

# Programming Curriculum

## Session 1 — Foundations Refresh + Level-Up

10 weekly sessions · Oct 3 – Dec 12, 2026

| Session              | Topics & Subtopics                                                                                                                                                                                                                                              | In-Class Lesson Breakdown (~75 min)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>1.1</b><br>Oct 3  | <b>Setup &amp; Python Core</b> <ul style="list-style-type: none"><li>– Install VS Code + Python; running scripts</li><li>– Variables &amp; data types (int, float, str, bool)</li><li>– Input / output; comments</li><li>– Arithmetic &amp; operators</li></ul> | <b>Welcome &amp; goals (5 min)</b> — Year overview; what we build by May; class norms.<br><b>Environment setup (15 min)</b> — Verify VS Code + Python; run first script; Run button vs terminal.<br><b>Variables &amp; types (20 min)</b> — int/float/str/bool; naming; type(); casting.<br><b>I/O &amp; operators (20 min)</b> — print(), input(), f-strings; arithmetic & precedence.<br><br><b>Examples / Activity:</b> "About Me" console app: collect user info, compute age in days, print a formatted profile card. |
| <b>1.2</b><br>Oct 10 | <b>Control Flow &amp; Logic</b> <ul style="list-style-type: none"><li>– if / elif / else; comparison operators</li><li>– Boolean logic (and, or, not)</li><li>– Nested conditionals; truthy / falsy</li></ul>                                                   | <b>Warm-up (5 min)</b> — Recap vars/I/O; predict-the-output quiz.<br><b>Conditionals (20 min)</b> — if/elif/else; comparison operators; indentation.<br><b>Boolean logic (15 min)</b> — and/or/not; truthy/falsy; combining conditions.<br><b>Nesting &amp; flow (15 min)</b> — Nested conditionals; flowcharts; avoiding deep nesting.<br><br><b>Examples / Activity:</b> Text-based "Choose Your Own Adventure" game with branching paths.                                                                               |
| <b>1.3</b><br>Oct 17 | <b>Loops &amp; Iteration</b> <ul style="list-style-type: none"><li>– for loops with range(); while loops</li><li>– break / continue; iterating strings &amp; lists</li><li>– Nested loops</li></ul>                                                             | <b>Warm-up (5 min)</b> — Recap conditionals.<br><b>for &amp; range() (20 min)</b> — Counting; iterating; accumulator pattern.<br><b>while loops (15 min)</b> — Conditions; break/continue; avoid infinite loops.<br><b>Nested loops (15 min)</b> — Grids & patterns; loop-in-loop reasoning.<br><br><b>Examples / Activity:</b> Number-guessing game + a pattern / multiplication-table printer.                                                                                                                           |
| <b>1.4</b><br>Oct 24 | <b>Functions &amp; Clean Code</b> <ul style="list-style-type: none"><li>– def, parameters, return values</li><li>– Local vs global scope; default arguments</li><li>– Docstrings, naming &amp; PEP8 style, DRY principle</li></ul>                              | <b>Warm-up (5 min)</b> — Recap loops.<br><b>Defining functions (20 min)</b> — def, parameters, return; calling; why functions.<br><b>Scope &amp; arguments (15 min)</b> — Local vs global; default args; return values.<br><b>Clean-code habits (15 min)</b> — Docstrings; naming; PEP8; DRY.<br><br><b>Examples / Activity:</b> temperature converter, tip calculator, password-strength checker.                                                                                                                         |

