

Module Handbook

5AAVC210

Introduction to Programming

Spring Semester 2017/18

Convenors:

Dr Gabriele Salciute Civiliene

Dr Giles Greenway

Credits: 15

1. Module Description

This module will teach students the basic structures and syntax of a common programming/scripting language, Python. Programming is at the heart of the human control of the digital world; through the use of programming and scripting languages, we can manipulate and gather data, create new applications and extend existing ones. Further, learning the logic, possibilities and limitations of programmatic structures allows us to better appreciate and understand the technology within the greater digital world.

Active engagement with data gives the researcher a unique appreciation of that data and the meaning encapsulated within it. As the amount of data that a researcher must deal with increases the methods of interaction, analysis and management become increasingly computational. This module aims to introduce students to the fundamental methods needed to design and build their own programs and allow them to further their understanding as a tool maker in addition to a tool user.

Python was selected as the introductory language as it is widely used in Digital Culture research and offers a comprehensive suite of data analytic libraries. Further it is one of the main components of the Django framework which is a popular tool for interactive website development. In the future the choice of language may need to be revised, but in the current digital ecosystem, Python represents the most appropriate language to introduce Digital Culture students to programming. It also offers a strong transferable skill.

At the end of this module, students will be expected to:

* have a systematic understanding of how to create and run a Python program;
* have a detailed knowledge of the basic structures that underlie the Python programming language;
* identify and analyse a problem and select the appropriate programmatic methods to solve that problem;
* harvest data from the web by using Python’s third-party libraries request and beautifulsoup;
* visualize and analyze harvested data by using Python’s third-party libraries matplotlib, bokeh and pandas.

2. Module Tutors

Convenors: Dr Gabriele Salciute Civiliene

Strand Bldg S3.15

gabriele.salciute-civiliene@kcl.ac.uk

Office Hours: Thursdays 12:00–14:00

Dr Giles Greenway

Strand Bldg Room XXXX

[giles.greenway@kcl.ac.uk](mailto:giles.greenway@kcl.ac.uk)

Office Hours: XXXXXXX

Seminar Leader: Dominic Weldon

Strand Bldg XXXXX

[dominic.weldom@kcl.ac.uk](mailto:dominic.weldom@kcl.ac.uk)

Office Hours: XXXXXXX

3. Module Location and Times

Lectures: Fridays 11:00–12:00, Strand Bldg S3.32

Seminars:

Group A & B: Fridays 13:30–15:30, Maughan’s Library 1.63

4. Readings

Each week has required readings. Please read these texts before each week’s session and come prepared to discuss the readings. Alongside the required texts, there are recommended readings. Read as many of these as you find interesting. Take the time to look them up. You are expected to read widely on this module, and in your assessment you must refer to a range of relevant readings.

**Core reading:**

Downey, Allen (2012) *Think Python*. O’Reilly Media.

Matthes, Eric (2016) *Python Crash Course*. No Starch Press.

Rossant, Cyrille (2013) *Learning IPython for Interactive Computing and Data Visualization*. Packt Publishing.

[[1]](#footnote-1)Beazley, David and Jones, Brian K. (2013) *Python Cookbook*. O’Reilly Media.

The Python Tutorial (2016). Available at: <https://docs.python.org/3.7/tutorial/index.html>

What’s New (2016). Available at: <https://docs.python.org/3.7/whatsnew/3.6.html>

**Online Python tutorials:**

Learn Python (anon.) *Interactive Python Tutorial*. Available at: <https://www.learnpython.org/>

DataCamp (2017) *Intro to Python for Data Science*. Available at:<http://bit.ly/2DTm47H>

Code School (2018) *Try Python*. Available at: <https://www.codeschool.com/courses/try-python>

Jupyter Nbviewer (anon.) *A Crash Course on Python*. Available at: <http://nbviewer.jupyter.org/github/ipython-books/minibook-2nd-code/blob/master/chapter1/14-python.ipynb>

5. KEATS

KEATS is essential to this module. All readings, lecture slides, website links and other digital materials will be found on this module’s KEATS page. The module convenor will post information to KEATS throughout the semester. It is therefore important that you check KEATS regularly.

