‹ Return to "Intro to Programming Nanodegree" in the classroom

Adventure Game

REVIEW | CODE REVIEW 6 | HISTORY

▾ adventure_game_vamshi_krishna_p.py 6

```
import time
import random
```

AWESOME

Awesome!! Great work importing (random) class.

```

# List of items to be used in the game, from line 5 - 130
shop_items = ["Health Portion          - Worth 50$",
```

AWESOME

Awesome!! Great work using lists to store your items

- Your list is well organized and sound good.
- Here to learn more about this.
  https://swcarpentry.github.io/python-novice-inflammation/03-lists/

```
              "Robes of Freya          - Worth 300$",
              "Dress of a princess     - Worth 500$",
              "Glittering steel armour - Worth 400$",
              "A grand Tribute         - Not for sale",
              "A Talisman              - Worth 300$",
              "Jewellary               - Worth 400$",
              "Durandal,a mighty sword - Worth 600$",
              "Steel Helmet            - Worth 200$",
              "Go back to village      - None      "]

player_items = ["trusty dagger",
                "Steel armour"]

gift_items = ["Leave"]

princess_items = []

bar_info_day = ["INFO: Don't go into the forest un-prepared",
                "INFO: King of this country like presents",
                "INFO: Check out the contests at night too",
                "INFO: The cave is nest to stronger enemies",
                "INFO: Forest is vast, explore out of options to "
                "find many things",
                "INFO: Did you know there is a legendary tomb in the forest?",
                "INFO: Animals in forest are faster than you, you got "
                "my point?"]

bar_info_night = ["INFO: Forest at night is dark. You may need fire source",
                  "INFO: Did you know that the forest has a range "
                  "between 1 to 50",
                  "INFO: The princess of this country, likes flowers",
                  "INFO: check out the secreat contests at night",
                  "INFO: Do not pick a fight with enemies stronger than you",
                  "INFO: The village next to us has been attacked by "
                  "Werewolves",
                  "INFO: Access to Fort is restricted at night",
                  "INFO: Sneaking into fort at night time might "
                  "result in imprisonment",
                  "INFO: Having full health is always an advantage"]

village_locations = ["Traven",
                     "Inn",
                     "Contest grounds",
                     "Fort",
                     "Cave",
                     "Forest"]

fort_locations_day = ["Library",
                      "Quaters",
 55                       "Garden",
 56                       "Central Hall",
 57                       "Go back to Village"]
 58 
 59 fort_locations_night = ["Library",
 60                         "Quaters",
 61                         "Garden",
 62                         "Central Hall"]
 63 
 64 contest_items = ["Unknown Key",
 65                  "Mystery Box"]
 66 
 67 rewards_day = ["Special Prize: Unknown Key",
 68                "Deposit Gold and win back Double",
 69                "Deposit 10 Gold, Win back 50 Gold"]
 70 
 71 rewards_night = ["Special Prize: Mystery Box",
 72                  "Deposit Gold and win back Double",
 73                  "Deposit no Gold, Win 50 back Gold"]
 74 
 75 player_offense_moves = ["You step forward to slash the enemy",
 76                         "You rise high to deliver a mighty blow",
 77                         "You make a lunge and slash attack",
 78                         "You deliver an counter attack",
 79                         "A power blow was directed towads the enemy",
 80                         "You rain down a series of short swings",
 81                         "A sharp delivery to penetrate the defense of enemy",
 82                         "You run towards the enemy at great speed",
 83                         "You duck down and deliver a low swing",
 84                         "You stand tall and attck the enemy from the front",
 85                         "You run around and penetrate the back of the enemy"]
 86 
 87 player_defense_moves = ["You hold the sword to counter the enemy attck",
 88                         "The enemy attacks are repelled by you",
 89                         "The enemy jumps on you and you roll to the side",
 90                         "You swing your sword in rythm to the enemy attacks",
 91                         "You step back away from the enemy",
 92                         "You dodge the enemy sword and ready to make an attack"
 93                         ]
 94 
 95 enemy_offense_moves = ["The enemy charges at you like a mad bull",
 96                        "The enemy has a arm streangth of a boulder",
 97                        "Enemy has wits and great strategy to corner you",
 98                        "The enemy attacks from behind",
 99                        "The enemy rolls around and attacks from the side",
100                        "The enemy jumps on you",
101                        "The enemy fights you head on"]
102 
103 enemy_defense_moves = ["Enemy has a strong defense",
104                        "Your enemy has brains to dodge your attacks",
105                        "Your enemy has speed advantage to evade your attacks",
106                        "Your enemy has a strength to with stand your attack",
107                        "Enemy is determined to take you head on",
108                        "Your enemy is a formidable opponent"]
109 
110 animal_offense_moves = ["The animal charges straight at you ",
111                         "The animal attacks you with ferocity ",
112                         "The animal attacks you from behind",
113                         "The animal tries to bite you",
114                         "The animal jumps on you",
115                         "The animal uses its claws"]
116 
117 animal_defense_moves = ["The animal has a thick skin",
118                         "The animal has brains to dodge your attacks",
119                         "The animal has speed advantage to evade your attacks",
120                         "The animal deflects your attacks with claws",
121                         "The animal steps back to counter"]
122 
123 animal_list = ["Boar", "Bear", "Wolf"]
124 
125 strong_animal_list = ["Leopard", "Tiger", "Werewolf"]
126 
127 monster_list = ["BARTENDER",
128                 "MINISTER",
129                 "SHOPKEEPER",
130                 "HOST of the contest grounds"]
131 
132 # triggers and declarations to be used in the code, from line 133 - 181
133 player = ""
134 shop_count = 0
135 freya_dress = "available"
136 princess_dress = "available"
137 armour = "available"
138 durandal = "available"
139 tribute = "available"
140 helmet = "available"
141 bar_count = 0
142 grapes_available = ""
143 player_health = 100
144 gold = 500
145 drink_count = 0
146 info_count = 0
147 time_zone = "day"
148 player_offense_min = 0
149 player_offense_max = 30
150 player_defense = 10
151 contest_day_count = 0
152 contest_night_count = 0
153 game_night_count = 0
154 glowing_blossom_day_count = 0
155 flowers_day_count = 0
156 grapes_day_count = 0
157 Keyword_count = 0
    forest_explore_count = 0
```

```
forest_trigger = "deactivate"
game_over = "deactivate"
grapes_available = ""
trade_contract_sequense = ""
```

