MAHONING COUNTY CAREER & TECHNICAL CENTER

Object-Oriented Programming

Syllabus & Curriculum Map · ODE Subject Code 145065

AI Automation & Software Development | Junior Year, Semester 2

2026–2027 · February 1 – June 10, 2027

Version 1.0 · Instructor: Michael Sekol · michael.sekol@mahoningctc.com

CONCEPT **What this course is**

Representing programs as objects with data and behavior. You will learn object-oriented design in Python, build a full web application with a database, then move to C# where the compiler enforces everything Python lets you fake.

You will finish having deployed a live web application and built a working industrial interface that reads real sensor data off a machine.

CHECKPOINT **College credit**

Object-Oriented Programming 145065 is CTAG-eligible for college credit at Ohio public colleges and universities. Credit is triggered by the acknowledgement survey at the end of your WebXam post-test.

Course Overview

Official course description

Students will learn to represent programming concepts as "objects" that have data fields and associated procedures known as methods. Students will implement classes such as support static, instance method, inheritance, polymorphism, exception handling, and object serialization. A variety of commercial and opensource programs and applications will be used.

— Ohio Department of Education & Workforce, Information Technology Career Field

Where this sits

YEAR	SEMESTER 1	SEMESTER 2	LANGUAGE
Junior	145060 Programming	145065 Object-Oriented Programming	Python → Python + C#
Senior	145130 Applications of AI	145010 Web Design + Capstone	C# + Python → Web

You arrive from 145060 able to write Python, use Git, and deploy. This course adds object-oriented design, a real database, and a second language. Seniors arrive at 145130 already knowing C#, which is what that course builds with.

Two languages, and why

Python, Weeks 1–10	Object-oriented concepts introduced in a language you already know. Then Flask, a database, and a deployed CRUD application.
C#, Weeks 11–19	The same concepts in a typed, compiled language. private actually means private. An interface is a contract the compiler checks. Then event-driven programming and industrial interfaces.

CONCEPT **Why both**

Python teaches OOP softly. There are no real access modifiers, only a naming convention. There are no interfaces, only duck typing. Nothing enforces the concepts, so a student can pass without internalizing them.

C# enforces all of it. Learning the ideas in Python first and then meeting them in a language that checks your work is the fastest route to actually understanding them.

Structure

Length	19 calendar weeks, 88 instructional days
Grading Period 3	February 1 – April 9 · Units 0–5 · Python
Grading Period 4	April 12 – June 10 · Units 6–9 · C#

Daily block	Periods 1–3, 8:00–10:22. About 118 working minutes.
Languages	Python 3.14 with uv · C# 14 on .NET 10 LTS
Editors	VS Code for Python · Visual Studio 2026 for C#
Database	MySQL Workbench to learn · PostgreSQL on Render to deploy
Hardware	Raspberry Pi with sensor kits, on the isolated lab network

Assessments and hard dates

ASSESSMENT	WINDOW
WebXam 145065 Pretest	Administered in January, during 145060, before this course begins
BPA State pre-submission	February 4
BPA State Leadership Conference	February 21–23
BPA National Leadership Conference	May 4–9, Colorado
WebXam 145065 Post-test	Week 16 — May 17–21. Post-test window closes May 28.

WARNING **Two scheduling constraints**

The WebXam winter pretest window closes January 29, three days before this course begins. The 145065 pretest must be administered in January while students are still finishing 145060.

The post-test window closes May 28. The post-test is scheduled for Week 16, not the final week, so it lands inside the window with room to spare.

BPA alignment

EVENT	SUPPORTED BY
330 C# Programming	Units 6–9. State and National both fall inside the C# block.
345 SQL Database Fundamentals	Unit 5 Databases and CRUD
391 Computer Programming Concepts	Units 1–3 object-oriented concepts
V04 Virtual Web Application Team	Units 4–5 Flask and deployment

How Class Runs

The daily block

MIN	SEGMENT	WHAT HAPPENS
15	Bell ringer & Gate 1 rep	A question on the board when you walk in, then a timed rep. No AI, no autocomplete.
15	Instruction	One concept. Live coding, including a deliberate mistake debugged in front of you.
35	Build 1	Hands-on.
5	Reset	Stand up, walk, reset.
40	Build 2	Hands-on or pair work.
8	Commit & close	Push your work. Someone demos or names what broke.

Friday introduces no new content. Fridays are BPA preparation, credential work, side quests, and catch-up. In this course, February and April Fridays carry heavy competition load.

