Student ID-

University of Essex Online

[Fundamentals of Programming]

[Unit 9 – Adventure Game]

Contents

Section A – Python Code

The full-size screenshot is also attached.

```python
##Imports.
import random
import time
import sys

##Text formatting. References:
  #https://stackoverflow.com/questions/287871/how-do-i-print-colored-text-to-the-terminal
  #https://stackoverflow.com/questions/4842424/list-of-ansi-color-escape-sequences
  #https://www.101computing.net/python-typing-text-effect/
textColours = {
    "DEFAULT": '\033[0m',  #White (fallback).

    "PROMPT": '\033[96m',  #Cyan. For player prompts / descriptions.
    "NEGATIVE": '\033[31m',  #Red. For warnings / dangers.
    "NEUTRAL": '\033[33m',  #Yellow. For neutral information.
    "POSITIVE": '\033[32m',  #Green. For success / positive choices.

    "SPEECH": '\033[95m',  #Light magenta. For any speech.

    "ITEM": '\033[30;45m',  #Black on magenta background. For item names.
    "GOLDENKEY": '\033[33;45m'  #Yellow on magenta background. For the golden key.
}

def textEffects(text, colour="DEFAULT", clean=False, typewriter=False):
    #Clean text (for user inputs). Return early so the other effects don't ruin input.
    if clean:
        return text.strip().lower()

    #Colour text.
    code = textColours.get(colour, textColours["DEFAULT"])
    colouredText = f"{code}{text}{textColours['DEFAULT']}"

    #Typewriter effect.
    if typewriter:
        for char in colouredText:
            sys.stdout.write(char)
            sys.stdout.flush()
            time.sleep(0.03)  #Typewriter speed.
        return ""
    return colouredText

##World setup. References:
  #https://www.w3schools.com/python/python_dictionaries.asp
roomsList = {  #List of rooms and their details.
    "Entrance": {  #Start location. No traps or items.
        "description": "You stand outside HelixCore's datacentre. The building is guarded, but you slip past into the side entrance.",
        "directions": {"Forward": "Lobby"},
        "items": [],
        "trap": None,
    },
    "Lobby": {
        "description": "You stand in the main lobby of HelixCore- no staff, just the piercing silence and an empty front desk.",
        "directions": {"Back": "Entrance", "Left": "Staff Offices", "Right": "Security Office"},
        "items": [],
        "trap": None,
    },

    "Staff Offices": {
        "description": "You enter the staff offices. Empty desks surround you with the dull sound of computers whirring.",
        "directions": {"Back": "Lobby", "Left": "Break Room", "Right": "Manager's Office"},
        "items": [],
        "trap": None,
    },
    "Break Room": {
        "description": "You peek inside the employee break room, the smell of stale coffee overwhelming you.",
        "directions": {"Back": "Staff Offices"},
        "items": [],
        "trap": None,
    },
    "Manager's Office": {
        "description": "Surprisingly, the manager's office is left unlocked. Just a desk and dull company mottos stand inside.",
        "directions": {"Back": "Staff Offices"},
        "items": [],
        "trap": None,
    },


    "Security Office": {
        "description": "The security office glows bright with monitors watching every angle of the vault.",
        "directions": {"Back": "Lobby", "Left": "Server Room", "Forward": "CCTV Data Room", "Right": "Supply Closet"},
        "items": [],
        "trap": None,
    },
    "CCTV Data Room": {
        "description": "Screens and racks whir, recording every movement outside the building.",
        "directions": {"Back": "Server Room", "Left": "Server Room"},
        "items": [],
        "trap": None,
    },
    "Supply Closet": {
        "description": "Nothing interesting here, just cleaning tools and dust.",
        "directions": {"Back": "Security Office"},
        "items": [],
        "trap": None,
    },