AWESOME

Your variables are very descriptive. 👏

```
troll = "alive"
treasure_box = "not_opened"
chest = "not_opened"
monster = "alive"
huge_monster = "alive"
fort_count = 0
toll_status = ""
inner_fort_count = 0
fort_access = "restricted"
bell = "deactivated"
library_status = ""
princess_count = 0
princess_introduction = ""
name = ""
princess_quest_status = ""
player_status = ""
tomb_password = ""
witch_room_count = 0
kings_status = ""


# function to print messages with a 2 second delay
def print_pause(message):
    print(message)
    time.sleep(2)


# function to print messages with a 1 second delay
def print_pause1(message):
```

SUGGESTION

Instead of making 2 functions:

```
# function to print messages with a 2 second delay
def print_pause(message):
    print(message)
    time.sleep(2)


# function to print messages with a 1 second delay
def print_pause1(message):
    print(message)
    time.sleep(1)
```

You could make one function that takes another argument to take the delay time that you want:

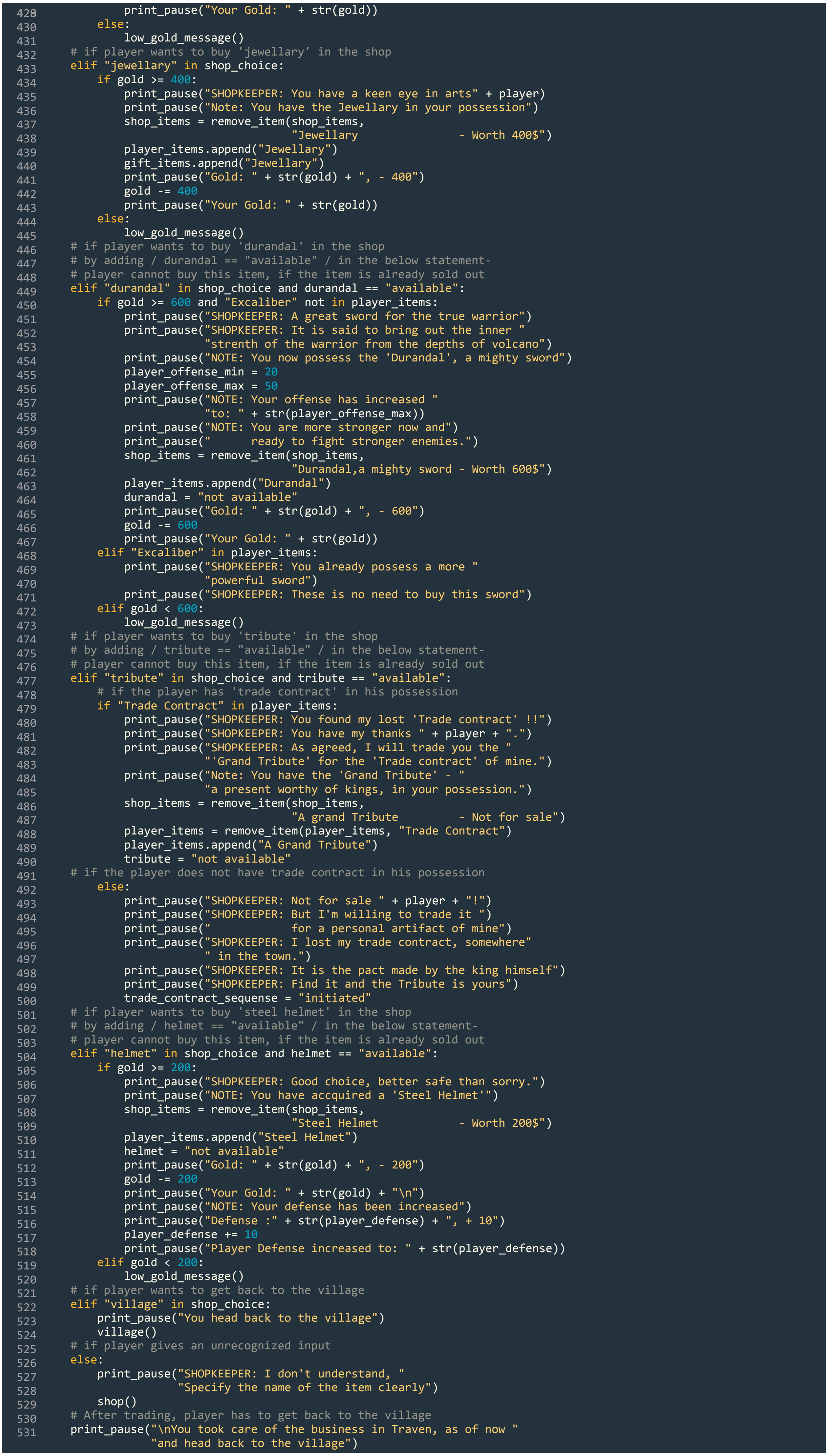
```
 def print_pause1(message, delay):
    print(message)
    time.sleep(delay)
```

```
    print(message)
    time.sleep(1)


# function to display the 'items in the list' with a serial number
def list_serial(list):
    for index in range(len(list)):
        print_pause1(str(index+1) + ". " + list[index])
    print_pause1("\n")
    return ""


# function to remove a specific item from a list
def remove_item(list, item):
    new_list = []
    for index in range(len(list)):
        if list[index] != item:
            new_list.append(list[index])
    list = new_list
    return list


# function to increase player's health within the limit of 100
def health_increase(num):
    global player_health
    if player_health < 100 and player_health > 0:
        player_health += num
        if player_health > 100:
            player_health = 100
        return player_health


# function to decrease player's health within the threshold of 0
def health_decrease(health, num):
    if health > 0:
        health += num
        if health < 0:
            health = 0
