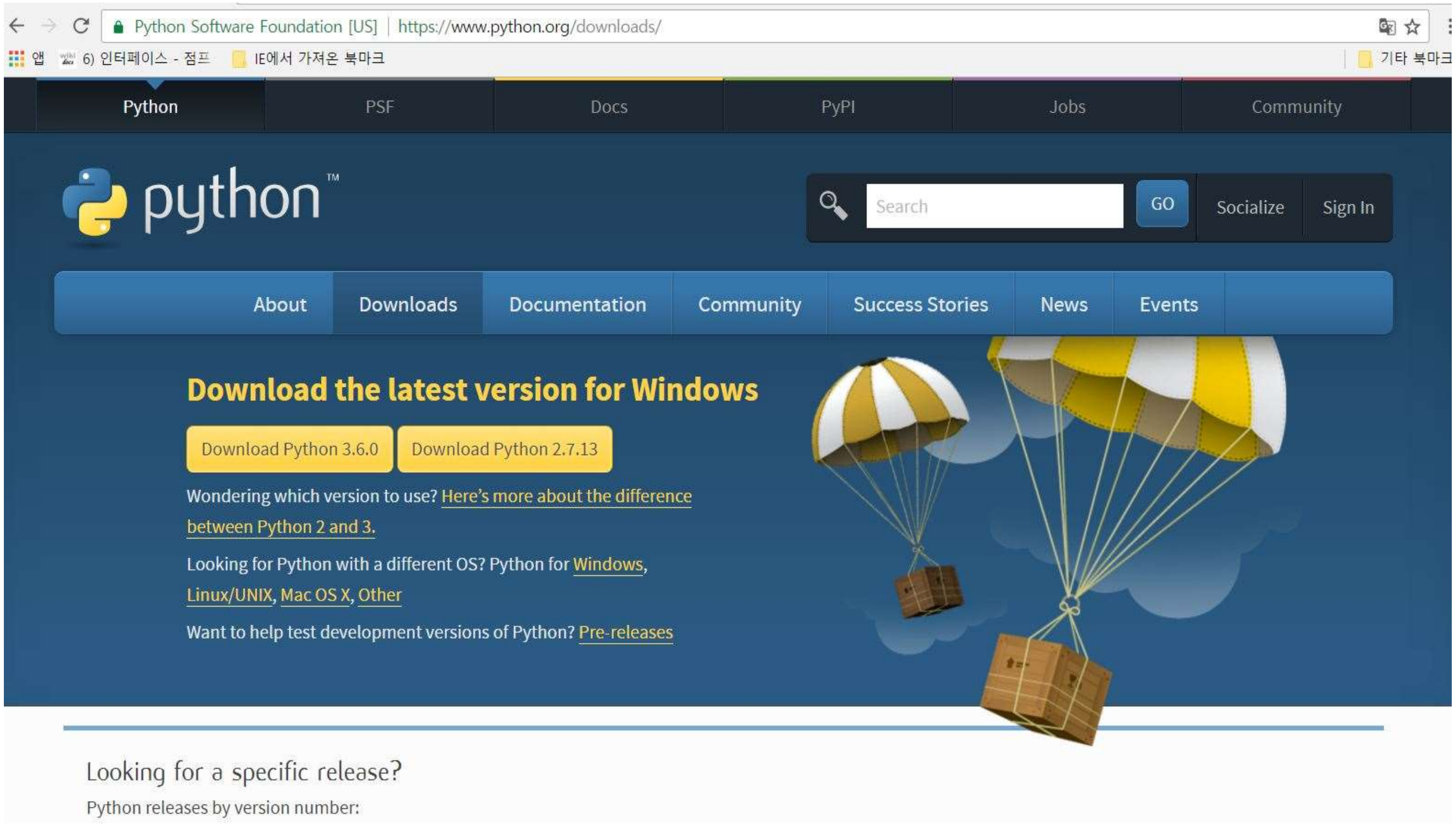
python™

- General-purpose, High-level, Scripting Language
- First appeared 1991, invented by Guido van Rossum
- Easy to use, easy to learn
- Widely used as
  - Scientific libraries
  - Web Frameworks
  - Backend Frameworks
  - UI Frameworks
  - Graphic Frameworks
  - Data Mining Frameworks
  - And many others…

Why Python?: Advantages vs Disadvantages

- Advantages
  - Fast prototype testing
  - Minimal development effort
  - High readability
- Disadvantages
  - As a scripting language, it requires a interpreter
  - Performance might be an issue (memory, computation)
  - Weak typing might be harder to debug

Python Installation on your PC


Python Software Foundation [US] | https://www.python.org/downloads/
앱 6) 인터페이스 - 점프 IE에서 가져온 북마크 기타 북마크
Python PSF Docs PyPI Jobs Community
python™
Search GO Socialize Sign In
About Downloads Documentation Community Success Stories News Events

Download the latest version for Windows

Download Python 3.6.0 Download Python 2.7.13

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

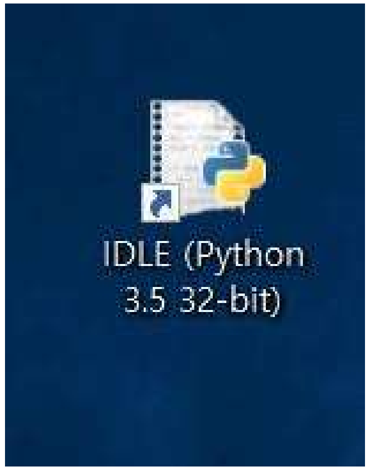
Looking for Python with a different OS? Python for Windows, Linux/UNIX, Mac OS X, Other

