

Introduction

Prof.Dr. Bahadır AKTUĞ
BME362 Introduction to Python

**Compiled from sources given in the references.*

Python

- Python programming/scripting language developed by Guido Van Rossum has roots in the ABC programming language invented by CWI (Centrum Wiskunde & Informatica) for which Van Rossum also used to work.
- The first version (version 0.9.0) was published in 1991. Since the first version, Python has always been considered as a object-oriented programming language and several fundamental data structures such as lists, dictionaries have been a part of it.

Python

- The next version (1.0) was published in 1994 and functional programming tools such as lambda, map, filter, reduce were incorporated into the core modules.

- A sample lambda function:

```
>>> f = lambda x, y : x + y
```

```
>>> f(1,1)
```

```
2
```

- A sample combination of filter/ lambda functions:

```
>>> fib = [0,1,1,2,3,5,8,13,21,34,55]
```

```
>>> result = filter(lambda x: x % 2, fib)
```

```
>>> print result
```

```
[1, 1, 3, 5, 13, 21, 55]
```

Python

- ▶ Python 2.0 was published in 2000. The most significant improvements are «list comprehension», «garbage collector», and support for unicode.

- ▶ An example of list comprehension:

```
>>> S = [x**2 for x in range(10)]
```

```
>>> V = [2**i for i in range(13)]
```

```
>>> M = [x for x in S if x % 2 == 0]
```

```
>>>
```

```
>>> print S; print V; print M
```

```
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

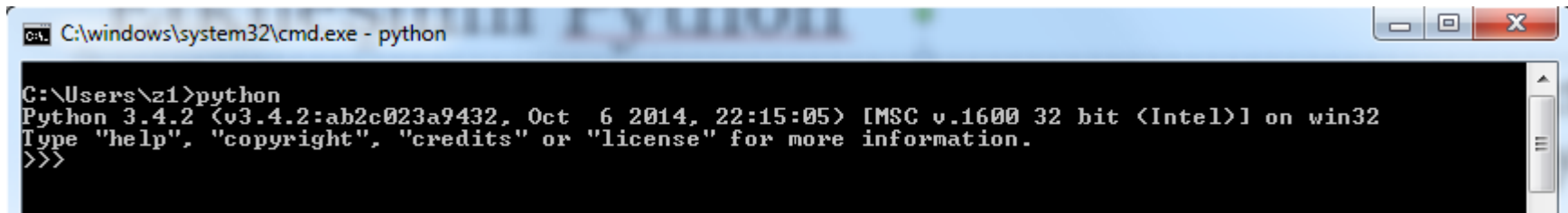
```
[1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096]
```

```
[0, 4, 16, 36, 64]
```

- ▶ While several other Python'un 2.x versions were published until 2008, the most significant update starts with Python version 3.0.

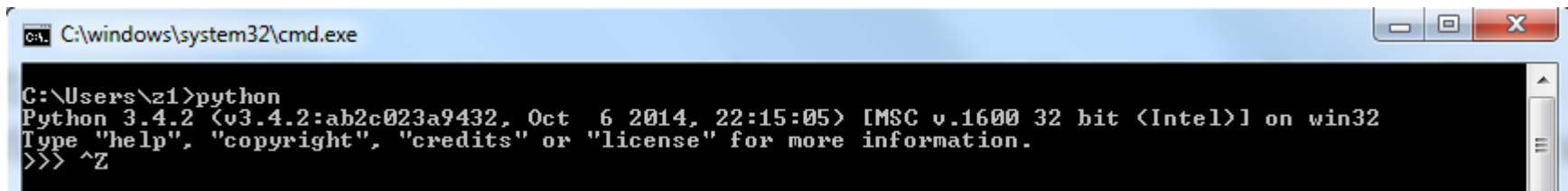
Interactive Python

- While there are several development tools for Python, Python itself offers a powerful shell for programming interactively.
- If Python is properly installed, «python» command can be given directly to enter the interactive environment.



