

Introduction to Programming

MET CS201 SB1 (Summer 2019)

Syllabus

Course Description

MET CS201 provides an introduction to computer programming. While the skills taught in this course apply to any programming language, the class discussions, homework exercises, exams, etc., utilize the Python language.

The Python language features to be covered include, but are not limited to: strings, lists, dictionaries, sets, tuples and iterators; flow of control constructs such as 'if' and 'while' statements; functions; file input and output; graphics, graphing and graphical user interface programming. Additionally the course introduces fundamentals of object-oriented programming, application design and software development.

Time permitting, additional Python features and programming techniques may be covered, including comprehensions, classes and recursion.

The course also will require students to formulate solutions for certain types of problems (for instance, to efficiently locate a value in a list), to write clear and efficient Python code to implement these algorithms, and to produce fully-tested, debugged and working programs.

Class time will be divided between lecture and in-class exercises.

Homework exercises will consist mostly of programming. Students will be required to meet regularly with the instructor to discuss their projects, to demonstrate their working programs, and to explain the thinking behind the code they write.

By the end of the course, students will be able to:

- (1) explain basic concepts of how programs work;
- (2) explain how to solve certain types of problems with a program;
- (3) write short programs in Python without referring to documentation;
- (4) explain the nature of the Python features covered in this course, and to provide examples of their use.

Prerequisites

This course is designed for students who are interested in learning to program. No prior programming experience is required. Good algebra and logic skills are strongly recommended. This course will move quickly through a wide range of programming language features that students will be expected to master quickly. Please contact me before registering if you would like to better understand the depth and pace of this course.

Laptops are required at each class.

Location and Time

Wednesdays 6:00pm-8:45pm in CGS 521 from Wednesday, September 4 to Wednesday, December 11, 2019. A final will be held on Wednesday, December 18 from 6:00pm-8:45pm in CGS 521.

Instructor

John Keklak	TBA
Office hours:	no fixed hours; send an email to request a time to meet
Office location:	via teleconferencing only
Email:	jkeklak[at]bu[dot]edu

Required Texts

No textbook is required. We will be using [the Python Software Foundation on-line documentation](#) and online notes provided in Blackboard.

Readings and Exercises

Reading and exercises will be assigned for each week. Students must reproduce the Python examples provided in the online readings, and also must complete all exercises at the end of each chapter. This code must be submitted to Blackboard.

Solutions to the exercises are provided following the exercises.

It is strongly recommended that students devote time every day to do a portion of assigned readings and exercises.

Additionally students must submit code written for class exercises.

Assignment Grading

Assignments are graded as "done" or "not done". Written responses and code in submissions must be clearly-written and well-formatted. Poorly formatted or incomplete code and written responses will be considered "not done". It is recommended that you consider the following practices when preparing your submissions:

- (1) complete and well-written sentences and paragraphs for written responses
- (2) a consistent coding style
- (3) a consistent naming convention for variables, functions and files
- (4) use of ample white space
- (5) use of names that convey intended meanings
- (6) hiding unnecessary detail
- (7) comments that explain code that you suspect will puzzle someone other than you
- (8) rewriting (refactoring) your code once it works

Timely Completion of Assignments

With the very rapid pace of the course, it is extremely important that students complete homework and class exercises on time. Class exercises, quizzes and exams will be very difficult for students who have not properly worked through the examples and exercises in the reading.

In-Class Exercises

A portion of class time will be reserved for in-class exercises to create applications using Python. All students are required to bring laptop computers suitable for writing programs in Python to each class meeting. During this time, students will regularly confer with the instructor to discuss their class exercise work.

Exams and Quizzes

There will be four short quizzes during the semester (one quiz approximately every other week), and a final exam during finals week. All quizzes and the final exam are closed book, closed computer.

Grading

Grades will be determined by the following weighting:

Quizzes	-	40%
Final	-	60%
Instructor's discretion	-	+/-10%

Incompletes will not be given.

Quiz grades may be viewed on the course grade sheet in BlackBoard.

