VIDHYAVISION

AI & Coding Starter Guide

A beginner-friendly, offline-capable learning workbook for students

CODING	AI	PROJECTS
Think step-by-step	Understand, question, create	Build something useful

Designed for VidhyaVision sessions and independent practice

Version 1.0 • 2026

How to use this guide

This guide is meant to support a class, volunteer-led session, or independent practice. You do not need to finish everything at once. A good pace is one module per session, with time to discuss and build.

1. Learn — Read the short explanation and ask questions when a word or idea is unfamiliar.

2. Try — Complete the small exercise without worrying about being perfect on the first attempt.

3. Build — Use the project prompts to turn the idea into something you can demonstrate.

4. Reflect — Write down what worked, what failed, and what you would change next time.

Four ground rules

- It is okay not to know. Good technology work begins with questions.
- Do not copy code or AI output you cannot explain in your own words.
- Protect personal information. Never paste passwords, private photos, school records, phone numbers, or other sensitive information into public tools.
- A failed attempt is useful data. Debugging means learning why something did not work.

Module 1 — Computational Thinking

Before learning a programming language, learn to break a problem into clear steps. Computers are fast, but they are literal: they need instructions that are precise enough to follow.

The four habits

Habit	What it means
Decompose	Break one large problem into smaller problems.
Recognize patterns	Notice what repeats or looks similar.
Abstract	Focus on the details that matter and ignore the ones that do not.
Design an algorithm	Write an ordered set of steps that solves the problem.

Try it: explain a familiar task

Write an algorithm for one task below. Pretend the reader has never done it before.

- Make a cup of tea
- Send a photo to a family member
- Check tomorrow's weather on a phone
- Find the largest number in a list

1. _____

2. _____

3. _____

4. _____

5. _____

Module 2 — Python Foundations

Python is a programming language designed to be readable. The goal here is not memorization. The goal is to learn how data, decisions, repetition, and reusable instructions fit together.

Variables

A variable gives a name to a value.

```
name = "Asha"
age = 14
print(name, age)
```

Conditionals

A conditional lets a program choose what to do.

```
temperature = 34
if temperature > 30:
    print("Hot day")
else:
    print("Not too hot")
```

Loops

A loop repeats an instruction.

```
for number in range(1, 6):
    print(number)
```

Functions

A function packages steps so they can be reused.

```
def greet(name):
    return "Hello " + name

print(greet("Asha"))
```

Practice: predict before you run

What prints?

```
x = 4
y = 3
print(x + y)
```

My prediction: _____

What prints?

```
score = 72
if score >= 60:
    print("Pass")
```

```
else:
    print("Keep practicing")
```

My prediction: _____

How many times does this loop run?

```
for i in range(5):
    print(i)
```

My prediction: _____

Module 3 — What AI Actually Does

Artificial intelligence is a broad term for computer systems that perform tasks associated with human intelligence, such as recognizing patterns, generating language, or making predictions. Different AI systems work in different ways, so avoid treating “AI” as one single machine or mind.

Concept	Student-friendly meaning
Training data	Examples or information used to help a model learn patterns.
Model	A learned mathematical/computational representation used to make predictions or generate outputs.
Prompt	An instruction or input given to a generative AI system.
Inference	Using a trained model to produce an output for a new input.
Hallucination	When a generative model produces information that sounds plausible but is incorrect or unsupported.
Bias	Systematic patterns that can create unfair or misleading outcomes, often influenced by data, design choices, or use context.

AI is useful — and it can be wrong

- Ask AI for explanations, brainstorming, examples, or help finding mistakes — then verify important facts.
- For schoolwork, follow your teacher’s rules about when AI is allowed and what must be your own work.
- Do not treat an AI answer as an authority simply because it sounds confident.
- For health, legal, financial, safety, or other high-stakes decisions, involve a qualified adult or professional rather than relying on a chatbot.

Try it: improve a prompt

Weak prompt: “Teach me Python.”

Stronger prompt: “I am a beginner. Explain Python loops using a simple real-life analogy, then give me one short example and a three-question practice exercise. Do not give the answers until I try.”