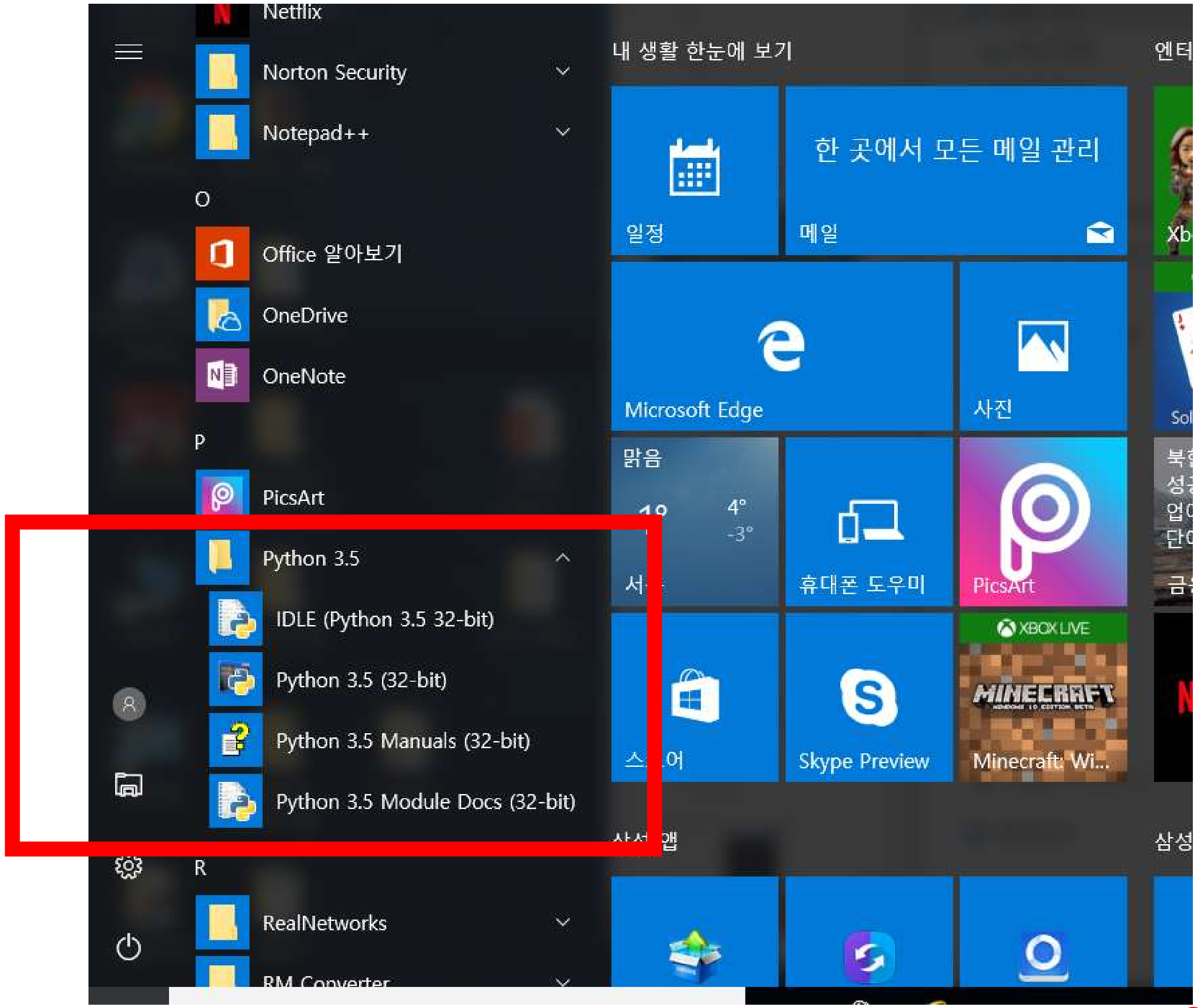
Want to help test development versions of Python? Pre-releases

Looking for a specific release?

Python releases by version number:

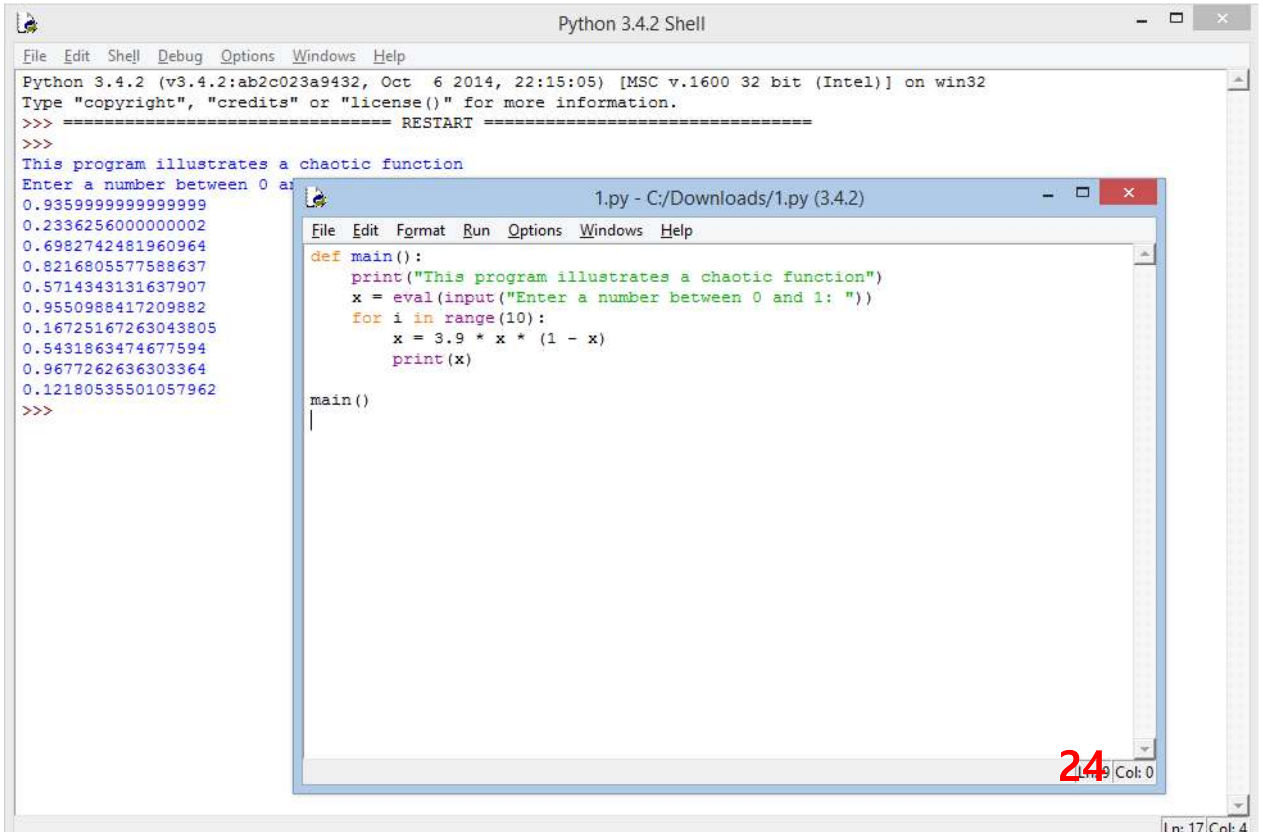
After Installing Python





Ways to Use Python: Python IDLE [1/6]