Grade conversions:

A	93-100
A-	90-92.999...
B+	87-89.999...
B	83-86.999...
B-	80-82.999...
C+	77-79.999...
C	73-76.999...
C-	70-72.999...
D	60-69.999...
F	59.999... and below

Grades are not curved. The quizzes and the final exam are designed such that a student who has diligently completed the homework and in-class assignments, and has participated in all class meetings can realistically earn an A. The quizzes and final exam will not contain any "trick questions", but rather will determine a student's level of fluency with Python and programming.

Attendance

Class meetings will be a crucial component of the learning process, and therefore attendance is essential. Class will include one-on-one discussions with the instructor. Lecture will complement and supplement the reading material. Attendance will be taken at the beginning of each class. Absences will significantly affect both your Python skill and your grade.

Be sure to arrive to class on time, since it is highly disruptive to have students flowing in throughout the class period. Late arrivals will adversely affect your grade.

Attention

Students are expected to pay attention to the lecture, and not to text or engage in any other activity that diverts their attention from the lecture. Such activity also distracts others, reducing their ability to follow the lecture. All phones, etc. must be off during class.

Collaboration

All course participants must adhere to the Boston University Metropolitan College academic conduct code. Printed copies of the code are available from the college. All instances of academic dishonesty will be reported to the appropriate academic conduct committee.

The material you submit for assignments must be your own original work and it is an act of plagiarism to represent the work of another as your own. You are welcome to discuss the general concepts in assignments with other students in the course, but it is not acceptable to share, to post, or to copy code or written answers to homework questions. It is not acceptable to post homework or class exercise questions or solutions anywhere, including on tutoring websites. If you are uncertain whether an action constitutes a violation of the academic conduct policy, I will be glad to discuss the matter with you.

In other words, you may discuss concepts with other students, and you may help (or receive help from) other students in preparing your programs and written responses. However, generally* all the material you submit must be code and/or prose that you -- and only you -- wrote. Your code and prose may not include material written by others, or derived from the material of others, regardless of how much you edit such material. In addition to avoiding violations of the collaboration policy, writing your materials from scratch will help you gain a clearer understanding of the concepts and principles presented in this course. (*There may be exceptions for team work where each member of a team contributes an assigned portion of a program.)

Email Responses

I will check my email each morning at about 9am, and will reply to your emails within 24 hours, barring some unforeseen circumstance.

Expected Course "Roadmap" (subject to change)

This course is a rapid introduction to the Python programming language and to the software development process. The first six weeks, in particular, will require a very significant amount of time for homework -- as much as 4-8 hours each week, not including time spent in class on in-class exercises. The remainder of the semester will focus on using Python to create a range of applications.

		Topics	Assignment (to be completed before next class meeting)
September	4	Intro to programming and Python printing messages strings and lists basic math and logic the IDLE debugger	Introduction, Chapters 1-4
	11	Applications and class exercises using strings, lists, math	Chapters 5-7
	18	More math, including random numbers functions Python and memory More on strings and lists Quiz 1 (covers Introduction through Chapter 4)	
	25	Applications and class exercises using strings, lists, math and functions	Chapters 8-9

October	2	<p>dictionaries</p> <p>file input and output</p> <p>JSON, CSV, TSV file formats and data exchange</p> <p>Quiz 2 (covers Chapters 5-7)</p>	
	9	<p>Applications and class exercises using dictionaries and files</p>	Chapters 10-11
	16	<p>Graphics, graphing, and graphical user interfaces</p> <p>Quiz 3 (covers Chapters 8 and 9)</p>	
	23	<p>Applications and class exercises using graphics, graphics and graphical user interfaces</p>	
	30	<p>Applications and class exercises: tbd</p> <p>Quiz 4 (covers Chapters 10 and 11)</p>	
November	6	<p>Applications and class exercises: tbd</p>	
	13	<p>Applications and class exercises: tbd</p>	
	20	<p>Applications and class exercises: tbd</p>	

	27	No class	
December	4	Applications and class exercises:tbd	
	11	Applications and class exercises:tbd Where to go from here	
	18	Final	

Applications include utilities, games and problem solvers such as:

* image generators

* loan payment calculator

* curve fitting

* tic-tac-toe

* poker

* sudoku solver

* subway solver