Using Canvas Quizzes to Provide Interactive Learning Adventure GUIDES

Faculty Professional Development Workshop

Thursday September 19, 2024 ZOOM

3:15 pm - 4:15 pm

Denise Case, PhD, PE - CSIS

Overview

This presentation shows how to use Canvas Quizzes to take students on interactive learning adventures.

Each quiz becomes a GUIDE (a Guided Understanding and Interactive Discovery Event) that allows us to provide content, engage students, and enable auto-graded assessments that can be taken as often as needed for understanding.

GUIDE: Guided Understanding and Interactive Discovery Event

Learning is fun. :)

Objective: Enhance Student Learning

As instructors, we are here to:

- Facilitate **student learning**
- Assist students in building an **understanding** of course material.
- Help students **master** key concepts and retain critical knowledge.
- Support students while they develop skills in our field.

All learning is active learning - **the work must be done by the students.**

Our role is to provide the structure, tools, and guidance to support them on that journey. ^ Work, ^ Benefit

Approach: Guided Interactive Learning Adventures

Canvas provides a built-in step-by-step process for leading students through content like stepping stones: Quizzes - one question at a time.

This feature enables us to guide students through interactive journeys that break down complex content into manageable parts. We can:

- Select a clear objective.
- Teach a bit, ask a question
- Continue the teach-ask cycle to help students engage with and absorb the material.
- Optional: Conclude with a question bank to evaluate understanding.

Lots of flexibility! Each question can incorporate video, readings, web links, and more - for very dynamic and engaging learning experiences.

Fits all levels of students! Some students get it on the first pass, other students may revisit the journey multiple times. Taken as often as needed (more like our adult continuing education courses).

Benefits for Students

Enhanced, Required, Rewarded Engagement

- Interactive quizzes require active participation, keeping students engaged with the course material.

Low-Fear, On-Demand Learning

- Ample time limits reduce pressure, allowing students to focus on learning rather than racing the clock.
- Quizzes can be retaken without penalty, promoting mastery over performance.
- Can be used as a pre-test, during learning, or as a post-test for review and reinforcement.

Easy Access to Content

- Quizzes are accessible on various devices, including mobile phones, giving students flexibility to learn on the go.
- Ideal for learning in short bursts—whether at soccer practice, waiting in line, or between activities.

Benefits for Faculty

Front-Loaded Effort

- Initial setup requires effort to create the guides, but subsequent maintenance is minimal, saving time in the long run.

Scales Well

- The process is scalable, accommodating larger student numbers without increasing your workload.

Easy to Modify

- Clearly labeled guides, aligned with specific learning objectives, are easy to locate and update.
- Content and questions can be effortlessly adjusted as the course material and teaching strategies evolve.

Quick Start: The Goal & The Plan

The Goal

Fewer point-less pages (they don't show on the syllabus or deadline pages and students tend to miss them).

The Plan

Take a lecture page and copy and paste to a quiz, then ask a question at the end to verify they've accessed the learning content.

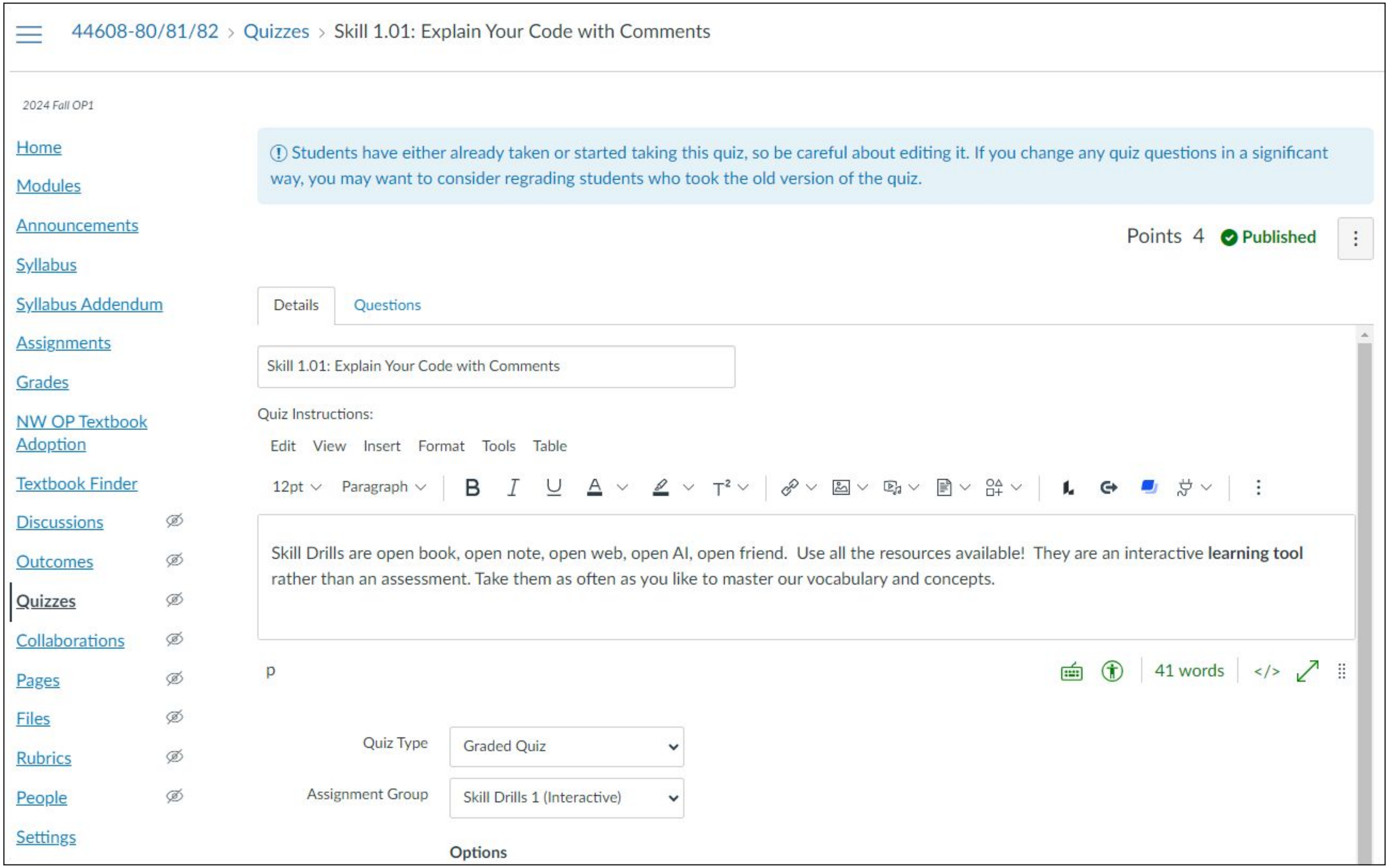
- Now the lecture content appears on the students task schedules.
- We know which students are accessing the material.
- We can verify they've seen the lecture content.