        return health
    else:
        health = 0
        return health


# function to display a message whenever the gold is insufficient
def low_gold_message():
    global gold
    global player
    print_pause("NOTE: You do not have enough cash " + player + "!")
    print_pause("Your Gold : " + gold)
    print_pause("NOTE: Try earning some in the town contest")


# ------------main function to trigger the game------------
def adventure_game():
    global player
    # input to ask for the player's name
    player = input("\nWhat is your name?\n")
    print_pause("\nOn gaurd " + player + ".")
    print_pause("This is a royal quest to slay the mighty monster")
    village()


# A branch sequence to the funstion 'adventure_game'
def village():
    # trigger to end the game from the function 'fight_sequence'
    if game_over == "activate":
        print_pause("\n\nGame over. Try again ........")
        exit()
    # village intro
    # player's choice
    print_pause("\nwelcome to the fabled village of Yorkshire")
    print_pause("Where would you like to venture into:")
    # displays list of village locations in a serial list
    village_choice = input(list_serial(village_locations)).lower()
    # if player decides to enter into the traven
    if "traven" in village_choice:
        print_pause("\nYou have entered Traven to make business")
        shop()
    # if player decides to enter into the Inn
    elif "inn" in village_choice:
        print_pause("\nYou have checked into an Inn")
        inn()
    # if player decides to enter into the contest grounds
    elif "contest" in village_choice:
        print_pause("\nYou head towards the contest grounds with curiosity")
        contest_grounds()
    # if player decides to enter into the forest
    elif "forest" in village_choice:
        forest()
    # if player decides to enter into the cave
    elif "cave" in village_choice:
        print_pause("You step into the darkness of cave courageously !!")
        print_pause("The cave is a labyrinth of multiple partitions")
        cave()
    # if player decides to enter into the fort
```