```

```python

    "Server Room": {
        "description": "You slip undetected into the server room. Server racks hum around you.",
        "directions": {"Back": "Security Office", "Left": "Data Backup", "Right": "CCTV Data Room", "Forward": "Server Cooling Room"},
        "items": [],
        "trap": None,
    },
    "Data Backup": {
        "description": "The exact same layout as the server room, backing up the data from those servers.",
        "directions": {"Back": "Server Room", "Forward": "Vault"},
        "items": [],
        "trap": None,
    },
    "Server Cooling Room": {
        "description": "Pumps whine, pushing coolant into the previous room.",
        "directions": {"Back": "Server Room", "Forward": "Vault"},
        "items": [],
        "trap": None,
    },

    "Vault": {  #Location of the golden key. No traps or directions as the key ends the game.
        "description": "You stand in HelixCore's vault, the golden key glistening in the centre of the room.",
        "directions": {},
        "items": ["Golden Key"],
        "trap": None,
    },
}

itemsList = {  #List of items and their details.
    "EMP": {
        "description": "A single electromagnetic pulse. Good for removing traps.",
        "effects": "Disable one trap.",
    },
    "Quickhack": {
        "description": "Hack HelixCore's security protocol once, slows their tracking.",
        "effects": "Grants 1 extra move.",
    },
    "Vault Key 1": {
        "description": "A glowing key, with the number 1 etched onto it.",
        "effects": "Vault Key 1.",
    },
    "Vault Key 2": {
        "description": "A glowing key, with the number 2 etched onto it.",
        "effects": "Vault Key 2.",
    },
    #Golden key is not here as it only spawns in the vault.
}

trapsList = {  #List of traps and their details.
    #Dodge is for using an item to disable the trap.
    #Detect is for not using / not having an item to disable the trap.
    #DecrementMoves is how many moves the trap deducts from the player.
    "CCTV": {
        "description": "As you enter, you look to the ceiling and spot a CCTV camera swivelling your way.",
        "dodge": "You act quickly, disabling the camera before it can spot you.",
        "detect": "The camera spots you, immediately flagging security.\nLose 1 move.",
        "decrementMoves": 1,
    },
    "Tripwire Alarm": {
        "description": "You step into the room, noticing that you have stepped into a tripwire.",
        "dodge": "You disable the tripwire, allowing you to progress.",
        "detect": "You step into the tripwire, alerting security to your presence.\nLose 2 moves.",
        "decrementMoves": 2,
    },
    "Laser Grid": {
        "description": "You vaguely notice a laser grid blocking your path.",
        "dodge": "You hastily disable the laser grid, unblocking your path.",
        "detect": "You trigger the laser grid, causing a silent alarm.\nLose 2 moves.",
        "decrementMoves": 2,
    },
}

#Randomly place items and traps in the rooms. References:
  #https://www.w3schools.com/python/ref_random_sample.asp
availableRooms = [
    room for room in roomsList.keys() if room not in ["Entrance", "Lobby", "Vault"]
    ]  #Excludes the first two rooms and vault.

#Distribute items throughout the rooms randomly.
itemRooms = random.sample(availableRooms, len(itemsList))
for i, item in enumerate(itemsList.keys()):
    roomsList[itemRooms[i]]["items"].append(item)

#Add 2 traps to random rooms.
trapRooms = random.sample(availableRooms, 2)
for room in trapRooms:
    trap = random.choice(list(trapsList.keys()))
    roomsList[room]["trap"] = trap

##Player setup.
Player = {
    "Room": "Entrance",
    "Inventory": [],
    "MovesLeft": 0,
    "LastRoom": None,  #To make traps only trigger once per room.

    "hasKey1": False,  #Vault keys to unlock vault.
    "hasKey2": False,
    "Victory": False,  #Victory condition (having the golden key).
}
```

```python
def difficultySelect():  #Function for difficulty selection.
    while True:
        difficulty = textEffects(input(
            textEffects(
                "Please select a difficulty.\n", colour="PROMPT") +
                    textEffects("1. Easy\n", colour="POSITIVE") +
                    textEffects("2. Medium\n", colour="NEUTRAL") +
                    textEffects("3. Hard\n", colour="NEGATIVE") +
                    textEffects("» ", colour="PROMPT")
                    ), clean=True)
        #Check user input and assign moves depending on difficulty.
        if difficulty in ["1", "easy"]:  #Easy difficulty:
            print("You have selected Easy.\n")
            Player["MovesLeft"] = 20
            break
        elif difficulty in ["2", "medium"]:  #Medium difficulty:
            print("You have selected Medium.\n")
            Player["MovesLeft"] = 17
            break
        elif difficulty in ["3", "hard"]:  #Hard difficulty:
            print("You have selected Hard.\n")
            Player["MovesLeft"] = 15
            break
        else:
            print(textEffects("Invalid input, please try again.\n", colour="NEGATIVE"))
difficultySelect()

##Game functions.
def printRoom(roomName):  #Displays current room's description
    room = roomsList[roomName]
    print("\nLocation: " + textEffects(roomName, colour = "PROMPT"))
    print(textEffects(room['description'], colour="PROMPT") + "\n")

def trapHandler(currentRoom):  #Handles traps in the player's room.
    while currentRoom["trap"] is not None:
        trap = trapsList[currentRoom["trap"]]
        print(textEffects(trap["description"], colour="NEUTRAL") + "\n")
        #If player has an EMP, give them the choice to use it.
        if "EMP" in Player["Inventory"]:
            useEMP = textEffects(input(
                "You have an EMP in your inventory, would you like to use it? (Y/N)\n» "), clean=True)
            if useEMP in ["y", "yes"]:  #Uses EMP to disable trap.
                print(textEffects(trap["dodge"], colour="POSITIVE") + "\n")
                Player["Inventory"].remove("EMP")
                currentRoom["trap"] = None
                break
            elif useEMP in ["n", "no"]:  #EMP not used, trap affects player.
                print(textEffects(trap["detect"], colour="NEGATIVE") + "\n")
                Player["MovesLeft"] -= trap["decrementMoves"]
                break  #Trap stays active.
            else:
                print("Invalid input. Please enter Y or N.")
        #If player doesn't have EMP, trap affects player.
        else:
            print(trap["detect"] + "\n")
            Player["MovesLeft"] -= trap["decrementMoves"]
            break
        print("\n")

def choiceHandler(choice, currentRoom): #Handles player choices.
    #Help menu.
    if choice == "help":
        print(textEffects(
"""
> 'Go [direction]'
> 'Take [item]'
> 'Use [item]'
> 'Inventory'
> 'Look'
""",
        colour = "PROMPT", typewriter=True))
        time.sleep(1)

    #Go command (to move between rooms).
    elif choice.startswith("go "):
        direction = choice[3:].capitalize()
        if direction in currentRoom["directions"]:
            #If the direction is the vault, check for keys.
            if currentRoom["directions"][direction] == "Vault":
                if not (Player["hasKey1"] and Player["hasKey2"]):
                    print(textEffects(
                        "The vault is locked, you need both vault keys to enter.",
                        colour="NEGATIVE"))
                    return
            #Otherwise, move the player.
            Player["Room"] = currentRoom["directions"][direction]
            Player["MovesLeft"] -= 1
        else:
            print(textEffects(
                "Invalid direction. Please try again.",
                colour="NEGATIVE"))