The three gates

GATE	AI USE	IN THIS COURSE
Gate 1 — Closed	None	Timed reps. In the C# half these include compiler-error reading, which is a distinct skill from reading a Python traceback.
Gate 2 — Adversarial	AI is the opponent	Weekly. AI-generated object-oriented code has characteristic failure modes: leaky encapsulation, inheritance where composition belongs, and interfaces that promise more than they deliver.
Gate 3 — Open	Full tooling	Ambitious builds. Decision log required, and in this course it must name the design pattern you chose and the one you rejected.

> **NOTE** **Hardware safety**
> Units 8 and 9 involve physical hardware on the isolated lab network. A signed safety agreement must be on file before you handle any equipment. ESD precautions are posted at the bench and are not optional.

Grading

CATEGORY	WEIGHT	WHAT COUNTS
Projects	35%	The CRUD web application, the HMI build, team sprints.
Written & Documentation	20%	Gate 2 reviews, AI usage logs, decision logs, UML and design docs, READMEs.
Lab & Practice	20%	Gate 1 reps, bell ringers, daily labs, problem drops.
Quiz & Exam	15%	WebXam-aligned quizzes and unit exams.
BPA / Credential / Capstone	10%	Competition prep and events, credentials, side quests.

Policies

- No work is graded that is not in a repository. Every period ends with a commit.
- AI is permitted on Gate 2 and Gate 3 work and prohibited on Gate 1 reps.
- Every project includes an AI usage log and a decision log naming the design choice you made and the one you rejected.
- No personal information, real names, or school data enters any AI tool.
- Any project may be revised once within two weeks of return if the original was on time.
- When stuck: struggle 10 minutes, ask a peer, ask an AI, then ask me.
- The only way to fail outright is to submit work you cannot explain.

Unit Sequence

Ten units across 19 weeks. PRIMARY tags are 145065 competencies. LATENT tags are 145130 competencies you will be tested on senior year.

UNIT 0 Onboarding & Emerging Technologies

WEEKS 1 **DATES** Feb 1 – Feb 5

Focus: Reset the environment, re-establish the rhythm, and survey what is actually changing in this industry.

Students will:

- Identify emerging technologies applicable to the marketplace
- Describe the fundamental architectures of emerging technologies and how they integrate into existing IT systems
- Research the value of an emerging technology on the marketplace
- Describe emerging technologies including IoT, large language models, machine learning, and additive manufacturing

Deliverables:

- Emerging technology brief: pick one, explain its architecture, argue its market value
- Environment verified, repository created, Git rhythm resumed
- BPA State pre-submission February 4

PRIMARY 145065 2.4.1, 2.4.2, 2.4.3, 2.4.4, 1.2.1, 1.2.12
LATENT 145130 2.14.1, 2.14.5, 2.4 emerging tech overlap

UNIT 1 Classes, Objects & Encapsulation

WEEKS 2–3 **DATES** Feb 8 – Feb 19

Focus: The core idea. Data and the behavior that operates on it, bundled together and protected.

Students will:

- Write code that creates classes, objects, and methods
- Distinguish instance methods from static methods and know when each applies
- Identify the scope of data across global, local, instance, and class levels
- Describe and contrast procedural, structured, object-oriented, and event-driven programming

Deliverables:

- Refactor a 145060 procedural project into classes, with a written justification for every class boundary
- UML class diagram produced before any code is written
- Gate 2: AI-generated classes with leaking encapsulation

PRIMARY 145065 5.3.12, 5.2.2, 5.1.4, 5.1.5, 5.5.5, 5.4.1, 5.4.2
LATENT 145130 2.14.4

UNIT 2 Inheritance & Polymorphism

WEEKS 4–5 **DATES** Feb 22 – Mar 5

Focus: Reuse without duplication, and one interface over many implementations. Also when inheritance is the wrong answer.

Students will:

- Implement inheritance and explain the relationship it creates
- Implement polymorphism and describe why it reduces conditional logic
- Explain when composition is a better choice than inheritance
- Analyze the strengths and weaknesses of different approaches for a specific problem

Deliverables:

- Class hierarchy project with at least three levels and one polymorphic method
- Written analysis: one place you used inheritance and one place you deliberately did not
- BPA State Leadership Conference February 21–23

PRIMARY 145065 5.3.12, 5.1.4, 5.1.6, 5.5.2, 5.6.9
LATENT 145130 2.14.6

UNIT 3 Exceptions & Serialization

WEEKS 6 **DATES** Mar 8 – Mar 12

Focus: What happens when things fail, and how objects survive being turned off.

Students will:

- Code error handling techniques including custom exception classes
- Serialize and deserialize objects to and from persistent storage
- Identify the need for disaster recovery policies and procedures
- Preserve, convert, or migrate existing data to a new format

Deliverables:

- Exception hierarchy built for a real failure domain
- Object serialization to JSON with full round-trip verification
- A written recovery plan for a system that loses its data store

PRIMARY 145065 5.3.10, 5.5.1, 3.2.8, 3.2.1, 5.5.6
LATENT 145130 2.6.4, 8.2.3

UNIT 4 Flask: Web Application Framework

WEEKS 7–8 **DATES** Mar 15 – Mar 26

Focus: Your objects, on the internet. Routing, templates, forms, and the request cycle.

