CSIS-116E: PYTHON PROGRAMMING - LEVEL 1

Effective Term
Fall 2026

BOT Approval Date
06/12/2025

Discipline
CSIS - Computer Sci/Info Systems

Course Number
116E

Course Title
Python Programming - Level 1

Short Title
Python Programming 1

Credit Status (CB 04)
D - Credit - Degree applicable

Units

Lecture Units
3.00

Total Units
3.00

Hours

Lecture Contact Hours
48-54

Out of Class Hours Lecture
96-108

Total Contact Hours
48 - 54

Total Out of Class Hours
96 - 108

Total Student Learning Hours
144 - 162

Distance Education
Yes

Distance Education

Distance Education Type
Both Fully Online and Hybrid Online

Fully Online Delivery Requirements:
a. Students must be notified via the college schedule of classes and the syllabus for the class if proctored tests are required for this course
b. Any planned face-to-face meetings, such as an orientation or study session, must be optional

c. The MSJC curriculum committee requires the use of accessible, asynchronous discussion as a component of every fully online course

Are you using publisher content?

No

Regular Substantive Interaction

Instructor-to-student, student-to-student, and student-to-content contact shall be distributed in a manner that will ensure regular substantive interaction (RSI) is maintained not only weekly but also for the duration of the course. Instructor interactions should be equivalent in time, quality and consistency to the engagement students would receive in a face-to-face course—for example, through announcements, discussion participation, personalized feedback, video messages, or other instructional communications. Instructors are expected to be responsive to student inquiries within a 24-72 hour time frame excluding holidays. RSI will be maintained through a variety of methods: • Orientation - Students must be oriented to the online and face-to-face portions at the start of the course. • Announcements - Announcements using the course management system must be posted at least weekly to keep students current on course events, due dates, materials, etc. • Discussion Forums - Content-related discussion forums within the designated CMS will include appropriate instructor participation and will be initiated and maintained, with timely feedback provided. Instructors are required to monitor the discussions and related activities to provide guidance and direction. Open-ended discussion boards may be used to further learning and interaction. • MSJC Communication – Instructor will use MSJC-administered communication tools (such as email or the designated CMS communication tools) to regularly communicate with and provide ease of access to students. • Timely Feedback - Timely feedback on student work will be provided by the instructor, which may include comments and/or rubrics, within the Grades area of the designated CMS. • Scheduled Meetings – Hybrid and synchronous courses will meet at regularly scheduled times. ADA Compliance - All course interactions and content must be designed with accessibility in mind and meet ADA requirements.

Catalog Description

This course introduces the principles of object-oriented programming using the Python programming language. Students will investigate and evaluate various programming design methodologies and apply them to programming problems in Python. Python features that will be covered include language syntax, class definitions, control structures, function definitions, and basic data collections. No prior programming experience required.

Codes

TOP Code (CB 03)

0707.10* - *Computer Programming

CIP Code

11.0201 - Computer Programming/Programmer, General.

Repeatability Code

No

Grading Option

Letter grade OR P/NP

Minimum Qualifications

Minimum Qualifications	And/Or
Computer Science (Masters Required)	Or
Computer Information Systems	End

GE Information

Check GE Types Requested

AA/AS D2 - Communication & Analytical Thinking

Learning Objectives

Learning Objectives

Learning Objective
1. Analyze the basic structure of a Python application and be able to document, debug, and run a simple application.
2. Create, name, and assign values to variables of different types.
3. Use common control statements to implement flow, repetition, and selection control.
4. Create Python functions that can return values and take parameters.
5. Create, initialize, and use Python Lists, Sets, and Dictionary objects to solve data storage and retrieval problems.
6. Explain the basic concepts and terminology of object-oriented programming and differentiate between object-oriented, structured, and functional programming methodologies.
7. Analyze and trace the execution of Python computer programs.
8. Implement inclusive coding practices including the use of color, design, and accessibility standards to insure applications are usable by diverse populations, including individuals with disabilities.

Learning Outcome Information

Course Learning Outcomes

Learning Outcome
1. Student will be able to transform user input into formatted output.
2. Student will be able to choose and implement appropriate data types including casting to solve a given problem.
3. Student will be able to utilize if...elif...else selection control to choose between multiple alternatives.
4. Student will be able to utilize nested loops to generate a complex geometric shape or design.
5. Student will be able to add items to a Python list, and display items from a Python list using iteration.