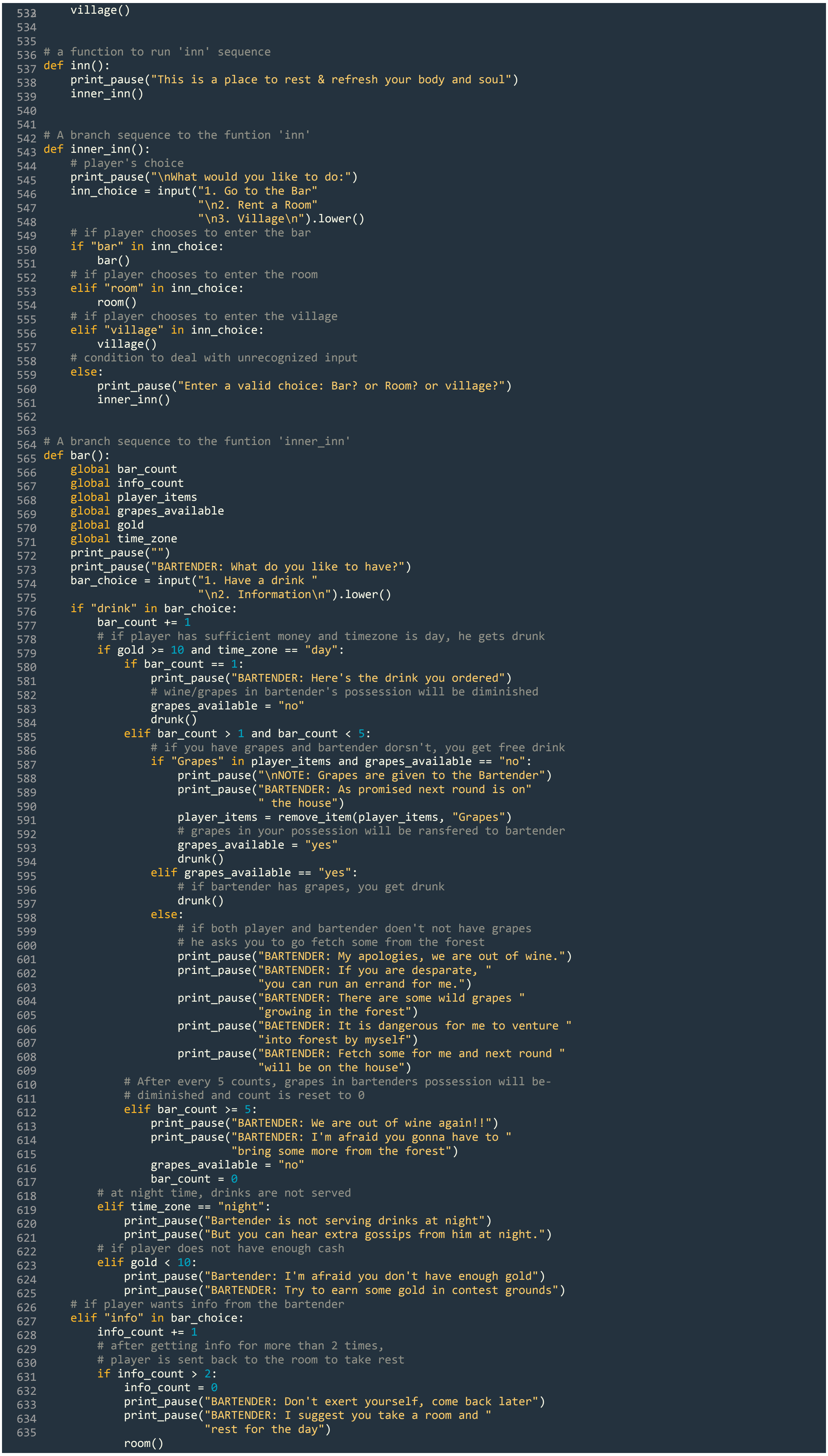
SUGGESTION

You don't need that, let your code explain itself, and it explains itself very well.