- Easy to use Interactive Development Environment (IDE)
- De-facto standard IDE for learning Python
- Provides simple debugging tool
- Provides simple **code completion**


Python 3.4.2 Shell
File Edit Shell Debug Options Windows Help
1.py - C:/Downloads/1.py (3.4.2)
File Edit Format Run Options Windows Help
Ln: 9 Col: 0
Ln: 17 Col: 4

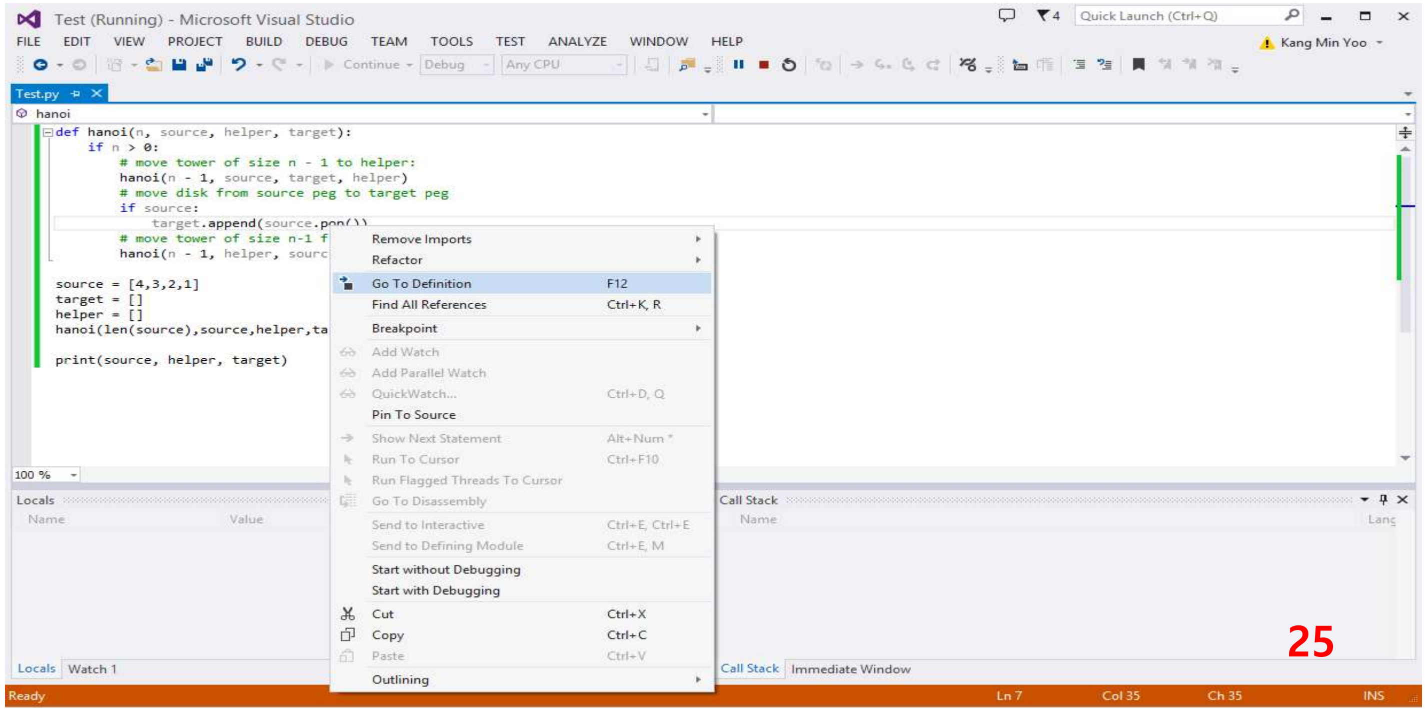
```
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct  6 2014, 22:15:05) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
>>>
This program illustrates a chaotic function
Enter a number between 0 ar
0.9359999999999999
0.2336256000000002
0.6982742481960964
0.8216805577588637
0.5714343131637907
0.9550988417209882
0.16725167263043805
0.5431863474677594
0.9677262636303364
0.12180535501057962
>>>
```

```python
def main():
    print("This program illustrates a chaotic function")
    x = eval(input("Enter a number between 0 and 1: "))
    for i in range(10):
        x = 3.9 * x * (1 - x)
        print(x)

main()
```

Ways to Use Python: Python Tools for Visual Studio [2/6]