You might want to have a percentage-based gradebook - we'll add a point or two and we don't always know how many points in advance.

Quick Start: The Implementation

1. Create a new quiz.
2. Take content from a lecture page and copy it into a quiz.
3. At the end of the content, include a simple question to confirm students have engaged with the material.

Quick Start: New Quiz (Guide, Skill Drill, etc.)



44608-80/81/82 > Quizzes > Skill 1.01: Explain Your Code with Comments

2024 Fall OP1

- Home
- Modules
- Announcements
- Syllabus
- Syllabus Addendum
- Assignments
- Grades
- NW OP Textbook Adoption
- Textbook Finder
- Discussions
- Outcomes
- Quizzes
- Collaborations
- Pages
- Files
- Rubrics
- People
- Settings

Students have either already taken or started taking this quiz, so be careful about editing it. If you change any quiz questions in a significant way, you may want to consider regrading students who took the old version of the quiz.

Points 4 Published

Details | Questions

Skill 1.01: Explain Your Code with Comments

Quiz Instructions:

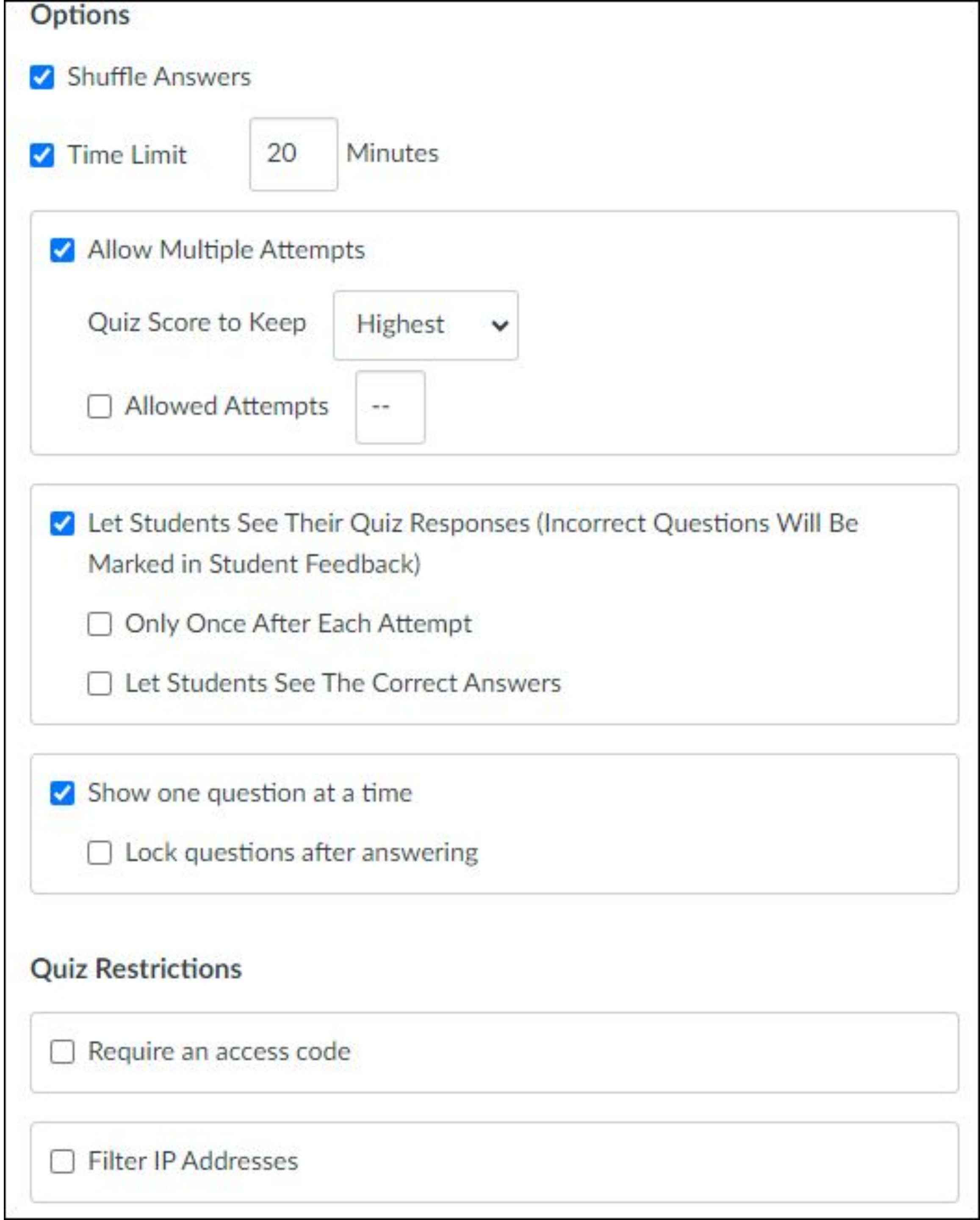
Skill Drills are open book, open note, open web, open AI, open friend. Use all the resources available! They are an interactive **learning tool** rather than an assessment. Take them as often as you like to master our vocabulary and concepts.