6. Email Communication Policy

King’s communicates to students via their KCL email. It is therefore vital that you check your King’s email account regularly.

King’s staff will only reply to student emails if they are sent through the student’s King’s email address. Please do not use your personal email when emailing your tutors or module convenors.

7. Student Participation and Attendance

Active student participation in this module is crucial. Students are expected to contribute to debates whenever requested, to do assigned reading and tasks to show their progress throughout the module.

Students are expected to attend all lectures and seminars, and to show up on time. Attendance will be tracked via KEATS. If you are unable to attend a session for any reason, please contact the convenor/tutor straight away via email.

8. Weekly Schedule

|  |  |  |
| --- | --- | --- |
| Week 1 | 12 January | **Introduction to Python & Programming: Overview of the Aims and Concepts of the Module**  Dr Gabriele Salciute Civiliene |

**Lecture:**

This introductory lecture will outline the major objectives and basic concepts that we you will be learning in greater depth throughout the module. You will also be introduced to programming in general and role it plays in our digital culture. Within this wider frame work, we will have a look at Python as a programming language, including the Python environment and the steps of running a program.

**Seminar:**

**Required reading:**

* Downey, Allen (2012) Chapter 1: The Way of the Program in *Think Python*. O’Reilly Media, pp. 1-10.
* Matthes, Eric (2016) Chapter 1: Getting Started in *Python Crash Course*. No Starch Press, pp. 3-18.
* Matthes, Eric (2016) Appendix A: Installing Python in *Python Crash Course*. No Starch Press, pp. 485-490.
* Matthes, Eric (2016) Appendix B: Text Editor in *Python Crash Course*. No Starch Press, pp.491-497.
* The Python Tutorial (2016). Available at: <https://docs.python.org/3.7/tutorial/index.html>
* What’s New (2016). Available at: <https://docs.python.org/3.7/whatsnew/3.6.html>

|  |  |  |
| --- | --- | --- |
| Week 2 | 19 January | **Variables, Basic Data Types and Some Control Structures**  Dr Gabriele Salciute Civiliene |

**Lecture:**

This lecture will focus on how Python function syntax allows for more complex program structures as variables and will explore the manipulation of basic data types such as strings and integers. It will also introduce basic control structures such as if/ifelse statements with the different types of operators.

**Seminar:**

**Required reading:**

* Downey, Allen (2012) Chapter 2: Variables, Expression and Statements in *Think Python*. O’Reilly Media, pp. 11-18.
* Downey, Allen (2012) Chapter 3: Functions in *Think Python*. O’Reilly Media, pp. 19-30.
* Downey, Allen (2012) Chapter 5: Conditionals and Recursion in *Think Python*. O’Reilly Media, pp. 41-50.
* Matthes, Eric (2016) Chapter 2: Variables and Simple Data Types in *Python Crash Course*. No Starch Press, pp. 19-36.
* Matthes, Eric (2016) Chapter 5: If Statements in *Python Crash Course*. No Starch Press, pp. 73-94.

|  |  |  |
| --- | --- | --- |
| Week 3 | 26 January | **Complex Data Types**  Dr Gabriele Salciute Civiliene |

**Lecture:**

By building on the work done in week 2, we will explore and manipulate more complex data structures such as dict, list, tuples, indexes vs keys and we will do some string processing.

**Seminar:**

**Required reading:**

* Downey, Allen (2012) Chapter 10: Lists in *Think Python*. O’Reilly Media, pp. 87-100.
* Downey, Allen (2012) Chapter 11: Dictionaries in *Think Python*. O’Reilly Media, pp. 101-112.
* Downey, Allen (2012) Chapter 12: Tuples in *Think Python*. O’Reilly Media, pp. 112-122.
* Matthes, Eric (2016) Chapter 3: Introducing Lists in *Python Crash Course*. No Starch Press, pp. 35-52.
* Matthes, Eric (2016) Chapter 4: Working with Lists in *Python Crash Course*. No Starch Press, pp. 53-74.
* Matthes, Eric (2016) Chapter 6: Dictionaries in *Python Crash Course*. No Starch Press, pp. 95-115.

|  |  |  |
| --- | --- | --- |
| Week 4 | 2 February | **More Control Structures**  Dr Gabriele Salciute Civiliene |

**Lecture:**

This lecture will focus on the major types of control structures such as iteration and selection along with standard statements such as while/for and if/ifelse, which allow us to ask questions and then perform different actions as in string processing. We’ll do more practice to understand how loops can be used to iterate a block of code for the number of items in lists, dictionaries, string variables or tuples.

