**“Structured and Object Oriented Programming with MATLAB and PYTHON”**

**Syllabus CT.01/1**

**Ľubomír Černický**

October, 20th, 2018

Version 1

# Name of the course

**Structured and Object Oriented Programming with MATLAB and PYTHON**

# ECTS credits

8 Credits, (**45 hours of Theory + 30 hours of Exercises & Lab**), 1st semester

Requirements: Before this course student should have mathematical knowledge such as linear algebra, probability theory and statistics, matrix, and graph theory.

# Objectives

This course gives students basic knowledge of object-oriented Programming with the use of Python and Matlab.

Students will acquire basic knowledge of language structure and important parts of Python (Mathematics, Expressions, Methods, Classes).

Students learn how to design and program simple applications. Having this knowledge students understand various parts in software like Aimsun (API), QGIS scripts etc.

Students will acquire basic knowledge of the MATLAB and understand how to convert simple tasks into MATLAB

# Learning outcomes

The general expectation regarding the knowledge to be provided/acquired is as follows:

* Understanding of basics programming in Python/Matlab
* Acquiring basic knowledge in programs, functions, recursions, arrays, texts, strings, lists, and graphic in Python, and variables, matrices, vector and matrix data visualizing in MATLAB
* Mastering of the script creation, which can be used in other modelling programs ( Aimsun API Programming)
* Mastering of base graphic in Tkinter and turtles

# Contents

1. Introduction
	1. Starting with Python
		1. How to get python
		2. IDLE
		3. How it works
	2. Data types
		1. int
		2. float
		3. str
	3. Data variables
		1. Definition of a variable
		2. Execute a Python script
	4. Exercises
2. Iterations
	1. For Loops
		1. Count-controlled for loop
		2. Iterator-based for loop
		3. The range() function
	2. Importing Modules
		1. math
		2. random
	3. Nested loops
	4. Exercises
3. Graphic
	1. Tkinter
		1. Coordinates
		2. Text
		3. Rectangle
		4. Color
		5. Oval
		6. Line
		7. Polygon
		8. PhotoImage
		9. Canvas setting
	2. Changes in printed objects
		1. delete
		2. move
		3. parameters change
	3. Exercises
4. Conditions
	1. Conditional statements
		1. True and False
	2. While loop
		1. Endless / Infinite loop
	3. Exercises
5. Functions
	1. Syntax
	2. Optional Parameters
	3. Namespaces
	4. Return statement
	5. Local and global variables in functions
	6. Arbitrary number of parameters
	7. Arbitrary number of keyword parameters
	8. Exercises
6. Strings
	1. Unicode Encodings
	2. String, Unicode and Python
	3. Some operators and functions for strings
		1. Concatenation
		2. Repetition
		3. Indexing
		4. Slicing
		5. Size
	4. Immutable Strings
	5. A String Peculiarity
	6. Escape Sequences in Strings
	7. Byte Strings
	8. Exercises
7. File Management
	1. Reading from the file
		1. Open
		2. Read
		3. Close
		4. Finding end
			1. For loop
			2. Whole file into 1 line
		5. Foreign language in the file
	2. Read and Write to the Same File
	3. Copy file
	4. With open
	5. Exercises
8. Lists
	1. pop and append
	2. extend
	3. Extending and Appending Lists with the '+'-Operator
	4. Removing an element with remove
	5. Find the Position of an Element in a List
	6. insert
	7. Exercises
9. Lists and tuples
	1. Using Lists
		1. split()
		2. join()
		3. enumerate()
	2. tuples
		1. tuple
		2. work with tuple
		3. tuple()
		4. For loop and tuple
		5. enumerate()
		6. Indexing
		7. len, sum, min, max
		8. tuples and graphic
	3. Exercises
10. Tkinter events and binds
	1. <Button>
	2. <Motion>
	3. <ButtonRelease>
	4. <Enter>
	5. <Leave>
	6. After
	7. Exercises
11. Turtle
	1. Turtle motion
	2. Pen control
	3. Turtle state
	4. Special methods
	5. Exercises
12. Recursion
	1. Definition of Recursion
	2. Recursive Functions in Python
	3. The Pitfalls of Recursion
	4. Exercises
13. Two-dimensional array
	1. Nested lists
	2. Two-dimensional array: examples
	3. Two-dimensional arrays: nested generators
	4. Arrays and various length of rows
	5. Exercises
14. Classes
	1. Class Creation
		1. Attributes
		2. Mutable objects
		3. Functions
	2. Magic methods
	3. Exercises
15. Classes and methods
	1. Magic methods
	2. \_init\_
	3. \_\_str\_\_
	4. Calling method from another method
	5. Attributes
		1. Some examples with graphic objects
			1. Circle
			2. Rectangle
	6. Exercises
16. Classes and inheritance
	1. Object-Oriented Programming
	2. Inheritance
		1. Syntax and Simple Inheritance Example
		2. Overloading and Overriding
	3. Exercises
17. Errors and Exceptions
	1. try - except
		1. Multiple Except Clauses
	2. Custom-made Exceptions
	3. Clean-up Actions
	4. Combining try, except and finally
	5. else Clause
	6. Exercises
18. Polymorphism
	1. Polymorphism with a function
	2. Polymorphism with abstract class
	3. Example
	4. Exercises
19. Matlab basic
	1. Introduction
	2. User interface
		1. Reading data from file
		2. Saving and loading variables
		3. Plotting data
20. Variables and expressions
21. Analyzing vectors and matrices
	1. Vectors
	2. Matrices
22. Automating commands with scripts
	1. Creating scripts
	2. Running scripts
23. Working with data files
	1. Import
	2. Export
	3. Conversions amongst various types
24. Working with data types
	1. MATLAB data types
	2. Integers
	3. Structures
	4. Converting types
25. Writing programs with logic and flow control
26. Writing functions
	1. Creating functions
	2. Subfunctions
	3. Workspaces
	4. Path and dependence
	5. Calling subfunctions