Quiz Type: Graded Quiz

Assignment Group: Skill Drills 1 (Interactive)

Options

Quick Start Quiz Options



Options

- [✓] Shuffle Answers
- [✓] Time Limit 20 Minutes
- [✓] Allow Multiple Attempts
  - Quiz Score to Keep: Highest
  - [ ] Allowed Attempts --
- [✓] Let Students See Their Quiz Responses (Incorrect Questions Will Be Marked in Student Feedback)
  - [ ] Only Once After Each Attempt
  - [ ] Let Students See The Correct Answers
- [✓] Show one question at a time
  - [ ] Lock questions after answering

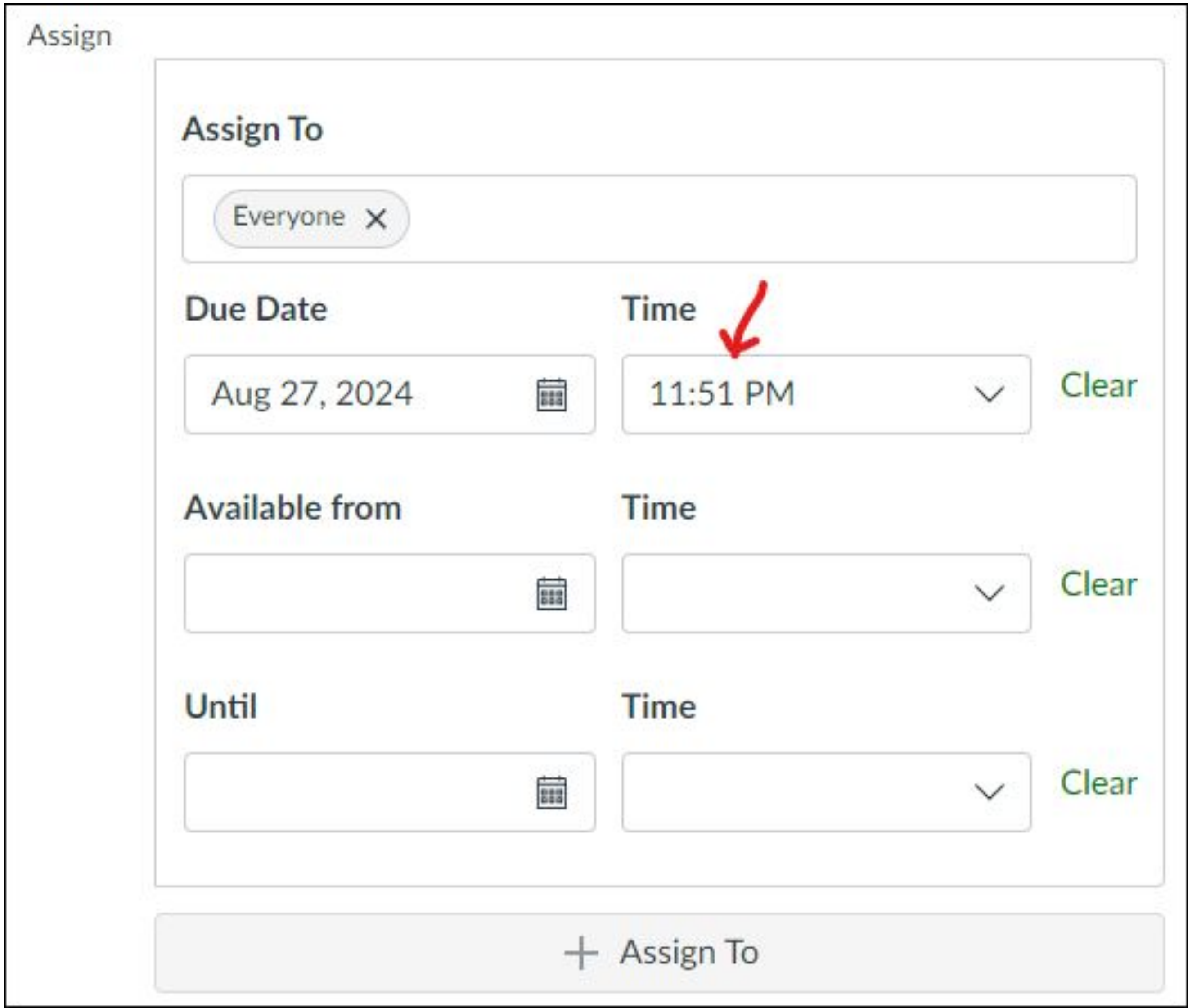
Quiz Restrictions

- [ ] Require an access code
- [ ] Filter IP Addresses

Quick Start Quiz Assign

I update the due date and time if multiple just so they appear on the calendar in the correct order.

There may be 3-5 interactive guides on a school day.



Assign

Assign To

Everyone ×

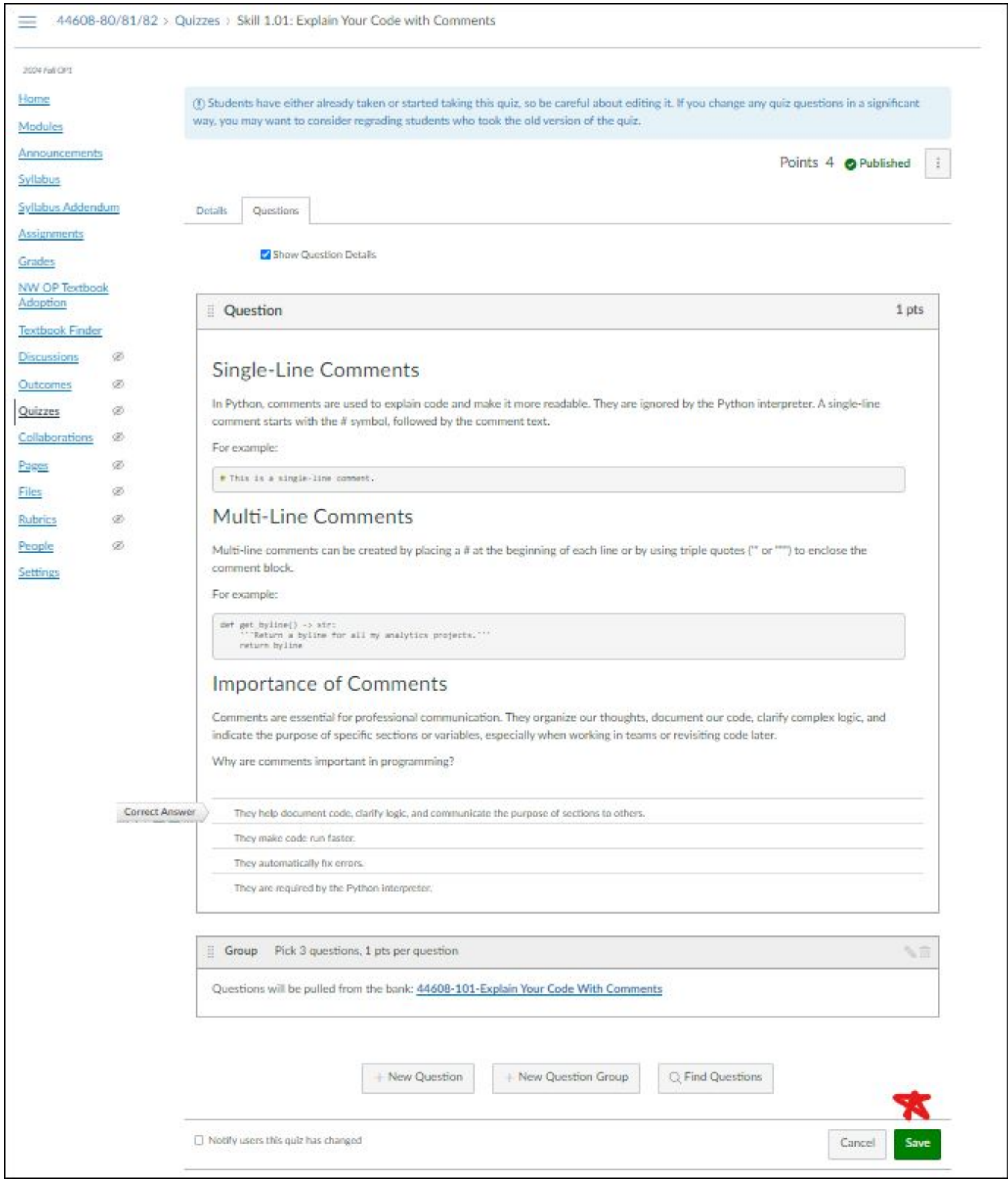
Due Date	Time	
Aug 27, 2024	11:51 PM	Clear
Available from	Time	
		Clear
Until	Time	
		Clear

