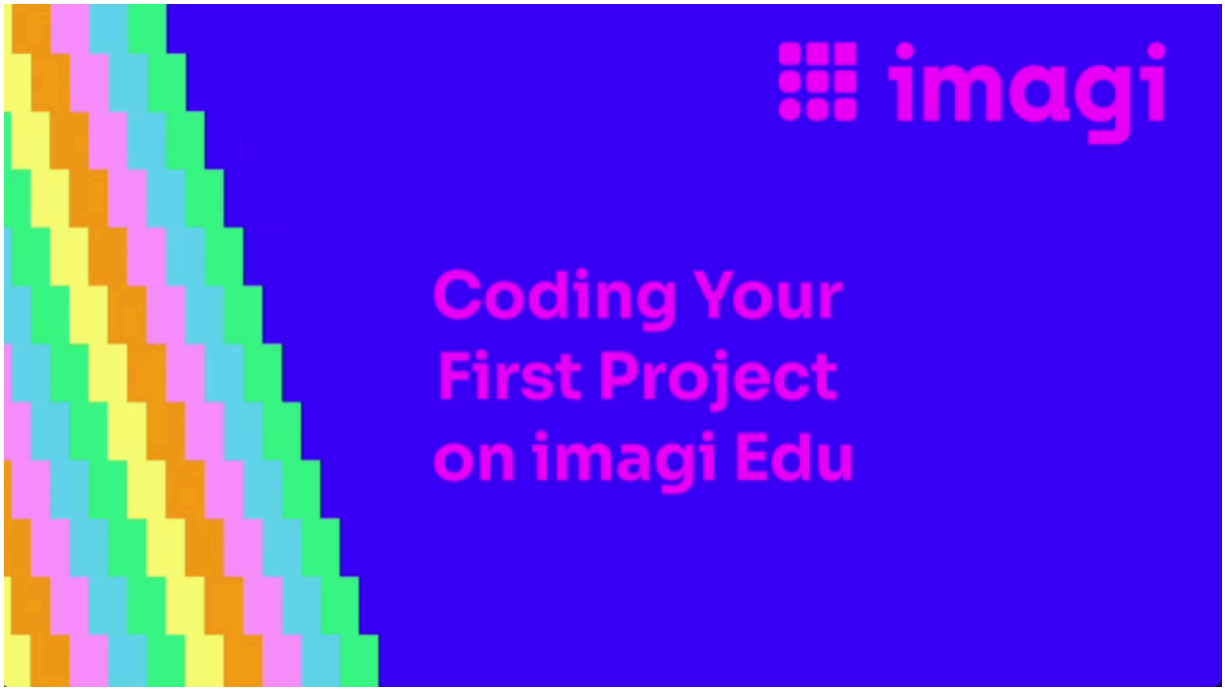
Creative Coding with Python and AI



Overview and Purpose

In this one-hour lesson, students will explore how AI can support creative coding with Python. They'll learn how to write simple code to create pixel art and animations while using an AI-powered Debugging Buddy to identify and fix errors. Students will practice computational thinking, creativity, and responsible use of AI.

Duration	1 hour (with follow-on activities)
Grades	3-8
Level	Beginner
Platform	Web/tablet (internet is required)
Student privacy	Built for COPPA compliance
Materials and Resources	imagi Edu - Setup Guide Creative Coding with Python and AI - Teacher Slides imagi Edu - Python Learning Center FAQ

Suggested Learning Targets

1. Understand what AI is and how it supports coding.

© 2026 imagi Education Inc. All rights reserved.

2. Create simple Python programs using variables, RGB colors, functions, loops, and animations.
3. Use the **Debugging Buddy** responsibly to identify and fix coding errors.
4. Demonstrate perseverance and a growth mindset.

Glossary

- **AI (Artificial Intelligence)**: A type of computer technology that can learn from information and make decisions like humans.
- **Matrix**: A grid of pixels used for creating digital images or animations.
- **Debugging**: The process of finding and fixing errors (bugs) in code.
- **Variable**: A name that stores a piece of information in a program.
- **Function**: A reusable block of code that performs a specific task.
- **Animation**: A sequence of frames that creates the illusion of movement.

Documentation

```
m[2][3] = on  # Turns on a pixel
m[2][3] = (255, 0, 0)  # Colors the pixel red using RGB
eye = (25, 200, 207)  # Creates a variable for eye color

def smiley(eye, mouth):
    m[2][2] = eye
    m[2][5] = eye
    m[5][1] = mouth
smiley(A, R)

a = Animation()  # Creates an animation
a.add_frame(m)   # Adds a frame to the animation
```

Teacher Tips

- Encourage students to debug independently before using the Debugging Buddy.
- Highlight how AI provides guidance, not answers — students must evaluate its responses.
- Remind them to check indentation and syntax carefully when using loops and functions.
- This lesson doesn't use Lovable. If you've turned on Lovable for this classroom, switch it off for this lesson so students stay in the Python editor.
- Ask your IT team to allow imagilabs.com so pages load on school devices.