https://www.visualstudio.com/en-us/features/python-vs.aspx

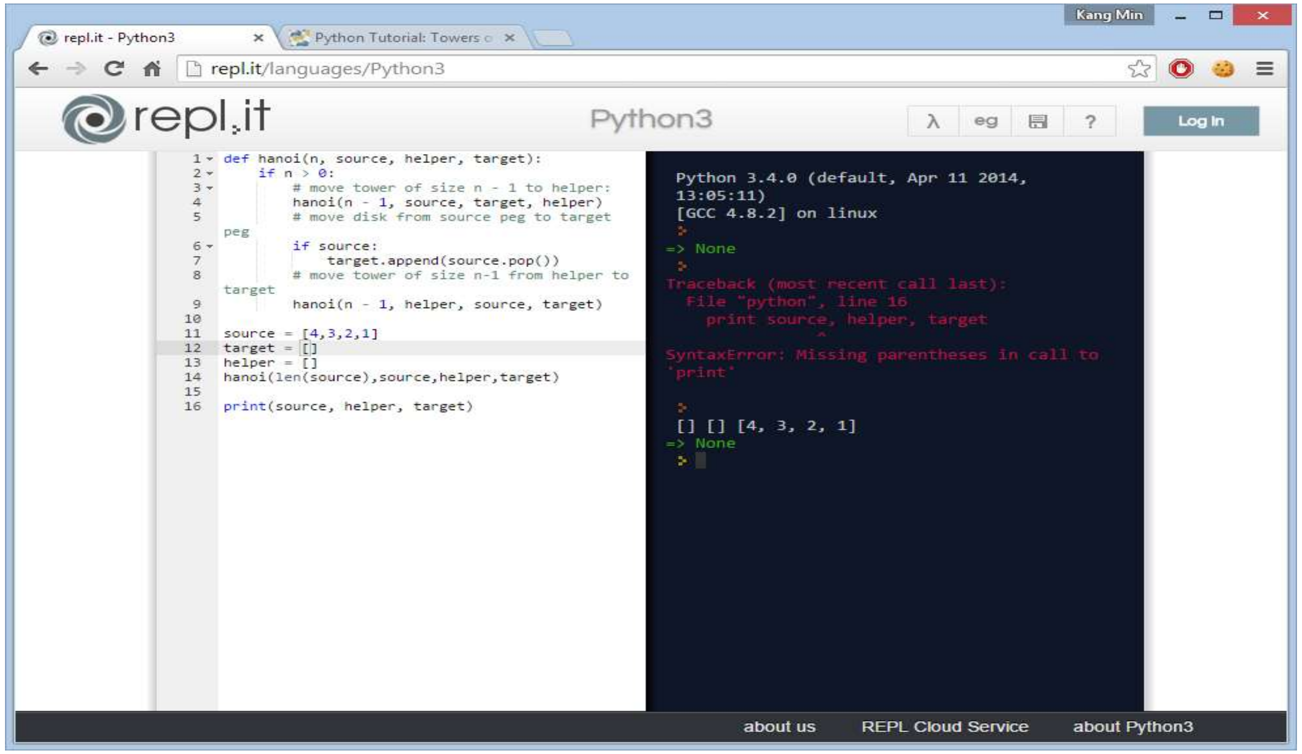
- Has a steep learning curve, but very useful if used right
- Might be difficult for beginners in programming
- Supports most visual studio features
  Finding references // Code completion // Syntax checking
  Simple semantics checking // Full stack Debugging
  Inspection // And many others…


Test (Running) - Microsoft Visual Studio

```
def hanoi(n, source, helper, target):
    if n > 0:
        # move tower of size n - 1 to helper:
        hanoi(n - 1, source, target, helper)
        # move disk from source peg to target peg
        if source:
            target.append(source.pop())
        # move tower of size n-1 f
        hanoi(n - 1, helper, sourc

source = [4,3,2,1]
target = []
helper = []
hanoi(len(source),source,helper,ta

print(source, helper, target)
```

Ways to Use Python: http://repl.it/ [3/6]

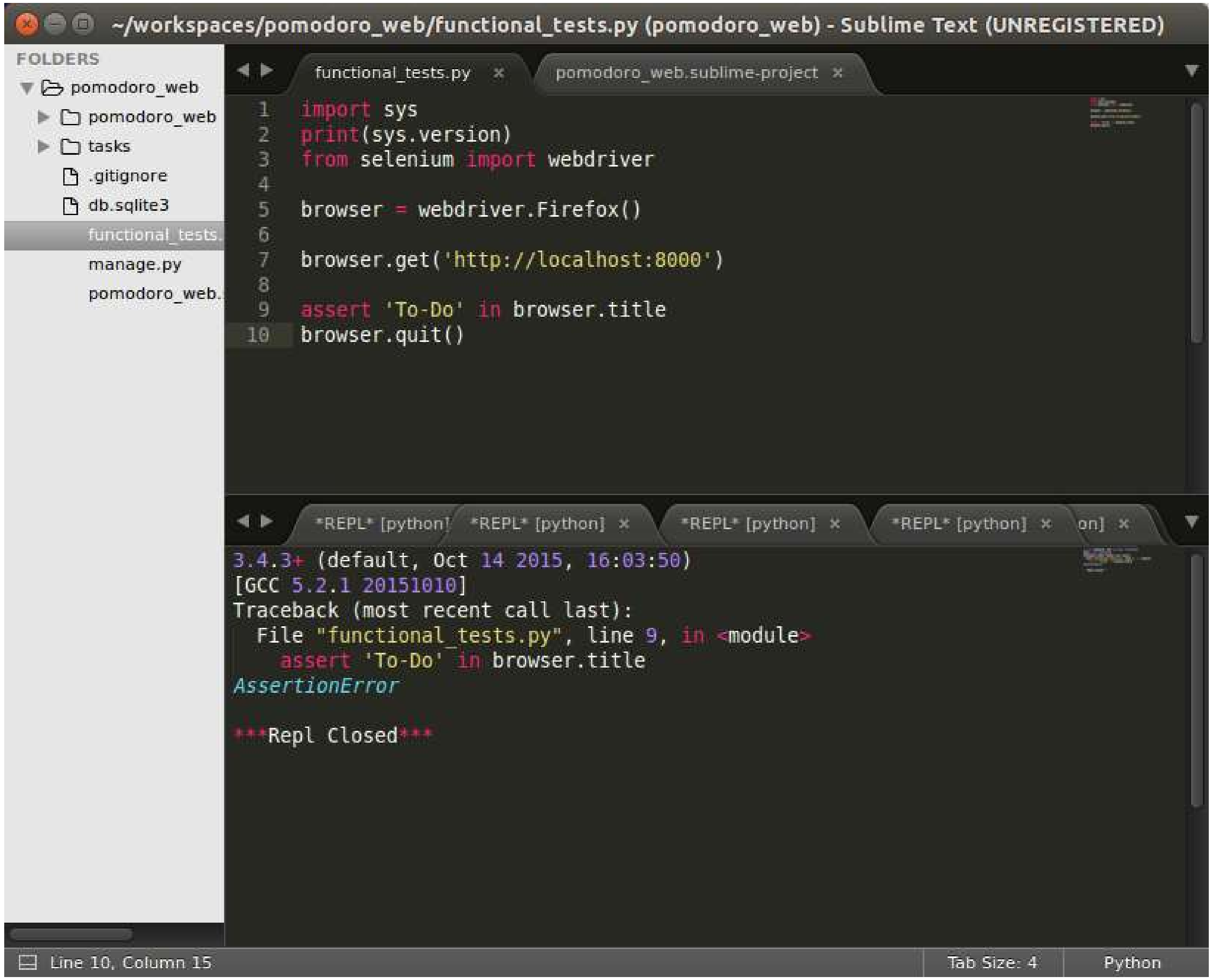
- Surprisingly good and very easy to use
- Requires no installation of the interpreter on the machine
- Can be used interactively
- However, only Python 3.4.0 is available
  - The latest Python version is 3.4.3
- Scripts might be interpreted differently



```python
def hanoi(n, source, helper, target):
    if n > 0:
        # move tower of size n - 1 to helper:
        hanoi(n - 1, source, target, helper)
        # move disk from source peg to target peg
        if source:
            target.append(source.pop())
        # move tower of size n-1 from helper to target
        hanoi(n - 1, helper, source, target)

source = [4,3,2,1]
target = []
helper = []
hanoi(len(source),source,helper,target)

print(source, helper, target)
```

```
Python 3.4.0 (default, Apr 11 2014,
13:05:11)
[GCC 4.8.2] on linux
>
=> None
>
Traceback (most recent call last):
  File "python", line 16
    print source, helper, target
               ^
SyntaxError: Missing parentheses in call to
'print'

>
[] [] [4, 3, 2, 1]
=> None
>
```

