Python Foundations

Ages 13-15 | Tech Builders

Write a small text-based quiz game in Python.

How to use this guide

Read each section, complete its four lab steps, then answer the questions in the online course. The online progress bar advances only after four correct answers.

Section 1: Variables

LEARNING GOAL: Variables. Your finished work should demonstrate: A quiz that asks one question.

EXPLAIN THE CONCEPT: A variable stores a value under a name. Meaningful names make a program easier to read and change. Think about where you might use this skill in a real or imaginary project. Describe the problem it helps you solve and how you would recognize a successful result.

ORIGINAL PROJECT LAB: 1. Open a Python editor and create quiz.py. 2. Store a question and answer in named variables. 3. Print the question and collect input. 4. Run the program and check what was stored. Finished result: A quiz that asks one question.

EXPLAIN THE MECHANISM: A variable name refers to a value. Assignment evaluates the right-hand expression and binds the name to the result. Updating a score does not mean that the old value changes everywhere it was previously printed. Trace state in execution order.

WORKED CASE: For score = 2 followed by score = score + 3, the second line reads the current 2, adds 3, then assigns 5. If bonus = score ran before the update, bonus still refers to 2. Meaningful names help show which value you intend to use.

TEST AND JUSTIFY: Write a two-column trace of line and value before running a tiny program in an adult-approved environment. Predict the output, then compare. Explain the difference between score = 5, which assigns, and score == 5, which compares. Never use personal passwords as sample values.

GO DEEPER: Identify an assumption, test a safe alternative, and explain the trade-off. Keep a short before-and-after record that someone else could follow.

SAFETY CHECK: Recognize phishing and social engineering, secure accounts, understand privacy settings, and perform cyber labs only with authorization. Use pretend data,

never real passwords or private information. Stop and ask a trusted adult if anything seems unsafe.

REVIEW QUESTIONS: What is the main idea? Why does the first step matter? How would you detect a mistake? What evidence proves the result? What would you improve? Answer in your own words before taking the quiz.

PORTFOLIO REFLECTION: Save a drawing, sanitized screenshot, or short note describing the goal, your approach, your evidence, and one lesson learned. Expected result: A quiz that asks one question.

Investigate with pictures and choices

score starts at 2, then score = score + 3.

What is score afterward?

1. 2
2. 5
3. 23

Worked reasoning

Answer: 5. The numeric addition computes 2 + 3 and assigns the new value.

Try a changed example

Contrast a number with the text string "2".

Picture and meaning matches

Variable: Names a value

Condition: Chooses a branch

Function: Groups reusable work

Ways to respond

Choose, point, draw, demonstrate, type, speak or use your usual communication tool. A helper can operate the controls while you choose.

Counting support if useful

Use zero to ten counters or dots. Touch each once. The last count names the quantity. Rearrange the same set and check whether anything was added or removed. Choose this foundation practice only when it fits the learner goal.

My notes: _____

Hands-on lab

1. Open a Python editor and create quiz.py.
2. Store a question and answer in named variables.
3. Print the question and collect input.
4. Run the program and check what was stored.

Finished result: A quiz that asks one question.

Check your understanding

1. Which idea best describes Variables?

A. A variable stores a value under a name.

B. A function groups steps into a reusable action.

C. It has no purpose in this project.

D. An if statement chooses what happens when a condition is true or false.

2. What is the first step in this lab?

A. Print the question and collect input.

B. Open a Python editor and create quiz.py.

C. Run the program and check what was stored.

D. Store a question and answer in named variables.

3. What should you do after step two?

A. Run the program and check what was stored.

B. Open a Python editor and create quiz.py.

C. Print the question and collect input.

D. Store a question and answer in named variables.

4. Which result shows that your lab is finished?

A. A copied answer without checking it

B. A skipped lab

C. A blank page with no work

D. A quiz that asks one question.

Answer review

1. A. A variable stores a value under a name.
2. B. Start with: Open a Python editor and create quiz.py.
3. C. Step three is: Print the question and collect input.
4. D. The goal is: A quiz that asks one question.

Section 2: Logic

LEARNING GOAL: Logic. Your finished work should demonstrate: A quiz with two possible results.