```

```python

    #Take command (to pick up items).
    elif choice.startswith("take "):
        item = textEffects(choice[5:], clean=True)
        foundItem = None
        for roomItem in currentRoom["items"]:
            if textEffects(roomItem, clean=True) == item:
                foundItem = roomItem
                break
        if foundItem:
            Player["Inventory"].append(foundItem)
            currentRoom["items"].remove(foundItem)
            print(textEffects(
                f"You have picked up: {item}",
                colour="PROMPT"))
            #Check for vault keys / golden key.
            if foundItem == "Vault Key 1":
                Player["hasKey1"] = True
            elif foundItem == "Vault Key 2":
                Player["hasKey2"] = True
            elif foundItem == "Golden Key":
                Player["Victory"] = True
        else:
            print(textEffects(
                "Invalid item. Please try again.",
                colour="NEGATIVE"))

    #Use command (to use items in inventory).
    elif choice.startswith("use "):
        item = textEffects(choice[4:], clean=True)
        foundItem = None
        for invItem in Player["Inventory"]:
            if textEffects(invItem, clean=True) == item:
                foundItem = invItem
                break
        if foundItem:
            if foundItem == "Quickhack":
                Player["MovesLeft"] += 1
                Player["Inventory"].remove("Quickhack")
                print(textEffects(
                    "You use the Quickhack, halting HelixCore's tracking momentarily.\nYou gain 1 extra move.",
                    colour="POSITIVE"
                ))

    #Inventory command (to view inventory).
    elif choice == "inventory":
        if Player["Inventory"]:
            for item in Player["Inventory"]:
                itemDetails = itemsList.get(item)
                if itemDetails:
                    print(textEffects(f"- {item}: {itemDetails['description']}",
                                      colour="ITEM"))
        else:
            print("Your inventory is empty.")

    #Look command (describes the room).
    elif choice == "look":
        if currentRoom["items"]:
            print("You see the following items: " +
                  textEffects(", ".join(currentRoom["items"]), colour="ITEM"))
            print()
        else:
            print(textEffects("There are no items in this room.\n",
                              colour="PROMPT"))

    #Invalid command otherwise.
    else:
        print(textEffects("Invalid command, please try again.\n",
                          colour="NEGATIVE"))

    print("")

##Game start.
print(textEffects(
f"""
> Your goal, runner, is to infiltrate HelixCore's datacentre and retrieve the golden key from their vault.
> Beware though, HelixCore will quickly notice your presence once you're inside.
> Be sure to <<look>> around each room and <<take>> any useful items you find.
> You will only get <<{Player['MovesLeft']}>> moves to complete your mission, so plan wisely. Good luck.
""",
colour="SPEECH", typewriter=True
))
time.sleep(1)