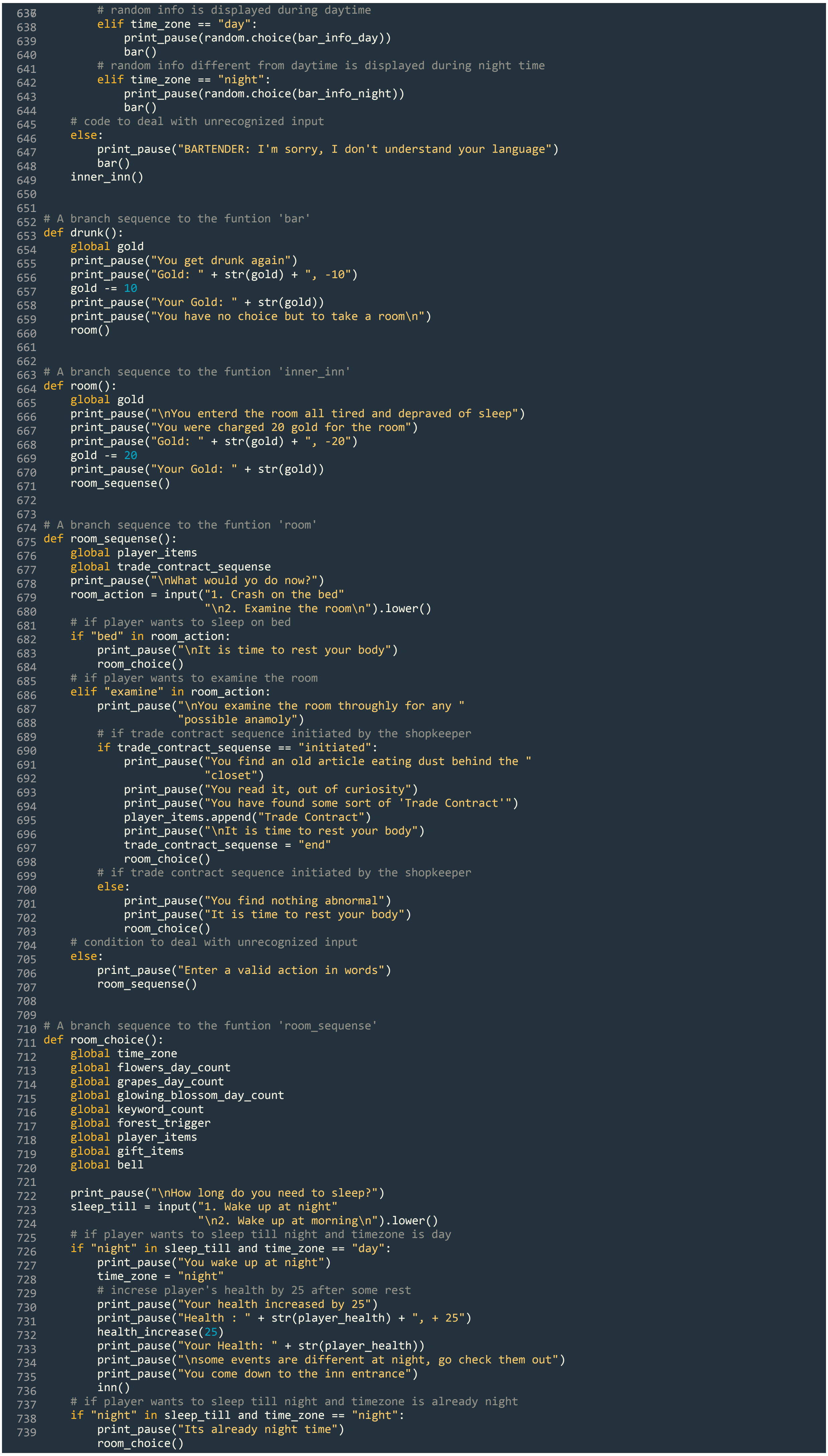
```
    elif "fort" in village_choice:
        fort()
    # if player decides to enter into the secret passage
    elif "secret" in village_choice and "Secret Passage" in village_locations:
        secret_passage()
    # condition to deal with unrecognized input
    else:
        print_pause("The place you are looking for does not exist "
                    "in this village")
        village()


# funtion to trigger shop sequence in the game
def shop():
    global gold
    global shop_items
    global player_items
    global player
    global player_status
    global player_offense_min
    global player_offense_max
    global player_defense
    global player_health
    global trade_contract_sequense
    global freya_dress
    global princess_dress
    global armour
    global durandal
    global tribute
    global helmet
    global gift_items
    global shop_count
    shop_count += 1
    # An introduction to be used only one time in the game
    if shop_count == 1:
        print_pause("SHOPKEEPER: Haah.., a fine warrior.")
        print_pause("SHOPKEEPER: Welcome to the Traven of trade and purchase.")
        print_pause("SHOPKEEPER: We have a large collection from far lands "
                    "of Euthopia.")
        print_pause("SHOPKEEPER: Best and most rarest of all, if you can "
                    "afford that is!")
        print_pause("SHOPKEEPER: Feel free to browse through them.")
    # A message to be displayed only if the player is wanted
    elif player_status == "wanted":
        print_pause("SHOPKEEPER: I am not allowed to trade with "
                    "criminals " + player + ".")
        print_pause("SHOPKEEPER: But I am willing to keep this as a "
                    "secret between us")
    # A message to be used throughout the game, if player revisit the shop
    else:
        print_pause("\nSHOPKEEPER: Welcome back " + player + " !!")
        print_pause("SHOPKEEPER: Back for more?")
        print_pause("SHOPKEEPER: Let me see if I can find, what you need.")

    print_pause("SHOPKEEPER: What do you want to purchase? Name it.")
    # Displays shops items in a serial list and take desired input from player
    shop_choice = input(list_serial(shop_items)).lower()
    # if player wants to buy health potion in the shop
    if "health" in shop_choice:
        if gold >= 50 and player_health < 100:
            print_pause("SHOPKEEPER: One must stock up on health or face "
                        "certain death")
            print_pause("Gold: " + str(gold) + ", - 50")
            gold -= 50
            print_pause("Your Gold: " + str(gold))
            health_increase(25)
            print_pause("Your health increased +25, feel stronger now ?")
            print_pause("Your Health: " + str(player_health))
        elif player_health == 100:
            print_pause("SHOPKEEPER: You already have full health")
        elif gold < 50:
            low_gold_message()
    # if player wants to buy 'freya dress' in the shop
    # by adding /freya_dress == "available"/ in the below statement-
    # player cannot buy this item, if the item is already sold out
    elif "freya" in shop_choice and freya_dress == "available":
        if gold >= 300:
