Course Manual

Programming Skills

SOW-BS85

2015 - 2016

Written by:

Dr. Thijs Verwijmeren

Dr. Ron Dotsch

CONTENTS

CONTACT INFORMATION

COORDINATOR:

Dr. Thijs Verwijmeren t.verwijmeren@psych.ru.nl
Office: A.09.24

SCHEDULE

All lectures will be on Tuesdays from 13:45 – 15:30 in room TvA 6.00.15. Lab sessions are on Thursday from 13:45 – 15:30 in room SP.A.-1.55.

Note that there are no lectures on 3-5 and no practicum on 5-5.

Note that lecture 4 is not on the usual day, time and place but on **Wednesday 11-5 on 10:45 at TvA 1.0.35**.

Week	Date	Time	Topic
15	12/4	13:45	How to think like a computer scientist and intro to python
16	19/4	13:45	Simple OO, Functions, Modules, Math, Random
17	26/4	13:45	Decisions, Selection, Iteration, Images
19	**11/5**	**10:45**	Strings, Lists, Tuples
20	17/5	13:45	Files, Dictionaries, Classes, and Objects
21	24/5	13:45	PsychoPy: Basics
22	31/5	13:45	PsychoPy: Programming experiments I
23	7/6	13:45	PsychoPy: Programming experiments II
25	23/6	23:59	**Deadline take-home exam**
26	30/6	12:45	**Written exam**
34	25/8	12:45	**Retake written exam**

GOAL OF THE COURSE

Conducting experimental and non-experimental research in the psychological and educational sciences requires more advanced computer skills than ever before. The modern researcher ought to be able to program in general purpose as well as specialized statistical programming languages in order to efficiently process and combine raw data, visualize those data in meaningful ways, and create new experimental tasks without being limited by the tools conventional programs offer. Programming skills make a researcher's life much easier. Manually processing data will be a thing of the past. This course focuses on learning to program experiments, although the acquired skills will be useful for data processing, analysis and visualization as well.

You will learn how to think like a computer scientist. You will learn how to program in Python, a free, open-source, platform independent, and continuously maintained programming language. Once you know how to program in python, it will be much easier for you to learn other, more specialized, languages (such as Matlab, R, or Presentation), although Python can do (or will be able to do so in the near future)most of what many of these other languages do.

LEARNING GOALS

At the end of this course:

1. You will be able to think in terms of **algorithms** (i.e., like a computer scientist)
2. You will have a working **knowledge** of the Python programming language
3. You will be able to **program experiments** in Python
4. You will be able to **debug** (fix) Python code

These learning goals will be assessed and trained as follows:

Assessment	**1: algorithms**	**2: knowledge**	**3: programming**	**4: debugging**
Homework	X	X	X	X
Take-home exam	X	X	X	X
Written exam	X	X		X

Training	**1: algorithms**	**2: knowledge**	**3: programming**	**4: debugging**
Self-assessments e-book	X	X		
Homework	X	X	X	X
Grading each other's homework		X		X
Practice exam	X	X		X

General structure of the course

This course is divided into two parts: Part I: How to think like a computer scientist and Part II: Programming experiments. In Part I you will learn how to think algorithmically, acquire basic knowledge of the Python programming language and how to debug python code. In the second part, you will learn how to program experiments, using PsychoPy, a free and open source module that gives Python all the functionality needed to program psychological experiments.

There are 8 weekly lectures, each followed by a lab session (see schedule above). In the lab sessions, you first grade someone else's homework, after which you start working on your homework assignments. The advantage of sticking around in the lab sessions is that I will be there to help and answer questions. After that, you're on your own (but don't forget to ask help and help other students on the Blackboard forum).

Grades for the course will be based on a grade for a final programming assignment (take home exam) as well as a grade for a written exam (see below). Homework assignments are not graded, but need to be handed in and assessed as sufficient in order to pass the course.

Literature

There are no books to buy, you will be using online documentation for reference (http://www.python.org/doc/).

For Part I, you will be using the excellent online and free book "How to think like a computer scientist" (http://interactivepython.org/courselib/static/thinkcspy/index.html).

For Part II, you will rely mostly on the documentation provided by the PsychoPy developers (http://www.psychopy.org/documentation.html).

Homework

For the first five homework assignments, you do not need to install Python on your computer. They all work directly in your browser.

Each week, during the lecture, you will receive the homework assignment for that week. For the first five weeks, these can be completed on-line in the tutorial in any modern web browser. Once you complete an assignment, you print your answers on hard copy and bring them to the lab session (for that, you probably need to copypaste it into a document. Please don't use Microsoft Word, because it screws up the indentation, but a nice and simple text editor).

The hard copy will be graded by your peers based on the answer model provided through Blackboard. Importantly, there are multiple ways in which a problem can be solved, so sometimes deviations from the answer model are fine too. When in doubt, call in my help.

You will not receive grades for the assignments, but for any mistake identified by your peers, you will have to write down on the hard copy print out what will go wrong because of the mistake and how it needs to be fixed to show that you really understand what your mistake was. After that, hand in your assignment with me.

The homework assignments are **required** and they need to be handed in during the lab session. You only pass the course if you hand in all homework assignments. If you miss a lab session or homework assignment for a good reason (and I will be the judge of that), I will provide you with an alternative assignment (which may be a lot harder than the original homework).

GRADES

Your grades consist of the average of a written exam (50%) and a programming assignment (50%). Additionally, you need to fulfill the homework requirements (see above).

WRITTEN EXAM

The written exam with open questions will test your knowledge of python, your ability to think algorithmically, and your ability to debug. You may for example be asked to explain what a particular block of python code does, or to explain why a given block of python code does not work.

The exam will encompass all material from the lectures and the 'How to think like a computer scientist' e-book. The best way to prepare for the exam is to program a lot and to practice with the practice exams (last year's first and second attempt, which I will make available through Blackboard later in the course).

TAKE HOME EXAM: PROGRAMMING ASSIGNMENT

The take home exam entails programming an experiment. The precise experiment that you will need to program will be announced during the last lecture (week 23). Submit the take home exam on Blackboard. The deadline for the programming assignment is June 23, 23:59. Needless to say: you do the take home exam on your own and you are not allowed to help each other.

EVALUATION CRITERIA TAKE HOME EXAM

The programming assignment script will be graded on:

1. Functionality (does the code do what it needs to do? No exceptions raised?)
2. Style (is the code efficient, readable, elegant?)
3. Reusability (is the code modular, general instead of specific solution, easy to re-use for other purposes?)
4. Documentation (is the code readable? Are comments provided? Are the dependencies clear?)

If the code does not run properly, it will be graded with a 1, so make sure your code works before submitting.

PYTHON CHALLENGE

Last, you will get a special Pythonista certificate, stating that you are an awesome Python programmer, if you are able to complete the optional python challenge (http://www.pythonchallenge.com). Provide evidence that you completed the challenge by providing screenshots of every python challenge riddle and scripts for your solutions. Importantly, only try this if you think the course is too easy for you. Last year, no one managed to complete the challenge.