```

```python

##Main loop.
while Player["MovesLeft"] > 0 and not Player["Victory"]:  #While player has moves and hasn't won:
    print(textEffects(
        ">" * 50,  #Allows player to process the information.
        colour="PROMPT", typewriter=True))
    currentRoom = roomsList[Player["Room"]]
    #Room description function.
    printRoom(Player["Room"])

    #Trap handler.
    if Player.get("LastRoom") != Player["Room"]:
        trapHandler(currentRoom)
        Player["LastRoom"] = Player["Room"]

    #Show directions & moves left.
    print("You can go: " +
          textEffects(", ".join(currentRoom["directions"].keys()), colour="PROMPT") + "\n" +
          "Moves left: " + textEffects(Player["MovesLeft"], colour="PROMPT") + "\n")

    #Get player's choice / handle it.
    choice = textEffects(input(
        "What's your next move?\nType 'Help' for all commands.\n» "),
        clean=True)
    choiceHandler(choice, currentRoom)

#Check for victory condition.
if Player["Victory"]:
    print(textEffects(
f"""
> *You hold the golden key in your hands, feeling its weight and power.*
> Great job runner, your mission was successful with {Player['MovesLeft']} moves to spare.
> Now, get yourself out of there swiftly.
""",
colour="POSITIVE", typewriter=True))

else:
    print(textEffects(
"""
> HelixCore have managed to locate your position, you need to get out of there, now.
> Game over. (You have run out of moves.)
""",
colour="NEGATIVE", typewriter=True))

exitInput = input("Press enter to exit.")

#Bugs:
    #If player uses a command that doesn't use a move (i.e. inventory) in a room
    #with a trap, the trap will trigger again.
    #Fixed by adding "LastRoom" to Player dictionary and comparing it to current room.

    #Vault will allow player in even if they don't have both keys.
    #Fixed by adding a check before moving
    #and adding "HasKey1" and "HasKey2" to Player dictionary.
```

Section B – Description (250 Words) & References

This game is a text-based adventure set inside a Cyberpunk and Watch Dogs themed world. The player is a net-runner and their objective is to seek out the golden key within a mega corporation's datacentre, known as HelixCore. The world is a structured list of dictionaries, each with their own description, directions, possible items and traps. The player begins at the entrance and navigates these connected rooms in hopes of finding the golden key.

The world is dynamically populated: items (EMP, Quickhack and Vault Keys) are placed randomly at the start of each playthrough and traps (CCTV, Tripwires and Laser Grids) are also assigned to random rooms to increase playability and increase the challenge. The items each provide functional benefits: the EMP can disable traps, the Quickhack grants the player an extra move, and the Vault Keys are required to access the vault where the golden key lies.

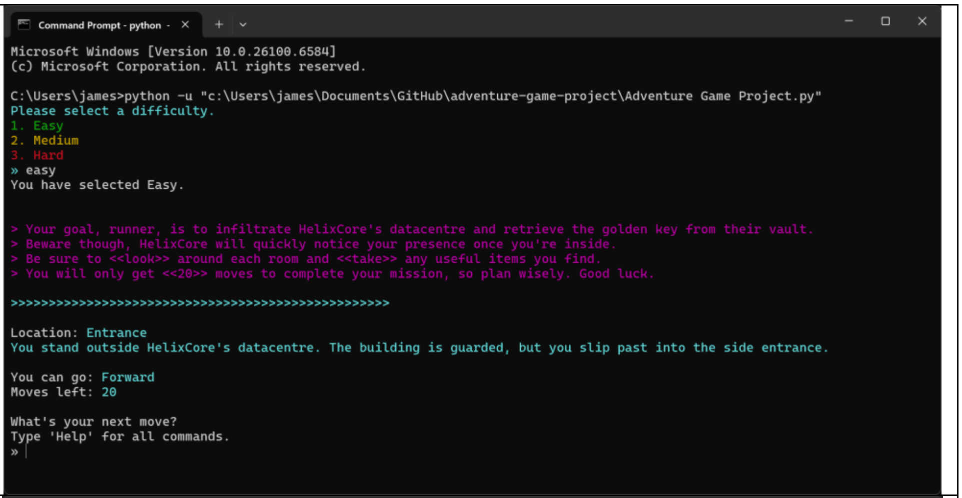
Player progress is tracked through a player dictionary, storing their current room, inventory, moves left and victory conditions. Moves are assigned based on the difficulty selected by the player (Easy, Medium and Hard), allowing them to select their desired pace. Functions such as printRoom() and trapHandler() handle the game's flow and choiceHandler() handles the player's inputs. The textEffects() function is used heavily to recolour and/or add a typewriter effect to the game's text to increase immersion.

Overall, the game combines structured room navigation, randomised elements, and inventory-based interaction, resulting in a concise but replayable adventure that balances exploration, resource management, and risk.