Content

Course Lecture Content

1. Language Background and History
   a. Language Purpose
   b. Common Uses and Applications
2. Language Structure and Design
   a. Unique Features
   b. Dynamic vs Static Typing
   c. Interpreter vs. Compiled
3. Available IDE's and Tools
4. Primitive Variable Types
   a. Integers
   b. Floats
   c. Strings
   d. Boolean
5. Sequence Control
   a. Input Function
   b. Print Function
   c. Numeric Formatting
   d. String Formatting
   e. Type Casting
6. Looping/Repetition Control Statements
   a. While Statement
   b. For Statement
   c. Iterating with each and in range Statements
7. Selection/Decision Control Statements

   a. If statement
   b. If -else statement
   c. If -elif – else statement
8. Python Data Structures
9. Lists
   a. Tuples
   b. Sets
   c. Dictionaries
10. Functions
   a. Built-in Functions
   b. User-defined Functions
      i. Defining Functions
      ii. Calling Functions
      iii. Function Arguments
      iv. Return Values
11. Python Standard Library
   a. Turtle Module
   b. Random Module
   c. TKinter Module
   d. Datetime Module
   e. Math Module
   f. Os Module
12. Advance Libraries and Packages
   a. Overview of available Libraries and Packages
   b. Using PyPi to find Libraries
   c. Using Pip to download and install Libraries
13. Inclusive Programming Practices
   a. Use Inclusive Language in Code
   b. Keyboard Short-cuts and Keyboard Only Navigation
   c. Accessible Forms
   d. Localization & Internationalization
   e. Screen Readers & Automation

Methods of Instruction

Method

Discussion

Integration

Weekly discussion will be conducted as a means to compare and contrast the use and implementation of the for and while looping constructs; the if, if else, conditional, and elif decision constructs; efficiency of algorithms that implement lists and dictionaries for storage and data retrieval; and Inclusive Programming Practices such as implementing keyboard short-cuts and accessible forms.

DE Adaptations for MOI

Discussions will occur using the course management system's (CMS) discussion tool, designed with accessibility in mind including the use of captions for video content, clear and structured formatting for written materials, and descriptive text for visuals. Discussions will take place using the course's discussion platform. Feedback on discussion contributions will be provided within one week via the CMS, ensuring timely and accessible engagement for everyone.

Method

Lecture

Integration

Lecture, with supporting visual materials will introduce conceptual and practical skills such as coding and implementing for and while constructs; implementing if, if else, conditional, and elif decision constructs; the development of algorithms that implement lists and dictionaries for data storage and retrieval; demonstrate the use of primitives to develop complex types; and inclusive programming practices such as implementing keyboard short-cuts and accessible forms.

DE Adaptations for MOI

The CMS will be a central hub for online students, offering accessible recorded lectures, online presentations, and supplementary materials. To promote inclusivity, live webinars and recorded lectures will be provided with closed captions (CC), enabling real-time interaction and discussions.

Method

Observation and Demonstration

Integration

Guided practice and demonstration will illustrate lecture principles and reading assignments with a focus on fundamental Python programming concepts and applying them to problem solving and potential solutions to homework assignments. Guided practice will include coding and implementing for and while constructs; implementing if, if else, conditional, and elif decision constructs; the development of algorithms that implement dictionaries and lists for data storage and retrieval; demonstrate the use of primitives to develop complex types; the development of algorithms that show the appropriate creation and use of abstract data types.

DE Adaptations for MOI

Observation and demonstrations will be accessible through the CMS, offering closed-captioned videos, web-based tutorials, and hands-on activities. Interactive software simulations hosted within the CMS will provide students with real-time engagement opportunities and practical experience, ensuring accessibility and active learning for all participants.

Methods of Evaluation

Method

Exams/Tests

Integration

Up to four exams will be given. The exams will be designed to evaluate the mastery of such topics as looping constructs; decision structures; data storage and retrieval with lists; and the creation of abstract data types. Exams will be evaluated based on the program running to completion without error or warning; how well directions are followed; program correctness; proper input and output; and inclusive programming practices such as implementing keyboard short-cuts and accessible forms.

DE Adaptations for MOE

Exams will be administered through the course management system (CMS), consisting of multiple-choice, true/false, and short-answer coding questions. Students will submit their completed exams directly through the CMS. All accessibility requirements including captioning should be met. Immediate feedback will be provided for objective questions, such as multiple-choice and true/false, while feedback on short-answer and coding responses will be given within 48 hours. This ensures timely reinforcement of learning and offers clarity on areas for improvement.