Technical accommodations

imagi Edu runs in the browser, so students can use their device's built-in accessibility tools: zoom or a magnifier, high-contrast mode, and a screen reader or

© 2026 imagi Education Inc.

select-to-speak to read prompts and AI responses aloud. Students can dictate prompts with their device's own voice typing tool.

Instructional accommodations

Pair students so one reads the code aloud and one types, and give students who need support a printed copy of the code examples.

Standards

AI Literacy Competencies (AILit Framework)

1. **Manage 1:** Decide whether to use AI systems based on the nature of the task.
    a. **Manage 4:** Monitor and evaluate AI use throughout a problem-solving process.
2. **Create 3:** Direct generative AI systems to elicit feedback, refine results and support reflection.
3. **Engage 3:** Evaluate whether AI outputs should be accepted, revised, or rejected.
    a. **Engage 2:** Describe how AI systems perform tasks using language that addresses and clarifies common misconceptions.

CSTA 2026 Standards

<u>Grades 3 to 5</u>

E3-PRO-PD-04, E5-PRO-VD-06, E4-PRO-TR-08, E5-PRO-TR-08, E5-SOC-HU-18,

<u>Middle school</u>

MS-ALG-PS-05, MS-SOC-ET-40, MS-PRO-TR-19, MS-PRO-RD-17

ISTE

1.1 Empowered Learner

- 1.1.a – Students connect their learning needs, strengths and interests to their goals and use technology to help achieve them and reflect on their progress.
- 1.1.b – Students build networks and customize their learning environments in ways that support the learning process.
- 1.1.c – Students use technology to seek feedback that informs and improves their practice and to demonstrate their learning in a variety of ways.
- 1.1.d – Students understand fundamental concepts of how technology works, demonstrate the ability to choose and use current technologies effectively, and are adept at thoughtfully exploring emerging technologies.

1.2 Digital Citizen

- 1.2.b – Students demonstrate empathetic, inclusive interactions online and use technology to responsibly contribute to their communities.

© 2026 imagi Education Inc.

- 1.2.c – Students safeguard their well-being by being intentional about what they do online and how much time they spend online.

1.3 Knowledge Constructor
- 1.3.a – Students use effective research strategies to find resources that support their learning needs, personal interests and creative pursuits.
- 1.3.c – Students curate information from digital resources using a variety of tools and methods to create collections of artifacts that demonstrate meaningful connections or conclusions.
- 1.3.d – Students build knowledge by exploring real-world issues and gain experience in applying their learning in authentic settings.

1.4 Innovative Designer
- 1.4.a – Students know and use a deliberate design process for generating ideas, testing theories, creating innovative artifacts or solving authentic problems.
- 1.4.b – Students select and use digital tools to plan and manage a design process that considers design constraints and calculated risks.
- 1.4.c – Students develop, test and refine prototypes as part of a cyclical design process.
- 1.4.d – Students exhibit a tolerance for ambiguity, perseverance and the capacity to work with open-ended problems.

1.5 Computational Thinker
- 1.5.a – Students formulate problem definitions suited for technology-assisted methods such as data analysis, abstract models and algorithmic thinking in exploring and finding solutions.
- 1.5.b – Students collect data or identify relevant data sets, use digital tools to analyze them, and represent data in various ways to facilitate problem-solving and decision-making.
- 1.5.c – Students break problems into component parts, extract key information, and develop descriptive models to understand complex systems or facilitate problem-solving.
- 1.5.d – Students understand how automation works and use algorithmic thinking to develop a sequence of steps to create and test automated solutions.

1.6 Creative Communicator
- 1.6.a – Students choose the appropriate platforms and digital tools for meeting the desired objectives of their creation or communication.
- 1.6.b – Students create original works or responsibly repurpose or remix digital resources into new creations.
- 1.6.c – Students use digital tools to visually communicate complex ideas to others.
- 1.6.d – Students publish or present content that customizes the message and medium for their intended audiences.

1.7 Global Collaborator
- 1.7.a – Students use digital tools to connect with learners from a variety of backgrounds, recognizing diverse viewpoints and broadening mutual understanding.

© 2026 imagi Education Inc.

- 1.7.b – Students use collaborative technologies to work with others, including peers, experts or community members, to examine issues and problems from multiple viewpoints.
- 1.7.c – Students contribute constructively to project teams, assuming various roles and responsibilities to work effectively toward a common goal.
- 1.7.d – Students explore local and global issues and use collaborative technologies to work with others to investigate solutions.