| Session              | Topics & Subtopics                                                                                                                                                                                                                                                           | In-Class Lesson Breakdown (~75 min)                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>1.5</b><br>Oct 31 | <b>Data Structures</b> <ul style="list-style-type: none"> <li>– Lists (methods, slicing), tuples, sets</li> <li>– Dictionaries (key/value), nesting</li> <li>– List &amp; dictionary comprehensions</li> </ul>                                                               | <b>Warm-up (5 min)</b> – Recap functions.<br><b>Lists, tuples, sets (20 min)</b> – Create/modify; methods; slicing; when to use each.<br><b>Dictionaries (20 min)</b> – Key/value; access, update, iterate; nesting.<br><b>Comprehensions (10 min)</b> – List & dict comprehensions (intro).<br><hr/> <b>Examples / Activity:</b> Contact book / gradebook: add, search, update, and summarize entries.                         |
| <b>1.6</b><br>Nov 7  | <b>Strings, Files &amp; Data Formats</b> <ul style="list-style-type: none"> <li>– String methods &amp; f-strings</li> <li>– Reading / writing text files</li> <li>– Parsing CSV; intro to JSON</li> </ul>                                                                    | <b>Warm-up (5 min)</b> – Recap data structures.<br><b>String toolkit (15 min)</b> – Methods; slicing; f-strings; formatting.<br><b>File I/O (20 min)</b> – Open/read/write text; with-blocks.<br><b>CSV &amp; JSON (20 min)</b> – Parse CSV; read/write JSON.<br><hr/> <b>Examples / Activity:</b> Read a movies CSV, filter/sort it, write results out; save & load a to-do list as JSON.                                      |
| <b>1.7</b><br>Nov 14 | <b>Errors, Debugging &amp; Testing</b> <ul style="list-style-type: none"> <li>– Syntax vs runtime vs logic errors; reading tracebacks</li> <li>– try / except; common exceptions</li> <li>– Print-debugging + VS Code debugger; simple asserts / tests</li> </ul>            | <b>Warm-up (5 min)</b> – Recap files.<br><b>Types of errors (15 min)</b> – Syntax vs runtime vs logic; reading tracebacks.<br><b>try / except (20 min)</b> – Catching exceptions; common ones; validation.<br><b>Debugging tools (20 min)</b> – print-debugging; VS Code debugger; breakpoints; assert.<br><hr/> <b>Examples / Activity:</b> Fix a buggy program; add input validation and error handling.                      |
| <b>1.8</b><br>Nov 21 | <b>Object-Oriented Programming (New)</b> <ul style="list-style-type: none"> <li>– Classes &amp; objects; __init__</li> <li>– Attributes &amp; methods; why OOP for larger apps</li> <li>– Simple inheritance (intro)</li> </ul>                                              | <b>Why OOP (10 min)</b> – Modeling real things; objects around us.<br><b>Classes &amp; objects (20 min)</b> – class, __init__, attributes; instances.<br><b>Methods (20 min)</b> – Behaviors; self; calling methods.<br><b>Inheritance intro (10 min)</b> – Subclasses; reuse.<br><hr/> <b>Examples / Activity:</b> Model a real thing: a BankAccount or Pet class with methods; build a small inventory of objects.            |
| <b>1.9</b><br>Dec 5  | <b>Libraries, Environments &amp; APIs</b> <ul style="list-style-type: none"> <li>– Built-in libs (math, random, datetime)</li> <li>– pip &amp; virtual environments; installing packages</li> <li>– The requests library; calling a public REST API; reading JSON</li> </ul> | <b>Warm-up (5 min)</b> – Recap OOP.<br><b>Built-in libraries (15 min)</b> – math, random, datetime; import styles.<br><b>pip &amp; virtual envs (20 min)</b> – Install packages; venv; why isolation.<br><b>Calling an API (20 min)</b> – requests; GET; reading JSON; keys/rate limits (intro).<br><hr/> <b>Examples / Activity:</b> Pull live data from a free public API (weather, jokes, or Pokémon) and display it nicely. |

| Session                      | Topics & Subtopics                                                                                                                                                                                                                                | In-Class Lesson Breakdown (≈75 min)                                                                                                                                                                                                                                                                                                     |
|------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>1.10</b><br><b>Dec 12</b> | <b>Git, GitHub &amp; Mini-Capstone</b> <ul style="list-style-type: none"> <li>– What is version control; git init / add / commit / log</li> <li>– Branches (intro); good commit messages</li> <li>– GitHub: repos, push / pull, README</li> </ul> | <b>What is version control (10 min)</b> – Why Git; snapshots of your work.<br><b>Git basics (20 min)</b> – init/add/commit/log; branches (intro); messages.<br><b>GitHub (20 min)</b> – Repos; push/pull; README; profile.<br><hr/> <b>Examples / Activity:</b> Publish a project to GitHub with a README; small capstone + short demo. |

## Session 2 — Build & Ship Real Apps with Claude Code

17 weekly sessions · Feb 6 – May 29, 2027 · Prereq: Session 1 (or equivalent Python)