```
C:\windows\system32\cmd.exe - python
C:\Users\z1>python
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct  6 2014, 22:15:05) [MSC v.1600 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

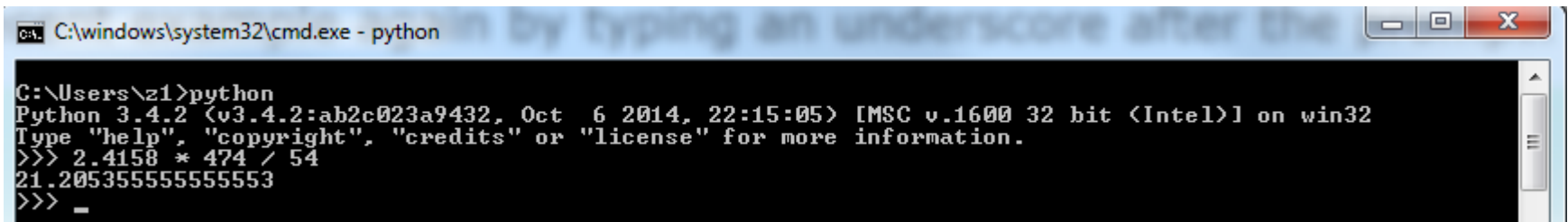
- As seen above, Python interactive shell has a command prompt «>>>» similar to Matlab's «>>». To exit the shell, use «CTRL+Z» for DOS, and «CTRL+D» for Unix-based OS.



```
C:\windows\system32\cmd.exe
C:\Users\z1>python
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct  6 2014, 22:15:05) [MSC v.1600 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> ^Z
```

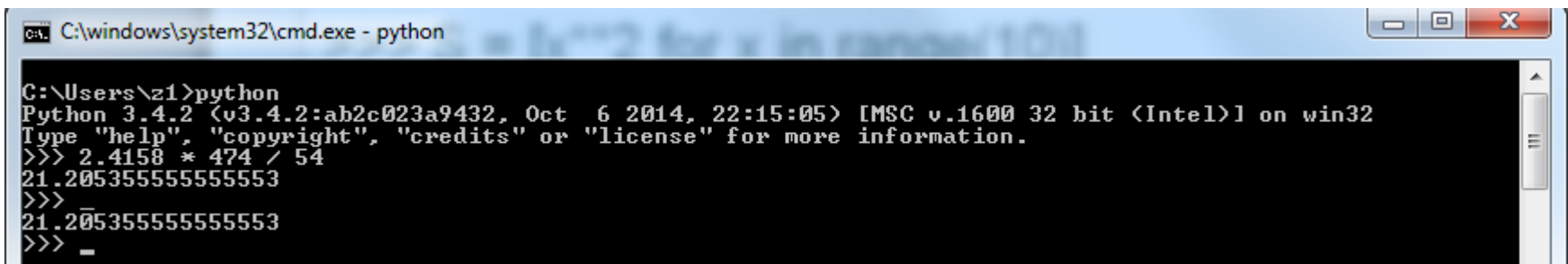
Interactive Python

- ▶ Many simple mathematical operations can be directly performed within Python interactive shell.



```
C:\windows\system32\cmd.exe - python
C:\Users\z1>python
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct 6 2014, 22:15:05) [MSC v.1600 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 2.4158 * 474 / 54
21.205355555555553
>>> _
```

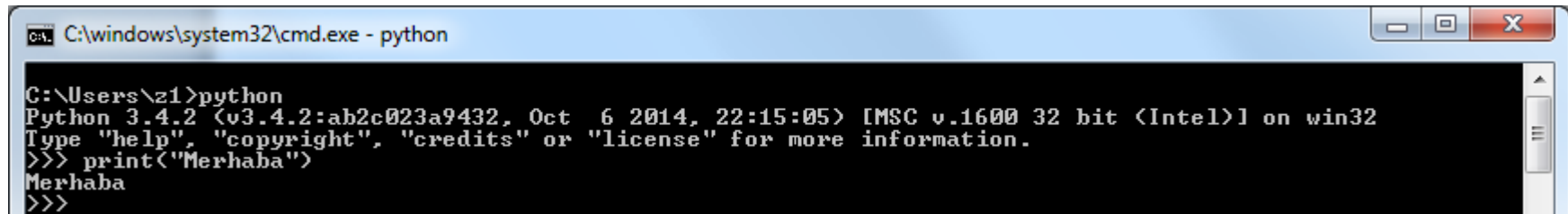
- ▶ The last command output can be recalled by using the "_" variable, similar to the «ans» variable in Matlab.



```
C:\windows\system32\cmd.exe - python
C:\Users\z1>python
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct 6 2014, 22:15:05) [MSC v.1600 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> 2.4158 * 474 / 54
21.205355555555553
>>> _
21.205355555555553
>>> _
```