© 2026 imagi Education Inc.

Suggested Lesson Structure

Part 1: Coding Basics and Debugging Buddy

1. Intro to Python and the Matrix [Direct Instruction] 10 mins

Introduce students to the imagi Edu platform and explain how code interacts with the pixel matrix. Show the interface: code editor, error messages, and output grid. Explain how a pixel can be turned on and off using simple Python commands such as `m[row][col] = on`. Provide quick examples and have students predict what will happen before running code.

2. Run Your First Code! [Guided Practice] 5 mins

Students create a new project and run the sample code. Ask: “What does this line do?” Discuss coordinates and matrix structure (rows and columns). Guide students through common errors (e.g., syntax, index out of range) and introduce the Debugging Buddy as a tool to help identify and fix errors.

```python
m[2][3] = on
```

3. Debugging Buddy [Unplugged + Interactive Discussion] 5 mins

Define “bugs” and “debugging.”Explain why the Debugging Buddy doesn’t fix code but gives hints. Have students identify an intentional syntax error on the board and work together to debug it. Reinforce responsible AI use — ask: “When should we ask the Debugging Buddy for help?”

4. Add Some Color! [Guided Practice] 10 mins

Students modify their code to replace `on` with color variables (e.g., `R`, `G`, `B`). Demonstrate how to use RGB color values such as `(255, 0, 0)` for red. Show how to use the Color Picker to find RGB combinations. Discuss computational thinking over memorization: patterns > memorizing values.

5. Variables for Efficiency [Direct Instruction] 5 mins

Introduce variables as a way to store color values. Show how changing one variable updates multiple parts of the code. Ask students to describe why this makes code easier to manage.

```python
eye = (25, 200, 207)
mouth = (255, 165, 150)
```

© 2026 imagi Education Inc.

```
m[2][2] = eye
m[2][5] = eye
m[5][1] = mouth
m[6][2] = mouth
m[6][3] = mouth
m[6][4] = mouth
m[6][5] = mouth
m[5][6] = mouth
```

6. My Colorful Smiley [Guided Practice] 10 mins

Students recreate a colorful smiley face using variables for different elements (eyes, mouth). Encourage experimentation with colors and patterns. Walk around and assist students using Debugging Buddy tips.

Part 2: Functions, Loops, and Animation

1. Functions for Reusable Code [Direct Instruction]

Explain what functions are and why they're useful. Walk through defining and calling a function. Call the function using `smiley(A, R)` and discuss parameters.

Python

```
def smiley(eye, mouth):
    m[2][2] = eye
    m[2][5] = eye
    m[5][1] = mouth
    m[6][2] = mouth
    m[6][3] = mouth
    m[6][4] = mouth
    m[6][5] = mouth
    m[5][6] = mouth
smiley(A, R)
```

2. Animate It! [Guided Practice]

Introduce the concept of animation using frames. Students observe how frames create motion. Discuss: "What happens if we swap the colors or add more frames?"

Python

```
a = Animation()
def smiley(eye, mouth):
    m[2][2] = eye
```

© 2026 imagi Education Inc.

```
    m[2][5] = eye
    m[5][1] = mouth
    m[6][2] = mouth
    m[6][3] = mouth
    m[6][4] = mouth
    m[6][5] = mouth
    m[5][6] = mouth
smiley(R, A)
a.add_frame(m)
smiley(A, R)
a.add_frame(m)
```

3. For Loops and Patterns [Direct Instruction + Guided Practice]

Explain how loops reduce repetition. Show how to fill a full row of pixels with a loop. Discuss indentation and the concept of iteration. Prompt students to predict output before running.

Python

```
for col in range(0, 8):
    m[0][col] = R
```

4. Rainbow Challenge [Guided Practice]

Students expand their loop to create multiple colored rows. Encourage exploration: What happens if we change the range or colors? Introduce adding animation frames inside the loop.

Python

```
for col in range(0, 8):
    m[0][col] = R
    m[1][col] = O
    m[2][col] = Y
    m[3][col] = G
    m[4][col] = A
    m[5][col] = B
    m[6][col] = P
    m[7][col] = M
```

5. Animate Your Rainbow! [Guided Practice]

© 2026 imagi Education Inc.

Students combine loops, animation, and color to create a moving rainbow. Encourage creativity—students can make wave effects or flashing lights. Have them debug independently or with the Debugging Buddy.

```python
a = Animation()
for col in range(0, 8):
    m[0][col] = R
    m[1][col] = O
    m[2][col] = Y
    m[3][col] = G
    m[4][col] = A
    m[5][col] = B
    m[6][col] = P
    m[7][col] = M
    a.add_frame(m)
```

6. Responsible AI Use [Additional Activity]

Discuss how to use AI responsibly:

- Try problem-solving before asking the AI.
- Ask clear, specific questions.
- Always evaluate the AI's responses critically.