Quick Start: Add a Question

Just copy and paste from the Canvas lecture page and add a question at the bottom.

1 point is fine.

Optional: This example includes drawing 3 more questions from a bank to ensure they understand the lesson content.



Students have either already taken or started taking this quiz, so be careful about editing it. If you change any quiz questions in a significant way, you may want to consider regrading students who took the old version of the quiz.

Points 4 Published

☑ Show Question Details

Question 1 pts

Single-Line Comments

In Python, comments are used to explain code and make it more readable. They are ignored by the Python interpreter. A single-line comment starts with the # symbol, followed by the comment text.

For example:

```
# This is a single-line comment.
```

Multi-Line Comments

Multi-line comments can be created by placing a # at the beginning of each line or by using triple quotes (''' or """) to enclose the comment block.

For example:

```
def get_byline() -> str:
    '''Return a byline for all my analytics projects.'''
    return byline
```

Importance of Comments

Comments are essential for professional communication. They organize our thoughts, document our code, clarify complex logic, and indicate the purpose of specific sections or variables, especially when working in teams or revisiting code later.

Why are comments important in programming?

Correct Answer: They help document code, clarify logic, and communicate the purpose of sections to others.

They make code run faster.

They automatically fix errors.

They are required by the Python interpreter.

Group Pick 3 questions, 1 pts per question

Questions will be pulled from the bank: 44608-101-Explain Your Code With Comments

☐ Notify users this quiz has changed

Implementation: Planning

- Quizzes/Guides **easily map to learning objectives** - or break content into smaller skills.
- In my fundamentals class, we moved from 3 submissions per week to **3 per day**, with positive student feedback. (This was quite a surprise, actually.) :)
- More assignments, smaller stakes fosters a sense of **mastery** and **curiosity**.
- Serves as a **Table of Contents** - easy to find what they need

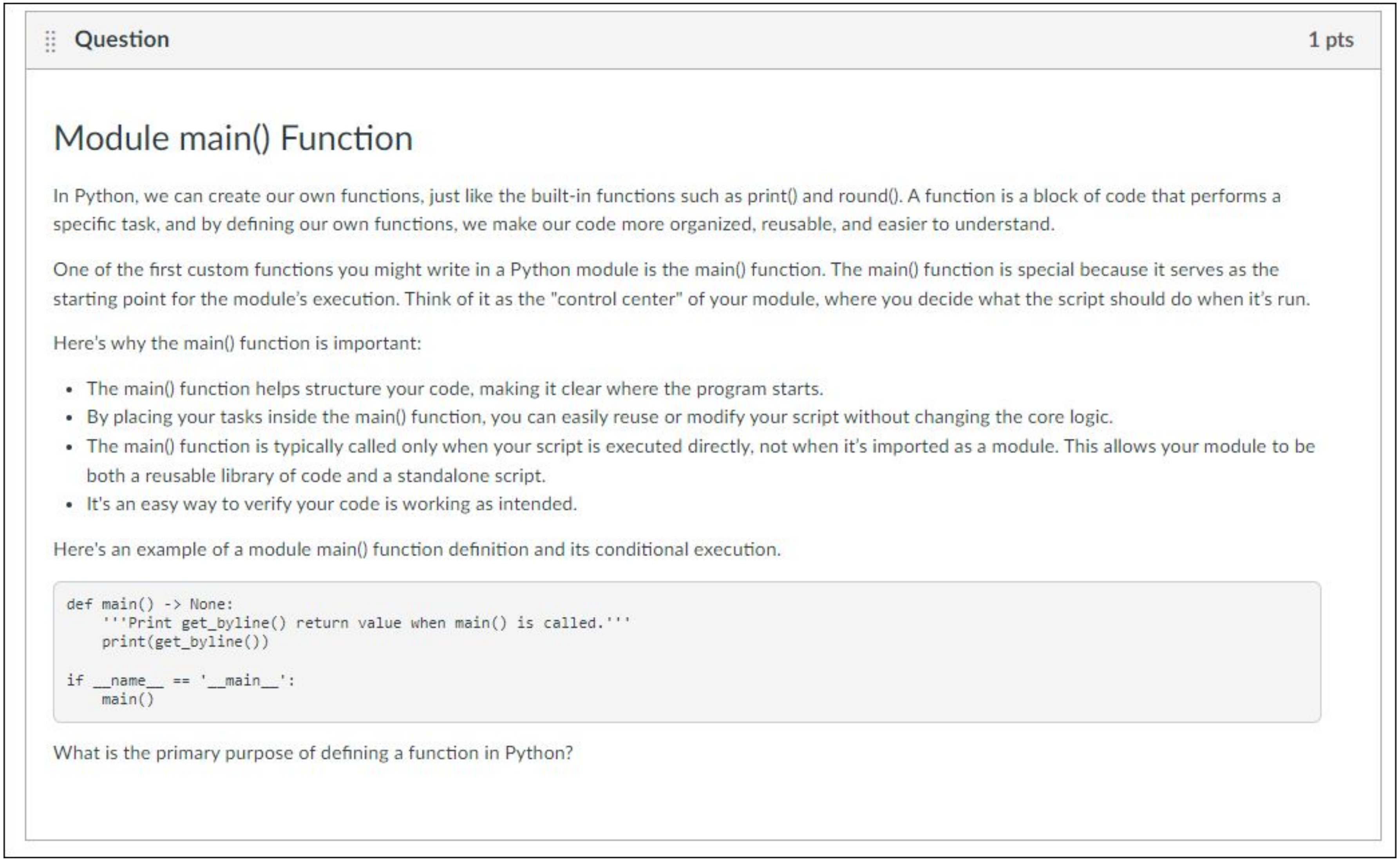
Guide 1.01: Explain Your Code with Comments