EXPLAIN THE CONCEPT: An if statement chooses what happens when a condition is true or false. Compare answers in a predictable way. Think about where you might use this skill in a real or imaginary project. Describe the problem it helps you solve and how you would recognize a successful result.
ORIGINAL PROJECT LAB: 1. Add an if statement to compare the typed answer. 2. Show a success message when it matches. 3. Show a retry message when it does not. 4. Test both a correct and an incorrect answer. Finished result: A quiz with two possible results.
EXPLAIN THE MECHANISM: A condition selects a branch. The exact boundary matters: greater than is different from greater than or equal to. Indentation groups the Python statements belonging to a branch. A program can run without an error and still use the wrong rule.
WORKED CASE: A level requires at least five points. Under score >= 5, scores 4, 5, and 6 produce not yet, complete, complete. Under score > 5, the boundary score 5 incorrectly produces not yet. Testing only 4 and 6 would miss the bug.
TEST AND JUSTIFY: Write expected outcomes for all three scores before running. Keep the message outside the comparison itself so you can explain the condition and result separately. If a branch behaves unexpectedly, inspect the comparison and indentation before changing several lines.
GO DEEPER: Identify an assumption, test a safe alternative, and explain the trade-off. Keep a short before-and-after record that someone else could follow.
SAFETY CHECK: Recognize phishing and social engineering, secure accounts, understand privacy settings, and perform cyber labs only with authorization. Use pretend data, never real passwords or private information. Stop and ask a trusted adult if anything seems unsafe.
REVIEW QUESTIONS: What is the main idea? Why does the first step matter? How would you detect a mistake? What evidence proves the result? What would you improve? Answer in your own words before taking the quiz.
PORTFOLIO REFLECTION: Save a drawing, sanitized screenshot, or short note describing the goal, your approach, your evidence, and one lesson learned. Expected result: A quiz with two possible results.

Investigate with pictures and choices

The rule completes a level when score >= 5. score is 5.

Which branch runs?

1. Complete
2. Not yet
3. Neither branch can run

Worked reasoning

Answer: Complete. The greater-than-or-equal comparison includes the boundary value five.

Try a changed example

Test scores four, five and six.

Picture and meaning matches

Variable: Names a value

Condition: Chooses a branch

Function: Groups reusable work

Ways to respond

Choose, point, draw, demonstrate, type, speak or use your usual communication tool. A helper can operate the controls while you choose.

Counting support if useful

Use zero to ten counters or dots. Touch each once. The last count names the quantity. Rearrange the same set and check whether anything was added or removed. Choose this foundation practice only when it fits the learner goal.

My notes: _____

Hands-on lab

1. Add an if statement to compare the typed answer.
2. Show a success message when it matches.
3. Show a retry message when it does not.
4. Test both a correct and an incorrect answer.

Finished result: A quiz with two possible results.

Check your understanding

1. Which idea best describes Logic?

A. An if statement chooses what happens when a condition is true or false.

B. A function groups steps into a reusable action.

C. It has no purpose in this project.

D. A variable stores a value under a name.

2. What is the first step in this lab?

A. Show a retry message when it does not.

B. Add an if statement to compare the typed answer.

C. Test both a correct and an incorrect answer.

D. Show a success message when it matches.

3. What should you do after step two?

A. Test both a correct and an incorrect answer.

B. Add an if statement to compare the typed answer.

C. Show a retry message when it does not.

D. Show a success message when it matches.

4. Which result shows that your lab is finished?

A. A copied answer without checking it

B. A skipped lab

C. A blank page with no work

D. A quiz with two possible results.

Answer review

1. A. An if statement chooses what happens when a condition is true or false.

2. B. Start with: Add an if statement to compare the typed answer.

3. C. Step three is: Show a retry message when it does not.

4. D. The goal is: A quiz with two possible results.

Section 3: Functions

LEARNING GOAL: Functions. Your finished work should demonstrate: A two-question quiz organized with a function.

EXPLAIN THE CONCEPT: A function groups steps into a reusable action. Its name should describe the job it performs. Think about where you might use this skill in a real or imaginary project. Describe the problem it helps you solve and how you would recognize a successful result.