Students reflect: "Which AI skill do I want to improve—problem-solving, communication, or perseverance?"

Common Misconceptions

Address misconceptions that might have come up. Examples:

Misconception	Explanation
AI will fix my code for me.	The Debugging Buddy helps students understand their mistakes - it doesn't fix them automatically. Students must still apply problem-solving and reasoning skills to correct errors.
I should ask the Debugging Buddy right away when I see an error.	Students should first try to solve the problem themselves, review their code, and think critically before asking for help. This develops resilience and strengthens problem-solving skills.
Debugging is only about finding syntax errors.	Debugging also includes fixing logic errors, improving sequencing, and testing for unexpected outcomes. Teaching this helps students develop computational thinking.

© 2026 imagi Education Inc.

Checking for Understanding

Administer a quick exit ticket at the end of class or assign any of the following as homework:

Quiz

1. What does the Debugging Buddy do?
   a) Writes your code for you
   b) Helps you fix and understand errors
   c) Adds colors automatically
   d) Makes your project faster

2. What is a variable used for?
   a) To repeat code
   b) To store information
   c) To color the matrix
   d) To debug programs

3. What should you do before asking AI for help?
   a) Give up
   b) Ask a friend
   c) Try to fix the issue yourself
   d) Restart the project

Answer Key:

1. b
2. b
3. c

Exercises

1. Write code that colors 3 rows of pixels in different RGB colors.
2. Modify the code in Exercise 1 to animate the colors using frames.
3. Identify and fix the bug in the following code:

```python
for col in range(0, 8):
m[0][col] = (255, 0, 0)
```

Solutions:

1.

© 2026 imagi Education Inc.

Python

```python
# Three distinct RGB colors (you can change these!)
row0 = (224, 182, 34)   # gold
row1 = (143, 88, 152)   # purple
row2 = (25, 200, 207)   # aqua

# Color the top three rows
for col in range(8):
    m[0][col] = row0
    m[1][col] = row1
    m[2][col] = row2
```

2.

Python

```python
row0 = (224, 182, 34)
row1 = (143, 88, 152)
row2 = (25, 200, 207)

a = Animation()

for col in range(8):
    m[0][col] = row0
    m[1][col] = row1
    m[2][col] = row2
    a.add_frame(m)
```

3. **Error message:** expected an indented block after 'for' statement on line 1 (<string>, line 2).

   Debugging Buddy: The error occurs because the line inside your for loop is not indented. In Python, code blocks that follow statements like `for` need to be indented to show that they belong to that block. To fix this, you need to add an indentation (usually 4 spaces) before the line that sets the color of the pixel.

Python

```python
for col in range(0, 8):
    m[0][col] = (255, 0, 0)
```

© 2026 imagi Education Inc.

Student Reflection

1. What went well today?
2. What problem(s) came up?
3. What did you find most interesting in the lesson?
4. What did you find most challenging in the lesson?
5. What do you still have questions about?

Rubric

Suggested rubric for evaluating how well the student has met each learning target.

Area	Learning Target	Beginning	Developing	Proficient	Exceptional
Knowledge	Understand what AI is and how it supports coding	Cannot explain the role of AI in coding	Vaguely understands the role of AI in coding	Explains the AI's purpose clearly	Connects AI's role to real-world applications
Skill	Create code using variables, RGB colors, and animation	Requires significant help	Completes steps with guidance	Works independently	Adds creative features
Thinking	Debug using DB responsibly	Relies on DB for all help	Occasionally consults with DB	Evaluates and applies DB suggestions critically	Demonstrates independent reasoning when applying DB

© 2026 imagi Education Inc.

Intrapersonal	*Demonstrate perseverance and growth mindset*	*Easily gives up*	*Needs encouragement*	*Shows persistence*	*Reflects on challenges and growth*

Follow-on activities

For educators and students eager to continue exploring creative coding and AI, imagi Edu offers full-length courses and cross-curricular lessons that build upon the concepts introduced in this lesson.

Grades 3-5	Creative Computer Science with Python ◆ Beginner 1	15 weeks (15-30 hours)
Grades 3-5	Creative Computer Science with Python ◆ Beginner 2	15 weeks (15-30 hours)
Grades 6-8	Creative Computer Science with Python ◆ Beginner	15 weeks (30-45 hours)
Grades 6-8	Creative Computer Science with Python ◆ Intermediate	15 weeks (30-45 hours)
Grades 3-5, 6-8	Cross-curricular Lessons	

Browse the curriculum library on the imagi Edu platform: https://edu.imagilabs.com/

© 2026 imagi Education Inc.