# Import necessary libraries

import json

import os

# Define the main function

def main():

# Initialize variables

keepGoing = True

game = None

# Main loop to manage user interaction

while keepGoing:

# Get user choice from menu

userChoice = getMenuChoice()

# Process user choice

if userChoice == 0:

# Exit the program

keepGoing = False

print("Thank you for playing!")

elif userChoice == 1:

# Load default game

print("Loading default game")

game = getDefaultGame()

elif userChoice == 2:

# Load a game file

game = loadGame()

elif userChoice == 3:

# Save the current game

saveGame(game)

elif userChoice == 4:

# Edit or add a node

if game is not None:

editNode(game)

else:

print("Please load a game first.")

elif userChoice == 5:

# Play the current game

if game is not None:

playGame(game)

else:

print("Please load a game first.")

# Function to display menu and get user choice

def getMenuChoice():

# Display menu options

print("""

0) Exit

1) Load default game

2) Load a game file

3) Save the current game

4) Edit or add a node

5) Play the current game

""")

# Get user input for choice

userChoice = input("What would you like to do? ")

return int(userChoice)

# Function to load the default game

def getDefaultGame():

# Define default game data

game = {

"start": {

"description": "You are in a dark room",

"menus": ["Play Again", "Exit"],

"nodes": ["start", "quit"]

}

}

return game

# Function to play the game

def playGame(game):

# Initialize game state

keepGoing = True

currentNode = "start"

# Main game loop

while keepGoing:

# Check if the game should continue

if currentNode == "quit":

keepGoing = False

else:

currentNode = playNode(game, currentNode)

# Function to play a node in the game

def playNode(game, currentNode):

try:

# Retrieve node data

nodeData = game[currentNode]

description = nodeData["description"]

menus = nodeData["menus"]

nodes = nodeData["nodes"]

# Display node information and options

print(f"{description}")

for i, menu in enumerate(menus, start=1):

print(f"{i}) {menu}")

# Get user choice

while True:

response = input("What will you do? (Enter the number of your choice) ")

if response.isdigit():

response = int(response)

if response >= 1 and response <= len(nodes):

nextNode = nodes[response - 1]

break

else:

print("Invalid choice. Please select a valid option.")

else:

print("Invalid input. Please enter a number.")

# Return the next node

return nextNode

except KeyError:

# Handle illegal node encountered

print("Illegal node encountered. Exiting the game.")

return "quit"

# Function to save the current game

def saveGame(game):

try:

# Get file name from user

filename = input("Enter the filename to save (must end with .json): ")

if not filename.endswith('.json'):

print("File name must end with .json extension.")

return

# Save game data to file

with open(filename, "w") as outFile:

json.dump(game, outFile, indent=2)

print(f"Saved game data to {filename}")

except Exception as e:

# Handle errors while saving the game

print(f"Error occurred while saving the game: {e}")

# Function to load a game from a file

def loadGame():

try:

# Get list of available game files

files = os.listdir()

print("Available game files:")

for file in files:

if file.endswith(".json"):

print(file)

# Get file name from user

filename = input("Enter the filename to load: ")

# Load game data from file

with open(filename, "r") as inFile:

game = json.load(inFile)

print(f"Loaded game from {filename}")

return game

except FileNotFoundError:

# Handle file not found error

print("File not found.")

except Exception as e:

# Handle other errors while loading the game

print(f"Error occurred while loading the game: {e}")

return None

# Function to edit or add a node in the game

def editNode(game):

try:

# Display current game status

print("Current status of game:")

print(json.dumps(game, indent=2))

print("Current Nodes:")

for nodeName in game.keys():

print(nodeName)

# Get node name from user

newNode = input("Name of new node? ")

# Check if node already exists

if newNode in game.keys():

print("Node already exists. Editing existing node.")

nodeData = game[newNode]

else:

print("Creating new node.")

nodeData = {"description": "", "menus": [], "nodes": []}

# Get node details from user

nodeData["description"] = input("Enter the description for this node: ")

numMenus = int(input("How many menus do you want for this node? "))

for i in range(numMenus):

menu = input(f"Enter menu option {i + 1}: ")

nodeData["menus"].append(menu)

# Get corresponding nodes for each menu option

print("For each menu option, enter the corresponding node name:")

for menu in nodeData["menus"]:

node = input(f"Enter the node for menu option '{menu}': ")

nodeData["nodes"].append(node)

# Update game with new/edited node

game[newNode] = nodeData

print("Node updated successfully.")

except Exception as e:

# Handle errors while editing node

print(f"Error occurred while editing node: {e}")

# Call the main function when the script is executed directly

if \_\_name\_\_ == "\_\_main\_\_":

call main()