Name: Talha Asif

Reg No: Sp23-bai-042

**Text-Based Adventure Game**

**Overview**

This is a Python-based text adventure game where the player explores various rooms, solves puzzles, manages an inventory, and interacts with the game world. The player can move between rooms, take or drop items, solve puzzles to unlock areas, and save or load game progress.

**Features**

* **Room Navigation:** Players can move between different rooms by entering directions.
* **Puzzle Solving:** Some rooms have puzzles (like locked doors or riddles) that need to be solved before progressing.
* **Inventory Management:** Players can pick up and drop items found in rooms.
* **Save & Load:** Players can save their game progress to a file and load it later.

**Game Structure**

**Rooms and Game Map**

Each room has a description, exits (directions leading to other rooms), and possibly items or puzzles. The game map is defined in a game\_map object imported from an external module (map.py).

**Player Object**

The player has the following attributes:

* **Name:** Default is 'Player'.
* **Inventory:** A list of items the player has picked up.
* **Score:** The player's score, which increases when they solve puzzles and progress.
* **Dropped Inventory:** A list of items the player has dropped.

**How to Play**

1. **Movement:**
   * Players can move between rooms by typing a command in the format go [direction] (e.g., go north).
   * If the room in the chosen direction has a puzzle, the player must solve it to proceed.
2. **Inventory Management:**
   * To manage inventory, after moving between rooms, the game will prompt the player to take or drop items. Players can also explicitly manage inventory by using the inventory command.
   * **Take Items:** Players can take items from the room and add them to their inventory.
   * **Drop Items:** Players can drop items from their inventory back into the room.
3. **Puzzles:**
   * Some rooms have puzzles (like locked doors or riddles) that the player must solve to enter.
   * The puzzles are handled through the puzzel.py module.
4. **Save & Load:**
   * Players can save their progress using the save command, which stores the current game state (room, inventory, score) to a JSON file.
   * Players can load a saved game using the load command.
5. **Commands:**
   * inventory: View and manage the player's inventory.
   * direction: Move between rooms by entering a direction.
   * examine: Examine the surroundings or solve a puzzle.
   * save: Save the game progress to a file.
   * load: Load a previously saved game.
   * exit: Exit the game.

**Functions**

**Main Functions**

* **direction()**
  + Handles player movement between rooms. Prompts the player to enter a direction or interact with the room (look around, examine, or solve puzzles).
* **inventory\_player()**
  + Manages inventory operations (take or drop items). Allows the player to pick up items from the room or drop items back into it.
* **show\_inventory()**
  + Displays the player's current inventory.
* **save\_game(filename='savefile.json')**
  + Saves the current game state (room, inventory, and score) to a JSON file.
* **load\_game(filename='savefile.json')**
  + Loads a previously saved game state from a JSON file.

**Puzzle Integration**

The puzzel.py module handles the logic for solving puzzles. The player must solve puzzles to unlock rooms and progress in the game.

**Sample Game Flow or sample output 1:**

**Enter a command ("inventory", "exit","direction","examine","save","load"): direction**

**Description of Entrance: You are at the entrance of a dark and mysterious castle.**

**enter the direction (e.g:go north): go north**

**solve the puzzel then move**

**The door to the Hallway is locked. Solve the Puzzle.**

**Enter the answer (type "exit" to give up): stone**

**Incorrect answer! Try again.**

**Your score is now: -1**

**Enter the answer (type "exit" to give up): key**

**Correct! You solved the locked door puzzle.**

**You have moved to: Hallway**

**Description of Hallway: A long hallway stretches before you, dimly lit by flickering torches.**

**Player score is 9**

**Do you want to manage your inventory? (yes/no): no**

**Sample gameflow or output 2:**

**Enter a command ("inventory", "exit","direction","examine","save","load"): inventory**

**Your inventory is empty.**

**Enter a command ("inventory", "exit","direction","examine","save","load"): direction**

**enter the direction (e.g: go east): go east**

**solve the puzzel then move**

**The door to the Secret Room is locked. Solve the Puzzle.**

**Enter the answer (type "exit" to give up): echo**

**Correct! You solved the riddle puzzle.**

**You have moved to: Secret Room**

**Description of Secret Room: A hidden chamber filled with treasure and ancient artifacts.**

**Player score is 19**

**Sample output3:**

**Do you want to manage your inventory? (yes/no): yes**

**Do you want to "take" or "drop" an item? take**

**Items in the room:**

**1. Golden Chalice**

**2. Ancient Scroll**

**Enter the number of the item you want to take: 1**

**You have taken: Golden Chalice**

**Sample output 4:**

**Enter a command ("inventory", "exit","direction","examine","save","load"): inventory**

**Your inventory:**

**1. Golden Chalice**

**Do you want to "take" or "drop" an item? drop**

**Our inventory:**

**1. Golden Chalice**

**Enter the number of the item you want to drop: 1**

**You have dropped: Golden Chalice**

**Sample output 5:**

**Enter a command ("inventory", "exit","direction","examine","save","load"): save**

**Game saved successfully to savefile.json.**

**Enter a command ("inventory", "exit","direction","examine","save","load"): load**

**Game loaded successfully from savefile.json.**

**You are currently in Secret Room.**

**Sample output 6:**

**Enter a command ("inventory", "exit","direction","examine","save","load"): examine**

**What would you like to examine?**

**Enter your choice: table**

**The table is covered with dust and a key is hidden beneath it.**

**Running the Game**

To start the game, run the main Python script that contains the game loop. You will be prompted to enter commands to navigate, interact, manage inventory, save, and load the game.

**Modules**

* map.py: Contains the game map with room descriptions, exits, items, and puzzles.
* puzzel.py: Handles puzzle-solving logic.
* game.py: Main game logic and loop.

**Notes**

* The game\_map is defined properly in map.py with all necessary room details, including directions, items, and puzzles.
* The puzzel.py module should contain the logic for different types of puzzles such as solve\_locked() and solve\_riddle() and examine () functions.