**Seminar:**

**Required reading:**

* Downey, Allen (2012) Chapter 7: Iteration in *Think Python*. O’Reilly Media, pp. 63-70.
* Downey, Allen (2012) Chapter 8: Strings in *Think Python*. O’Reilly Media, pp. 71-80.
* Matthes, Eric (2016) Chapter 7: User Input and while Loops in *Python Crash Course*. No Starch Press, pp. 117-132.
* Matthes, Eric (2016) Chapter 11: Testing Your Code in *Python Crash Course*. No Starch Press, pp. 117-132.

|  |  |  |
| --- | --- | --- |
| Week 5 | 9 February | **Libraries and Files**  Dr Gabriele Salciute Civiliene |

**Lecture:**

In this lecture, we will get familiar with the basic third-party libraries to HTML code to further learn how to scrape data from the web. You will be introduced to the basic steps of using requests and BeautifulSoup libraries to create parsed HTML trees, to extract data of your choice from a few websites and import it into csv files that can be used for further manipulation and analysis.

**Seminar:**  
  
**Required reading:**

* Downey, Allen (2012) Chapter 14: Files in *Think Python*. O’Reilly Media, pp. 133-142.
* Matthes, Eric (2016) Chapter 10: Files and Exceptions in *Python Crash Course*. No Starch Press, pp. 189-214.
* Beautiful Soup Documentation (2015). Available at: <https://www.crummy.com/software/BeautifulSoup/bs4/doc/>
* The Python Standard Library(2018). Available at: <https://docs.python.org/3/library/index.html>.

|  |  |  |
| --- | --- | --- |
| Week 6 | 16 February | **Querying Structured Data: Pandas**  Dr Giles Greenway |

**Lecture:**

Sometimes data we are interested comes to us in a structured form, although often noisy and incomplete. There are libraries that help manage these problems, without resorting to interminable loops and conditionals. Pandas dataframes are the most ubiquitous example, and are particularly  
useful as they are modeled on the similar dataframe objects in R. The equally ubiquitous Titanic survivor dataset will provide a suitable challenge.

**Seminar:**

**Required reading:**

* The Pandas Cook-Book. Available at: <https://github.com/jvns/pandas-cookbook>
* Pandas Documentation. Available at: <http://pandas.pydata.org>

|  |  |
| --- | --- |
| **Reading Week 7**  **19 February – 25 February** | **No lecture or seminar** |

|  |  |  |
| --- | --- | --- |
| Week 8 | 2 March | **Querying Structured Data: SQLite**  Dr Giles Greenway |

**Lecture:**

Much discussion in the Digital Humanities centres around databases, in any case, some familiarity with structured Query Language (SQL) will be required if any non-trivial use of the many Python web frameworks to be used. Fortunately, the Python standard library allows the use of SQLite without any additional database software. A small database of third-party libraries used by Android apps aimed at refugees will be explored.

**Seminar:**

**Required reading:**

* Python3 SQLite Documentation. Available at: [https://docs.python.org/3/library/sqlite3.html](https://docs.python.org/3/library/sqlite3.html" \t "_blank)
* W3Schools Introduction to SQL. Available at: [https://www.w3schools.com/sql/sql\_intro.asp](https://www.w3schools.com/sql/sql_intro.asp" \t "_blank)

|  |  |  |
| --- | --- | --- |
| Week 9 | 9 March | **Querying Unstructured Data: Regular Expressions**  Dr Giles Greenway |

**Lecture:**

Regular expressions allow complex search patterns for text to be specified as a string of characters. They are sometimes described as "write-only", having written and tested them, it is sometimes difficult to discern what they do or how they work afterwards! However, they are very useful and powerful, they'd have to be, in order for us to tolerate them. We will show how they can provide an "easy win" for some text-analysis tasks, without resorting to full-blown Natural Language Processing, providing you know their limitations. Analysing newspaper articles with respect to the style-guide of the Trans Media Watch charity will build on our web-scraping skills.