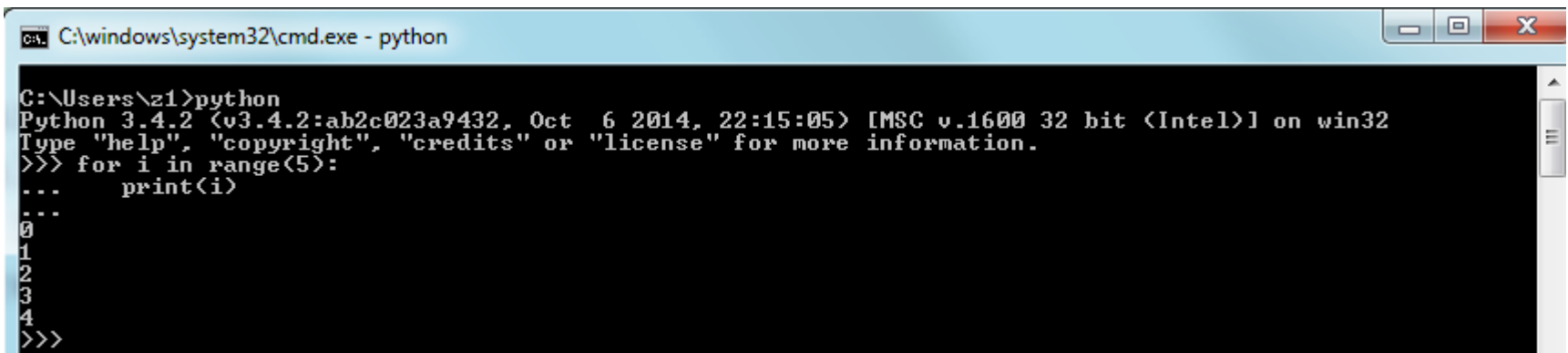
Interactive Python

- ▶ The commands can be given in the shell and the results can be output to the screen.



```
C:\windows\system32\cmd.exe - python
C:\Users\z1>python
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct  6 2014, 22:15:05) [MSC v.1600 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Merhaba")
Merhaba
>>>
```

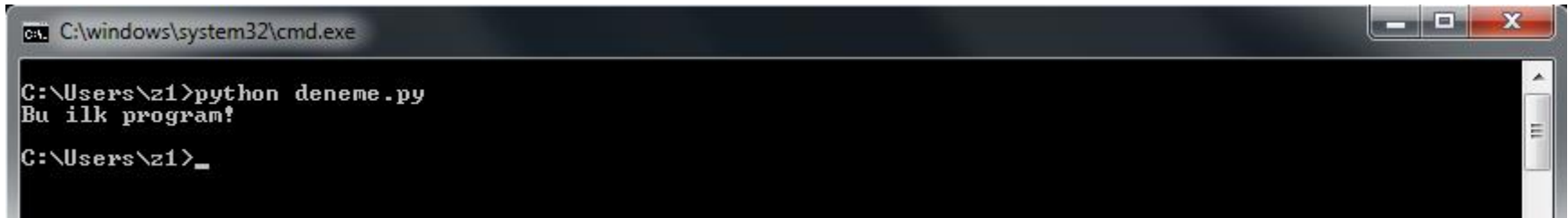
- ▶ Multi-line commands can be given by using the «...» operator.



```
C:\windows\system32\cmd.exe - python
C:\Users\z1>python
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct  6 2014, 22:15:05) [MSC v.1600 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> for i in range(5):
...     print(i)
...
0
1
2
3
4
>>>
```

Running a Python Program

- ▶ Python command can be packed into a batch file for executing externally. Long scripts are particularly executed in this way.
- ▶ For instance, let's write the following command into a file named as "deneme.py":
 - ▶ `print("This is the first script!")`
- ▶ We can then execute the file "deneme.py" in the operating system shell (not Python shell!!!) by the using the syntax below.



```
C:\windows\system32\cmd.exe
C:\Users\z1>python deneme.py
Bu ilk program!
C:\Users\z1>_
```

- ▶ As seen above, a Python script can be executed directly within the operating system shell.

Compiler / Interpreter

▶ **Compiler**

Compilers are computer programs which convert the source code of a programming language, to the instruction sets of the target platform (the computer architecture and the operating system). They convert the high level commands to the low level (assembly or machine code) commands.

▶ **Interpreters**

Interpreters are computer programs which run the source code of a programming language. They do it by executing the source code line by line. Just as they can run the source code directly, they might transform the source code into an intermediate format before execution.

