Syllabus sample 15 Apr. 2014

Programming Fundamentals (50:198:111) — Spring 2014 or other

Jean-Camille Birget
Office: Business and Science Bldg. 320, (856) 225-6653
`birget@camden.rutgers.edu` `http://clam.rutgers.edu/~birget/`

Class times and room: M. W. 1:20 - 2:40pm, BSB335.

Office hours: M. W. 2:50 - 4:00pm, or by appointment.

Course work: Programming assignments and other homework, worth 20% of the total grade. Three in-class exams, worth 10% (exam 1), 35% (exam 2), 35% (exam 3).

Exam dates: *Exam 1,* Wed. **26** Feb., in class. *Exam 2,* Wed. 2 Apr., in class. *Exam 3 (Final),* 11:30 - 2:30 Fri. 9 May, in classroom.
[Deadline to drop with W: W. 9 Apr. Last day of class: M. 5 May.]

Course description: Fundamental concepts of structured programming and algorithmic problem solving: primitive data types, control structures, functions and parameter passing, files, and the mechanics of writing, running, testing, and debugging programs. These concepts will be taught using the high-level language Python. File systems, and some commands in the Unix operating system.

Learning goals:
- Understanding the fundamental instructions of the most common programming languages:
    - arithmetic and string operations,
    - data types,
    - conditionals,
    - iterations,
    - functions (and procedures) and function calls,
    - basic data structures,
    - usage of files in programming.
- Writing and executing simple Python programs, in accordance with a specification.
- Analyzing a simple specification and a simple program, in order to check whether the program meets the specification.
- Using simple Unix operating system commands.
- Knowing the basic structure of a computer: central processor, storage (memory), internal communications (e.g., buses), input-output.

Connection with learning goals of General Education as a whole:
Goal *1. Knowledge of Human Cultures and the Physical and Natural Worlds, through study in the sciences and mathematics, ...* :

The course fits into mathematics.

Goal *2. Intellectual and practical skills, including*
Inquiry and analysis
Critical and creative thinking
Written and oral communication
Quantitative literacy
Information literacy
Problem solving:

The course fits into *Inquiry and analysis* since program specifications and programs need to be analyzed, so that one can check that the program satisfies the specification. *Critical and creative thinking*, as well as *Problem solving*, play a crucial role in the development of programs. In programming and in most of computer science, mathematical and logical skills are essential, so *Quantitative literacy* matters; however, the ability to think mathematically is much more important here than mathematical knowledge.

Connection with learning goals of in Logical and Quantitative Reasoning:

1. Analyze and evaluate mathematical or logical arguments.

This is done in the analysis of program specifications, and of programs, with the goal of checking whether the program satisfies the specifications.

2. Demonstrate an understanding of the scope and limitations of logical reasoning, including the nature of rational norms, formal languages, and logical paradoxes.

Programming uses a formal language. Logical reasoning is used in the development of a program.

Textbook (required):

P. Wentworth, J. Elkner, A. Downey, Ch. Meyers:
How to Think Like a Computer Scientist, Learning with Python 3,
`http://openbookproject.net/thinkcs/python/english3e/` (online, in html and in pdf).

Other books (not required):

- John Zelle, *Python Programming* (**2**nd ed.), Franklin, Beedle and Associates (2010).
- Mark Lutz, *Learning Python* (**4**th edition, 2009), O'Reilly Media. ISBN-13: 978-0-596-51398-6.
- John V. Guttag, *Introduction to Computation and Programming Using Python*, rev. ed., MIT Press (2013).
- M. Pilgrim, *Dive into Python*, 2nd ed., Apress (2009). [Advanced.]

On-line references:

- See `http://clam.rutgers.edu/~birget/cs211/UnixCRefs` , where you can find:
  Python tutorial (very useful), index, standard library, documentation overview.
- Richard Gruet's Python page (up to v2.7): `http://rgruet.free.fr/#QuickRef`

On Unix:

- List of Unix utilities: `http://en.wikipedia.org/wiki/List_of_Unix_utilities`
- Guide to Unix/Commands: `http://en.wikibooks.org/wiki/Guide_to_Unix/Commands`
- M.A. Thomas, Intro. to Unix: `http://www.ucblueash.edu/thomas/Intro_Unix_Text/TOC.html`

Pre-requisites: None.

General Education course: The course can be taken by students in all subjects. It has no official pre-requisites, and the mathematical knowledge expected is at the middle-school level; however, mathematical skills and the ability to think logically are important. The course has intellectual content that is useful in many fields.

Grading policy: Programming assignments are expected to be done individually. Discussions about the ideas of an assignment and about general information with fellow students is encouraged; but the actual coding and writing of the programs should be done completely independently. Copying (or jointly writing) significant portions of a program is considered cheating. Many exam questions will be very similar to homework problems; the homework is intended in large part to prepare you for the exams. Grading scale: [0 F [60 D [65 D+ [70 C [75 C+ [80 B [85 B+ [90 A 100].

Class and test attendance is required; having missed classes is not an acceptable excuse for not having information that was given in class. Homework due dates are firm. Unless you have a major medical or personal emergency, late homework will not be accepted. The grade "incomplete" is given only when justified according to University policy.

Class etiquette

In class the following are to be avoided, due to the noise or distraction that they cause: Cell-phone ringing or talking or cell-phone listening, texting, gaming, earphones, radios, computer sounds and noisy typing, chatting, eating. Feel free to (quietly) leave the room if you absolutely need to do any of the above.