Required for every student: a Claude Code subscription, plus free accounts on GitHub, Supabase, Stripe (test mode), and Vercel/Netlify. Class project stack: HTML / CSS / JavaScript + Supabase (database & auth) + Stripe (payments). Session 1 (Feb 6) is a survey lesson on how the web is built and why we chose this stack.

| Session              | Topics & Subtopics                                                                                                                                                                                                                                                        | In-Class Lesson Breakdown (≈75 min)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>2.1</b><br>Feb 6  | <b>How the Web Is Built &amp; Choosing Your Stack (New)</b> <ul style="list-style-type: none"><li>– The three layers of an app</li><li>– The main stack options &amp; trade-offs</li><li>– Where React, Python &amp; no-code fit</li></ul>                                | <p><b>The 3 layers (10 min)</b> — Frontend / backend / database; what runs in the browser vs on a server.</p> <p><b>The main options (20 min)</b> — Plain HTML/JS + Supabase · React + Supabase · Python (Flask/FastAPI) backend · No-code (Wix/Bubble) — trade-offs of each.</p> <p><b>Stack detective (15 min)</b> — Tour real sites, "guess the stack," discuss why.</p> <p><b>Where React &amp; Python fit (10 min)</b> — React = frontend layer; Flask/FastAPI = backend layer; Supabase = backend-in-a-box.</p> <p><b>Code vs no-code (10 min)</b> — When to hand-code vs Wix/Squarespace/Bubble; why code wins for a real startup.</p> <p><b>How Claude Code fits any path (5 min)</b> — The agent can build in any stack — you still choose &amp; review.</p> <hr/> <p><b>Examples / Activity:</b> "Stack detective" tour of real sites + class commits to the path we build on.</p> |
| <b>2.2</b><br>Feb 13 | <b>Modern Dev Setup &amp; the Claude Code Workflow</b> <ul style="list-style-type: none"><li>– Terminal / command-line basics; Git &amp; GitHub refresher</li><li>– Installing &amp; configuring Claude Code</li><li>– The plan → build → review → iterate loop</li></ul> | <p><b>Terminal basics (15 min)</b> — Navigate folders; run commands; why the CLI.</p> <p><b>Git/GitHub refresher (10 min)</b> — Repo, commit, push recap.</p> <p><b>Install Claude Code (20 min)</b> — Setup; auth; first run; project folder.</p> <p><b>The agent loop (15 min)</b> — plan → build → review → iterate.</p> <p><b>First page (15 min)</b> — Scaffold a "hello world" page; commit to GitHub.</p> <hr/> <p><b>Examples / Activity:</b> Set up the environment; use Claude Code to scaffold a "hello world" web page; commit to GitHub.</p>                                                                                                                                                                                                                                                                                                                                    |
| <b>2.3</b><br>Feb 20 | <b>Working with an AI Coding Agent</b> <ul style="list-style-type: none"><li>– Writing clear prompts &amp; specs</li><li>– Reading &amp; reviewing generated code; debugging AI output</li><li>– When to accept vs redirect; avoiding blind "vibe coding"</li></ul>       | <p><b>Prompting well (20 min)</b> — Clear specs; context; constraints; examples.</p> <p><b>Reading generated code (20 min)</b> — Skim, understand; ask the agent to explain.</p> <p><b>Review &amp; redirect (15 min)</b> — Accept vs redo; iterating; not "vibe coding."</p> <p><b>Debugging AI output (10 min)</b> — Spot bugs; feed errors back to the agent.</p> <hr/> <p><b>Examples / Activity:</b> Guide Claude Code to build an interactive page, then read through and explain every part.</p>                                                                                                                                                                                                                                                                                                                                                                                      |