**Seminar:**

**Required reading:**

* The Trans Media Watch Style Guide: Available at: [http://www.transmediawatch.org/Documents/Media%20Style%20Guide.pdf](http://www.transmediawatch.org/Documents/Media%20Style%20Guide.pdf" \t "_blank)
* The Python3 Regular Expression HOWTO. Available at: [https://docs.python.org/3/howto/regex.html](https://docs.python.org/3/howto/regex.html" \t "_blank)
* Your Python Regular Expression’s Best Buddy. Available at: <http://www.pyregex.com/>)

|  |  |  |
| --- | --- | --- |
| Week 10 | 16 March | **Querying Unstructured Data: An Introduction to Natural Language Processing (NLP)**  Dr Giles Greenway |

**Lecture:**

Natural Language Processing is a complex field, but libraries and tools do allow novices to obtain useful results with relative ease. The Natural Language Toolkit (NLTK) provides a very complete set of algorithms for Python. A newer library, Spacy, takes the approach of providing one implementation for each common NLP task that will be effective for most cases. We will use it to analyze some of the novels freely available from Project Gutenberg.

**Seminar:**

**Required reading:**

* PyCon 2016 workshop Natural Language Processing in 10 Lines of   
  Code. Available at: [https://github.com/cytora/pycon-nlp-in-10-lines](https://github.com/cytora/pycon-nlp-in-10-lines" \t "_blank)
* Natural Language Processing with Python. Available at: [http://www.nltk.org/book/](http://www.nltk.org/book/" \t "_blank)

|  |  |  |
| --- | --- | --- |
| Week 11 | 23 March | **Visualizations**  Dr Giles Greenway |

**Lecture:**

In the previous sessions we have shown that it is possible to use Python to collect data and gain insights from it with relative ease. Many of these would have been more clearly expressed graphically. We will use some of our existing results to explore some of the visualization libraries available for Python. ggplot has the advantage of having an interface that will be very familiar to users of R. Matplotlib is a very fully featured library, but with a deserved reputation for being hard to use. Bokeh and Seaborn allow novices to obtain informative and attractive results with very little effort.

**Seminar:**

**Required reading:**

* Bokeh Quickstart Guide. Available at:   
  [https://bokeh.pydata.org/en/latest/docs/user\_guide/quickstart.html#userguide-quickstart](https://bokeh.pydata.org/en/latest/docs/user_guide/quickstart.html" \l "userguide-quickstart" \t "_blank)
* Seaborn Tutorial. Available at: [https://seaborn.pydata.org/tutorial](https://seaborn.pydata.org/tutorial" \t "_blank)
* Matplotlib Documentation. Available at: <https://matplotlib.org/>
* Matplotlib Basemap Toolkit Documentation. Available at: <https://matplotlib.org/basemap/>
* Rossant, Cyrille (2013) Chapter 4: Interactive Plotting and Graphical Interfaces in *Learning IPython for Interactive Computing and Data Visualization*. Packt Publishing, pp. 67-87.

9. Schedule Summary

|  |  |  |  |
| --- | --- | --- | --- |
| Week | Date | Lecture | Seminar |
| 1 | **12 January** | **Introduction to Python & Programming** |  |
| 2 | **19 January** | **Variables, Basic Data Types and Some Control Structures** |  |
| 3 | **26 January** | **Complex Data Types** |  |
| 4 | **2 February** | **More Control Structures** |  |
| 5 | **9 February** | **Libraries and Files** |  |
| 6 | **16 February** | **Querying Structured Data: Pandas** |  |
| Reading Week 7 19-25 February / No classes | | | |
| 8 | **2 March** | **Querying Structured Data: SQLite** |  |
| 9 | **9 March** | **Querying Unstructured Data: Regular Expressions** |  |
| 10 | **16 March** | **Querying Unstructured Data: An Introduction to Natural Language Processing (NLP)** |  |
| 11 | **23 March** | **Visualizations** |  |

10. Assessment

The module is assessed through three components of evaluations over the semester. These components include 10 Labs (20% altogether) as a form of continuous assessment, Mid-term Project 1 (30%) and End-term Project 2 (50%).

