Lesson 4.06: Tic Tac Toe ## Learning Objectives Students will be able to... * Use project planning skills to complete a larger project * Utilize loops, lists, and nested loops/lists to create a Tic-Tac-Toe game ## Materials/Preparation * [Project Spec - Tic-Tac-Toe] ([printable project Spec]) ([editable project spec]) * [Alternate Project Spec - 2 Player Tic-Tac-Toe] ([printable alternate project Spec]) ([editable alternate project spec]) * [Tic Tac Toe Starter Code](https://github.com/TEALSK12/2nd-semester-introduction-to-computer-science/raw/master/units/4_unit/06_lesson/tictactoe_starter_code.py) * Solution (access protected resources by clicking on "Additional Curriculum Materials" on the [TEALS Dashboard]) * Read through the handout, lesson, and project so that you are familiar with the requirements and can assist students * Try creating your own variation on the Tic-Tac-Toe game so you are familiar with the potential challenges and bugs your students will hit * Review [4 Steps to Solve Any CS Problem] * [Editable Grading Rubric] (https://github.com/TEALSK12/2nd-semester-introduction-to-computer-science/raw/master/units/4_unit/06_lesson/rubric.docx) ## Pacing Guide ### Day 1 | **Duration** | **Description** | | ---------- | ---------- | | 5 Minutes | Do Now | | 10 Minutes | Project Overview | | 30 Minutes | Planning/Design | | 10 Minutes | Debrief | ### Days 2-9 |**Duration** | **Description** | |---|---| | 5 Minutes | Day Plan | | 10 Minutes | Review | | 35 Minutes | Project Work| | 5 Minutes | Debrief | ## Instructor's Note ### 1. Do Now * Hand out the project spec and have students start reading through it. ### 2. Project Overview * Demo the Tic-Tac-Toe game. * Go over all of the game rules and program requirements. ### 3. Planning/Design #### Have students do the following in their notebook * Write down the 4 Steps to Solve Any CS Problem from memory. * Draw out game play and consider the overall design. * How will they represent the board? * How will they have users input their spots? * Create function names for each user interaction. * Figure out which variables are needed. #### Have students plan out their next 7 days (suggested plan below) 1. Set up the game board, basic game loop asking players for input, and dummy functions for each player's turn. 2. Create variables necessary to run the game, start implementing basic functions. 3. Focus on game play: 2 players should be able to play a game against each other. 4. Create functions for checking if the game is over. Create a horizontal checker, vertical checker. 5. Create the diagonal checkers. 6. Connect the functions together and test functionality. 7. Use multiple tests that game over check is correct, finalize project. ### 4. Debrief * Take time to go over questions and confusion relating to project requirements. * Make sure to look over individual student project plans and check that they have outlined the project in a way that makes sense. ### 5. Day Plan * At the start of Days 2-9, have each student refer to their original project plan, write down what they hope to accomplish that day, and assess their schedule to see if they are on track. ### 6. Review * Go over any parts of the program that the majority of the class is struggling with. * Provide scaffolding and tips to students that are not on track for completion. ## Accommodation/Differentiation * Students can create a variable sized board. * The checkers can actually be done using one function, taking in the start x and y and the movement of the x and y. * The planning phase of this project will be essential, especially for students who you think may struggle with this project. * Provide more guidance and scaffolding to those students that need it. ## Grading [Editable Grading Rubric] (https://github.com/TEALSK12/2nd-semester-introduction-to-computer-science/raw/master/units/4_unit/06_lesson/rubric.docx) ### Objective Scoring Breakdown | Points | Percentage| Objective | Lesson | | :---: | :---: | --- | --- | | 6 | 19 | Student uses for loops | 4.01 | | 3 | 10 | Students can c use the `range` and `len` functions |4.02 | | 3 | 10 | Students can use nested for loops |4.03 | | 3 | 10 | Student can use nested lists |4.04 | | 5 | 16| Student can decompose a problem to create a program from a brief|| | 5 | 16 | Student uses naming/ syntax conventions and comments to increase readability | | 25 | |**Total Points**|| | ## Forum discussion [Lesson 4.06: Project 4 (TEALS Discourse Account Required)](https://forums.tealsk12.org/c/unit-4-looping/lesson-4-06-tic-tac-toe) [Project Spec - Tic-Tac-Toe]: project.md.html [Alternate Project Spec - 2 Player Tic-Tac-Toe]: alternate_project.md.html [TEALS Dashboard]:http:/www.tealsk12.org/dashboard [4 Steps to Solve Any CS Problem]:https://github.com/TEALS-IntroCS/2nd-semester-introduction-to-computer-science-principles/raw/master/units/4%20Steps%20to%20Solve%20Any%20CS%20Problem.pdf [printable project Spec]: https://github.com/TEALSK12/2nd-semester-introduction-to-computer-science/raw/master/units/4_unit/06_lesson/project.pdf [editable project spec]: https://github.com/TEALSK12/2nd-semester-introduction-to-computer-science/raw/master/units/4_unit/06_lesson/project.docx [printable alternate project Spec]: https://github.com/TEALSK12/2nd-semester-introduction-to-computer-science/raw/master/units/4_unit/06_lesson/alternate_project.pdf [editable alternate project spec]: https://github.com/TEALSK12/2nd-semester-introduction-to-computer-science/raw/master/units/4_unit/06_lesson/alternate_project.docx