Students will:

- Explain the request and response cycle between a browser and a server
- Build routes that map URLs to functions and return rendered templates
- Design a data entry form and process the submitted data server-side
- Develop programs using data validation techniques on both client and server

Deliverables:

- Flask application with at least five routes and template inheritance
- Working form with server-side validation that rejects bad input
- Gate 2: a generated Flask route with an injection vulnerability

PRIMARY 145065 5.5.1, 5.5.4, 5.5.7, 5.3.11, 9.3.1, 9.3.3
LATENT 145130 2.8.9, 2.15.3

UNIT 5 Databases, CRUD & Deployment

WEEKS 9–10 **DATES** Mar 29 – Apr 9

Focus: A real database behind a real application, live on the internet. Grading period closes April 9.

Students will:

- Design a normalized schema with tables, fields, relationships, and keys
- Write SQL to create, read, update, and delete records
- Import a data set using SQL script or CSV
- Generate reports and query results including calculated fields
- Deploy an application to a publicly accessible URL and maintain it

Deliverables:

- FULL CRUD APPLICATION deployed live on Render with a PostgreSQL database
- ER diagram and data dictionary
- Someone outside the class used it and you fixed what they broke
- GP3 ENDS April 9

PRIMARY 145065 5.3.11, 5.5.7, 5.6.11, 5.6.16, 5.6.17, 5.1.5, 1.4.2
LATENT 145130 2.8.1, 2.8.2, 2.8.3, 2.8.4, 2.8.8, 8.3.1, 8.3.2, 8.5.4

UNIT 6 The C# Transition

WEEKS 11–13 **DATES** Apr 12 – Apr 30

Focus: Same concepts, a language that checks your work. Static typing, the compiler, and access modifiers that mean something.

Students will:

- Compare and contrast compilers and interpreters, and explain what the compiler catches
- Write C# using correct syntax, static typing, and namespaces
- Implement access modifiers and explain what each protects
- Implement interfaces as compiler-enforced contracts
- Configure options, preferences, and tools in Visual Studio 2026

Deliverables:

- Port your Unit 2 class hierarchy from Python to C#, with a written comparison of what the compiler caught
- Interface-driven design with at least two implementations
- Gate 1 reps shift to reading compiler errors

PRIMARY 145065 5.1.7, 5.1.4, 5.1.6, 5.3.12, 5.4.1, 5.4.2, 5.4.3, 5.4.6, 5.5.5
LATENT 145130 2.4.2

UNIT 7 Event-Driven Programming & WPF

WEEKS 14–15 **DATES** May 3 – May 14

Focus: Programs that wait for something to happen. Handlers, data binding, and the visual designer. BPA Nationals falls in Week 14.

Students will:

- Describe and contrast event-driven programming with procedural and object-oriented approaches
- Build a desktop interface using XAML and the visual designer
- Write event handlers and explain the event loop
- Bind interface elements to underlying data so the display updates itself

- Identify processor, memory, storage, power, and environmental requirements

Deliverables:

- WPF desktop application with at least four interactive controls and working data binding
- Event flow diagram for your own application
- BPA NATIONAL LEADERSHIP CONFERENCE May 4–9

PRIMARY 145065 5.1.4, 5.3.12, 5.4.2, 5.4.7, 2.10.2, 5.5.6
LATENT 145130 2.15.2, 2.15.3, 7.3.1

UNIT 8 Industrial Human-Machine Interface

WEEKS 16–17 **DATES** May 17 – May 28

Focus: The interface a factory operator actually uses. Live sensor data, alarm states, and what the screen shows when the machine stops answering.

Students will:

- Read live data from a physical device and display it in real time
- Design for industrial conditions: high contrast, large touch targets, glare and glove tolerance
- Implement alarm states and fail-safe display behavior on connection loss
- Distinguish stale data from missing data and show the difference to the operator
- Document a design using dataflow diagrams

Deliverables:

- HMI PANEL: a WPF interface reading live Raspberry Pi sensor data
- Alarm handling with a documented threshold you can defend
- Fail-safe behavior demonstrated by disconnecting the sensor while it runs
- WEBXAM 145065 POST-TEST — Week 16, May 17–21

PRIMARY 145065 5.5.7, 5.6.6, 5.6.7, 5.6.5, 2.10.2, 5.6.14
LATENT 145130 2.15.2, 2.15.3, 2.11.1, 2.11.8

UNIT 9 Ship, Document & Demonstrate

WEEKS 18–19 **DATES** May 31 – Jun 10

Focus: Finish it, hand it off, and defend it. Memorial Day May 31.