Method

Other

Integration

Discussion and written guided practice which consists of Python syntax and logic problems; looping structures (for, while); decision structures (if, if else, elif, conditional); the use of lists to solve storage and retrieval problems; the development of algorithms that demonstrate the proper use primitives to create complex types; the development and proper use of abstract data type problems will be required. Guided practice / discussion projects will be evaluated based on program completeness and how well directions are followed.

DE Adaptations for MOE

Guided Practice problems may be written or recorded using a screen-capture tool. These problems should be stored and maintained in the CMS system. Discussions will occur using the CMS discussion tool. Guided practice problems may be written or recorded using a screen-capture tool and uploaded to the course. All instructional materials developed by the instructor will be created with accessibility in mind, including the use of captions for video content, clear and structured formatting for written materials, and descriptive text for visuals. Discussions related to guided practice will take place using the course's discussion platform, allowing students to engage with the material and with each other in multiple formats. All accessibility requirements including captioning

should be met. Timely feedback will be provided to students via the course management system within five days of submission and should be based on a published rubric.

Method
Homework

Integration
Weekly programming assignments will apply important concepts and skills to the student's experience outside the classroom. The focus might be on the development and correct implementation of abstract data types, correct usage of the different looping constructs like the for and while loops, the implementation of if, else if, elif, and conditional decisions, and storage and data retrieval techniques using lists and dictionaries. Weekly assignments will be evaluated based on the program running to completion without error or warning; how well directions are followed; program correctness; proper input and output; and inclusive programming practices such as implementing keyboard short-cuts and accessible forms.

DE Adaptations for MOE
Homework will be submitted by students via the CMS assignment tool. All accessibility requirements including captioning should be met. Timely feedback will be provided to students via the course management system within five days of submission and should be based on a published rubric.

Assignments

Assignment Type
Other

In-class and/or outside of class
Outside of class

Formative or Summative
Formative

Assignment
Objective: To create a menu-driven Python console application that utilizes one or more lists to manage structured data items.

General Specification: Your application should utilize a simple sentinel loop with the following menu options. A. Add Item; B. List Items; C. Search for items, and Q. Quit. Your database should have a minimum of three fields. Each field can have its own list, or you can utilize a list that combines data elements.

The data concept is up to you, but should follow a cohesive data theme (items that make logical sense and go together). Your data theme should have at least 3 data elements. Your data theme should include at least one string, and one numeric element.

For example, if you decide to implement a vehicle data theme, you might use make, model, year, and price for your data fields:

How assignment will be adapted in online format
The assignment will be distributed and collected using the assignments area of the course management system. All accessibility requirements including captioning should be met. Assignment instructions may be recorded using a screen-capture tool and uploaded to the course. All instructional materials developed by the instructor will be created with accessibility in mind, including the use of captions for video content, clear and structured formatting for written materials, and descriptive text for visuals. Discussions related to assignments will take place using the course's discussion platform, allowing students to engage with the material and with each other in multiple formats. Timely feedback will be provided to students via the course management system within five days of submission and should be based on a published rubric.

Assignment Type
Writing

In-class and/or outside of class

Outside of class

Formative or Summative

Formative

Assignment

Improving System Usability and Access for Diverse Users

Choose a computer system and accompanying software. This could be a stand alone application, a web application, a mobile app, or even an embedded system (i.e. a system embedded in your automobile, or a point-of-sale kiosk at a local fast food establishment). Based on the selected system, write a one paragraph description of the system with an emphasis on the accessibility of the system. Identify at least three hardware or software features that promote inclusion and access in the existing system (language, culture, disability). In addition, please propose at least two additional features or enhancements to the system, hardware or software, that would enhance or improve access. At least one of the proposed features should enhance the user-interface. Be specific in how the proposed enhancements will improve usability and access for diverse users.

How assignment will be adapted in online format

The assignment will be distributed and collected using the assignments area of the course management system. All accessibility requirements including captioning should be met. Assignment instructions may be recorded using a screen-capture tool and uploaded to the course. All instructional materials developed by the instructor will be created with accessibility in mind, including the use of captions for video content, clear and structured formatting for written materials, and descriptive text for visuals. Discussions related to assignments will take place using the course's discussion platform, allowing students to engage with the material and with each other in multiple formats. Timely feedback will be provided to students via the course management system within five days of submission and should be based on a published rubric.