Write your own stronger prompt for a topic you are learning:

_____

_

_____

_

_____

_

Module 4 — A No-Code Machine Learning Experiment

A simple way to understand machine learning is to train a model yourself and observe where it succeeds and fails. Google’s Teachable Machine lets you create image, sound, or pose classifiers in a browser.

1. Choose two or three categories that are easy to demonstrate safely (for example: pen / notebook / water bottle).
2. Collect several examples of each category. Vary angle, distance, lighting, and background.
3. Train the model.
4. Test it with new examples that were not in your training set.
5. Record one mistake the model made and propose a reason.
6. Add better or more diverse training examples and test again.

Reflection

What pattern did the model seem to learn?

_____

_

What caused the model to make mistakes?

_____

_

How did changing the training examples change the result?

_____

_

What could go wrong if a real-world system was trained on incomplete examples?

_____

_

Module 5 — Build a Small Technology Project

A good beginner project is small enough to finish but useful enough to explain. Start with a problem, not a technology buzzword.

Project idea	What you could build
School FAQ helper	Design a simple question-and-answer tool or prototype for school schedules, subjects, or common student questions.
Weather learning dashboard	Use sample weather data to practice tables, charts, averages, and simple rules. Keep it educational—not a real agricultural decision system.
Study quiz	Create a Python quiz that asks questions, checks answers, and reports a score.
Community information page	Build a basic webpage that organizes useful public information such as school contacts, learning links, or local services.
Sensor idea	Sketch how a temperature, light, or moisture sensor could collect data and what a student might learn from the readings.

Project canvas

Problem I want to solve:

_____

_

Who is this useful for?

_____

_

What is the smallest version I can build?

_____

_

What information or data does it need?

_____

_

What could go wrong?

_____

_

How will I test whether it works?

_____

_

What will I show in my final demo?

_____

_

Module 6 — Make a Mentor Session Count

A mentor session is most valuable when you arrive with questions and leave with one action to try. You do not need to impress the mentor. Your job is to learn how they think and how they built their path.

- What did you know at my age, and what did you not know yet?
- What problem do you spend the most time solving in your work?
- Which skill matters more in your job than students usually expect?
- What project could a beginner build to understand your field?
- What mistake taught you something important?
- How is AI changing your work?

After the session

One surprising idea:

_____

_

One skill I want to practice:

_____

_

One resource or person I want to explore:

_____

_

My next action this week:

_____

_

Module 7 — Free Learning Library

These are external resources selected as useful starting points. They are not owned by VidhyaVision, and their availability or terms can change. Choose the resource that matches your current level rather than trying to complete everything.

Resource	Best for	Link
Code.org / CodeAI — How AI Works	Beginner videos and lessons on machine learning, neural networks, LLMs, bias, and ethics.	https://code.org/en-US/curriculum/how-artificial-intelligence-works
Google — Teachable Machine	No-code image, sound, and pose machine-learning experiments.	https://teachablemachine.withgoogle.com/
Khan Academy — Intro to Computer Science: Python	Beginner Python and computational thinking.	https://www.khanacademy.org/computing/intro-to-python-fundamentals
Harvard CS50 — Introduction to Programming with Python	Free lectures, practice problems, and a final project.	https://cs50.harvard.edu/python/
Google for Developers — Python Class	Written lessons, videos, and exercises for learners with some programming experience.	https://developers.google.com/edu/python
DIKSHA	School-focused digital learning resources in India.	https://diksha.gov.in/
ePathshala	NCERT digital textbooks and supporting resources.	https://epathshala.nic.in/
MIT Scratch	Creative coding tutorials and projects.	https://scratch.mit.edu/ideas
Arduino Learn	Official Arduino tutorials for physical computing and electronics.	https://docs.arduino.cc/learn/

Final challenge — Explain, Build, Share

Choose one concept from this guide and create a 3–5 minute demonstration for another student. The demonstration can be a short Python program, a Scratch project, an AI experiment, a sensor idea, or a visual explanation.

Your demo should answer:

- What problem or concept are you showing?
- How does it work?
- What did you try that did not work at first?
- What limitation should the audience understand?
- What would you build next with more time or resources?

Curiosity → Practice → Projects → Opportunity

Keep the work small enough to finish and clear enough to explain.