Section C – Sample Gameplay

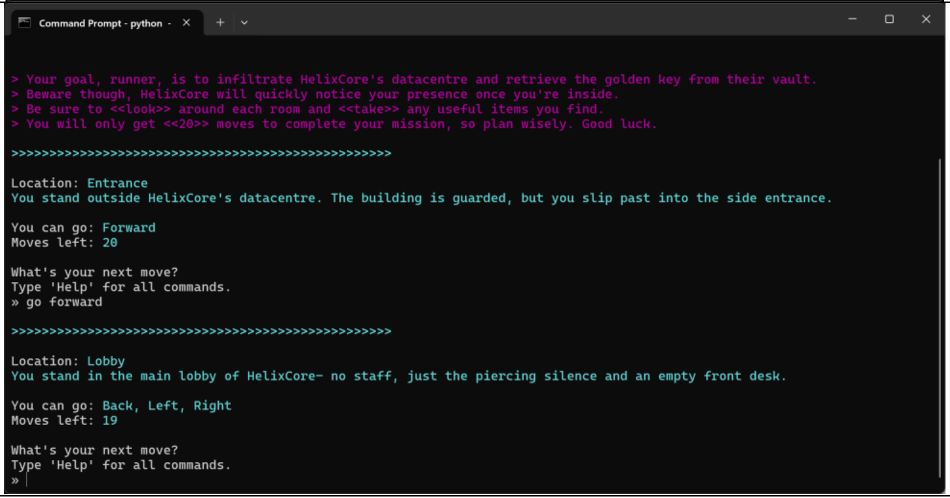
Here are the screenshots of the game in action:

Difficulty Selection



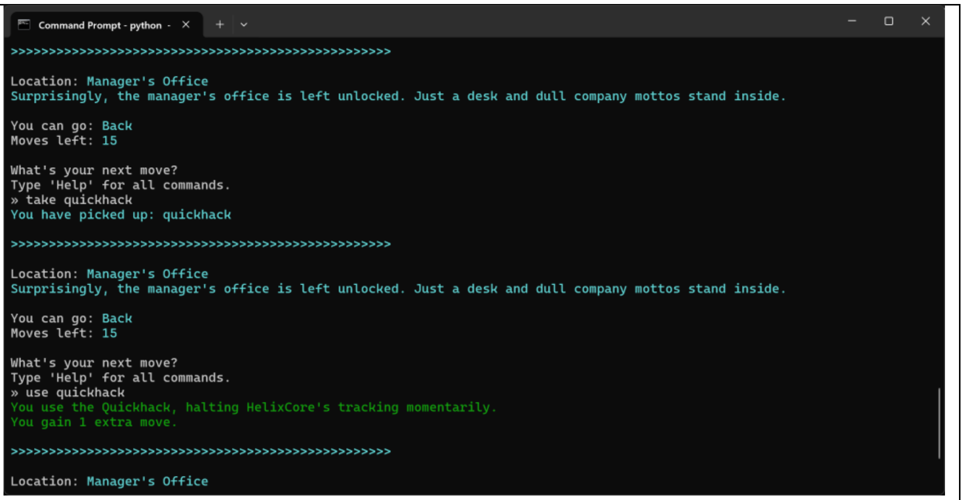
```
Microsoft Windows [Version 10.0.26100.6584]
(c) Microsoft Corporation. All rights reserved.

C:\Users\james>python -u "c:\Users\james\Documents\GitHub\adventure-game-project\Adventure Game Project.py"
Please select a difficulty.
1. Easy
2. Medium
3. Hard
» easy
You have selected Easy.


> Your goal, runner, is to infiltrate HelixCore's datacentre and retrieve the golden key from their vault.
> Beware though, HelixCore will quickly notice your presence once you're inside.
> Be sure to <<look>> around each room and <<take>> any useful items you find.
> You will only get <<20>> moves to complete your mission, so plan wisely. Good luck.

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Location: Entrance
You stand outside HelixCore's datacentre. The building is guarded, but you slip past into the side entrance.

You can go: Forward
Moves left: 20

What's your next move?
Type 'Help' for all commands.
»
```

Go



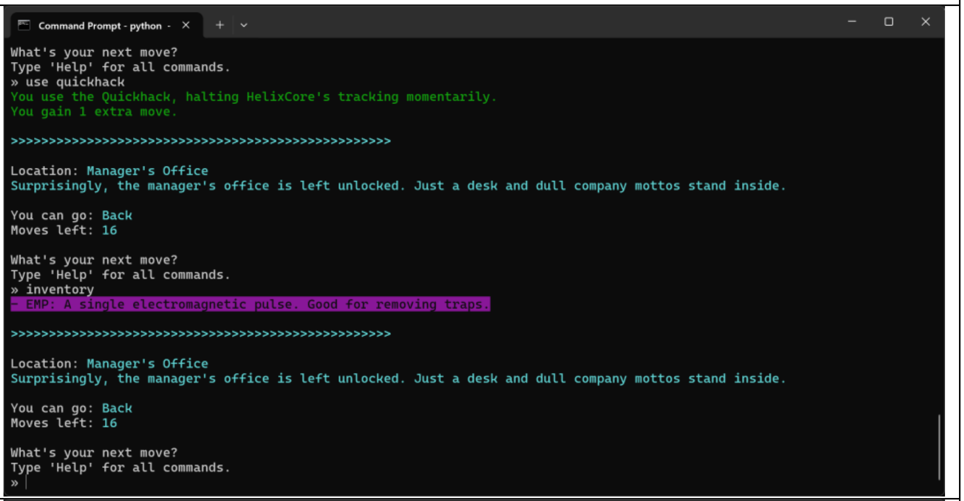
```
> Your goal, runner, is to infiltrate HelixCore's datacentre and retrieve the golden key from their vault.
> Beware though, HelixCore will quickly notice your presence once you're inside.
> Be sure to <<look>> around each room and <<take>> any useful items you find.
> You will only get <<20>> moves to complete your mission, so plan wisely. Good luck.

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Location: Entrance
You stand outside HelixCore's datacentre. The building is guarded, but you slip past into the side entrance.

You can go: Forward
Moves left: 20

What's your next move?
Type 'Help' for all commands.
» go forward

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Location: Lobby
You stand in the main lobby of HelixCore- no staff, just the piercing silence and an empty front desk.

You can go: Back, Left, Right
Moves left: 19

What's your next move?
Type 'Help' for all commands.
»
```