Assignment Type

Other

In-class and/or outside of class

In-class
Outside of class

Formative or Summative

Summative

Assignment

The Final Exam presentation is open to any Python topic, with an emphasis on external libraries. Unique Python language features or the demonstration of an external IDE or other Python tool or feature is also acceptable.
Presentation (Virtual or Written Report): 100 points
Final Exam Participation: 50 points

The following four step process is recommended as you prepare your presentation -

1. Research and read up on possible libraries in the sub-area you are interested in. (i.e. Game Development, Image Processing, Music, etc.). There is usually more than one library in a particular domain, and often many.

2. Use PIP or the built-in Package manager in your IDE to download and install your library.

3. Write some simple code that demonstrates your library or language feature. Look for more complex examples that demonstrate the capability of your library.

4. Present your findings in a 4-6 minute Zoom presentation at the Final Exam meeting. OR Submit a two-page written summary of your findings (see presentation requirements below). Written presentation should include the six topic elements below, and include your Python Code if relevant (IDE & Tool presentations do not require code). Attendance and participation at the Final Exam Presentation session is required (even if you are not presenting).

Required Elements for In-class and Written Presentations –

1. Start by discussing your interest in the library or topic you chose. (i.e. Why did you choose this topic?)

2. Discuss your approach to researching and finding a library. Was it difficult to choose between libraries? Why did you eventually choose the library you did?

3. Discuss the process of downloading and installing your library. Did you have any difficulties with this step?

4. Describe what the library is capable of in general terms. What are some of the use-case scenarios for using this particular library?

5. Show and demonstrate a simple code example of how your library works (i.e. what are some of the simple functions that the library supports). Be sure to include at least one 'Hello World' example.

6. Show a more complex example of how someone might use this library (a demonstration of one or more of the advance features of the library). This could be a demo or an actual application that utilizes this library.

How assignment will be adapted in online format

The assignment will be distributed and collected using the assignments area of the course management system. All accessibility requirements including captioning should be met. Assignment instructions may be recorded using a screen-capture tool and uploaded to the course. All instructional materials developed by the instructor will be created with accessibility in mind, including the use of captions for video content, clear and structured formatting for written materials, and descriptive text for visuals. Discussions related to assignments will take place using the course's discussion platform, allowing students to engage with the material and with each other in multiple formats. Timely feedback will be provided to students via the course management system within five days of submission and should be based on a published rubric.

Assignment Type

Reading

In-class and/or outside of class

Outside of class

Formative or Summative

Formative

Assignment

Refer to the official 'Python Language Reference' guide at Python.org. Under the documentation tab please read and review section 8 - Compound Statements. Based on this reading, write a simple code example or code snippet that utilizes each of the following three statements - if statement; while statement; and for statement.

Please use your own variables and data scenario to complete this assignment. Your code should be original, and represent your own concept. Ideas can be simple, with each code example not to exceed 10 lines of code. Examples that are 5 to 8 lines of code should be sufficient. A typical example might have one or two lines of input, followed by the compound statement, and conclude with one or two lines of output.

Please write a one paragraph example of what your code does, and why you used the statement you did (given the three available choices).

How assignment will be adapted in online format

The assignment will be distributed and collected using the assignments area of the course management system. All accessibility requirements including captioning should be met. Assignment instructions may be recorded using a screen-capture tool and uploaded to the course. All instructional materials developed by the instructor will be created with accessibility in mind, including the use of captions for video content, clear and structured formatting for written materials, and descriptive text for visuals. Discussions related to assignments will take place using the course's discussion platform, allowing students to engage with the material and with each other in multiple formats. Timely feedback will be provided to students via the course management system within five days of submission and should be based on a published rubric.

Course Materials

Textbooks

Slatkin, Brett (2024). Effective Python Addison-Wesley. ISBN: 978-0138172183

Other Resources

Official Python Documentation: https://docs.python.org/3/

Class Size Information

Class Size

35

Class Size Category

B - Facilitated learning 35

Diversity and Inclusion

Course Design and Materials: Course will incorporate diverse perspectives, authors, and examples that reflect a variety of cultures, identities, and lived experiences. Instructional materials will be reviewed regularly to ensure representation, inclusivity, and alignment with equity-minded practices across disciplines.