ORIGINAL PROJECT LAB: 1. Move the question logic into a function called ask_question. 2. Call the function from the main program. 3. Add a second question using the same pattern. 4. Run both questions and fix any errors. Finished result: A two-question quiz organized with a function.

EXPLAIN THE MECHANISM: A function groups a job with named inputs and a result. Defining a function does not run its body immediately; calling it supplies

arguments. Returning a value lets another part of the program use it, while printing only displays information.

WORKED CASE: Consider def add_bonus(score, bonus): return score + bonus, written with the return statement indented on its own line. Calling add_bonus(4, 2) returns 6. The names score and bonus describe roles inside the function; the caller can store the returned value under a different name.

TEST AND JUSTIFY: Predict calls with (0,2), (4,2), and (4,0). Explain which argument changes and why each answer follows. Keep the example limited to numbers; input checking is a separate design decision. Test the returned value rather than assuming that displayed text is a reusable result.

GO DEEPER: Identify an assumption, test a safe alternative, and explain the trade-off. Keep a short before-and-after record that someone else could follow.

SAFETY CHECK: Recognize phishing and social engineering, secure accounts, understand privacy settings, and perform cyber labs only with authorization. Use pretend data, never real passwords or private information. Stop and ask a trusted adult if anything seems unsafe.

REVIEW QUESTIONS: What is the main idea? Why does the first step matter? How would you detect a mistake? What evidence proves the result? What would you improve? Answer in your own words before taking the quiz.

PORTFOLIO REFLECTION: Save a drawing, sanitized screenshot, or short note describing the goal, your approach, your evidence, and one lesson learned. Expected result: A two-question quiz organized with a function.

Investigate with pictures and choices

A function returns n + 2. Its input is 4.

What value does it return?

1. 42
2. 2
3. 6

Worked reasoning

Answer: 6. Substitute the input into the function rule. Returning a value differs from only printing it.

Try a changed example

Call it with zero and a negative number.

Picture and meaning matches

Variable: Names a value

Condition: Chooses a branch

Function: Groups reusable work

Ways to respond

Choose, point, draw, demonstrate, type, speak or use your usual communication tool.

A helper can operate the controls while you choose.

Counting support if useful

Use zero to ten counters or dots. Touch each once. The last count names the quantity. Rearrange the same set and check whether anything was added or removed.

Choose this foundation practice only when it fits the learner goal.

My notes: _____

Hands-on lab

1. Move the question logic into a function called ask_question.
2. Call the function from the main program.
3. Add a second question using the same pattern.
4. Run both questions and fix any errors.

Finished result: A two-question quiz organized with a function.

Check your understanding

1. Which idea best describes Functions?

A. A function groups steps into a reusable action.

B. An if statement chooses what happens when a condition is true or false.

C. It has no purpose in this project.

D. A variable stores a value under a name.

2. What is the first step in this lab?

A. Add a second question using the same pattern.

B. Move the question logic into a function called ask_question.

C. Run both questions and fix any errors.

D. Call the function from the main program.

3. What should you do after step two?

A. Run both questions and fix any errors.

B. Move the question logic into a function called ask_question.

C. Add a second question using the same pattern.

D. Call the function from the main program.

4. Which result shows that your lab is finished?

A. A copied answer without checking it

B. A skipped lab

C. A blank page with no work

D. A two-question quiz organized with a function.

Answer review

1. A. A function groups steps into a reusable action.

2. B. Start with: Move the question logic into a function called ask_question.

3. C. Step three is: Add a second question using the same pattern.

4. D. The goal is: A two-question quiz organized with a function.

Section 4: Guided Project & Worked Examples

COURSE INVESTIGATION: Validate before changing the score

YOUR PRACTICE INPUT: A paper quiz accepts a, b, or c. Correct answer b earns one point; an empty answer should ask again without changing the score.

WHY IT WORKS: Normalize surrounding spaces and case before comparing a valid answer. Validation distinguishes an invalid input from a valid but wrong choice.

PLAN BEFORE YOU ACT: Draw or write the starting state, the question you need to answer, and the evidence you expect to see. A paper trace is acceptable; label it as a prediction unless you actually run a test.

WORK THE CASE: Write a function plan with input, allowed values, returned outcome, and a test table. Predict before executing any code.

COMPARE WITH THE EXPECTED RESULT: Only the normalized b answer increases the score. Invalid input leaves the score unchanged.

