CityUniversity of Seattle

Syllabus

SCHOOL OF TECHNOLOGY AND COMPUTING

CS 151: Programming with Python

5 Credits
Effective: Winter 2019

Access to the Internet is required.
All written assignments must be in Microsoft-Word-compatible formats.
See the library's APA Style Guide tutorial for a list of resources that can help you use APA style.

FACULTY

Faculty Name: Maria Schenk, M.S.

- Blackboard - Questions & Answers Discussion Forum
    - If you have a question or concern that should be heard by all of the class then, whenever possible, please post your question in the Questions & Answers Discussion Forum.
- Email: mkschenk@cityu.edu
    - Please feel free to contact me with any questions or concerns. Please try contacting me by email first. I check email daily and always respond, so if you don't receive a reply, assume that I may not have received your email and either send another email or contact me at (360) 798-9825.

COURSE DESCRIPTION

This course provides an introduction to programming using Python. Students learn the fundamental programming concepts of process, iteration, selection, functions, classes, and objects. Students learn to apply key data structures and algorithms in their programs. Throughout the course students work in an immersive environment creating numerous programs to exercise their knowledge. At the end of the course, students have the ability to apply programming to many common problems and a solid foundation for more advanced programming concepts and challenges.

COURSE RESOURCES

Required and recommended resources to complete coursework and assignments are available from the Course Document Lookup.

CITYU LEARNING GOALS

This course supports the following City University learning goals:

- Professional competency and professional identity

COURSE OUTCOMES

In this course, learners:

- Explain a program as an organized set of instructions written in a programming language to solve a computing problem.
- Plan, design, and test an application using an integrated development environment.
- Develop simple programs using constants, variables, operators, data types, data structures, and simple algorithms.
- Design basic program flow using sequence, iteration, and selection control structures.
- Describe the relationship between data input, processing, and output.
- Utilize built-in functions to obtain program input, process data, and display output.
- Describe the relationship between classes and objects in object-oriented programming.
- Construct classes out of properties and methods.
- Instantiate objects from classes for use in object-oriented programs.
- Demonstrate quality coding practices to produce documented, legible, maintainable, reusable and efficient code.

CORE CONCEPTS, KNOWLEDGE, AND SKILLS

- Access the functions in a module by importing
- Apply building a new string
- Classes of Exceptions
- Closing Files
- Create a "Hello World!" program in python.
- Create and import a Python package
- Create functions
- Define algorithms
- Demonstrate counting with a for loop
- Demonstrate how to instantiate an class object in Python
- Demonstrate how to use inheritance to reuse a code set
- Demonstrate methods of expressing algorithms
- Demonstrate the use of an object's methods in code
- Demonstrate the use of parameters and return values
- Describe a Class
- Describe inheritence
- Describe the basic List methods
- Describe the process for researching a standard library or contributed package
- Download and install python.
- Employ the use of tuples
- Examine the history of Python.
- Explain a class' scope and why this is important to a programmer
- Explain a List, Tuple, and Dictionary
- Explain how lists can be used like arrays.
- Explain string immutability
- Explain the difference between a class and object
- Explain where python is used.
- Finding Your Way Around Files
- Handling Simple Exceptions
- Implement standard library functions
- List and give examples of common data types.
- List the three requirements for a Python package
- Opening Files
- Outline the functions available in the Standard Library.
- Practice the use of indexing with strings
- Reading from and Writing to Files
- Recognize that there are standard Python libraries of modules
- Review the use of Python IDLE.
- Show how variables are used to store data.
- Show slicing strings
- Show the use of global variables and constants
- Understand how and why python was created.
- Use executable statements to initialize a module
- Use keyword arguments and default parameter values
- Use sequence operators and functions with strings

- Use statements as the basic building block for python programs.
- Use the dir() function to identify defined modules
- Using the try Statement with an except Clause in Python

OVERVIEW OF COURSE GRADING

The grades earned for the course will be derived using City University of Seattle's decimal grading system, based on the following:

Overview of Required Assignments	***% of Final Grade***
Assignment 1: Core Objects, Variables, and IO	10%
Assignment 2: Conditional Statements and Iterative Statements	10%
Assignment 3: Structured Programming and Functions	15%
Assignment 4: Data Processing and Exception Handling	15%
Assignment 5: Object Oriented Programming	20%
Chapter Quizzes	10%
Instructor Determined Activities (incl. Discussion Board, Muddiest Point)	20%
TOTAL	**100%**

SPECIFICS OF COURSE ASSIGNMENTS

The instructor will provide grading rubrics that will provide more detail as to how this assignment will be graded.