| Session              | Topics & Subtopics                                                                                                                                                                                                                               | In-Class Lesson Breakdown (≈75 min)                                                                                                                                                                                                                                                                                                           |
|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>2.4</b><br>Feb 27 | <b>How the Web Works + HTML</b> <ul style="list-style-type: none"> <li>– Client / server, HTTP, URLs; hosting &amp; domains</li> <li>– HTML structure, tags, semantic elements, forms</li> </ul>                                                 | <b>Request lifecycle (15 min)</b> – Client/server; HTTP; URLs; DNS; hosting.<br><b>HTML structure (20 min)</b> – Tags, elements, nesting, semantic HTML.<br><b>Forms &amp; inputs (20 min)</b> – Inputs, labels, buttons, forms.<br><hr/> <b>Examples / Activity:</b> Build the HTML skeleton of a personal portfolio page.                   |
| <b>2.5</b><br>Mar 6  | <b>CSS, Styling &amp; Deployment</b> <ul style="list-style-type: none"> <li>– CSS selectors, box model, flexbox</li> <li>– Responsive design basics</li> <li>– Deploying a static site to Vercel / Netlify</li> </ul>                            | <b>CSS basics (15 min)</b> – Selectors, properties, colors, fonts.<br><b>Box model &amp; layout (20 min)</b> – Margin/padding/border; flexbox.<br><b>Responsive design (15 min)</b> – Mobile-friendly; media queries (intro).<br><hr/> <b>Examples / Activity:</b> Style and deploy the portfolio site live on a real URL.                    |
| <b>2.6</b><br>Mar 13 | <b>JavaScript &amp; Interactivity</b> <ul style="list-style-type: none"> <li>– JS basics: variables, functions</li> <li>– The DOM; events (clicks, input)</li> <li>– Updating the page dynamically</li> </ul>                                    | <b>JS basics (20 min)</b> – Variables, functions, conditionals in JS.<br><b>The DOM (20 min)</b> – Selecting elements; changing content.<br><b>Events (20 min)</b> – Clicks, input; event handlers.<br><hr/> <b>Examples / Activity:</b> Add interactivity: a working contact form, dark-mode toggle, or a quiz.                              |
| <b>2.7</b><br>Mar 20 | <b>Fetching Data &amp; Working with APIs</b> <ul style="list-style-type: none"> <li>– async basics; fetch()</li> <li>– Consuming a public API; displaying JSON</li> <li>– Handling loading &amp; error states</li> </ul>                         | <b>async &amp; fetch (20 min)</b> – Promises (intro); fetch(); await.<br><b>Consuming an API (20 min)</b> – Endpoints; JSON; displaying data.<br><b>Loading &amp; errors (15 min)</b> – Spinners; error states; try/catch.<br><hr/> <b>Examples / Activity:</b> Build a small app that fetches and shows live data (e.g., weather or search). |
| <b>2.8</b><br>Mar 27 | <b>Databases 101 + Supabase Setup</b> <ul style="list-style-type: none"> <li>– What a database is; tables, rows, columns</li> <li>– SQL basics: select / insert / update / delete</li> <li>– Creating a Supabase project &amp; tables</li> </ul> | <b>What is a database (15 min)</b> – Tables, rows, columns; the relational idea.<br><b>SQL basics (20 min)</b> – select / insert / update / delete.<br><b>Supabase project (20 min)</b> – Create project; make tables; the dashboard.<br><hr/> <b>Examples / Activity:</b> Design a schema for an app idea and create the tables in Supabase. |
| <b>2.9</b><br>Apr 3  | <b>Connecting the App to a Database (CRUD)</b> <ul style="list-style-type: none"> <li>– Connecting the frontend to Supabase</li> <li>– Create / read / update / delete data</li> <li>– Displaying database data in the UI</li> </ul>             | <b>Connect to Supabase (15 min)</b> – Client library; keys; connect.<br><b>Read &amp; display (20 min)</b> – Query data; render in the UI.<br><b>Create/update/delete (25 min)</b> – Full CRUD from the app.<br><hr/> <b>Examples / Activity:</b> Build a dynamic app (to-do list, notes, or a mini social wall) backed by Supabase.          |