EXPLAIN YOUR REASONING: Show the connection between the supplied input, each decision, and your result. If a number, label, or observation differs, return to the first differing step rather than changing several things together.

KEEP YOUR EVIDENCE: Save the initial plan, a drawing or sanitized output, and one explanation of what the result does and does not prove. Use pointing, speech, drawing, typing, or your usual communication method.

LEARNING GOAL: Guided Project & Worked Examples. Your finished work should demonstrate: A completed guided project with documented evidence: Source code, README with run steps, Test table, Debugging note.

EXPLAIN THE CONCEPT: Review the course concepts through two applied examples. Trace variables through a quiz program and predict output before execution. Compare repeated quiz logic with a reusable function and identify the maintenance benefit.
Guided project: Build a multi-question Python quiz with functions, score tracking, input normalization, validation, and test cases. Work with synthetic or fictional information and ask an adult for help when appropriate. Think about where you might use this skill in a real or imaginary project. Describe the problem it helps you solve and how you would recognize a successful result.
ORIGINAL PROJECT LAB: 1. Read both worked examples and explain the reasoning. 2. Build a multi-question Python quiz with functions, score tracking, input normalization, validation, and test cases. 3. Record the steps you completed and the result you observed. 4. Compare your result to the goal and correct one issue.
Finished result: A completed guided project with documented evidence: Source code, README with run steps, Test table, Debugging note.
EXPLAIN THE MECHANISM: Build a fictional checkpoint scorer using a variable, a function, and a clear completion condition. First write the rule in ordinary language: two points per checkpoint, level complete at six points. Do not add extra bonuses before the basic rule is tested.
WORKED CASE: A function returns checkpoints * 2. Inputs 0, 2, and 3 yield scores 0, 4, and 6. A condition score >= 6 labels only the final case complete. A mistaken > comparison would fail exactly at three checkpoints.
TEST AND JUSTIFY: Create a test table with input, expected score, expected label, and actual result. Fix one deliberate comparison bug and rerun all cases. Keep the original four lab steps and use this table as additional evidence of reasoning.
GO DEEPER: Identify an assumption, test a safe alternative, and explain the trade-off. Keep a short before-and-after record that someone else could follow.
SAFETY CHECK: Recognize phishing and social engineering, secure accounts, understand privacy settings, and perform cyber labs only with authorization. Use pretend data, never real passwords or private information. Stop and ask a trusted adult if anything seems unsafe.
REVIEW QUESTIONS: What is the main idea? Why does the first step matter? How would you detect a mistake? What evidence proves the result? What would you improve? Answer in your own words before taking the quiz.

PORTFOLIO REFLECTION: Save a drawing, sanitized screenshot, or short note describing the goal, your approach, your evidence, and one lesson learned. Expected result: A completed guided project with documented evidence: Source code, README with run steps, Test table, Debugging note.

Investigate with pictures and choices

A quiz score rises twice after one answer.

What should you inspect?

1. Whether to remove every test
2. Whether the update runs twice
3. Only the question wording

Worked reasoning

Answer: Whether the update runs twice. Trace the event and state update to find duplicated execution.

Try a changed example

Test one correct and one incorrect answer from a known start.

Picture and meaning matches

Variable: Names a value

Condition: Chooses a branch

Function: Groups reusable work

Ways to respond

Choose, point, draw, demonstrate, type, speak or use your usual communication tool.

A helper can operate the controls while you choose.

Counting support if useful

Use zero to ten counters or dots. Touch each once. The last count names the quantity. Rearrange the same set and check whether anything was added or removed.

Choose this foundation practice only when it fits the learner goal.

My notes: _____

Hands-on lab

1. Read both worked examples and explain the reasoning.
2. Build a multi-question Python quiz with functions, score tracking, input

normalization, validation, and test cases.

3. Record the steps you completed and the result you observed.

4. Compare your result to the goal and correct one issue.

Finished result: A completed guided project with documented evidence: Source code, README with run steps, Test table, Debugging note.