Assignment 1 – Core Objects, Variables, and IO

Students will develop a simple program written in Python to solve a problem statement provided by the instructor. The program must include 10 to 25 sequential instructions (excluding comments) to convert input into required output and apply the following concepts: Core Objects, Variables, and Input and Output. Students must provide evidence that they have followed all stages of the Programming Lifecycle Process, including: Problem Statement, Problem Analysis, Logical Design, Coding, Editing (Debugging and Testing), and Documentation. Debugging log must be included as an appendix and will include sorted date and time stamps. Students will also write a brief reflection (one or two paragraphs) about new skills or insights they learned about programming from developing this program and how they can apply them to their current or future work.

Components	*% of Grade*
Problem Statement	5%
Problem Analysis	15%
Logical Design	20%
Coding	20%
Editing (Debugging and Testing)	20%
Documentation	10%
Reflection	10%
TOTAL	**100%**

Assignment 2 – Conditional Statements and Iterative Statements

Students will develop a simple program written in Python to solve a problem statement provided by the instructor. The program must include 15 to 40 instructions (excluding comments) that include sequential, conditional, and/or iterative programming structures to convert input into required output and apply the following concepts: Conditional Statements and Iterative Statements. Students must provide evidence that they have followed all stages of the Programming Lifecycle Process, including: Problem Statement, Problem Analysis, Logical Design, Coding, Editing (Debugging and Testing), and Documentation. Debugging log must be included as an appendix and will include sorted date and time stamps. Students will also write a brief reflection (one or two paragraphs) about new skills or insights they learned about programming from developing this program and how they can apply them to their current or future work.

Components	*% of Grade*
Problem Statement	5%
Problem Analysis	15%
Logical Design	20%
Coding	20%
Editing (Debugging and Testing)	20%
Documentation	10%
Reflection	10%
TOTAL	**100%**

Assignment 3 – Structured Programming and Functions

Students will develop a simple program written in Python to solve a problem statement provided by the instructor. The program must include 15 to 40 instructions (excluding comments) that include sequential, conditional and/or iterative programming structures, and functions to convert input into required output and apply the following concepts: Structured Programming and Functions. Students must provide evidence that they have followed all stages of the Programming Lifecycle Process, including: Problem Statement, Problem Analysis, Logical Design, Coding, Editing (Debugging and Testing), and Documentation. Debugging log must be included as an appendix and will include sorted date and time stamps. Students will also write a brief reflection (one or two paragraphs) about new skills or insights they learned about programming from developing this program and how they can apply them to their current or future work.

Components	*% of Grade*
Problem Statement	5%
Problem Analysis	15%
Logical Design	20%
Coding	20%
Editing (Debugging and Testing)	20%
Documentation	10%
Reflection	10%
TOTAL	**100%**

Assignment 4 – Data Processing and Exception Handling

Students will develop a simple program written in Python to solve a problem statement provided by the instructor. The program must include 20 to 50 instructions (excluding comments) that include sequential, conditional and/or iterative programming structures, and customized and/or built-in functions from standard libraries to convert input into required output and apply the following concepts: Reading a file or set of files, Processing a file, Outputting a file, and Exception Handling. Students must provide evidence that they have followed all stages of the Programming Lifecycle Process, including: Problem Statement, Problem Analysis, Logical Design, Coding, Editing (Debugging and Testing), and Documentation. Debugging log must be included as an appendix and will include sorted date and time stamps. Students will also write a brief reflection (one or two paragraphs) about new skills or insights they learned about programming from developing this program and how they can apply them to their current or future work.

Components	*% of Grade*
Problem Statement	5%
Problem Analysis	15%
Logical Design	20%
Coding	20%
Editing (Debugging and Testing)	20%
Documentation	10%
Reflection	10%
TOTAL	**100%**

Assignment 5 – Object Oriented Programming

Students will develop a simple program written in Python to solve a problem statement selected from a list provided by the instructor. The program must include 20 to 50 instructions (excluding comments) that include sequential, conditional and/or iterative programming structures, customized and/or built-in functions from standard libraries, and classes, objects, properties, and methods to convert input into required output and apply the following concepts: Classes, Objects, and Inheritance. Extra credit may be offered for programs with additional relevant functionality. Students must provide evidence that they have followed all stages of the Programming Lifecycle Process, including: Problem Statement, Problem Analysis, Logical Design, Coding, Editing (Debugging and Testing), and Documentation. Debugging log must be included as an appendix and will include sorted date and time stamps. Students will also write a brief reflection (one or two paragraphs) about new skills or insights they learned about programming from developing this program and how they can apply them to their current or future work.

