UNIVERSITY OF WARSAW

UNIVERSITY'S INTEGRATED DEVELOPMENT PROGRAMME

Course syllabus

Course title	Introduction to programming in Python
Instructor(s)	Marcin Leśniak PhD
Contact details	m.lesniak2@uw.edu.pl
Affiliation	Faculty of Psychology, University of Warsaw
Course format	class
Number of hours	30 hours
Number of ECTS credits	**3 ECTS credits**
Brief course description	This is an introductory course to computer programming in Python. It does not require any previous programming experience. In the class students learn to think algorithmically and decompose problems into manageable parts. Through simple examples they become familiar with programming concepts such as variables, conditional statements, loops, functions, recursion. Basic Python syntax, standard data structures and flow control statements are introduced. Procedural style of programming is encouraged.
Full course description	Python is a high-level, general-purpose programming language which gained vast popularity among scientific community. It is a popular tool of choice for scientific computing, data analysis and computational modeling. In this course it is taught as the first programming language to an audience without prior experience with computer programming. Clear Python syntax helps the beginners to become acquainted with standard programming concepts—such as variables, conditional statements, loops, functions, recursion—without being distracted by implementation details. Standard data structures (list, dictionaries, sets) will be also introduced. Those form the necessary foundations for algorithmic thinking. As the class progresses, students should develop intuitions and become more familiar with this style of thinking, allowing them to express real-world scenarios in algorithmic terms. They should be able to solve simple problems on their own, creatively applying introduced concepts. This basic course focuses on imperative/procedural style of programming, where code is organized in (active) procedures and (static) data structures. Notions of object-oriented programming or functional programming are generally not introduced at this point. No additional software packages

	besides Python standard library are covered. Python 3 version of the language is used in classroom.
Learning outcomes	Student knows and understands: - syntax and semantics of Python programming language (K_W04) - basic programming concepts such as algorithm, computational complexity, recursion (K_W08) Student is able to: - describe and model particular cognitive tasks in algorithmic terms (K_U03, K_U05) - improve programming skills on her own through readings, exercises and individual exploration (K_K01, K_K02)
Learning activities and teaching methods	The class will be conducted in a computer laboratory. It will consist of programming exercises interspersed with short lectures and demonstrations. Short homework assignments and readings may be expected.
List of topics/classes and bibliography	Topics: 1. Ingredients. Basic data types: integers, floats, booleans, strings. Arithmetic expressions, string indexing. Basic string and number manipulation methods. Variables and assignments. Conditional branching (if statements). 2. Structures. Sequences (lists, tuples, ranges). List indexing. Assignment to variable vs assignment to list element. Bounded loop (for loop over elements of a sequence) vs unbounded loop (while loop). Using dictionaries to make key-value mappings. 3. Functions. Organizing code into functions. Parameters and returned value. Scope of variables. Nested function calls. Writing simple functions operating on text and numbers. 4. Programs. Writing a simple interactive program (for example, very simple text-based computer game). Parsing command line arguments, responding to user's input, dealing with malformed input. 5. Algorithms. Introducing simple algorithms (Sieve of Eratosthenes, Euclid's algorithm, etc.). Reading and analyzing algorithms. Thinking algorithmically. 6. Complexity. Concepts of computational complexity and space complexity of an algorithm. Sorting numbers. Bubble sort, insertion sort, pigeonhole sort. 7. Recursion.

	Writing simple recursive functions. Relation between recursion and loops. Divide-and-conquer strategy. Quicksort algorithm. 8. Style. Good Python programming practices. List and dictionary comprehension. Generators. Exceptions. Organizing code and data. Suggested reading: ● Algorithmics: The Spirit of Computing. David Harel. 3rd edition, Addison-Wesley, 2004. Chapters 1-2 (for topics 1-2), chapter 4 (for topic 5), chapter 6 (for topics 6-7). ● Python Programming: An Introduction to Computer Science. John Zelle. 3rd edition, 2016. Chapter 13 (for topics 5-7). Full bibliography: ● Algorithmics: The Spirit of Computing. David Harel. 3rd edition, Addison-Wesley, 2004. ● Python Programming: An Introduction to Computer Science. John Zelle. 3rd edition, 2016. ● Python Crash Course, a hands-on, project-based introduction to programming, Eric Matthes, 2016. ● Learn Python 3 the Hard Way: A Very Simple Introduction to the Terrifyingly Beautiful World of Computers and Code, Zed Shaw, 2013.
Assessment methods and criteria	**Laboratory exercises (40%)** At least two times during the semester students will be given simple problems to solve individually during class. They will be graded on the spot by the instructor. **Homework assignments (60%)** At least three times during the semester students will be assigned short problems to solve at home. These assignments may have more open format. Time to solve each assignment should be 1-2 weeks.
Attendance rules	Two unexcused absences are allowed in the semester. Further unexcused absences may result in lowering the grade. If due to absences students miss graded exercises in class, they may be given an additional homework assignment.
Prerequisites	Fluency in high-school level mathematics and logic.
Academic honesty	Students must respect the principles of academic integrity. Cheating and plagiarism (including copying work from other students, internet or other sources) are serious violations that are punishable and instructors are required to report all cases to the administration.
Remarks	