Course Specifications
Valid in the academic year 2017-2018

Course

Lecturers in academic year 2017-2018
De Neve, Wesley
KR01 lecturer-in-charge

Course offerings in academic year 2017-2018
A (year)
English

Offered in the following programmes in 2017-2018
crds offering
Bachelor of Science in Environmental Technology 10 A
Bachelor of Science in Food Technology 10 A
Bachelor of Science in Molecular Biotechnology 10 A
Joint Section Bachelor of Science in Environmental Technology, Food Technology and Molecular Biotechnology 10 A

Teaching languages
English

Keywords
Computational thinking, Command line, Creative problem solving, Programming, Python, SQL, Scientific problem solving, UNIX

Position of the course

Scientists and engineers are often confronted with time-consuming and repetitive tasks when processing and analysing data, namely collecting information from websites, converting files from one format to another, and analysing, summarizing and visualizing the information obtained. In addition, the exponential flow of newly incoming information requires present-day scientists and engineers to be able to automate these tasks, in order to speed up their daily job routines.

This course teaches students how to translate these time-consuming and repetitive tasks in such a way that they can be performed automatically by a computer. To that end, the necessary skills for computer-based creative problem solving will be acquired through learning to work and think in Python, a popular programming language, and in UNIX, the workhorse operating system of science and engineering. The programming problems that need to be solved are taken from different scientific disciplines, including mathematics, biology, chemistry, physics, and computer science.

To participate in this course, students do not need to have any prior programming experience. However, in order to successfully complete this course, students need to have an aptitude for mathematics and logic. In addition, given that this course follows a 'learning by doing' and a 'learning from mistakes' approach, students need to have a willingness to solve programming problems on a weekly basis.

Contents

Programming is the process of designing, writing, testing, debugging and maintaining the source code of computer programs. This requires knowledge of the syntax and semantics of a programming language and the ability to write programs in that language. Additionally, and maybe most importantly, when writing computer programs, one must learn how to think as a programmer. This process of computational thinking, or in other words, learning the skill of problem solving by programming, is a common theme throughout the whole course.

In this course, students learn how to make use of the Python programming language to solve a plethora of problems. To that end, attention is paid to.

(Approved)
Basic components: instructions, variables, data types, and operators;
control structures: conditional statements, repetitive statements, and functions;
data structures: strings, lists, tuples, dictionaries, sets, modules, and files;
text files: reading, processing, and writing data; and
object-oriented programming: objects, classes, attributes, methods,
encapsulation, polymorphism, and inheritance.

Furthermore, in this course, students learn how to make use of UNIX-based tools to
automate repetitive or complex tasks. To that end, attention is paid to:
- principles of UNIX-based operating systems;
- file systems;
- interactive command line usage;
- automated text editing;
- regular expressions; and
- the basics of scripting.

Finally, in this course, students learn how to make use of the Structured Query
Language (SQL) to communicate with a database.

Initial competences
- An aptitude for mathematics and logic.
- An interest in solving (scientific) problems.
- Prior programming skills are not required.
- Some basic computer knowledge is advantageous.

Final competences
- The students will be able to translate a task described in natural language into a
  program written in Python, and they will subsequently be able to execute this program
  by means of a computer, generating the required results.
- The students will be able to test and debug a program (module) and make the right
  choices between different alternatives when implementing a program, taking into
  account performance (efficiency), coding style, and correctness.
- The students will have a working knowledge of the basic principles of object-oriented
  programming.
- The students will be able to automate repetitive or complex tasks by making use of
  UNIX-based tools.
- The students will be able to deal smoothly with file systems, text editors, operating
  systems, computer networks, and databases.
- The student will be able to transfer the computational concepts learned to other
  computational environments (e.g., computational environments based on MATLAB or
  R).

Conditions for credit contract
- Access to this course unit via a credit contract is determined after successful competences
  assessment

Conditions for exam contract
- This course unit cannot be taken via an exam contract

Teaching methods
- Demonstration, lecture, seminar: practical PC room classes

Learning materials and price
  Slides shown during the lectures will be made available on Minerva, together with
  additional learning materials (e.g., background information and links to relevant
  websites).
- Free digital tools like Eclipse for writing and debugging Python source code, the Online
  Python Tutor for visualizing code execution, an online platform for automated
  verification of the correctness of solutions written in Python, and a remote UNIX
  environment (e.g., Helios).
  Students are required to have a personal laptop for use in this course.

References
  Free download @ http://diveintopython.org.

(Approved)
Course content-related study coaching

Evaluation methods
- end-of-term evaluation and continuous assessment

Examination methods in case of periodic evaluation during the first examination period
- Open book examination, skills test

Examination methods in case of periodic evaluation during the second examination period
- Open book examination, skills test

Examination methods in case of permanent evaluation
- Assignment

Possibilities of retake in case of permanent evaluation
- Examination during the second examination period is not possible

Calculation of the examination mark

During the first examination period, the score of the periodic evaluation (partial exams) accounts for 75% of the examination mark and the score of the non-periodic evaluation (hands-on sessions) accounts for 25% of the examination mark. To qualify for passing, both the score of the periodic and the non-periodic evaluation should be at least equal to 8/20. If that is not the case, the examination mark for the first examination period will be subject to an upper limit of 7/20.

The periodic evaluation consists of a partial examination at the end of the first term and a partial examination at the end of the second term. The partial examination at the end of the first term accounts for 75% of the final score obtained at the end of the first term, and the hands-on sessions that took place during the first term account for 25% of the final score obtained at the end of the first term. To qualify for passing, both the score of the partial examination at the end of the first term and the score of the hands-on sessions at the end of the first term should be at least equal to 8/20. If that is not the case, the final score obtained at the end of the first term will be subject to an upper limit of 7/20.

If the final score obtained at the end of the first term is higher than or equal to 10/20, then the partial examination at the end of the second term only covers the course content of the second term. In addition, the score of the periodic evaluation is then equal to the average of the score of the partial examination at the end of the first term and the score of the partial examination at the end of the second term. On a similar note, the score of the non-periodic evaluation is then equal to the average of the score of the hands-on sessions at the end of the first term and the score of the hands-on sessions at the end of the second term.

If the final score obtained at the end of the first term is lower than 10/20, then the partial examination at the end of the second term covers the course content of both the first
and the second term. In addition, the score of the periodic evaluation is then equal to the score of the partial examination at the end of the second term, whereas the score of the non-periodic evaluation is then equal to the average of the score of the hands-on sessions at the end of the first term and the score of the hands-on sessions at the end of the second term. During the second term, the hands-on sessions of the first term cannot be retaken.

Students who passed the partial examination at the end of the first semester may decide to retake this examination in the second term. When doing so, the last score obtained is assumed to be the final score for that part of the exam.

During the second examination period, the non-periodic evaluation cannot be retaken. Therefore, the examination mark for the second examination period is calculated twice. For the first calculation, the score of the non-periodic evaluation, as obtained during the first examination period, accounts for 25% of the examination mark and the score of the periodic evaluation, as obtained during the second examination period, accounts for the remaining 75% of the examination mark. For the second calculation, the examination mark is equal to the score of the periodic evaluation, as obtained during the second examination period. The final examination mark for the second examination period is then equal to the maximum of the above two calculations.

Scores for partial examinations can never be transferred to the second examination period.