Take / Use



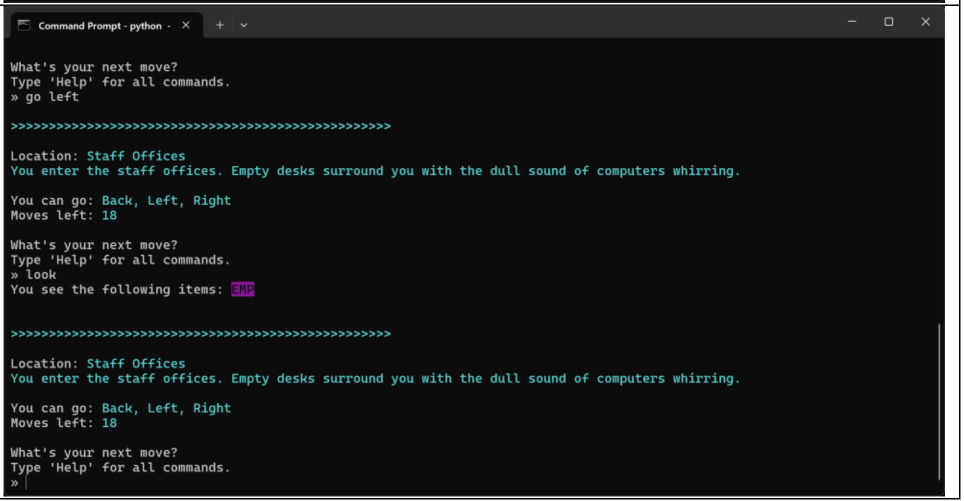
```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Location: Manager's Office
Surprisingly, the manager's office is left unlocked. Just a desk and dull company mottos stand inside.

You can go: Back
Moves left: 15

What's your next move?
Type 'Help' for all commands.
» take quickhack
You have picked up: quickhack

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Location: Manager's Office
Surprisingly, the manager's office is left unlocked. Just a desk and dull company mottos stand inside.

You can go: Back
Moves left: 15

What's your next move?
Type 'Help' for all commands.
» use quickhack
You use the Quickhack, halting HelixCore's tracking momentarily.
You gain 1 extra move.

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Location: Manager's Office
```

Inventory



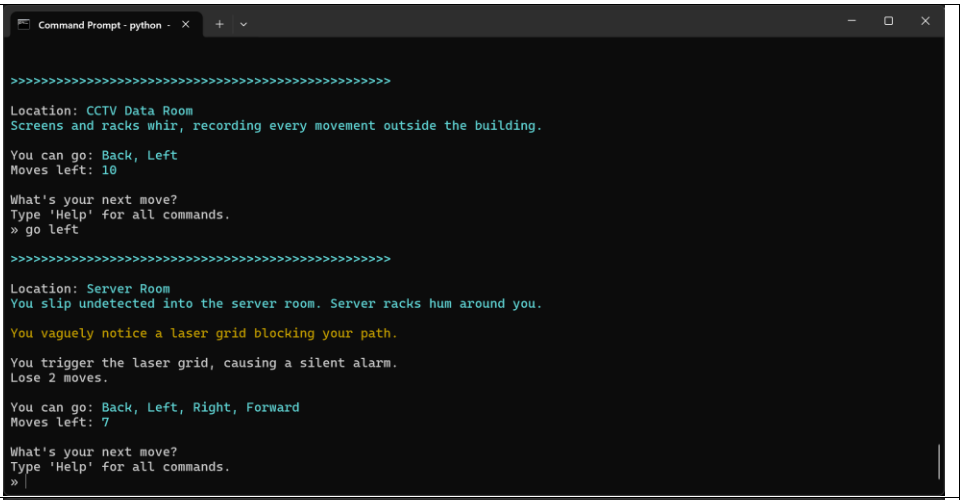
```
What's your next move?
Type 'Help' for all commands.
» use quickhack
You use the Quickhack, halting HelixCore's tracking momentarily.
You gain 1 extra move.

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Location: Manager's Office
Surprisingly, the manager's office is left unlocked. Just a desk and dull company mottos stand inside.

You can go: Back
Moves left: 16

What's your next move?
Type 'Help' for all commands.
» inventory
- EMP: A single electromagnetic pulse. Good for removing traps.

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Location: Manager's Office
Surprisingly, the manager's office is left unlocked. Just a desk and dull company mottos stand inside.

You can go: Back
Moves left: 16

What's your next move?
Type 'Help' for all commands.
»
```