Implementation: Planning

- Think of a course module - does it have **a Canvas page** of content, a video, or an article to read **with no points or submission associated** with it?
- What skill do you want students to demonstrate after covering the learning content? Action words and tasks, key concept.
- Works best with a **percentage-based gradebook**, allowing maximum flexibility to create as many 1-point questions as needed. (We can also vary question point values as desired.)
- Might use a naming system like Module Number, Skill Number, and Skill Name for clarity and easy access.

Guide 1.01: Explain Your Code with Comments

Implementation: Creating a GUIDE Content -> Question



Question 1 pts

Module main() Function

In Python, we can create our own functions, just like the built-in functions such as print() and round(). A function is a block of code that performs a specific task, and by defining our own functions, we make our code more organized, reusable, and easier to understand.

One of the first custom functions you might write in a Python module is the main() function. The main() function is special because it serves as the starting point for the module's execution. Think of it as the "control center" of your module, where you decide what the script should do when it's run.

Here's why the main() function is important:

- The main() function helps structure your code, making it clear where the program starts.
- By placing your tasks inside the main() function, you can easily reuse or modify your script without changing the core logic.
- The main() function is typically called only when your script is executed directly, not when it's imported as a module. This allows your module to be both a reusable library of code and a standalone script.
- It's an easy way to verify your code is working as intended.

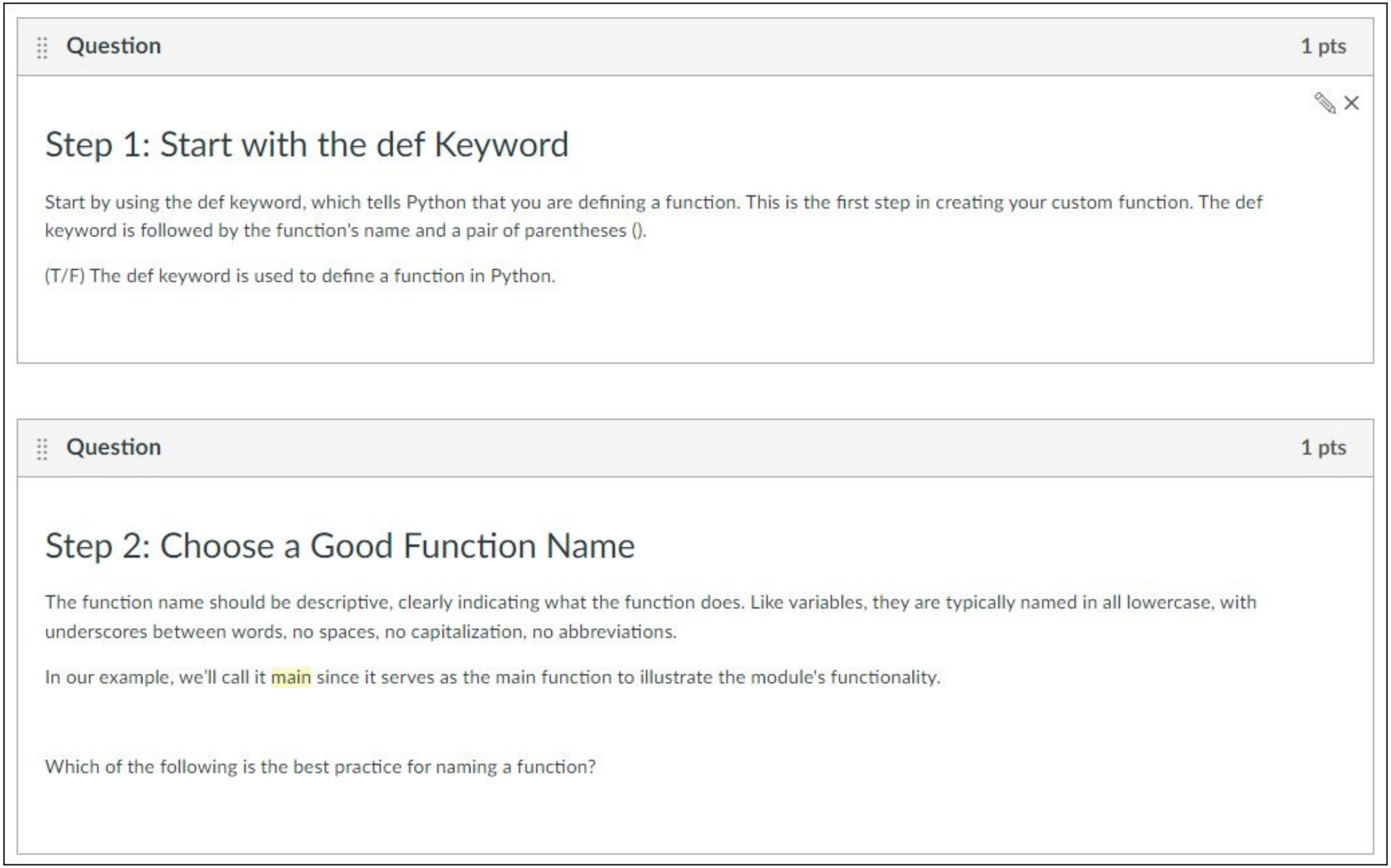
Here's an example of a module main() function definition and its conditional execution.

```
def main() -> None:
    '''Print get_byline() return value when main() is called.'''
    print(get_byline())

if __name__ == '__main__':
    main()
```

What is the primary purpose of defining a function in Python?

Implementation: Creating a GUIDE



Question 1 pts

Step 1: Start with the def Keyword

Start by using the def keyword, which tells Python that you are defining a function. This is the first step in creating your custom function. The def keyword is followed by the function's name and a pair of parentheses ().

