Reference Story No.	Story Pool	Story Source	Summary: 25-50 words
1	1	EECS 330	Implement the basic structure of a binary heap using an array. Define helper functions to calculate the parent and child indices for any given node. This establishes the array representation of the heap's tree-like structure.
2	1	EECS 330	Create a function to insert a new element into the heap. The new element is added to the end of the array, and a "heapify-up" (or "percolate-up") operation is performed to swap it with its parent until the heap property is restored.
3	1	EECS 168	Create a 2D array or list of lists to represent the grid. Initialize the grid with starting values, including marking a starting point and potentially obstacles or a target location. This sets up the environment for the recursive exploration.
4	1	EECS 348	Create functions to perform the four basic arithmetic operations: addition, subtraction, multiplication, and division. Each function will take two numbers as input and return the result of the corresponding calculation.
5	1	EECS 468	Implement the core game logic, including functions to handle player moves, update the board, and check for a win condition. This involves determining if three of the same player's marks are in a row, column, or diagonal.
6	2	EECS 330	Develop a function to extract the root element (min or max). This involves replacing the root with the last element in the array, reducing the heap's size, and then performing a "heapify-down" (or "percolate-down") operation to maintain the heap property.
7	2	EECS 168	Develop a recursive function that explores the grid from a given cell. The function should check for valid moves (within grid boundaries, not an obstacle), mark the current cell as visited, and then recursively call itself for neighboring cells.
8	2	EECS 268	Implement the three fundamental depth-first traversal algorithms: in-order (left, root, right), pre-order (root, left, right), and post-order (left, right, root). These will be recursive functions that visit each node in the specified order.
9	2	EECS 348	Based on the operator selected by the user, call the appropriate arithmetic function and display the result to the user. Include error handling for invalid input, such as division by zero.
10	3	EECS 268	Define a node structure for a binary tree that includes data and pointers to left and right children. Implement functions to create new nodes and build a basic binary tree structure by inserting these nodes.
11	3	EECS 468	Develop a simple command-line interface to allow two players to interact with the game. This includes displaying the board, prompting for user input for moves, and announcing the winner or a draw.
12	3	Personal Project	**Personal project inspired by EECS 348/468: Write an HTML code-based website that applied the skills learned from class and deploy using GitHub webpages (similar to people.eecs server). The website should be interactive and functional and user will be able to see the site without error.**
13	5	EECS 168	Define the base cases for the recursion, such as reaching the target destination or being unable to make any more valid moves. The function should return a value indicating whether a path was found, and potentially store the successful path.
14	5	EECS 348	Implement the user interface to get two numbers and an operator (+, -, *, /) as input from the user. This will involve prompting the user for input and storing their selections in variables.
15	5	EECS 468	Enhance the user interface to validate player input. The system should reject moves on already occupied squares and handle non-valid inputs gracefully, prompting the user to enter a valid move without crashing the program.
16	5	EECS 468	Implement a simple computer opponent that can play against a human. The AI's logic will select a random, unoccupied square for its turn, providing a basic, playable single-player experience and testing the game's core logic.
17	5	Personal Project	Created a markdown reader that you run from the terminal. This runs similar to that of jupyter notebooks and let's you read, edit, and render markdown files by simply running a command from the terminal.
18	5	Personal Project	Create an online platform to conduct speed math tests and compile user analytics to facilitate gradual improvement
19	5	Hackathon	Create an application to smoothen a doctor's visit. The application must capture a doctor's visit and use the recording to generate the correct SOAP notes. Then, use the SOAP notes to determine the insurance billing codes that are most likely to be approved
20	8	Hackathon	Create an application to help people with limited motor ability engage in 3D modelling. There is a vision engine that passes data to a 3D game engine in which users can place 3D objects
21	8	EECS 468	Create data types to represent the Tic Tac Toe board, the players (X and O), and the state of each cell (empty, occupied by X, or occupied by O). This will form the foundational structure for the game's logic and state management.
22	8	EECS 510	EECS 510 final project: create a real-life scenario using a non-trivial language with an application of online shopping. The program will involve enter, add, delete and checkout mechanisms. User will enter a string and program will determine if the stringm is acceptable or invalid.
23	8	EECS 678	EECS 678 (Fall) quash projects: Created the program with C language and the UNIX system calls. The user will be able to test basic commands similar to Linux command line such as file directories and replicating files (from scratch)
24	13	EECS 268	Write a main function to build a sample binary tree and then call each of the traversal functions. The output should clearly demonstrate the different ordering of node visits for each traversal method, allowing for verification of the algorithms' correctness.
25	13	EECS 678	Implement a shell in C. This shell runs whenever you run 'quash.' It has all the functionality of a simple shell being able to read, write, delete programs, use linux terminal commands, and much more.
26	13	CSP	Creating a CRUD app to track subscriptions for the KU audio reader. This meant to be able to create, read, update, and delete database entries of magazines.
27	13	EECS 665	Create a compiler for a custom language using multiple technologies. Most of the compiler is in C++ but there are assembly instructions at some point of time