Look



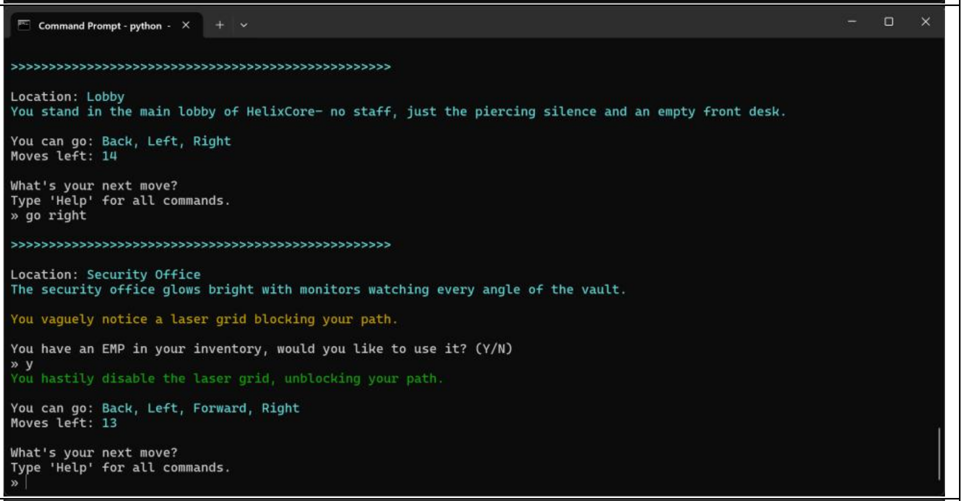
```

What's your next move?
Type 'Help' for all commands.
» go left

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Location: Staff Offices
You enter the staff offices. Empty desks surround you with the dull sound of computers whirring.

You can go: Back, Left, Right
Moves left: 18

What's your next move?
Type 'Help' for all commands.
» look
You see the following items: EMP


>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Location: Staff Offices
You enter the staff offices. Empty desks surround you with the dull sound of computers whirring.

You can go: Back, Left, Right
Moves left: 18

What's your next move?
Type 'Help' for all commands.
»
```

Trap Hit



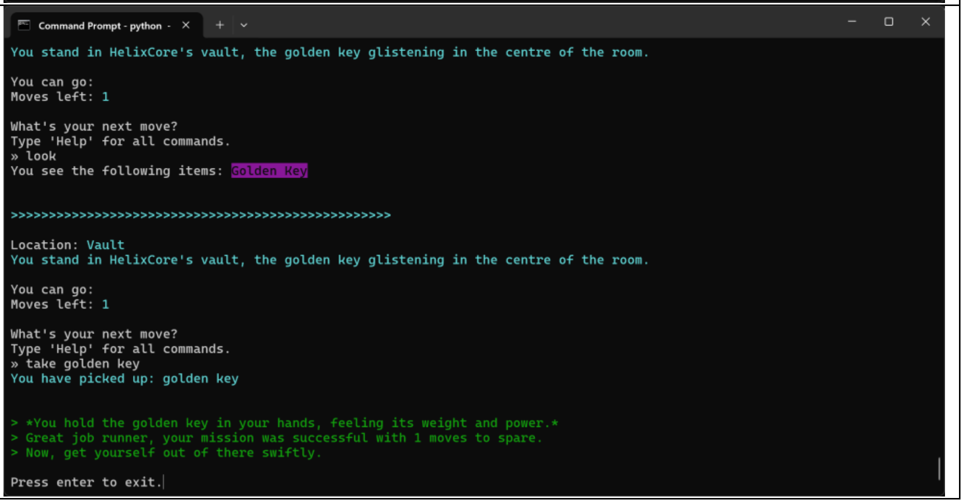
```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Location: CCTV Data Room
Screens and racks whir, recording every movement outside the building.

You can go: Back, Left
Moves left: 10

What's your next move?
Type 'Help' for all commands.
» go left

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Location: Server Room
You slip undetected into the server room. Server racks hum around you.

You vaguely notice a laser grid blocking your path.

You trigger the laser grid, causing a silent alarm.
Lose 2 moves.

You can go: Back, Left, Right, Forward
Moves left: 7

What's your next move?
Type 'Help' for all commands.
»
```