High Level/Low Level Programming Languages

High-level Language

```
temp = v[k];  
v[k] = v[k+1];  
v[k+1] = temp;
```

```
TEMP = V(K)  
V(K) = V(K+1)  
V(K+1) = TEMP
```

C/Java Compiler

Fortran Compiler

Assembly Language

```
lw St0, 0(S2)  
lw St1, 4(S2)  
sw St1, 0(S2)  
sw St0, 4(S2)
```

MIPS Assembler

Machine Language

```
0000 1001 1100 0110 1010 1111 0101 1000  
1010 1111 0101 1000 0000 1001 1100 0110  
1100 0110 1010 1111 0101 1000 0000 1001  
0101 1000 0000 1001 1100 0110 1010 1111
```

Compiler / Interpreter

- ▶ While Python is often considered an interpreter, technically it is both an interpreter and a compiler.
- ▶ Python transforms the source code into an intermediate command set (byte code) instead of transforming directly into the machine code.
- ▶ The intermediate command set is executed in the «Python Virtual Machine (PVM)» similar to the Java Virtual Machine (JVM) and .NET framework.
- ▶ The two-step execution method of Python lead to various implementations of the interpreter which aim to produce executable code directly in the target platform such as Jython for JVM and IronPython for .NET)
- ▶ It is usually faster to execute the already compiled code. Python uses an optimal scheme to make use of the compilation.

Compiler Aspects of Python

- ▶ Python takes care of the compilation step automatically. Before running a script, it checks whether an up-to-date version of the compiled code is available. If compiled code is found, it excludes those parts from transformation into byte code. The compiled code take place in the working directory with «.pyc» extension.
- ▶ If compiled code is found, «byte code» of the relevant source code parts are loaded to speed up the execution.
- ▶ If not found, compiled code is formed.
- ▶ Finally, the loaded intermediate "byte code" is executed in the "Python Virtual Machine".

Compiling Python Code

- ▶ If you wish to compile Python code anyway, you can still do it yourself;

```
>>> import py_compile
```

```
>>> py_compile.compile('test.py') veya
```

- ▶ or

```
>>> python -m py_compile test.py
```

- ▶ Both methods make a directory named «`__pycache__`» in the working directory and the file «`test.cpython-34.pyc`» is formed. The name of the file depends on the version of the installed Python.

Python and Indentation

- ▶ A sequential set of program commands which implements a specific task is called "a programming block".
- ▶ Many programming blocks are needed in a program (functions, control blocks, loops etc.)
- ▶ There are different styles in programming languages to bound the programming blocks. For instance;
 - ▶ "begin/end" reserved words for Pascal programming language
 - ▶ Curly braces ({, }) in C/C++/C#/Java programming languages
 - ▶ "do/done" reserved words for Bash/Bourne shell scripts
- ▶ The programming blocks can be independent or nested.
- ▶ In Python, programming blocks are identified with the indentation.
- ▶ While the amount of indentation in a Python program is arbitrary, they need to be consistent within a block. In other words, all the command lines in a programming block must have the same amount of indentation.

► References

- 1 Wentworth, P., Elkner, J., Downey, A.B., Meyers, C. (2014). *How to Think Like a Computer Scientist: Learning with Python (3rd edition)*.
- 2 Pilgrim, M. (2014). *Dive into Python 3 by*. Free online version: DiveIntoPython3.org ISBN: 978-1430224150.
- 3 Summerfield, M. (2014) *Programming in Python 3 2nd ed (PIP3)* :- Addison Wesley ISBN: 0-321-68056-1.
- 4 Summerfield, M. (2014) *Programming in Python 3 2nd ed (PIP3)* :- Addison Wesley ISBN: 0-321-68056-1.
- 5 Jones E, Oliphant E, Peterson P, et al. *SciPy: Open Source Scientific Tools for Python, 2001-*, <http://www.scipy.org/>.
- 6 Millman, K.J., Aivazis, M. (2011). *Python for Scientists and Engineers, Computing in Science & Engineering*, 13, 9-12.
- 7 John D. Hunter (2007). *Matplotlib: A 2D Graphics Environment, Computing in Science & Engineering*, 9, 90-95.
- 8 Travis E. Oliphant (2007). *Python for Scientific Computing, Computing in Science & Engineering*, 9, 10-20.
- 9 Goodrich, M.T., Tamassia, R., Goldwasser, M.H. (2013). *Data Structures and Algorithms in Python*, Wiley.
- 10 <http://www.diveintopython.net/>
- 11 <https://docs.python.org/3/tutorial/>
- 12 <http://www.python-course.eu>
- 13 <https://developers.google.com/edu/python/>
- 14 <http://learnpythonthehardway.org/book/>