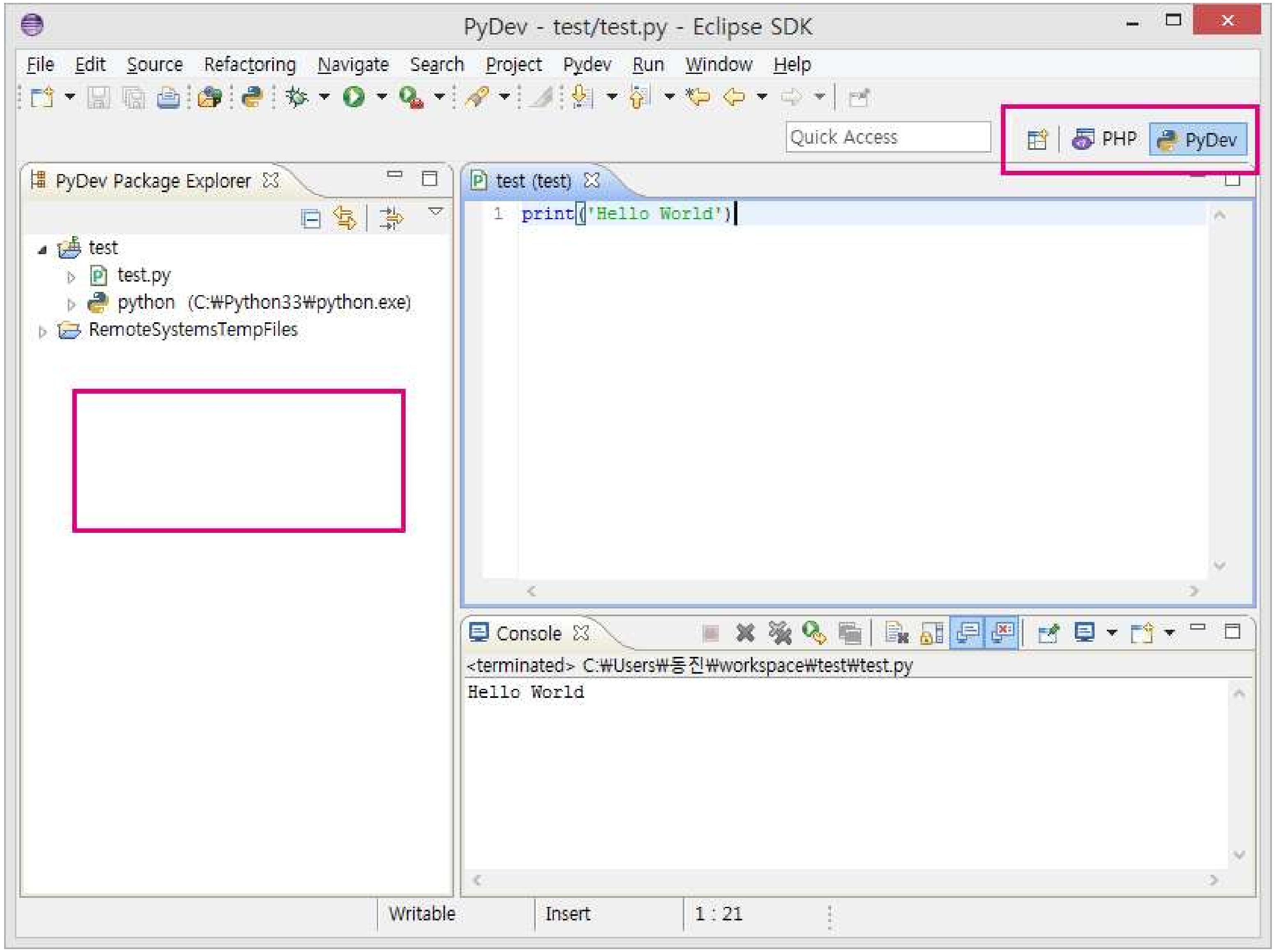
Ways to Use Python: SublimeText3 with Python [4/6]



```python
import sys
print(sys.version)
from selenium import webdriver

browser = webdriver.Firefox()

browser.get('http://localhost:8000')

assert 'To-Do' in browser.title
browser.quit()
```

```
3.4.3+ (default, Oct 14 2015, 16:03:50)
[GCC 5.2.1 20151010]
Traceback (most recent call last):
  File "functional_tests.py", line 9, in <module>
    assert 'To-Do' in browser.title
AssertionError

***Repl Closed***
```

Ways to Use Python: Eclipse with Python


PyDev - test/test.py - Eclipse SDK
File Edit Source Refactoring Navigate Search Project Pydev Run Window Help
Quick Access
PHP
PyDev
PyDev Package Explorer
test (test)
test
test.py
python (C:₩Python33₩python.exe)
RemoteSystemsTempFiles
Console
Writable
Insert
1 : 21

