LESSON PLAN – Computer Programming

Title: Hobbits vs. Nazgul – Build Arrays

Essential Questions	Can we model life-like behavior with Python? How can we apply our expanded understanding of data types and storage?

Learning Objectives:	Students will be able to create functions to: ● Build a 2D array ● Print the 2D array in a visually clear format ● Fill the 2D array with empty "cells"

Standards (CSDF)	
9-12.CT.4	Implement a program using a combination of student-defined and third-party functions to organize the computation.
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.7	Design or remix a program that utilizes a data structure to maintain changes to related pieces of data.
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.
9-12.CT.10	Collaboratively design and develop a program or computation artifact for a specific audience and create documentation outlining implementation features to inform collaborators and users.

Teaching Materials:
Student handouts (attached)
Student computers

Procedure:

I. Finalize group administration plan & feedback on student proposal as check of understanding
II. Division of labor
    A. Students will decide who will work on what portion of the code
    B. Each student should plan to complete at least one of the following functions, but collaboration is encouraged
        1. Build a 2D array
        2. Print the 2D array in a visually clear format
        3. Fill the 2D array with empty "cells" (represented as dictionaries)
III. Student work time

Cellular Automata – Hobbits vs. Nazgul

The goal for this programming project is to create a simple 2D predator–prey simulation implemented as a cellular automata. In this simulation, the prey are Hobbits, and the predators are Nazgul. These beings live in a fictional 2D world composed of a 30 X 30 grid of cells. Only one being may occupy a cell at a time and the Hobbits are constantly on the run from the voracious Nazgul who like nothing better than to feed on the plump, un-muscled, succulent Hobbit flesh. Around the edge of this world there is an ominous city wall. Due to the casting of some ancient black magic the Nazgul are not capable of crossing this wall. They are essentially repelled and must change direction. On the other hand, the Hobbits, who appeased the witches with hand crafted ales and excellent long bottom leaf, are capable of ascending these walls and tele-porting to the other side of town. When a Hobbit approaches a particular edge it wraps around to the opposite edge and appears there, similar to the mythological beast Pac Man. As this world is an automata, it does not require any human interaction. It is simply seeded with life and set on its inevitable path. As the seeding is random there are infinite scenarios that can play out. Time in this world is simulated in discrete steps which can be implemented as a loop iteration. A single iteration over the entire world is equivalent to a 24-hour Earth day. Each being performs some action every time step.

Task 1: Create the Board
The city consists of a 30x30 grid. To model this, we will create a 30x30 2D list. Define a function called `create_board` that returns a 30x30 2D list of cells, all of which should contain an empty cell structure. Note: Using list multiplication here to make 30 rows DOES NOT WORK.

Task 2: Print the Board
The default formatting for printing a 2D list is not conducive to visualizing a grid. Create a function called `print_board` that will take a board as an argument, then print the contents of the board as a grid. Be sure that the individual cells in each row are separated by something visible so that empty cells can still be differentiated from each other – think of vertical borders in a table. Horizontal borders are optional – if you have time, feel free to try them out to see if you prefer it with or without.

Day 2 Checklist: Constructing the arrays

You must have the following tasks accomplished before the end of class today.

Accomplished	Task
●	Finalize your group administration plan. Make sure all team members are able to agree with this. *You should only have to come back to this unless it is only absolutely necessary.*
●	Division of labor and assignment of tasks. ● Who works on what? ● Again, who should take on what responsibilities unless that/those person(s) are out for the day? ● What needs to happen if a team member is stuck or becomes frustrated?
●	Each students should expect to complete at least of one of the following below: ● Construct a two dimensional array ● Print the two dimensional array ● Fill the two dimensional array with empty

	cells (dictionaries should be used here) *Collaboration is encouraged if the team feels it necessary to be successful.*