Inclusive Learning Environment: Instructors will establish classroom norms and policies that promote respect, civil discourse, and belonging in both online and face-to-face classes. Clear expectations for communication and engagement will support a learning space where all students feel valued and heard.

Accessible Instruction: Course content will be delivered using accessible design. Materials will follow accessibility guidelines within the CMS, including captioned media, readable document formats, and multiple means of engagement to support diverse learning.

Equitable Teaching Practices: Instructors will use transparent and grading criteria, varied assessment methods, and timely feedback to promote fairness and student success. Opportunities for revision, reflection, and multiple demonstrations of learning may be incorporated when appropriate to the discipline.

Culturally Responsive Pedagogies: Instructional strategies will encourage critical thinking about diverse perspectives and systems of power relevant to the course content. Faculty will foster opportunities for students to connect course material to real-world contexts.

Ongoing Reflection and Improvement: Faculty will engage in continuous professional development related to IDEAA principles and reflect on course practices using metacognition to identify opportunities for increased equity, inclusion, and accessibility.

Explain how diversity and inclusion(e.g., gender, culture/race, sexuality, etc.) is infused into the course.

The discipline of Information Technology bridges cultures, genders, and race. Programming inherently connects people of diverse backgrounds through the applications that are created, and the multicultural communities that create them. Python coding projects and examples in this course should highlight and promote breadth and diversity (for example, a code example or lab that requires students to convert international currency, or translate an English phrase into a language of the students choice). Unfortunately Information Technology can also create barriers to access. Inclusive programming practices such as implementing keyboard short-cuts and accessible forms should be discussed and taught as an integral part of the course.

Describe how assignments, instruction, evaluation, course content, and interactions are contextualized for diverse learners promoting an equitable course.

Course content should be organized and delivered to effectively reach students with different learning styles. For example, video and audio content should be included to better reach visual and auditory learners. For projects and presentations students should be given a range of options for presenting their findings. (For example, students could be given an option to present their findings in a written report, or in an in-class presentation). In this way, multiple learning styles can be accommodated to create an equitable course. Course content should be formatted in an accessible manner and adhere to current standards. Finally, students should be encouraged to utilize available institutional resources (i.e. counseling, tutoring, etc.).

Interactions

Explain how the course will incorporate student-to-student interaction.

In face-to-face sections, CSIS 116E students will interact with each other primarily via in-class discussion, and small-group projects. In small-group projects students typically work in teams of two on in-class programming labs. Online and face-to-face students also

have significant interaction via the online discussion board through both specific and general discussion board threads. For example, some assignments require students to comment on, or provide peer-review of other student's code. In a programming course it is assumed that students in both face-to-face and online sections are engaging with each other via email, and/or discord or other technology mediated discussion forum tools.

Explain how the course will incorporate instructor-to-student interaction.

Instructor-to-Student interaction in CSIS 116E will occur primarily via lecture and instructor led in-class guided practice activities and labs. In a programming course, guided practice typically consists of demonstrating coding techniques on the screen, as students follow along on their own computer. For online sections, this interaction is facilitated via recorded lectures and labs that include video, audio, and screen-share technology. Additional interaction for both face-to-face and online sections occurs via instructor responses to discussion board questions, direct email response, and synchronous face-to-face or Zoom office hours. Weekly announcements posted concurrently via email and CMS announcement facility provide additional effective and regular Instructor-to-Student contact. For online students, a written or video welcome message posted at the start of the course introduces the instructor to the student in a supportive and encouraging manner.

Explain how the course will incorporate student-to-content interaction.

Student-to-Content interaction in CSIS 116E is facilitated through the use of online resources and/or OER materials including online documentation. In programming, online documentation is a primary means for student-to-content interaction. Live code software provides students web-based activities that they can use to practice coding techniques. Programming Labs provide students a real-world connection to the topics covered in the course.

Explain how the course will incorporate student-to-self interaction (metacognitive).

Specific assignments in this course are intended to engage students in ways that relate to their unique situation and personal interests. A data project in this course (lists and dictionaries) might require students to choose their own data theme and motif. In similar fashion, CSIS 116E students typically complete a project that requires the student to choose and research a Python library of their choice. This project provides an opportunity for students to engage in self-reflection as they research and elaborate on a topic of personal interest. (i.e. for example, students interesting in graphics can choose a graphics library, etc.)

Key: 1036