            print_pause("SHOPKEEPER: A rare robe, legend has it's "
                        "been worn by Goddess freya herself !!\n")
            print_pause("Note: Now you have 'Robes of Freya' in your "
                        "possession")
            shop_items = remove_item(shop_items,
                                     "Robes of Freya          - Worth 300$")
            player_items.append("Robes of Freya")
            gift_items.append("Robes of Freya")
            freya_dress = "not available"
            print_pause("Gold: " + str(gold) + ", - 300")
            gold -= 300
            print_pause("Your Gold: " + str(gold))
        else:
            low_gold_message()
    # if player wants to buy 'princess dress' in the shop
    # by adding /princess_dress == "available"/ in below statement-
    # player cannot buy this item, if the item is already sold out
    elif "princess" in shop_choice and princess_dress == "available":
        if gold >= 500:
            print_pause("SHOPKEEPER: A fine dress! " + player)
            print_pause("SHOPKEEPER: I guess it's not for you to wear. "
                        "Hmm., a gift perhaps!!")
            print_pause("Note: Now you have 'Dress of a princess' in"
                        " your possession\n")
            shop_items = remove_item(shop_items,
                                     "Dress of a princess     - Worth 500$")
            player_items.append("Dress of a princess")
            gift_items.append("Dress of a princess")
            princess_dress = "not available"
            print_pause("Gold: " + str(gold) + ", - 500")
            gold -= 500
            print_pause("Your Gold: " + str(gold))
        else:
            low_gold_message()
    # if player wants to buy 'steel armour' in the shop
    # by adding / armour == "available" / in the below statement-
    # player cannot buy this item, if the item is already sold out
    elif "armour" in shop_choice and armour == "available":
        if gold >= 400:
            print_pause("SHOPKEEPER: Shiny shiny, makes enemies whiny tiny")
            print_pause("Note: Now you have 'Steel Armour' in your possession")
            shop_items = remove_item(shop_items,
                                     "Glittering steel armour - Worth 400$")
            player_items.append("Glittering steel armour")
            gift_items.append("Glittering steel armour")
            armour = "not available"
            print_pause("Gold: " + str(gold) + ", - 400")
            gold -= 400
            print_pause("Your Gold: " + str(gold))
        else:
            low_gold_message()
    # if player wants to buy 'talisman' in the shop
    elif "talisman" in shop_choice:
        if gold >= 300:
            print_pause("SHOPKEEPER: I see you are in serious "
                        "business " + player)
            print_pause("SHOPKEEPER: Good luck. You are gonna need it.")
            print_pause("Note: You have the talisman, "
                        "it adds fire element to weapons")
            shop_items = remove_item(shop_items,
                                     "A Talisman              - Worth 300$")
            player_items.append("Talisman")
            print_pause("Gold: " + str(gold) + ", - 300")
            gold -= 300
```