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

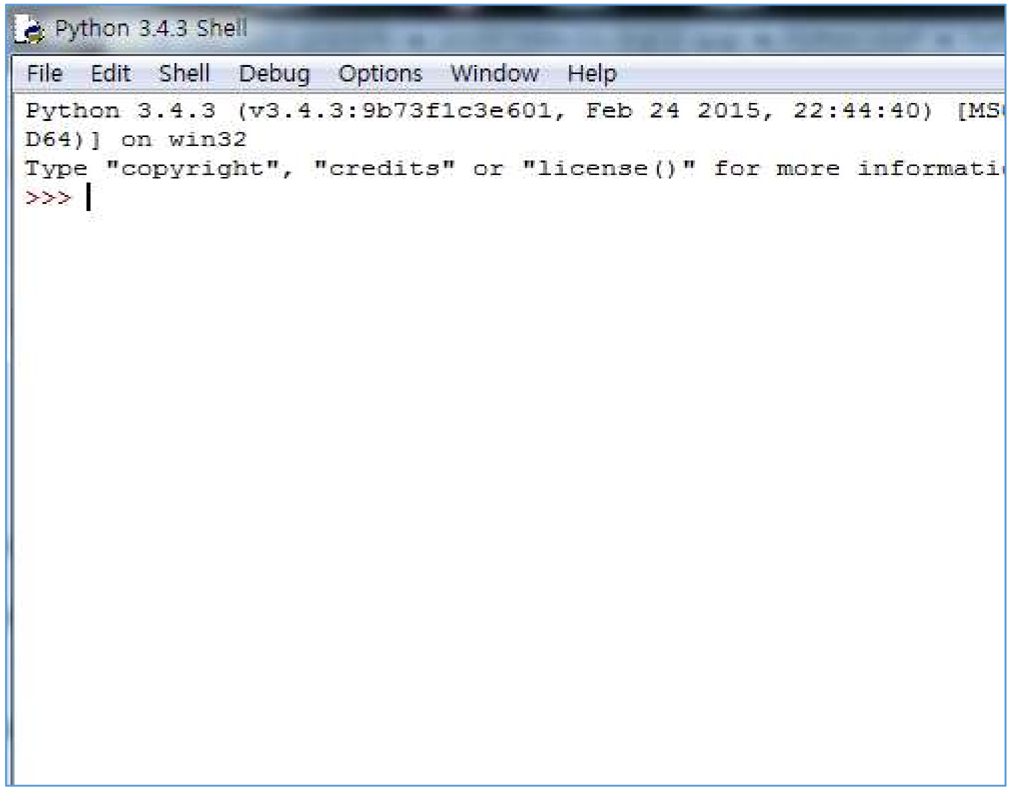
```
<terminated> C:₩Users₩동진₩workspace₩test₩test.py
Hello World
```

Ways to Use Python [6/6]

- Repl.it
  - Fast
  - Portable
  - Suitable for prototype testing
- Python IDLE
  - Readily available in the official python install package
  - Fairly easy to use
  - Features debugging
- Python Tools for Visual Studio
  - Contains the complete feature for programmers
  - The learning curve might be steep
  - Debugging, Refactoring, Syntax Checking, Syntax Highlighting, Dependency Management, and many more…
- SublimeText2 and Python
- Eclipse and Python

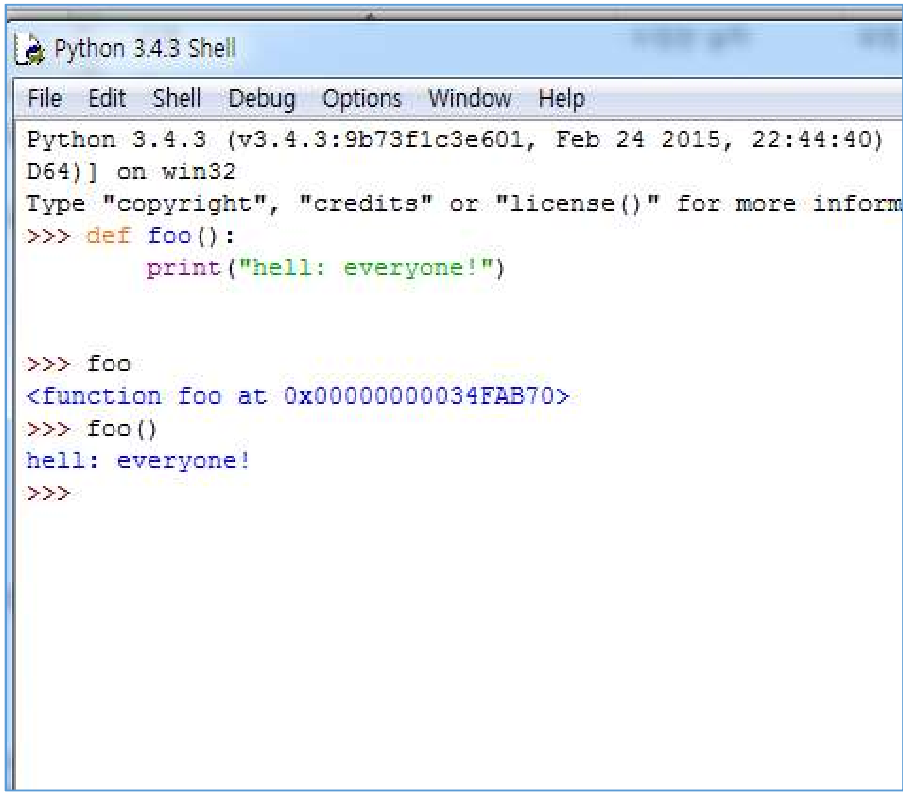
IDLE Screen Shots [1/5]

Initial Screen of IDLE: Python Shell



```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:44:40) [MSC
D64)] on win32
Type "copyright", "credits" or "license()" for more informati
>>> |
```