(T/F) The def keyword is used to define a function in Python.

Question 1 pts

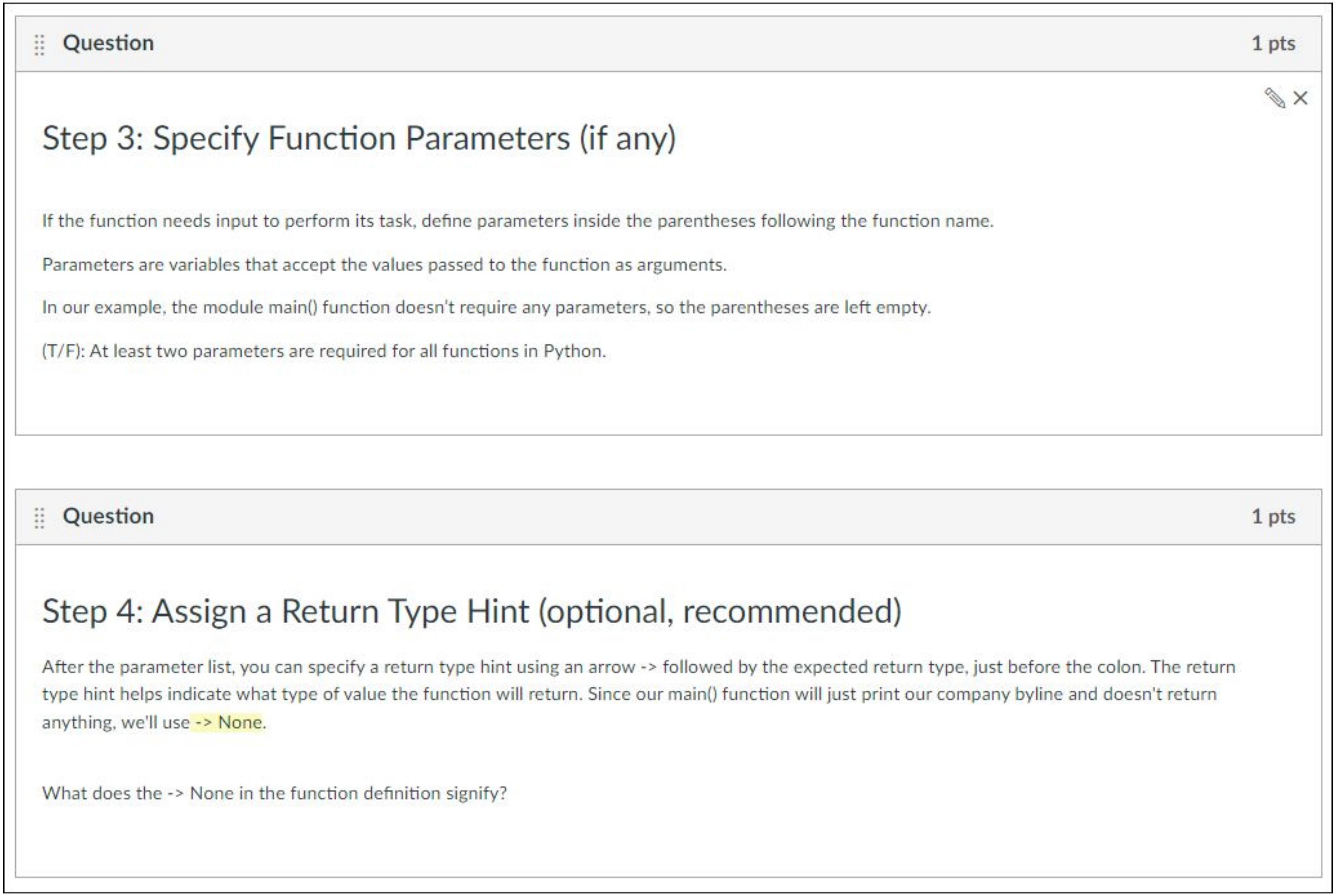
Step 2: Choose a Good Function Name

The function name should be descriptive, clearly indicating what the function does. Like variables, they are typically named in all lowercase, with underscores between words, no spaces, no capitalization, no abbreviations.

In our example, we'll call it main since it serves as the main function to illustrate the module's functionality.

Which of the following is the best practice for naming a function?

Implementation: Creating a GUIDE



Question 1 pts

Step 3: Specify Function Parameters (if any)

If the function needs input to perform its task, define parameters inside the parentheses following the function name.

Parameters are variables that accept the values passed to the function as arguments.

In our example, the module main() function doesn't require any parameters, so the parentheses are left empty.

(T/F): At least two parameters are required for all functions in Python.

