CSIS-126E: PYTHON PROGRAMMING - LEVEL 2

Effective Term
Fall 2026

BOT Approval Date
06/12/2025

Discipline
CSIS - Computer Sci/Info Systems

Course Number
126E

Course Title
Python Programming - Level 2

Short Title
Python Programming 2

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 advanced concepts of object-oriented programming (OOP) using the Python programming language. Students will investigate and evaluate various program design methodologies and apply them to programming problems using Python. Python features that will be covered include language syntax, encapsulation, inheritance, polymorphism, advanced O-O design principles, and exception handling.

Requisites

Prerequisite(s)

Prerequisite(s) (must be taken before)
CSIS-116E (with a grade of C or better).

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 and evaluate information systems in terms of their key components: hardware, software, data, procedures, and people.
2. Apply systems concepts in the investigation, evaluation, and resolution of information technology problems.
3. Construct O-O Python programming solutions for re-use that incorporate encapsulation, data abstraction, and information hiding.
4. Utilize and Implement Pandas to read, summarize, and visualize .csv data.
5. Implement a class hierarchy that utilizes polymorphism through inheritance.
6. Select and implement an algorithm that demonstrates the proper use of recursion.
7. Utilize and Implement a Machine Learning Library such as scikit-learn to implement a simple machine learning solution using supervised learning.
8. Evaluate potential negative consequences of algorithmic bias in a programming scenario, including but not limited to race and gender bias.

Learning Outcome Information

Course Learning Outcomes

Learning Outcome
1. Student will be able to read a .csv file and use the Pandas Library to determine min, max, mean, and standard deviation.
2. Student will be able to utilize Matplotlib or Seaborn to visualize two variable data using a scatter plot.
3. Student will be able to implement game or business logic in a class that utilizes AND, OR, or NOT Boolean logic operators.
4. Student will be able to implement two or more sub-classes that utilize inheritance and polymorphism to create multiple member functions with the same name.
5. Student will be able to apply scikit-learn or other Machine Learning library to visualize a linear regression relationship in a two variable dataset.

Content

Course Lecture Content

1. IDEs and Tools
2. Survey of use-cases for the Python Programming Language
3. Python Dictionaries
4. Introduction to Data Science with Python
    a. Reading .csv files
    b. Using Pandas for Data Exploration
    c. .ipynb files and Jupyter Notebooks
5. Introduction to Data Visualization with Python
    a. Simple Data Visualization with Matplotlib
    b. Statistical Data Visualization with Seaborn
6. Object-oriented Programming
    a. Python Classes
    b. Inheritance
    c. Polymorphism and Function Overloading
7. Recursion
8. Introduction to Machine Learning with Python
    a. Supervised Learning
    b. Unsupervised Learnin
    c. Reinforcement Learning
9. Advance Python Libraries
    a. Web Scripting
    b. Image Processing
    c. Web Scraping

10. Racial, Gender, and other Bias in Machine Learning
    a. Data Bias
    b. Algorithmic Bias
    c. Selection and Exclusion Bias

Methods of Instruction

Method

Discussion

Integration

Weekly discussion will be conducted as a means to compare and contrast the use and implementation of inheritance and an "isa" relationship, the proper uses for encapsulation and data abstraction, and identification of data, algorithmic, and other biases that can result in unfair and discriminatory outcomes.

DE Adaptations for MOI

Discussions will occur using the course management system's (CMS) discussion tool. 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 will take place using the course's discussion platform, allowing students to engage with the material and with each other in multiple formats. 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 reading and writing from data files, demonstrate the proper use of data abstraction and encapsulation, using inheritance to produce a class that has an "isa" relationship, and identification of data, algorithmic, and other biases that can result in unfair and discriminatory outcomes.

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. Lectures 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 lecture will take place using the course's discussion platform, allowing students to engage with the material and with each other in multiple formats.

Method

Observation and Demonstration

Integration

Guided practice and demonstration will illustrate lecture principles and reading assignments, with a focus on fundamental Java object oriented programming concepts and applying them to problem solving and potential solutions to homework assignments. Guided practice will include coding and implementing programs to read and write from data files, using inheritance to form an "isa" relationship between classes, the proper use of data abstraction and encapsulation, and the use of the common Python GUI components to build applications.

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. Guided practice and demonstration 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. Related discussion will take place using the course's discussion platform, allowing students to engage with the material and with each other.