| Session               | Topics & Subtopics                                                                                                                                                                                                         | In-Class Lesson Breakdown (~75 min)                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>2.10</b><br>Apr 10 | <b>User Accounts &amp; Authentication</b> <ul style="list-style-type: none"> <li>– Signup / login / logout with Supabase Auth</li> <li>– Protecting pages; user-specific data</li> </ul>                                   | <b>Why auth (10 min)</b> – Accounts, sessions, security basics.<br><b>Supabase Auth (25 min)</b> – Signup / login / logout.<br><b>Protecting pages (20 min)</b> – Auth checks; redirects; user-specific data.<br><hr/> <b>Examples / Activity:</b> Add real accounts so each user sees only their own data.                                                                                                                                          |
| <b>2.11</b><br>Apr 17 | <b>Security &amp; Secrets Basics</b> <ul style="list-style-type: none"> <li>– Environment variables &amp; API keys; never commit secrets</li> <li>– Row-Level Security (intuition); input validation</li> </ul>            | <b>Secrets &amp; env vars (20 min)</b> – API keys; .env; never commit secrets.<br><b>Row-Level Security (25 min)</b> – RLS intuition; rules; testing access.<br><b>Input validation (15 min)</b> – Sanitizing; preventing bad data.<br><b>Security lab (15 min)</b> – Lock down the app; test as different users.<br><hr/> <b>Examples / Activity:</b> Secure the app: move keys to env vars, add RLS rules, test that users can't see others' data. |
| <b>2.12</b><br>Apr 24 | <b>Payments with Stripe</b> <ul style="list-style-type: none"> <li>– How online payments work</li> <li>– Stripe test mode &amp; Checkout</li> <li>– Recording a successful payment</li> </ul>                              | <b>How payments work (15 min)</b> – Cards, processors, the checkout flow.<br><b>Stripe test mode (20 min)</b> – Account; test keys; test cards.<br><b>Checkout (25 min)</b> – Add a Checkout button; product/price.<br><b>Record payment (15 min)</b> – Handle success; mark user as paid.<br><hr/> <b>Examples / Activity:</b> Add a "Buy" / "Go Pro" button using Stripe test-mode checkout.                                                       |
| <b>2.13</b><br>May 1  | <b>Third-Party Integrations</b> <ul style="list-style-type: none"> <li>– Reading API docs</li> <li>– Adding a service (maps, email, image upload, or an AI feature)</li> <li>– Webhooks (intro)</li> </ul>                 | <b>Reading API docs (15 min)</b> – Find endpoints, auth, examples.<br><b>Add a service (30 min)</b> – Maps / email / image upload / AI feature.<br><b>Webhooks intro (15 min)</b> – Events from services (concept).<br><hr/> <b>Examples / Activity:</b> Integrate one external service into the app.                                                                                                                                                |
| <b>2.14</b><br>May 8  | <b>Full Deployment End-to-End</b> <ul style="list-style-type: none"> <li>– Deploying a full app (frontend + Supabase)</li> <li>– Custom domains (intro)</li> <li>– Testing in production; debugging live issues</li> </ul> | <b>Deploy full app (25 min)</b> – Frontend + Supabase live together.<br><b>Custom domains (15 min)</b> – Buying/pointing a domain (intro).<br><b>Test in production (20 min)</b> – Real-world testing; debugging live.<br><hr/> <b>Examples / Activity:</b> Ship the full app live and share the link.                                                                                                                                               |
| <b>2.15</b><br>May 15 | <b>Product Thinking, UX &amp; Project Management</b> <ul style="list-style-type: none"> <li>– Simple UI / UX principles</li> <li>– Breaking work into tasks; GitHub Issues / kanban; feedback</li> </ul>                   | <b>UI/UX basics (20 min)</b> – Clarity, hierarchy, accessibility (intro).<br><b>Task management (20 min)</b> – Break work down; GitHub Issues / kanban.<br><hr/> <b>Examples / Activity:</b> Scope a capstone MVP, create an issues board, and sketch the UI.                                                                                                                                                                                        |

| Session               | Topics & Subtopics                                                                                                                                                 | In-Class Lesson Breakdown (≈75 min)                                                                                                                                                                                    |
|-----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>2.16</b><br>May 22 | <b>Capstone Build (Guided)</b> <ul style="list-style-type: none"> <li>– Apply the full stack end-to-end</li> <li>– Instructor &amp; AI-assisted support</li> </ul> | <b>Standup (10 min)</b> – Plans; blockers.<br><b>Build sprint (50 min)</b> – Full-stack app with DB + auth + payment/integration.<br><b>Check-in &amp; review (15 min)</b> – Instructor + peer review.                 |
| <b>2.17</b><br>May 29 | <b>Capstone Polish</b> <ul style="list-style-type: none"> <li>– Final polish &amp; bug-fixing</li> <li>– Pitch / presentation</li> </ul>                           | <b>Polish &amp; fix (25 min)</b> – Bugs; final styling; edge cases.<br><b>Prep the pitch (15 min)</b> – What problem; demo script.<br><br><b>Examples / Activity:</b> Live demo and short pitch of each student's app. |