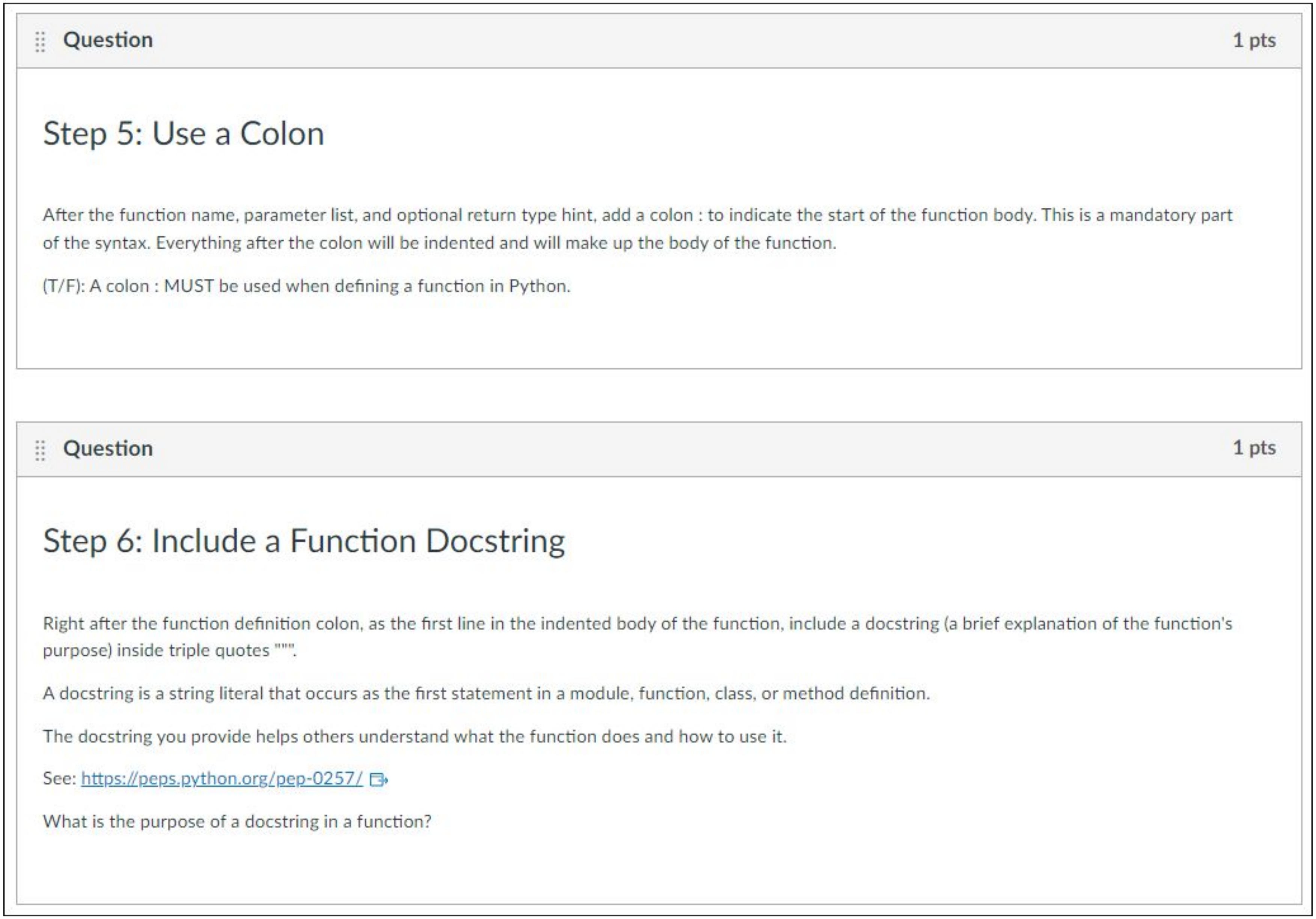
Question 1 pts

Step 4: Assign a Return Type Hint (optional, recommended)

After the parameter list, you can specify a return type hint using an arrow -> followed by the expected return type, just before the colon. The return type hint helps indicate what type of value the function will return. Since our main() function will just print our company byline and doesn't return anything, we'll use -> None.

What does the -> None in the function definition signify?

Implementation: Creating a GUIDE



Question 1 pts

Step 5: Use a Colon

After the function name, parameter list, and optional return type hint, add a colon : to indicate the start of the function body. This is a mandatory part of the syntax. Everything after the colon will be indented and will make up the body of the function.

(T/F): A colon : MUST be used when defining a function in Python.

Question 1 pts

Step 6: Include a Function Docstring

Right after the function definition colon, as the first line in the indented body of the function, include a docstring (a brief explanation of the function's purpose) inside triple quotes """.

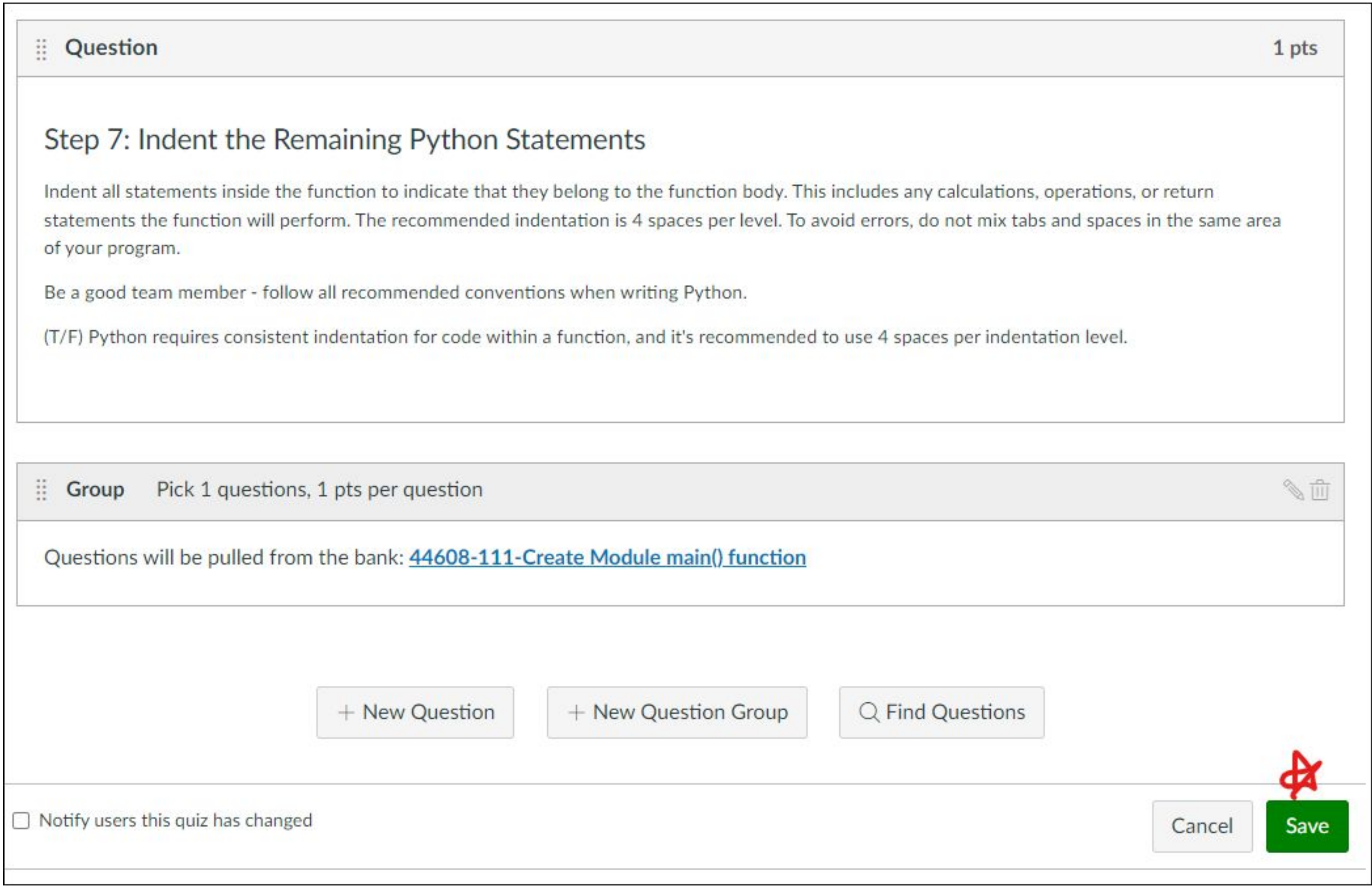
A docstring is a string literal that occurs as the first statement in a module, function, class, or method definition.

