Teacher: Jessica Novillo Argudo, Jing Xue, Richard Parker
Unit Plan: Introduction to Python
Topic of the Lesson: Summative Assessment / Building a Quiz App
Grade and Content: 10th - 12th / Computer Science
Timing/Pace: 1 class period

Learning Objective:
Students will be able to demonstrate mastery of the previous lessons by building an intermediate version of the quiz app.

NYS Standards:

9-12.CT.2 Collect and evaluate data from multiple sources for use in a computational artifact.

9-12.CT.5 Modify a function or procedure in a program to perform its computation in a different way over the same inputs, while preserving the result of the overall program.

9-12.CT.8 Develop a program that effectively uses control structures in order to create a computer program for practical intent, personal expression, or to address a societal issue.

9-12.CT.9 Systematically test and refine programs using a range of test cases, based on anticipating common errors and user behavior

Lesson Abstract:

Students have been learning how to program in Python using variables, conditionals, expressions and loops, the building of quiz apps will demonstrate student mastery of the concepts/skills presented in lessons 1-6.

Content-specific vocabulary/ concepts:

- variables
- conditionals
- lists
- loops
- boolean expressions
- casting
- user input
- sequencing
- concatenation
- print statements

Materials/Resources:

- Internet connection
- laptops
- Replit
- Nearpod
- smart board

Formative Assessments:

- Nearpod collaborative board and polls

Summative Assessment:

Quiz App program

- quiz_app_basic
- quiz_app_intermediate
- quiz_app_advanced
- quiz_app_challenge

Warm-up/ Mini lesson:

Students will post questions to the Nearpod Collaborative Board regarding difficulties encountered in the completion of the basic quiz app.

Activity / Sequence of Lesson:

Teacher will lead the class discussion regarding posts to the collaborative board. After class discussion, students will be asked to open their Repls and create a Python file called quiz_app_intermediate.

Day 2: Students will be instructed to complete a **intermediate quiz app** with the following guidelines:

- In Replit, create a new file called quiz_app_intermediate.py
- You may copy and paste the previous quiz app into this version
- Instead of changing/setting the score by one for a correct response, develop and implement logic that will change or set the score based on a grading scale of 0-100.
- Create a variable that will track the question number and should be printed for each question when asked (Question 1: How are you? Question 2: What time is it?...)
- Develop the program so that if the user gets the question incorrect the program will inform the user of the correct answer.
- The program should report the score after all the questions have been answered

Summary / Next Steps / Exit Slip:

- Students will submit quiz_app_basic and the teacher will use the rubric checklist to either give or not give an entrance/exit ticket to the next version of the quiz app. Students will have to receive checks for all 9 rubric categories to move onto the next quiz app.
- The goal is for students to complete as many, if not all of the quiz apps and challenges.

Quiz App Scoring and Rubric Checklist

No submission = 0

Approaching Quiz App (not complete) = 40-60

Quiz App Basic = 65 - 75

Quiz App Intermediate = 75 - 85

Quiz App Advanced = 85 - 95

Quiz App Challenge = 100

Students will receive checks for each skill category satisfactorily demonstrated

RUBRIC CHECKLIST	**Quiz Apps >>>>>**	**BASIC**	**INTERMEDIATE**	**ADVANCED**	**CHALLENGES 1-2**
variables	● Student is able to create and implement variables in the program				
functions	● Student is able to create and define a function				
conditionals	● Student is able to create an if /if else statement with proper syntax, indentation and case to be tested using operators and/or boolean expressions				
loops	● Student is able to create a for/ while loop with proper syntax, indentation and for while loop - a proper exit condition				

lists	● Student is able to make a list and fill it with values such as another list(sublist) for this program to function as intended.				
concatenation	● Student is able to demonstrate concatenation by casting variables and using +				
user input	● Student is able to take input from the user and implement it properly in the program for it to function as intended.				
sequencing	● Student is able to sequence the program in a way that allows the program to function as intended.				
print statements	● Student is able to execute print statements for the program to function as intended.				