**Labs**

Each weekly lab assignment involves Python coding based on the material taught during the lecture and seminars of a week (see Section 11 for details of submission deadlines).

**Mid-term Project 1**

Project 1 is an individual assignment. Using Jupyter notebooks and Python 3, write a program which requests a body of text from a source on the Internet and then calculates and displays to the user the total number of unique words in the text, and the mean average length of all words in the text. You may choose any text, and you may also provide other summary statistics of your choosing. 🡨 GSC Comment [the phrasing suggests that they may but do not have to do it. If they choose not to provide other summary stats, the code will be very lean. My suggestion is that we tell them to provide other stats summary for sure but they can choose what exactly they want to provide. We also need to define the length of their code so that their codes are not of wildly variable length, which would then cause a problem for us to mark them].

Your code should be accessible, easily readable, and well documented throughout.

You should complement code cells with text cells explaining what you are doing in your code and why, and discussing the rationale behind any decisions you have made. Submit your Project 1 assignment in the format of a Jupyter notebook via the designated area on the module’s page on KEATS before 4pm on 26th February.

**End-term Project 2**

Project 2 is an individual assignment that consists of two components. One is a Python3 code authored by you, and the other is a written document in which you need to write up your code as part of your research proposal for a big data harvesting project.

For the first component, use Jupyter notebooks and Python3 to harvest data from the web and to visualize the harvested data. Visualization has two aims - that is, analytical and illustrative. On the analytical side, you need to produce visuals by using Python in order to reveal hidden patterns in the harvested data. On the illustrative side, your visuals, produced with Python3, should summarize the findings of data analysis to back up your preliminary inferences. You may use third-party libraries other than taught during the classes of this module.

Make sure that, in your written proposal, you refer to your code in Jupyter notebooks and cite your code snippets in-text to explain what objectives the parts of your code serve to achieve. You may reuse the entire code written by somebody else, but only if you set out to improve it in a considerable way.  Be aware of plagiarism (see Section 12 for more details). Even when you work on improving the existing code, you need to acknowledge it.

Your written component (up to 1,500 words) has to make a research proposal addressed to some institution within or outside the academia. The document needs to be organized around the following rubrics:

* **Background** (a brief introduction to the problem area to explain how harvesting the data of your choice will help address that problem)
* **Method and Approach** (a brief outline of data type(s), data sources, steps of data collection, steps of data analysis, methodological/technical limitations)
* **Analysis** (a brief discussion of preliminary findings, such as patterns found in datasets , and a critical comment on how your findings relate to the existing body of research)
* **Expected Outcomes** (a brief summary to explain how this data harvesting exercise can grow and evolve into a big data project)

You may break the major headings into smaller sub-headings to signal the logical structure of your proposal. Use visuals mainly in the Analysis, but you may have an odd one in other parts as well. Your visuals need to be labelled as Figure 1, Figure 2, etc and to have captions that briefly explain the essence of a particular visual. Make sure your proposal is written in a coherent manner and formatted appropriately. Use the Harvard style for in-text citations and reference lists (see <http://libguides.kcl.ac.uk/reference/KingsHarvardV1>).

Both the components of the End-term Project 2 have to be submitted before 4 pm on 8th May 2018. Submit your code component in the format of a Jupyter notebook via the designated area on the module’s page on KEATS. The written proposal component needs to be submitted as a Word document also via the designated area on the module’s page on KEATS.

In case of failure,students are reassessed in the failed elements of assessment and by the same methods as the first attempt.

11. Assignment Deadlines

All the components of this module are to be submitted electronically via the Turnitin section on the module’s KEATS page.