Trap Dodge

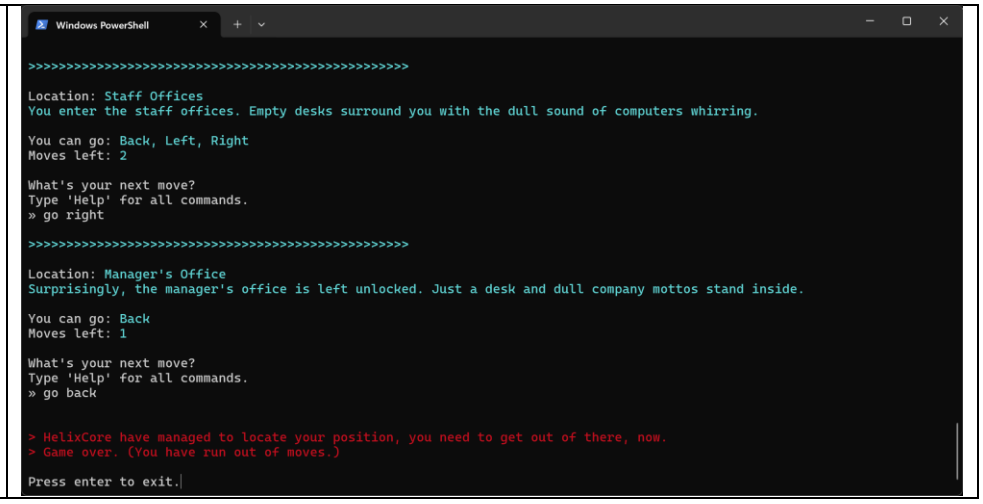


```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Location: Lobby
You stand in the main lobby of HelixCore- no staff, just the piercing silence and an empty front desk.

You can go: Back, Left, Right
Moves left: 14

What's your next move?
Type 'Help' for all commands.
» go right

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Location: Security Office
The security office glows bright with monitors watching every angle of the vault.

You vaguely notice a laser grid blocking your path.

You have an EMP in your inventory, would you like to use it? (Y/N)
» y
You hastily disable the laser grid, unblocking your path.

You can go: Back, Left, Forward, Right
Moves left: 13

What's your next move?
Type 'Help' for all commands.
»
```

Victory (Golden Key found)



```
You stand in HelixCore's vault, the golden key glistening in the centre of the room.

You can go:
Moves left: 1

What's your next move?
Type 'Help' for all commands.
» look
You see the following items: Golden Key


>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Location: Vault
You stand in HelixCore's vault, the golden key glistening in the centre of the room.

You can go:
Moves left: 1

What's your next move?
Type 'Help' for all commands.
» take golden key
You have picked up: golden key


> *You hold the golden key in your hands, feeling its weight and power.*
> Great job runner, your mission was successful with 1 moves to spare.
> Now, get yourself out of there swiftly.

Press enter to exit.
```

Loss (Run out of moves)	

```
>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Location: Staff Offices
You enter the staff offices. Empty desks surround you with the dull sound of computers whirring.

You can go: Back, Left, Right
Moves left: 2

What's your next move?
Type 'Help' for all commands.
» go right

>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Location: Manager's Office
Surprisingly, the manager's office is left unlocked. Just a desk and dull company mottos stand inside.

You can go: Back
Moves left: 1

What's your next move?
Type 'Help' for all commands.
» go back


> HelixCore have managed to locate your position, you need to get out of there, now.
> Game over. (You have run out of moves.)

Press enter to exit.
```

References:

- OpenAI (2025) ChatGPT (GPT-5). Available at: https://chat.openai.com/ (Accessed: 27 September 2025). (Used for references list).
- Stack Overflow (n.d.) *How do I print colored text to the terminal?* Available at: https://stackoverflow.com/questions/287871/how-do-i-print-colored-text-to-the-terminal (Accessed: 28 September 2025).
- Stack Overflow (n.d.) *List of ANSI color escape sequences.* Available at https://stackoverflow.com/questions/4842424/list-of-ansi-color-escape-sequences (Accessed: 28 September 2025).
- W3Schools (n.d.) *Python random.sample() Method.* Available at: https://www.w3schools.com/python/ref_random_sample.asp (Accessed 28 September 2025).
- W3Schools (n.d.) *Python dictionaries.* Available at: https://www.w3schools.com/python/python_dictionaries.asp (Accessed 29 September 2025).
- 101 Computing (n.d.). *Python Typing Effect.* Available at: https://www.101computing.net/python-typing-text-effect/ (Accessed 27 September 2025).