Check your understanding

1. Which idea best describes Guided Project & Worked Examples?

A. A variable stores a value under a name.

B. Review the course concepts through two applied examples.

C. An if statement chooses what happens when a condition is true or false.

D. It has no purpose in this project.

2. What is the first step in this lab?

A. Build a multi-question Python quiz with functions, score tracking, input normalization, validation, and test cases.

B. Record the steps you completed and the result you observed.

C. Read both worked examples and explain the reasoning.

D. Compare your result to the goal and correct one issue.

3. What should you do after step two?

A. Build a multi-question Python quiz with functions, score tracking, input normalization, validation, and test cases.

B. Compare your result to the goal and correct one issue.

C. Read both worked examples and explain the reasoning.

D. Record the steps you completed and the result you observed.

4. Which result shows that your lab is finished?

A. A completed guided project with documented evidence: Source code, README with run steps, Test table, Debugging note.

B. A copied answer without checking it

C. A skipped lab

D. A blank page with no work

Answer review

1. B. Review the course concepts through two applied examples.

2. C. Start with: Read both worked examples and explain the reasoning.

3. D. Step three is: Record the steps you completed and the result you observed.

4. A. The goal is: A completed guided project with documented evidence: Source code, README with run steps, Test table, Debugging note.

Section 5: Independent Challenge & Mastery

INDEPENDENT TRANSFER CASE: Validate before changing the score
KEEP THE ORIGINAL BASELINE: A paper quiz accepts a, b, or c. Correct answer b earns one point; an empty answer should ask again without changing the score.
CHANGE ONE CONDITION: Test ’ B ’, an empty string, ’d’, and ’a’. Expected results are correct, retry, retry, and wrong.
YOUR TASK: Write a function plan with input, allowed values, returned outcome, and a test table. Predict before executing any code.
PREDICT, THEN CHECK: State which part of your original reasoning should remain true and which part must be recalculated or retested. Keep the original and changed results side by side.
REVIEW CRITERIA: Only the normalized b answer increases the score. Invalid input leaves the score unchanged.
MENTOR DISCUSSION: Explain why the changed condition matters. Identify a tempting wrong conclusion, show the observation that would challenge it, and choose one next test. A statement of uncertainty is better than inventing a result.
PORTFOLIO NOTE: Keep your baseline, changed input, prediction, observed or paper-traced result, explanation, and one improvement. These practice prompts do not replace the course’s existing quiz or final assessment.
LEARNING GOAL: Independent Challenge & Mastery. Your finished work should demonstrate: An independent demonstration with evidence: Source code, README with run steps, Test table, Debugging note.
EXPLAIN THE CONCEPT: Apply what you learned without copying the guided solution. Independent challenge: Add a feature such as categories or retry logic, then document one bug found and how the test exposed it. Safety and responsibility: Never paste credentials, tokens, private data, or untrusted code into a program or AI helper; run only code you understand in an approved environment. Reflect on your work: What did you build or discover? What did you test or change after feedback? What safe or responsible choice did you make? Think about where you might use this skill in a real or imaginary project. Describe the problem it helps you solve and how you would recognize a successful result.
ORIGINAL PROJECT LAB: 1. Explain the challenge and make a short plan. 2. Add a feature such as categories or retry logic, then document one bug found and how the test exposed it. 3. Test the result and record evidence of your work. 4. Review the

safety scenario and describe one improvement. Finished result: An independent demonstration with evidence: Source code, README with run steps, Test table, Debugging note.

EXPLAIN THE MECHANISM: Design a new scoring function with your own stated rule. Choose clear numeric inputs, document the returned value, and decide the completion boundary. The goal is explaining and testing a small program, not adding as many lines as possible.

WORKED CASE: For three points per token and a twelve-point goal, token counts 3, 4, and 5 produce 9, 12, and 15. Expected labels are not yet, complete, complete. A trace should show the function result before the condition is checked.

TEST AND JUSTIFY: Write tests for zero, just below the goal, exactly at it, and above it. Explain one error you corrected. If negative or nonnumeric input is outside your planned scope, state that limitation rather than claiming the program handles every possible input.

GO DEEPER: Identify an assumption, test a safe alternative, and explain the trade-off. Keep a short before-and-after record that someone else could follow.

SAFETY CHECK: Recognize phishing and social engineering, secure accounts, understand privacy settings, and perform cyber labs only with authorization. Use pretend data, never real passwords or private information. Stop and ask a trusted adult if anything seems unsafe.