Lab 1 (2%) 18/01/2018– *no later than 4:00pm*

Lab 2 (2%) 25/02/2018 – *no later than 4:00pm*

Lab 3 (2%) 01/02/2018– *no later than 4:00pm*

Lab 4 (2%) 08/02/2018 – *no later than 4:00pm*

Lab 5 (2%) 15/02/2018 – *no later than 4:00pm*

Lab 6 (2%) 01/03/2018 – *no later than 4:00pm*

Lab 7 (2%) 08/03/2018 – *no later than 4:00pm*

Lab 8 (2%) 15/03/2018 – *no later than 4:00pm*

Lab 9 (2%) 22/03/2018 – *no later than 4:00pm*

Lab 10 (2%) 29/03/2018 – *no later than 4:00pm*

Mid-term Project 1 (30%): 26/02/2018 – *no later than 4:00pm*

End-term Project 2 (50%): 08/05/2018 – *no later than 4:00pm*

12. Citing and Plagiarism

The same rules and principles for citation and plagiarism apply to code as to essays. You may use and reuse any code that is authored by other programmers or your classmates. You may download and include Python libraries if they are not restricted by copyright); you may adapt parts of code you find online; you may include modules written by your classmates; you may even copy code directly. However, you must always acknowledge the authorship of other people and sources. How to cide code properly see the rubric ‘Citing within your source code’ in the King’s Guide to Referencing (<http://libguides.kcl.ac.uk/reference/KingsHarvardV1>). If you fail to acknowledge the provenance of the code that you copied, you may be guilty of plagiarism in exactly the same way as you would be for copying text in an essay.

If you cite your classmates, make sure you do so by candidate number as otherwise your classmates will no longer be anonymous to the markers. If you chose to include libraries or modules written by other people, you must include these libraries with your submission, otherwise your markers will not be able to test your code.

13. Essay/Report Marking Criteria

Marking criteria for Level 5 modules are set out in the Faculty of Arts & Humanities’ Undergraduate Generic Marking Criteria framework. See full details at <http://www.kcl.ac.uk/artshums/study/handbook/sguides/assessment/ugmarkcrit.pdf>.

**80 – 89 Outstanding** (First / Grade A+)

Highly thoughtful answer informed by wider reading, showing clarity of thought, personal insight and originality.

* Understanding: Thorough understanding of the relevant material and issues with evidence of evaluation in the discussion. Independent and critical evaluation.
* Depth of Knowledge: Full range of sources used and applied in a highly focused manner.
* Structure: Clear and fluent style. Very well focused and structured.

**70 – 79 Excellent** (1st / Grade A)

Thoughtful answer informed by wider reading showing clarity of thought and personal insight.

* Understanding: Thorough understanding of the relevant material and issues demonstrating insight and a good level of evaluation.
* Depth of Knowledge: Comprehensive range of relevant literature, evidence is used to support arguments, awareness of wider issues.
* Structure: Clear, logical and integrated presentation.

**60 – 69 Good** (Upper 2nd / Grade B)

Good understanding of basic principles and relevant evidence, with a coherent and logical argument.

* Understanding: Good understanding of the relevant material and issues with development of analytical thought.
* Depth of Knowledge: Good use of relevant literature.
* Structure: Coherent, well organised and logical presentation.

**50 – 59 Fair** (Lower 2nd / Grade C)

Sound understanding demonstrated with some analysis.

* Understanding: Sound understanding of the relevant material and issues is demonstrated, evaluative thought is apparent in some areas.
* Depth of Knowledge: Appropriate reading is demonstrated to support the discussion.
* Structure: Clearly presented but little development.

**40 – 49 Pass** (3rd / Grade D)

* Understanding: General knowledge demonstrated but the work is mainly descriptive.
* Depth of Knowledge: Sparse coverage of basic literature / material. Low quality in a number of areas and poor range of reading.
* Structure: Adequate presentation. Some unclear sections.

**33 – 39 Fail**

* Understanding: Some knowledge but does not focus on the question or is very limited. Descriptive work with little recognisable analysis.
* Depth of Knowledge: Inappropriate literature/material used in assignment. Key texts missing.
* Structure: Disorganised/unclear presentation with loose ends.

1. This book is meant for those who already have some knowledge and experience with Python. [↑](#footnote-ref-1)