

## Программа учебной дисциплины «Программирование»

Утверждена

Академическим советом ООП

Протокол № \_\_\_\_ от \_\_\_\_ августа 2018 г.

Автор	Ефремов С.Г., к.т.н.
Число кредитов	7
Контактная работа (час.)	84
Самостоятельная работа (час.)	182
Курс	1
Формат изучения дисциплины	С использованием онлайн курса

### **Abstract:**

In the modern highly technological world computer skills have become essential for specialists in almost all possible areas. Programming in particular has gone beyond its traditional borders of being just a prerogative of IT specialists, instead turning into an element of general computer literacy. In the last 15 years programming languages and tools have evolved significantly, which now enables people even without a solid technical background to successfully master related skills.

The present course is offered to 1st year students of the HSE and University of London Parallel Degree Programme in Management and Digital Innovation. It is based on the Python programming language, which is considered by many as the best first programming language to learn having clear and concise syntax and free cross-platform tools for development.

The course is taught in a blended format with part of the content provided from a MOOC on Coursera.

### **Course Objectives**

The course provides students with knowledge of fundamental programming principles and the corresponding practical skills. Although based on a specific toolset (Python), the course aims to give a general view of what can be done using a modern general-purpose programming language. On completing the course students are expected to:

- understand the key ideas behind programmatic problem solving
- know fundamentals of the Python programming language
- be able to select appropriate data structures for storing information
- perform basic processing of numeric and textual information
- know several techniques of automated data acquisition, including web queries

### **Methods**

The course is practice-oriented and requires active student involvement in its activities. The following methods and forms of study are used in the course:

- MOOC on Fundamentals of Python taken through Coursera
- Lectures
- Practice sessions conducted in computer classes
- Regular homework assignments. Each assignment takes from 2 to 4 hours to complete including the required readings.
- Team project (final month of the course)
- Online consultations from course instructor(s)
- Self-study activities: completing homework assignments, studying recommended resources, experimenting with the toolset, solving advanced tasks

In total the course comprises 266 academic hours, including 40 lecture hours, 44 practice hours and 182 hours of self-study activities.

### **Main Reading:**

The main textbook “Think Python” [1] is a well-structured introductory resource for Python programming. Students can study the book chapter by chapter, as the course closely follows its structure. Special attention should be paid to the glossary at the end of each chapter. “Learn Python” by M.Lutz [2] is a more advanced textbook and only selected reading is recommended in the beginning.

1. Allen B. Downey. Think Python. How to think like a computer scientist. Green Tea Press, 2008. Electronic version available for free download at: <http://greenteapress.com/wp/think-python-2e/> Interactive version with program execution in the browser available at: <http://interactivepython.org/runestone/static/thinkcspy/index.html>
2. Лутц М. Изучаем Python, 4-е издание. – Пер. с англ. – СПб.: Символ-Плюс, 2011. – 1280 с., ил.

### **Supplementary reading**

3. Kenneth A. Lambert. Fundamentals of Python: From First Programs Through Data Structures. – Course Technology, 2010.
4. W. McKinney. Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython. – O’Reilly Media, 2012.

### **Internet resources**

5. Coursera online course: Programming for Everybody (Getting started with Python): <https://www.coursera.org/learn/python>
6. Python practice book. <http://anandology.com/python-practice-book/index.html>
7. Python 3 installation guides for different OS: <https://pragprog.com/book/gwpy2/practical-programming>
8. Some of the basic differences between Python 2 and Python 3: <http://www.cs.carleton.edu/faculty/jgoldfea/cs201/spring11/Python2vs3.pdf>
9. Python regular expressions tutorial. [http://www.tutorialspoint.com/python/python\\_reg\\_expressions.htm](http://www.tutorialspoint.com/python/python_reg_expressions.htm)
10. Python requests library. <http://docs.python-requests.org/en/latest/>
11. API integration in Python – Part 1. <https://realpython.com/blog/python/api-integration-in-python/>
12. NumPy tutorial. [http://wiki.scipy.org/Tentative\\_NumPy\\_Tutorial](http://wiki.scipy.org/Tentative_NumPy_Tutorial)
13. OpenPyXL library. <https://openpyxl.readthedocs.org/en/latest/>
14. RESTful web services: <http://www.drdobbs.com/web-development/restful-web-services-a-tutorial/240169069>
15. Python database access: [http://www.tutorialspoint.com/python/python\\_database\\_access.htm](http://www.tutorialspoint.com/python/python_database_access.htm)

### **Required software**

The course is based on Python 3. The default toolset, which can be downloaded from <https://www.python.org/downloads/>, is open-source and cross-platform.

### **Grade determination**

The final grade (FG) for the course is calculated as a weighted average of in-class assessments, homework assignments, team project and the final exam. The percentage breakdown is the following:

<i>Component</i>	<i>Weight (%)</i>
Average grade for in-class assessments (HW)	30
Project (P)	30
Final exam (EX)	40

$$FG = 0,3 \times HW + 0,3 \times P + 0,4 \times EX$$

## **Course Outline**

### **1. Brief overview of modern computer architecture**

Hardware, software and firmware. Memory layers. Operating system. Information encoding.

### **2. Introduction to programming and the Python language.**

Purpose of programming. Source code and executable files. Programming languages. Algorithms and ways of defining them. Python application areas. Python versions 2 and 3. Overview of resources and development tools. Software installation. Interactive shell.

### **3. Python basics**

Types and variables. Integer, floating point and string types. Conversions between types. The “type” operator. Arithmetic operators. Console input-output. Formatted output.

### **4. Program flow**

Boolean expressions. Conditional execution. Code formatting. “while” loop. Code editor. Debugging programs.

### **5. Functions and modules**

Importing modules. Calling standard functions. Functions and methods. Math and random modules. Defining custom functions. Installing new packages.

### **6. Data structures**

Mutable and immutable types. Lists: creating a list, adding and removing elements, retrieving elements, sorting lists, slices. Dictionaries: creating a dictionary, adding and removing items, querying items by key. Tuples. Conversions between data structures. “for” loop and “in” operator. The datetime type. Special algorithms on strings. Comprehensions.

### **7. Processing text. Regular expressions**

Specialized methods for string manipulation and processing. Regular expression language: main capabilities and samples. Extracting data to groups.

### **8. File input-output**

File system. Absolute and relative paths. File access modes. Formats. Standard file operations.

## **9. Data acquisition**

Human-readable and machine-readable formats. Integration with the Web: basics of HTTP, extracting data from HTML, downloading files, querying RESTful services. Integration with SQL DBMS.

## **10. Python for data analysis and visualization**

Overview of available packages and their features. Integration with Excel.