The docstring you provide helps others understand what the function does and how to use it.

See: https://peps.python.org/pep-0257/

What is the purpose of a docstring in a function?

Implementation: Creating a GUIDE



Question 1 pts

Step 7: Indent the Remaining Python Statements

Indent all statements inside the function to indicate that they belong to the function body. This includes any calculations, operations, or return statements the function will perform. The recommended indentation is 4 spaces per level. To avoid errors, do not mix tabs and spaces in the same area of your program.

Be a good team member - follow all recommended conventions when writing Python.

(T/F) Python requires consistent indentation for code within a function, and it's recommended to use 4 spaces per indentation level.

Group Pick 1 questions, 1 pts per question

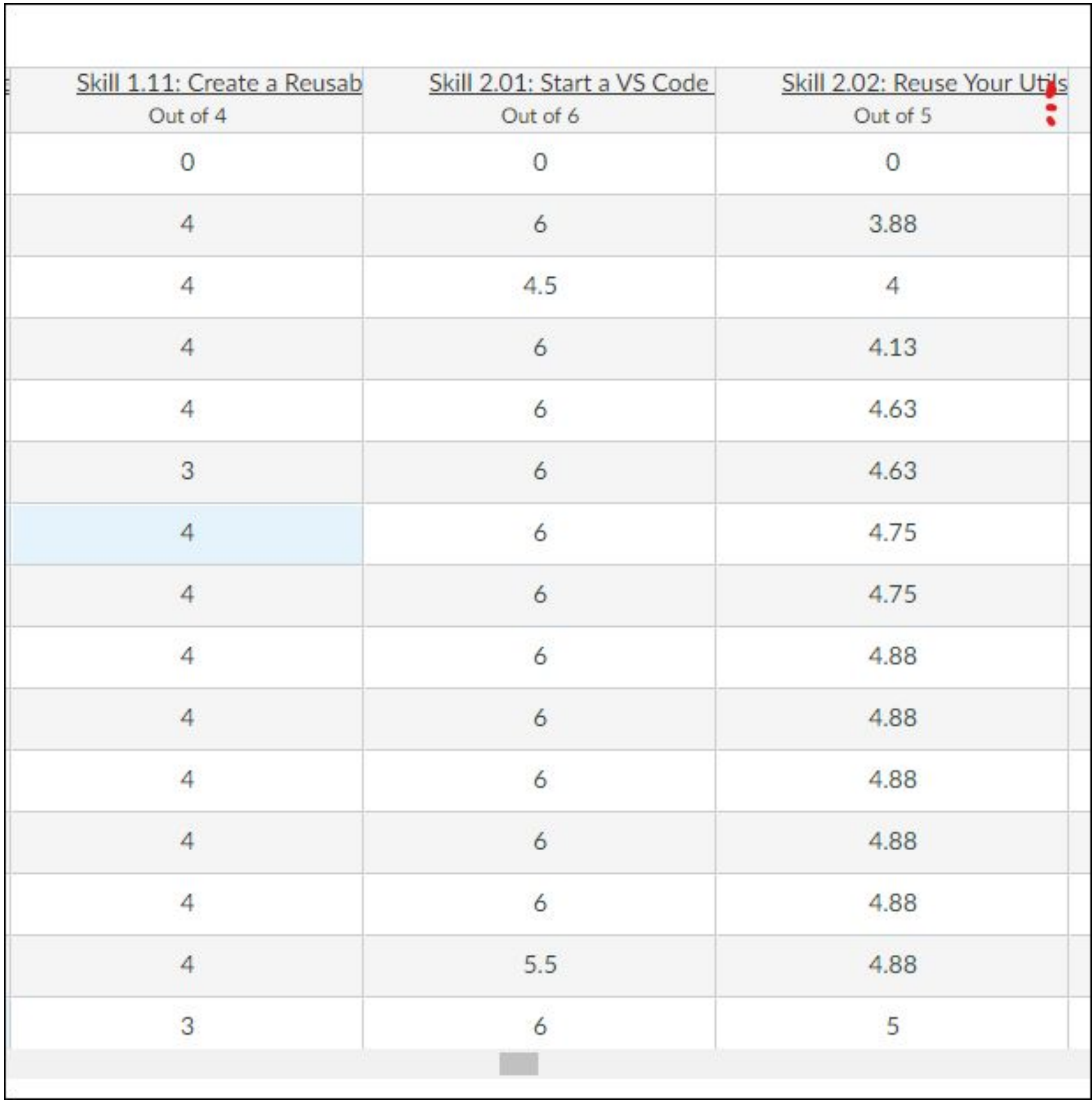
Questions will be pulled from the bank: 44608-111-Create Module main() function

Implementation: Review and Regrading

When things go well, **SpeedGrader** is still the best.

IMPORTANT: Only available via the Gradebook.

If there's an error, I just fix it. Award credit to the person who called it to my attention and the later attempts will reflect the correct question options.



Skill 1.11: Create a Reusab Out of 4	Skill 2.01: Start a VS Code Out of 6	Skill 2.02: Reuse Your Utils Out of 5
0	0	0
4	6	3.88
4	4.5	4
4	6	4.13
4	6	4.63
3	6	4.63
4	6	4.75
4	6	4.75
4	6	4.88
4	6	4.88
4	6	4.88
4	6	4.88
4	6	4.88
4	5.5	4.88
3	6	5

Tips: Students Can Listen to Pages

NaturalReader - AI Text to Speech Chrome Extension

Read aloud any text with realistic AI voices, compatible with webpages, kindle Ebooks, Google Docs, PDF, Emails, and more.

Canvas Supports Reading Pages (see your personal settings - not course settings)

This extension will read any web page.

Thanks and Q&A

Thank you for participating!

Many thanks to:

- Alyssa Bugbee, NW Online & LTC Specialist
- Dr. Gail Cullen, Assistant Director/Instructional Designer
- Dr. Darla Runyon, Director, Northwest Online | Learning & Teaching Center

Any Questions?

Denise Case

dcase@nwmissouri.edu