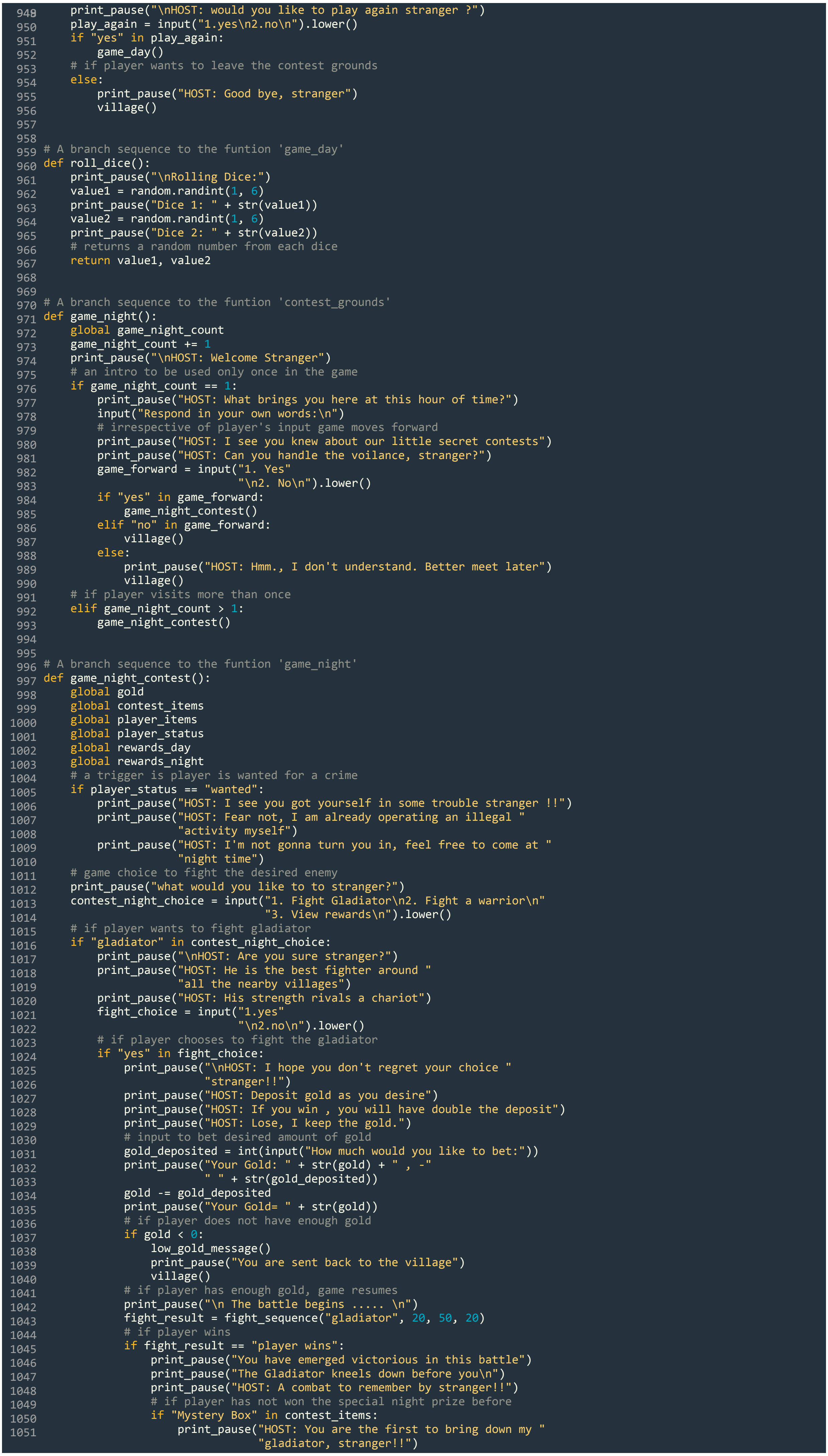
```
            print_pause("Your Gold: " + str(gold))
        else:
            low_gold_message()
    # if player wants to buy 'jewellary' in the shop
    elif "jewellary" in shop_choice:
        if gold >= 400:
            print_pause("SHOPKEEPER: You have a keen eye in arts" + player)
            print_pause("Note: You have the Jewellary in your possession")
            shop_items = remove_item(shop_items,
                                     "Jewellary               - Worth 400$")
            player_items.append("Jewellary")
            gift_items.append("Jewellary")
            print_pause("Gold: " + str(gold) + ", - 400")
            gold -= 400
            print_pause("Your Gold: " + str(gold))
        else:
            low_gold_message()
    # if player wants to buy 'durandal' in the shop
    # by adding / durandal == "available" / in the below statement-
    # player cannot buy this item, if the item is already sold out
    elif "durandal" in shop_choice and durandal == "available":
        if gold >= 600 and "Excaliber" not in player_items:
            print_pause("SHOPKEEPER: A great sword for the true warrior")
            print_pause("SHOPKEEPER: It is said to bring out the inner "
                        "strenth of the warrior from the depths of volcano")
            print_pause("NOTE: You now possess the 'Durandal', a mighty sword")
            player_offense_min = 20
            player_offense_max = 50
            print_pause("NOTE: Your offense has increased "
                        "to: " + str(player_offense_max))
            print_pause("NOTE: You are more stronger now and")
            print_pause("      ready to fight stronger enemies.")
            shop_items = remove_item(shop_items,
                                     "Durandal,a mighty sword - Worth 600$")
            player_items.append("Durandal")
            durandal = "not available"
            print_pause("Gold: " + str(gold) + ", - 600")
            gold -= 600
            print_pause("Your Gold: " + str(gold))
        elif "Excaliber" in player_items:
            print_pause("SHOPKEEPER: You already possess a more "
                        "powerful sword")
            print_pause("SHOPKEEPER: These is no need to buy this sword")
        elif gold < 600:
            low_gold_message()
    # if player wants to buy 'tribute' in the shop
    # by adding / tribute == "available" / in the below statement-
    # player cannot buy this item, if the item is already sold out
    elif "tribute" in shop_choice and tribute == "available":
        # if the player has 'trade contract' in his possession