# Teaching method

Lectures and exercises

* Students have access to slides of the lecture before or during the lecture. It is convenient that students use computers with Python/Matlab at both lectures and exercises. The full contents of each slide is systematically explained by the Lecturer.
* Lectures contain theory explanation, which is connected to examples with solutions. Everything is shown on example.
* At the end of each lecture there are exercises, which students should do themselves. Solving these exercises are part of the final exam.
* Exercises are focused on better understanding of the lectures (students solve examples with the helps of lecturer).
* Students also do 1 or 2 projects at the exercises. Students work in groups for the projects. This will help them to discuss solution with each other and improve team work.

# Assessment method

Exercises from the lectures and examination with the use of PC (student get exercise(s), which should be solved without any support)

# Textbooks - Publications - Software

**Textbooks**

Mark Summerfield: **Programming in Python 3 A Complete Introduction to the Python Language**, second edition, Pearson Education, Inc. ISBN 978-0-321-68056-3

Hahn, B., Valentine, D.: **Essential Matlab for Engineers and Scientists**, 5th ed., Academic Press, 2013.

Mathews, J. H., Fink, K. D.: **Numerical Methods Using Matlab. Prentice Hall**, 1999

Stormy Attaway: **MATLAB A practical Introduction to Programming and Problem Solving,** 4th edition, Todd Green, ISBN: 978-0-12-804525-1

Kiusalaas, J.: **Numerical Methods in Engineering with Matlab**. Cambridge University Press, 2005.

Mark Lutz: **Learning Python,** 5th edition, ISBN-10: 1449355730

Michael Dawson: **Python Programming for the absolute beginner**, Third Edition, ISBN-10: 1435455002

**Selected relevant Publications**

<http://interactivepython.org/runestone/static/thinkcspy/index.html>

Software

**\*** Numerical Computing: Algorithmic/Coding and Model-Based Design

* **MATLAB TAH FULL SUITE** for scientific computing, MathWorks (1984), USA.

https://de.mathworks.com/academia/student\_version.html

* Python 3

<https://www.python.org/> - free software