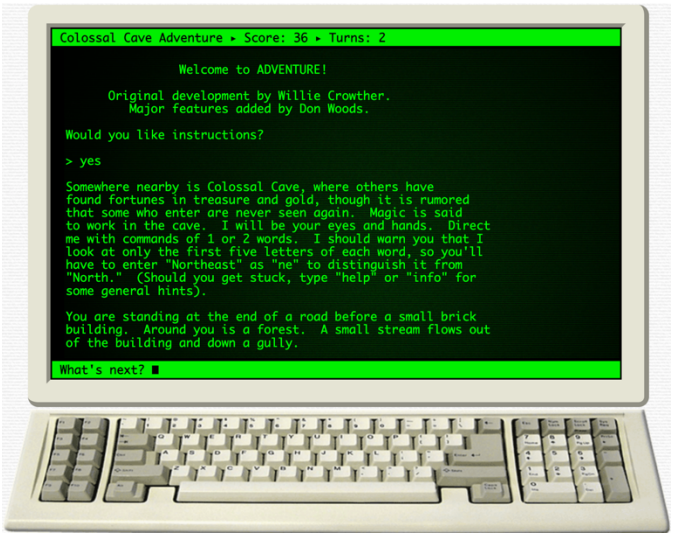
Text-Based Adventure Game

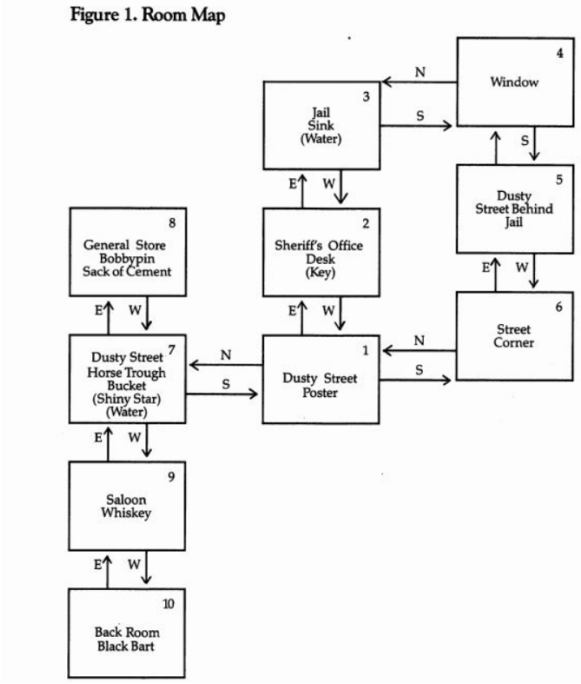
Introduction

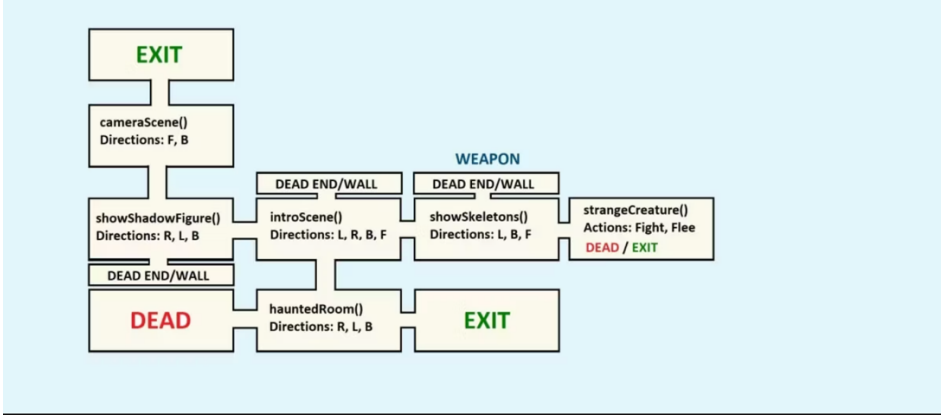


In this project, you will create your own text-based adventure game inspired by classic games such as The Oregon Trail, Colossal Cave Adventure, Zork, Star Trek, etc. The story for your game will contain several scenes or "rooms". You can create a function for each scene so that you can reuse it later if the player ends up entering the same room again. Each scene will also have different choices of where to go, which means it will have a list of valid directions and an if-statement for the multiple paths the player can take. Depending on the path the player takes in your Python adventure game, the script will call the next scene. This is an introduction to concepts like decision-making, branching narratives, and basic game mechanics. We will use conditionals, loops, functions, and user input.