Initial Coding in IDLE

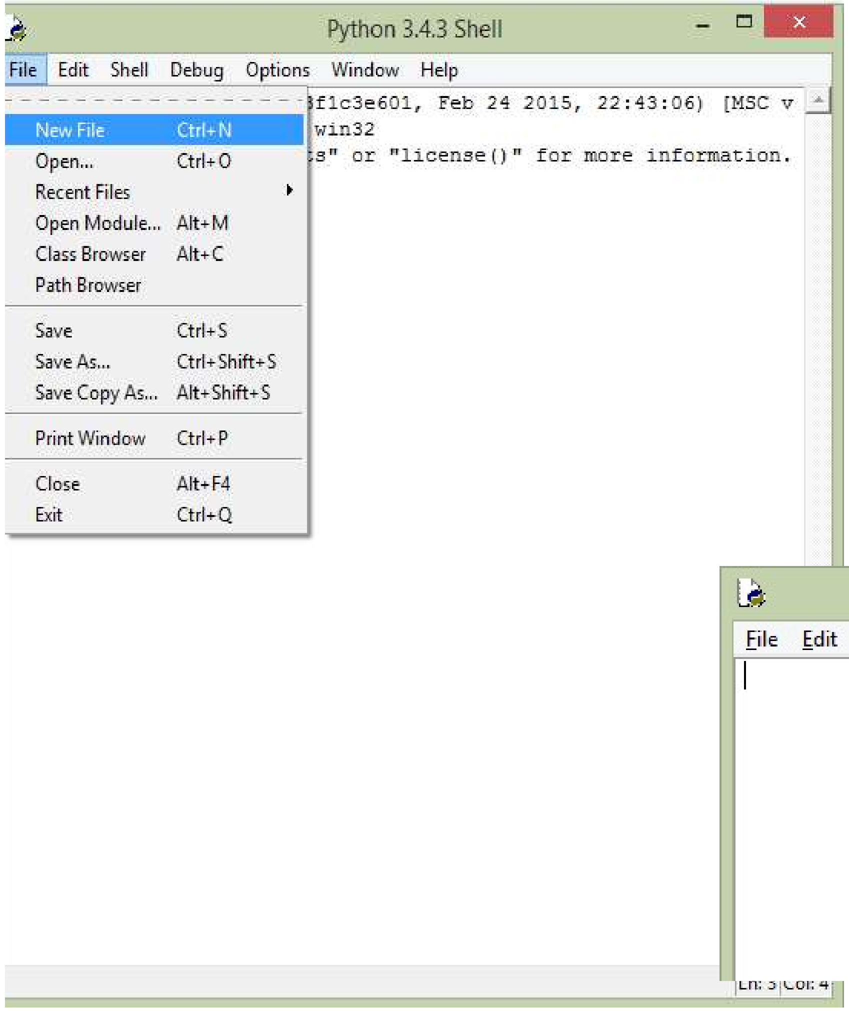


```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:44:40)
D64)] on win32
Type "copyright", "credits" or "license()" for more inform
>>> def foo():
        print("hell: everyone!")


>>> foo
<function foo at 0x00000000034FAB70>
>>> foo()
hell: everyone!
>>>
```

IDLE Screen Shots [2/5]

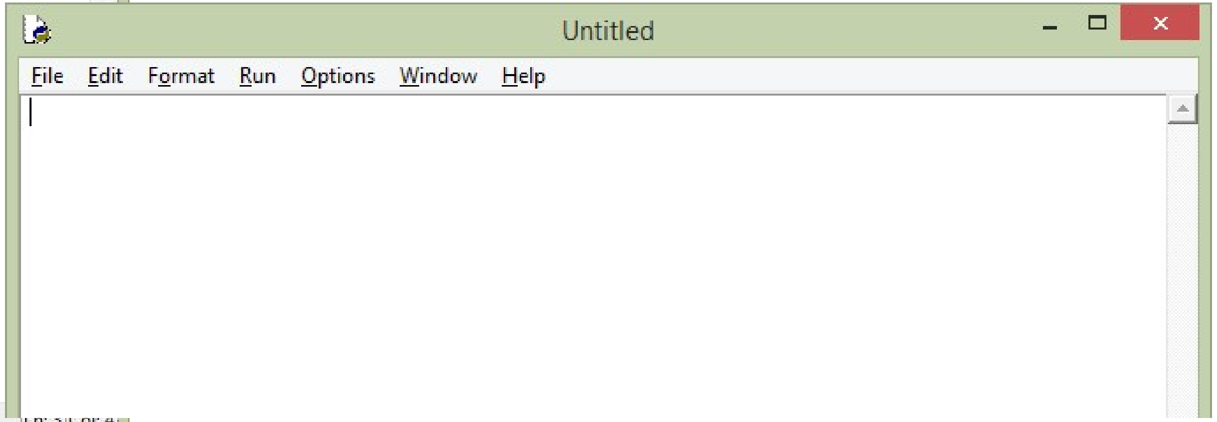
Suppose you finish up coding into IDLE and
you want to save your Python code in your directory!

Creating a new script file in IDLE



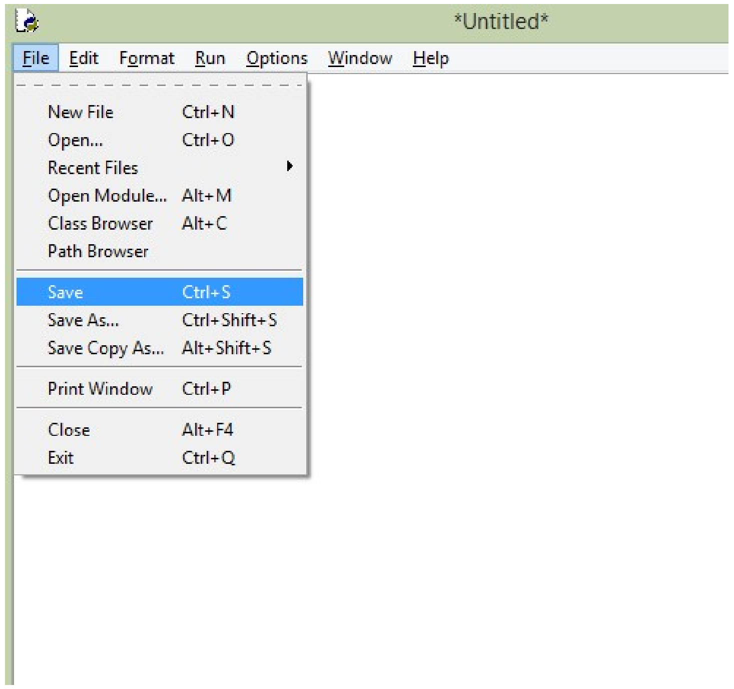
```
3f1c3e601, Feb 24 2015, 22:43:06) [MSC v
win32
s" or "license()" for more information.
```

A new "untitled" window for a new script is poped up



IDLE Screen Shots [3/5]