Components	*% of Grade*
Problem Statement	5%
Problem Analysis	15%
Logical Design	20%
Coding	20%
Editing (Debugging and Testing)	20%
Documentation	10%
Reflection	10%
TOTAL	**100%**

Chapter Quizzes

Students will complete five quizzes that consist of multiple choice or matching questions and test students on foundational concepts and methods of programming in a selected general programming language. Additional instructions will be provided by the course instructor.

Components	*% of Grade*
Accuracy of Answers	100%
TOTAL	**100%**

Discussion Board Activities

Students will complete a set of activities that support the course outcomes and major assignments of the class. These activities are done on the discussion board and could include: preparation for major assignments, discussion of relevant/current topics, knowledge checks, content exploration, peer-review, journals, other alternate online tools, and/or other activities as determined by your instructor. Descriptions are provided by the instructor in the course

Components	*% of Grade*
Engagement	100%
TOTAL	**100%**

COURSE POLICIES

Late Assignments

Assignments must be submitted on due date.

Participation

In an online course, participation in the weekly discussions/activities is essential. Make sure to log in often during the week. Start participating in the weekly discussion as soon as possible after you have had a chance to complete the weekly reading assignment. Your initial discussion post must be made by Thursday. Make sure to monitor your posts for comments. It is expected that students will make discussion board posts on at least 2 days during the week.

Professional Writing

Assignments require error-free writing that uses standard English conventions and logical flow of organization to address topics clearly, completely, and concisely. CityU requires the use of APA style.

UNIVERSITY POLICIES

You are responsible for understanding and adhering to all of City University of Seattle's academic policies. The most current versions of these policies can be found in the University Catalog that is linked from the CityU Web site.

Title IX Statement

City University of Seattle and its faculty are committed to supporting our students and seeking an environment that is free of bias, discrimination, and harassment. If you have encountered any form of sexual misconduct (e.g. sexual assault, sexual harassment, stalking, domestic or dating violence), we encourage you to report this to the University. If you speak with a faculty member about an incident of misconduct, that faculty member must notify CityU's Title IX coordinator and share the basic fact of your experience. The Title IX coordinator will then be available to assist you in understanding all of your options and in connecting you with all possible resources on and off campus.

To view CityU'S sexual misconduct policy and for resources, please visit the Campus Safety and Title IX Page in the my.cityu.edu portal.

Scholastic Honesty

Scholastic honesty in students requires the pursuit of scholarly activity that is free from fraud, deception and unauthorized collaboration with other individuals. You are responsible for understanding CityU's policy on scholastic honesty and adhering to its standards in meeting all course requirements. A complete copy of this policy can be found in the University Catalog in the section titled *Scholastic Honesty* under *Student Rights & Responsibilities*.

Attendance

Students taking courses in any format at the University are expected to be diligent in their studies and to attend class regularly.

Regular class attendance is important in achieving learning outcomes in the course and may be a valid consideration in determining the final grade. For classes where a physical presence is required, a student has attended if s/he is present at any time during the class session. For online classes, a student has attended if s/he has posted or submitted an assignment. A complete copy of this policy can be found in the University Catalog in the section titled *Attendance Policy for Mixed Mode, Online and Correspondence Courses*.

SUPPORT SERVICES

Disability Services Accommodations Statement

Students with a documented disability who wish to request academic accommodations are encouraged to contact Disability Support Services to discuss accommodation requests and eligibility requirements. Please contact Disability Support Services at *disability@cityu.edu* or 206.239.4752 or visit the Disability Support Services page in the my.cityu.edu portal. Confidentiality will be observed in all inquiries. Once approved, information about academic accommodations will be shared with course instructors.

Library Services

CityU librarians are available to help you find the resources and information you need to succeed in this course. Contact a CityU librarian through the *Ask a Librarian* service, or access library resources and services online, 24 hours a day, seven days a week.

Smarthinking

As a CityU student, you have access to 10 free hours of online tutoring offered through Smarthinking, including writing support, from certified tutors 24 hours a day, seven days a week. Contact CityU's Student Support Center at help@cityu.edu to request your user name and password.