Students will:

- Create documentation including an implementation plan and user help
- Train a stakeholder to use what you built
- Collect application feedback and maintain the application
- Explain version management, baseline and lifecycle phases, and the impact of changes
- Deliver a formal presentation of your work

Deliverables:

- HMI application shipped with full documentation and a user guide
- A non-technical person operated it successfully without your help
- Final demonstration: what it does, why built this way, what you rejected
- GP4 ENDS June 10

PRIMARY 145065 5.6.8, 5.6.15, 5.6.17, 5.7.1, 5.7.2, 5.7.3, 1.2.2, 1.2.5, 1.2.11
LATENT 145130 7.1.9, 2.12.6

Semester Calendar

Grading Period 3 · February 1 – April 9 · Python

WK	DATES	UNIT	MILESTONES
1	Feb 1–5	Unit 0	Feb 4 BPA State pre-submission. Feb 5 Waiver Day, no classes.
2	Feb 8–12	Unit 1	Feb 12 Presidents' Day Break, no classes. Classes and objects.
3	Feb 15–19	Unit 1	Feb 15 Presidents' Day, no classes. Refactor project due.
4	Feb 22–26	Unit 2	BPA STATE Feb 21–23. Inheritance opens.
5	Mar 1–5	Unit 2	Polymorphism. Class hierarchy project due.
6	Mar 8–12	Unit 3	Exceptions and serialization. Problem drop.
7	Mar 15–19	Unit 4	Flask opens. Routing and templates.
8	Mar 22–26	Unit 4	Mar 26 Spring Break begins. Forms and validation.
9	Mar 29 – Apr 2	Unit 5	Mar 29 Spring Break. Database design and SQL.
10	Apr 5–9	Unit 5	CRUD APP DEPLOYED. GP3 ENDS April 9.

Grading Period 4 · April 12 – June 10 · C#

WK	DATES	UNIT	MILESTONES
11	Apr 12–16	Unit 6	C# opens. Syntax, static typing, Visual Studio 2026.
12	Apr 19–23	Unit 6	Access modifiers, namespaces, the compiler as a teacher.
13	Apr 26–30	Unit 6	Interfaces. Python-to-C# port due.
14	May 3–7	Unit 7	BPA NATIONALS May 4–9. WPF and XAML open.
15	May 10–14	Unit 7	Event handlers and data binding. WPF app due.
16	May 17–21	Unit 8	WEBXAM 145065 POST-TEST. HMI build opens.
17	May 24–28	Unit 8	Alarm states and fail-safe behavior. Sensor integration.

WK	DATES	UNIT	MILESTONES
18	May 31 – Jun 4	Unit 9	May 31 Memorial Day. Documentation and handoff.
19	Jun 7–10	Unit 9	Final demonstrations. GP4 ENDS June 10.

NOTE **Three interruption-heavy weeks**

Weeks 1 through 4 lose four days to a waiver day, Presidents' Day Break, and the State conference. Weeks 8 and 9 lose two more to spring break. Week 14 is consumed by Nationals.

Units 0 and 3 are deliberately one week each so the February losses land on the lightest content. Unit 7 opens the week of Nationals because WPF setup absorbs interruption better than new conceptual material.

Competency Coverage

Primary — 145065

STRAND	NAME	UNITS
1	Business Operations / 21st Century Skills	0, 2, 9
2	IT Fundamentals	0, 7, 8
3	Information Security	3, 4
5	Programming & Software Systems	1, 2, 3, 4, 5, 6, 7, 8, 9
9	Cybersecurity	4

Latent — 145130 coverage carried here

You will take 145130 Applications of Artificial Intelligence senior year. Item counts are from the published WebXam 145130 blueprint.

OUTCOME	NAME	EXAM ITEMS	COVERED IN
8.5	Queries & Transactions	6	5
8.3	Data Entry & Access	5	5
2.4	Emerging Technologies	4	0, 6
2.6	Installation & Configuration	4	3
2.11	Troubleshooting	4	8
2.15	UX/UI Design	4	7, 8
2.8	Databases	3	4, 5
7.3	Production	3	7
7.1	Interactive Media	4	9
8.2	Design & Creation	1	3
2.14	Artificial Intelligence	21	0, 1, 2

CHECKPOINT **Exit criteria**

Design and implement a class hierarchy in two languages. Deploy a database-backed web application to a public URL. Build a desktop interface that reads live data off a physical machine and behaves correctly when that machine stops answering.

You arrive at senior year knowing C#, which is the language 145130 builds with.