Alejandro Perez

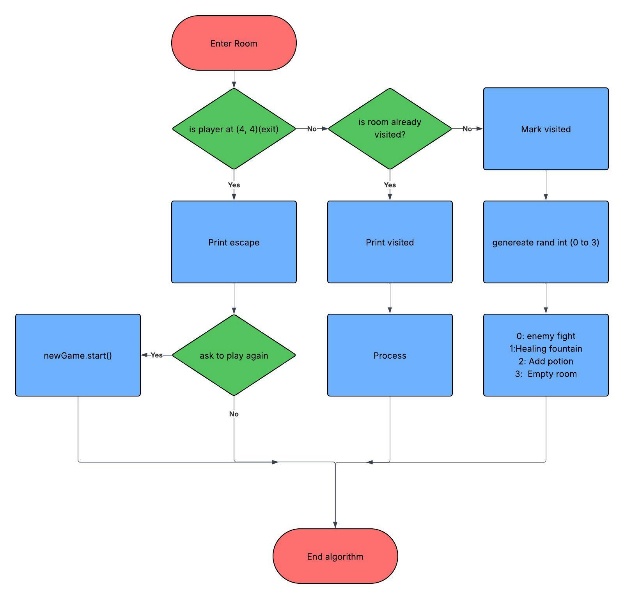
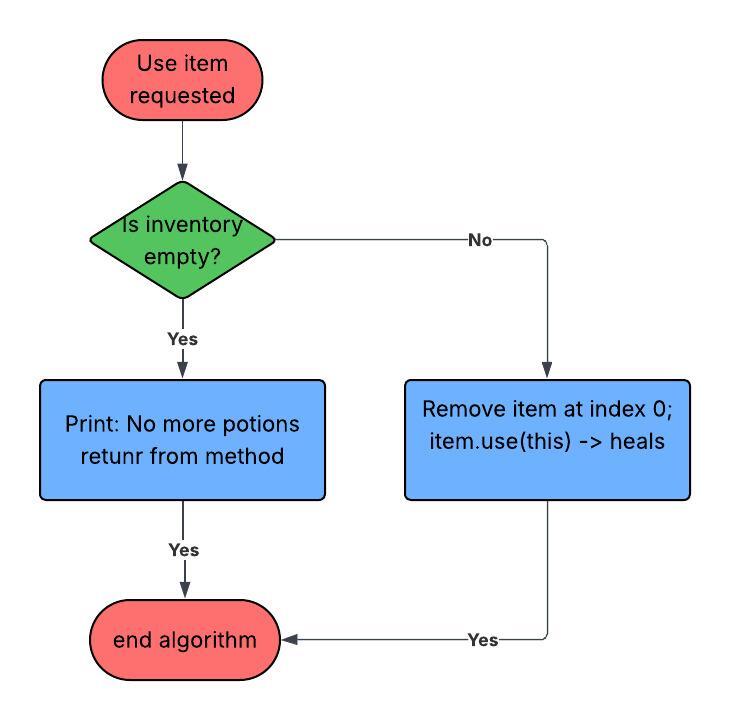
The dungeon crawler is a terminal based game where the player attempts to escape from the dungeon. Each room in the dungeon has a random event where the play may encounter an enemy, a healing fountain, a healing potion, or just an empty room. once the player enters the dungeon their goal is to find the exit at a specific coordinate all while encountering multiple enemies. The game is purely for entertainment allowing the player to move around freely while stumbling into enemies and finding healing items to help them along their journey. It is a useful Java project because it allowed me to practice the concepts we have been learning in class like algorithms, data structures, OOD, and user input handling. For the algorithms I chose to use them for the combat loop, the room event generator, and the item usage and inventory management. The combat loop manages the turn-based combat between the player and the enemy. when the player attacks or uses a potion the enemy attacks back if the player is still alive. This continues until one side health goes down to zero. For the big O time of the loop the best case is O(1) where one attack ends the battle, and the worst case is O(n) where n is the number of turns until one side dies. The room event generator decides what event happens in that room whether it is an enemy, a potion, a fountain, or nothing at all. The big O for this one is O(1). Lastly for the item usage and inventory management the algorithm checks if the player has items before using one. if there is a potion then it removes the item and applies the effect. if the inventory is empty a message is displayed warning the player. the big O time complexity for this algorithm is O(1). In the data structures portion of the project, I chose to use a 2D array and an ArrayList. the 2D array list was used to represent the dungeon grid. Each room object is stored in a 2D array based on its x and y coordinates. this allows easy lookup of the room the player is in and ensures consistent map structure. The ArrayList was used for the player’s inventory. I used the list to store healing potions, and it allowed easy adding and removing of items. the ArrayList was chosen over a regular array because the number of items can change during gameplay. Throughout making this game I discovered an opportunity where I could add some variety in healing the player where rather than finding potions, they could just find a healing fountain. This kept their inventory lighter. A problem I did encounter was the player could move past the border of the dungeon. This created the problem where the player would skip rooms entirely and rooms would be replaced. I fixed the problem by updating the Player.move method to check that the position remains between 0 and 4 and cancel movement letting the player know that the direction was invalid. In the next version I would consider adding diverse types of enemies dealing different amounts of damage and having varying max health.

A screenshot of a computer program

AI-generated content may be incorrect.A screen shot of a computer code

AI-generated content may be incorrect.

A screen shot of a computer code

AI-generated content may be incorrect.A diagram of a flowchart

AI-generated content may be incorrect.