Steps to create your game:

1. Discuss your game design:
    a. Brainstorm ideas about the game you would like to create.
    b. First come up with ideas for the storyline, setting, and characters of the game. Discuss basic game mechanics like player choices and branching narratives.
    c. Then discuss creating different locations, items, and characters in the game.
    d. Map out your story like in the images below to make sure the story for your text-based adventure game is well organized:





2. Set up the Python script:
    a. Refer to the example code below for ideas about how to set up your Python script for your game. You can adjust this accordingly.
    b. Consider organizing code into functions for readability and reusability.
3. Create the main game loop:
    a. Set up a main game loop where the player can make choices and navigate through the game.
    b. Use `input()` to get player choices.
4. Implement player choices:
    a. Create functions to handle different choices the player can make.
    b. Use conditionals (`if`, `elif`, `else`) to handle different scenarios based on player input.
5. Add challenges and interactions:
    a. Introduce random events or challenges that the player must overcome.

    b. Use loops (`while`) to repeat actions until a condition is met.
6. Testing and debugging:
    a. Test the game by playing through different scenarios.
    b. Debug any issues or errors.

Additional suggestions:

1. Adding polish:
    a. Add descriptive text and feedback to enhance the player experience.
2. Consider what graphics would go along with this game if you were to include a GUI (graphical user interface).
3. Have other GWC members play your game!
    a. This will allow for good feedback about how you can improve your game ☺

Example code structure:

```
# Sample code structure for a simple text-based adventure game
def start_game():
    print("Welcome to the Text Adventure Game!")
    # Initialize game state variables
    # e.g., current_location = "start"
    # e.g., player_inventory = []

def main_game_loop():
    while True:
        # Display current location and options
        # e.g., print("You are in the", current_location)
        # e.g., print("What do you want to do?")
        # e.g., print("1. Go north")
        # e.g., print("2. Go south")
        # Get player input
        # e.g., choice = input("Enter your choice: ")
        # Process player input and call appropriate functions

def explore():
    # Function to handle player exploration choices

def interact():
    # Function to handle player interactions with objects or
characters

# Call start_game() to begin
start_game()
# Call the main game loop
main_game_loop()
```

Simple example of a text-based adventure game where the player explores a haunted house:

```
import time

def explore_room(room):
    print("\nYou are now in the", room["name"])
    print(room["description"])
    time.sleep(1)
    if "item" in room:
        print("You found a", room["item"])
        inventory.append(room["item"])
        del room["item"]
    time.sleep(1)

def main_game_loop():
    current_room = rooms["entrance"]
    while True:
        explore_room(current_room)
        if current_room == rooms["treasure"]:
            print("Congratulations! You found the treasure and
won the game!")
            break
        next_room = input("Where do you want to go next?
(north/south/east/west): ").lower()
        if next_room in current_room:
            current_room = rooms[current_room[next_room]]
        else:
            print("You can't go that way. Try another
direction.")

# Define the rooms and their connections
rooms = {
    "entrance": {
        "name": "Entrance Hall",
        "description": "You are standing in the dusty entrance
hall of a haunted house.",
        "north": "kitchen",
        "east": "living room"
    },
    "kitchen": {
        "name": "Kitchen",
        "description": "You enter a dimly lit kitchen with
cobwebs covering the ceiling.",
        "south": "entrance"
    },
    "living room": {
```

```
        "name": "Living Room",
        "description": "You step into a spooky living room with
old furniture covered in sheets.",
        "west": "entrance",
        "north": "treasure"
    },
    "treasure": {
        "name": "Treasure Room",
        "description": "You find yourself in a room filled with
glittering treasure!"
    }
}

inventory = []

print("Welcome to the Haunted House Adventure Game!")
print("Your goal is to find the treasure hidden in the house.")

main_game_loop()
```