REVIEW QUESTIONS: What is the main idea? Why does the first step matter? How would you detect a mistake? What evidence proves the result? What would you improve? Answer in your own words before taking the quiz.

PORTFOLIO REFLECTION: Save a drawing, sanitized screenshot, or short note describing the goal, your approach, your evidence, and one lesson learned. Expected result: An independent demonstration with evidence: Source code, README with run steps, Test table, Debugging note.

Investigate with pictures and choices

A function works for one input.

What gives stronger evidence?

1. Tests for normal, boundary and invalid inputs
2. Copying the same run output
3. Saying it looks correct

Worked reasoning

Answer: Tests for normal, boundary and invalid inputs. A broader test set checks the stated contract and its limits.

Try a changed example

State how the function should handle empty or wrong-type input before testing.

Picture and meaning matches

Variable: Names a value

Condition: Chooses a branch

Function: Groups reusable work

Ways to respond

Choose, point, draw, demonstrate, type, speak or use your usual communication tool. A helper can operate the controls while you choose.

Counting support if useful

Use zero to ten counters or dots. Touch each once. The last count names the quantity. Rearrange the same set and check whether anything was added or removed. Choose this foundation practice only when it fits the learner goal.

My notes: _____

Hands-on lab

1. Explain the challenge and make a short plan.
2. Add a feature such as categories or retry logic, then document one bug found and how the test exposed it.
3. Test the result and record evidence of your work.
4. Review the safety scenario and describe one improvement.

Finished result: An independent demonstration with evidence: Source code, README with run steps, Test table, Debugging note.

Check your understanding

1. Which idea best describes Independent Challenge & Mastery?

A. It has no purpose in this project.

B. A variable stores a value under a name.

C. Apply what you learned without copying the guided solution.

D. An if statement chooses what happens when a condition is true or false.

2. What is the first step in this lab?

A. Review the safety scenario and describe one improvement.

B. Add a feature such as categories or retry logic, then document one bug found and how the test exposed it.

C. Test the result and record evidence of your work.

D. Explain the challenge and make a short plan.

3. What should you do after step two?

A. Test the result and record evidence of your work.

B. Add a feature such as categories or retry logic, then document one bug found and how the test exposed it.

C. Review the safety scenario and describe one improvement.

D. Explain the challenge and make a short plan.

4. Which result shows that your lab is finished?

A. A blank page with no work

B. An independent demonstration with evidence: Source code, README with run steps, Test table, Debugging note.

C. A copied answer without checking it

D. A skipped lab

Answer review

1. C. Apply what you learned without copying the guided solution.

2. D. Start with: Explain the challenge and make a short plan.

3. A. Step three is: Test the result and record evidence of your work.

4. B. The goal is: An independent demonstration with evidence: Source code, README with run steps, Test table, Debugging note.

Mastery & safe-use plan

Ages 13-15

Digital safety: Recognize phishing and social engineering, secure accounts, understand privacy settings, and perform cyber labs only with authorization.

AI literacy: Evaluate bias, hallucinations, deepfakes, privacy risks, attribution, and responsible model use.

Challenge: Complete an independent debugging, defense, data, or design scenario with documented reasoning.

Portfolio: Publish a sanitized project artifact with README-style purpose, steps, screenshots or evidence, and lessons learned.

Mastery check: Explain the design, demonstrate testing, identify a security or ethics risk, and defend one improvement.

Project worksheet

My project: Build a quiz game in Python

Who will use or review it? _____

My plan: _____

Checkpoint 1: A quiz that asks one question.

Checkpoint 2: A quiz with two possible results.

Checkpoint 3: A two-question quiz organized with a function.

Checkpoint 4: A completed guided project with documented evidence: Source code, README with run steps, Test table, Debugging note.

Checkpoint 5: An independent demonstration with evidence: Source code, README with run steps, Test table, Debugging note.

One thing I tested: _____

One thing I changed after testing: _____

What I learned: _____

Trusted adult or mentor feedback: _____

Family review

Show your finished work to a trusted adult. Explain one thing you learned and one thing you would improve.

Beat IT Kids | For learning at home. Do not put real passwords or private details in practice projects.