Methods of Evaluation

Method

Other

Integration

Participation in several written guided practice and discussion projects will be required. Guided practice Topics such as reading and writing from data files, proper use of data abstraction and encapsulation, the use of inheritance to form "isa" relationships, and the use of the common Python GUI components will be evaluated. 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. All accessibility requirements including captioning should be met. 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." 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

Exams/Tests

Integration

Up to four essay style exams will be given as a means to evaluate evaluate the mastery of such topics as: reading and writing from data files, using data abstraction and encapsulation, and identification of data, algorithmic, and other biases that can result in unfair and discriminatory outcomes. 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.

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 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. 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

Homework

Integration

Written programming assignments will be given that apply important concepts and skills to the student's experience outside the classroom. The focus might be on the use of abstract data types, the implementation of inheritance to create "isa" relationships, reading and writing from data files, and identification of data, algorithmic, and other biases that can result in unfair and discriminatory outcomes. 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.

DE Adaptations for MOE

Homework will be submitted by students via the CMS assignment tool. All accessibility requirements including captioning should be met. Assignment instructions 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 assignments will take place using the course's discussion platform, allowing students to engage with the material and with each other. 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
Simple Data Assignment with Visualization

For this assignment submit code that demonstrates the following four items.

1. Read a .csv data file via pandas; (recommend you use Kaggle to find your dataset)

2. Utilize .info(), .describe(), .head(), and .tail() to summarize your data;

3. Visualize two histograms (.hist()) – at least one histogram visualization should include sliced data (i.e. DataFrame subset) - children=df[df['Age']<16]

4. Visualize one Boxplot (.boxplot(), or .scatter);

No conclusions or data interpretation is required for this assignment. You can utilize this same dataset for your Final Presentation Project.

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
In-class
Outside of class

Formative or Summative
Summative

Assignment
Use Python to analyze and visualize diversity in a dataset:

Select a dataset from the list of suggested diversity datasets, OR select a diversity dataset of your own choosing from Kaggle. If you choose you own dataset, please include a reference as to the source of the dataset, and what diversity metric you are measuring. The selected dataset should include at least one field that tracks a diversity measure (race, gender, disability, etc.). Using this dataset, utilize Matplotlib or Seaborn to create at least three data visualizations that illustrates or shows disparities or progress in diverse participation over time, or that contrasts disparities against different measures (i.e. compare diversity disparities across different

geographic locations, different education levels, etc.). Write a one paragraph summary to accompany each visualization. Please make sure each visualization has a title, and appropriate X and Y labels. Be prepared to present your project in class.

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 open source book 'Python Data Science Handbook' at https://jakevdp.github.io/PythonDataScienceHandbook/. Please read and review Chapter 4 – Visualization with Matplotlib. Based on this reading, write a simple code example that generates a simple line plot. At a minimum your plot should include a title, X and Y axis labels, at least one color reference, and one unique line style.

Use your own variables and data scenario to complete this assignment. Please use a python list with no more than 100 data elements. Do not use a .csv file for this assignment. Your code should be original, and represent your own concept. Ideas should be simple, not to exceed 20 lines of code. Examples that are 10 to 12 lines of code should be sufficient for most scenarios.

Please write a one paragraph summary of what your code does and why you chose the data theme that you did.

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 Wesely. 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. Course content should celebrate the accomplishments and achievements of individuals from diverse backgrounds and traditionally underrepresented groups. In-class Python coding exercises and the accompanying datasets presented in this course should highlight and promote breadth and diversity. Specifically, datasets presented should contain data that fairly and accurately reflect diverse populations including race, gender, and other measures of inclusion. Unfortunately, Information Technology can also be used to control and divide individuals and groups. Issues such as bias in Machine Learning algorithms should be acknowledged and openly discussed in the context of avoiding or at least identifying biases that can have a negative impact on diversity and inclusion.

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

In CSIS 126E students are encouraged to select, research, and elaborate a Python dataset of their choosing that reflects the student's unique interests and cultural background. 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 an in-class presentation). 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 126E students will interact with each other primarily via in-class discussion, and small-group programming projects. In small-group projects students typically work in teams of two on completing 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 do and 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 126E 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 126E 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. In this course students will interact with course concepts through datasets and application themes that they have chosen. In this way, the lens through which students approach and view the course content will be data of personal interest and relevance that they have chosen to work with.

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. For example, a data project in this course might ask a students to choose, curate, and analyze a dataset of their own choosing. In this way, major assignments and labs incorporate datasets and project particulars that resonate with student's personal interest and values. For example, a student interested in healthcare would be encouraged to choose a medial related dataset for their final project (i.e. for example, a dataset that analyzes the risk of heart disease). This project approach provides an opportunity for students to engage in self-reflection as they research and elaborate topics of personal interest through programming.

Key: 1055