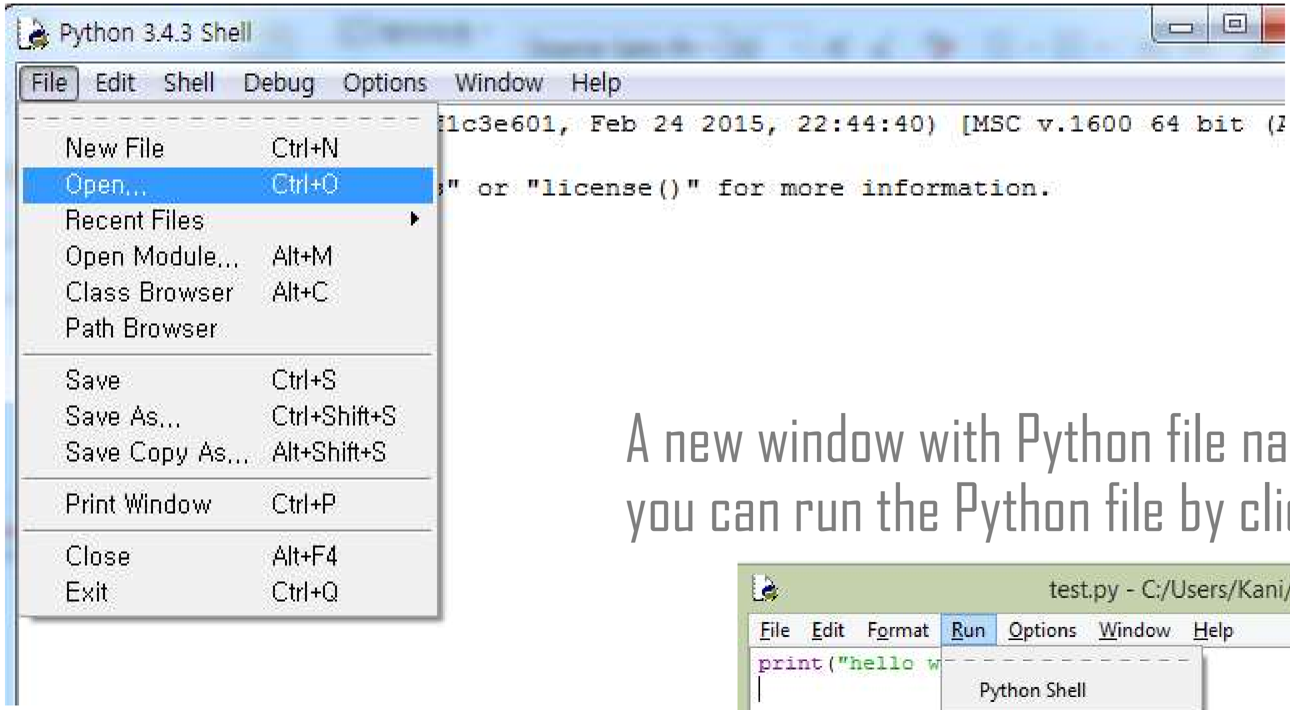
- Cut & Paste Python codes in IDLE window to "untitled" window
- Then, save the code as a new Python file (say, test.py)



- Now you have "test.py" in your directory

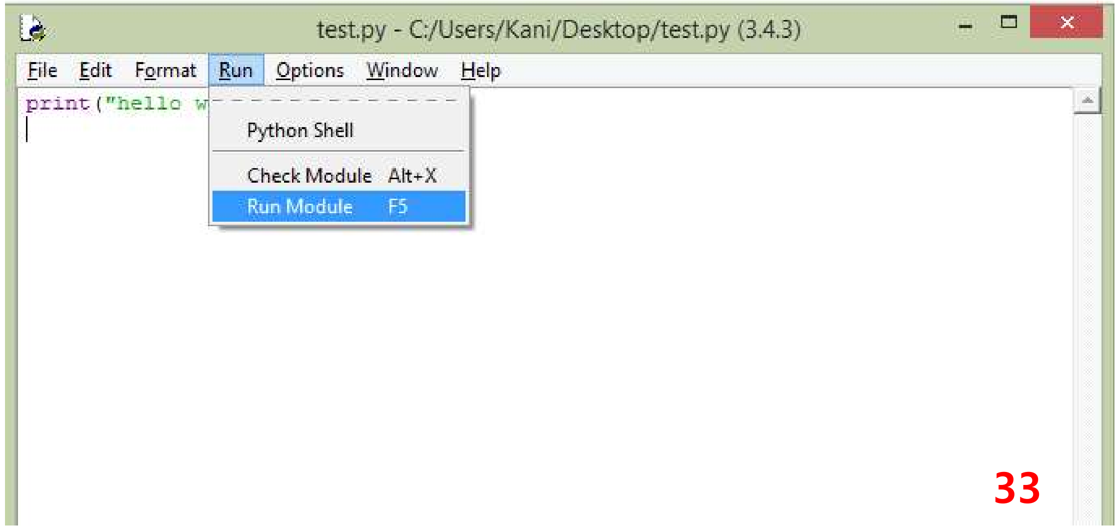
IDLE Screen Shots [4/5]

If you want to read an existing Python file (say, test.py) into IDLE



```
f1c3e601, Feb 24 2015, 22:44:40) [MSC v.1600 64 bit (A
s" or "license()" for more information.
```

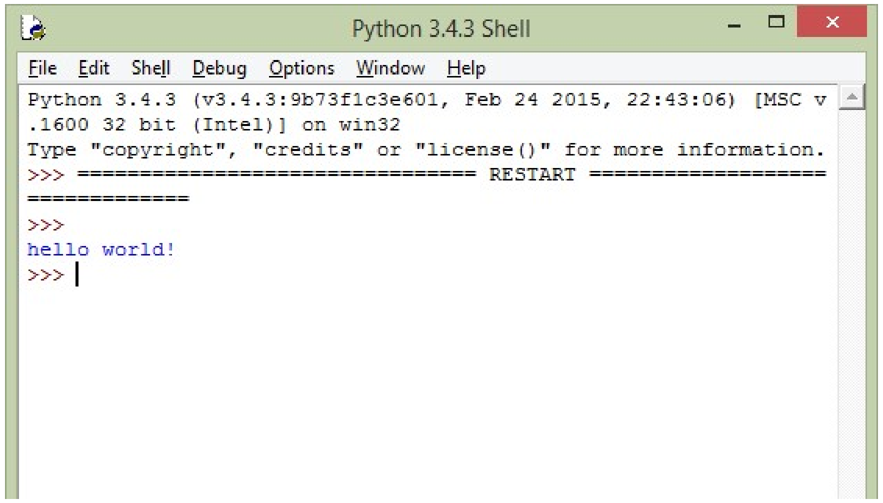
A new window with Python file name is poped up and you can run the Python file by clicking "Run Module"



```
print("hello w
```

IDLE Screen Shots [5/5]

Test results are displayed in a new (existing) shell window



```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v
.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ===================
=============
>>>
hello world!
>>> 
```