        if "Trade Contract" in player_items:
            print_pause("SHOPKEEPER: You found my lost 'Trade contract' !!")
            print_pause("SHOPKEEPER: You have my thanks " + player + ".")
            print_pause("SHOPKEEPER: As agreed, I will trade you the "
                        "'Grand Tribute' for the 'Trade contract' of mine.")
            print_pause("Note: You have the 'Grand Tribute' - "
                        "a present worthy of kings, in your possession.")
            shop_items = remove_item(shop_items,
                                     "A grand Tribute         - Not for sale")
            player_items = remove_item(player_items, "Trade Contract")
            player_items.append("A Grand Tribute")
            tribute = "not available"
    # if the player does not have trade contract in his possession
        else:
            print_pause("SHOPKEEPER: Not for sale " + player + "!")
            print_pause("SHOPKEEPER: But I'm willing to trade it ")
            print_pause("            for a personal artifact of mine")
            print_pause("SHOPKEEPER: I lost my trade contract, somewhere"
                        " in the town.")
            print_pause("SHOPKEEPER: It is the pact made by the king himself")
            print_pause("SHOPKEEPER: Find it and the Tribute is yours")
            trade_contract_sequense = "initiated"
    # if player wants to buy 'steel helmet' in the shop
    # by adding / helmet == "available" / in the below statement-
    # player cannot buy this item, if the item is already sold out
    elif "helmet" in shop_choice and helmet == "available":
        if gold >= 200:
            print_pause("SHOPKEEPER: Good choice, better safe than sorry.")
            print_pause("NOTE: You have accquired a 'Steel Helmet'")
            shop_items = remove_item(shop_items,
                                     "Steel Helmet            - Worth 200$")
            player_items.append("Steel Helmet")
            helmet = "not available"
            print_pause("Gold: " + str(gold) + ", - 200")
            gold -= 200
            print_pause("Your Gold: " + str(gold) + "\n")
            print_pause("NOTE: Your defense has been increased")
            print_pause("Defense :" + str(player_defense) + ", + 10")
            player_defense += 10
            print_pause("Player Defense increased to: " + str(player_defense))
        elif gold < 200:
            low_gold_message()
    # if player wants to get back to the village
    elif "village" in shop_choice:
        print_pause("You head back to the village")
        village()
    # if player gives an unrecognized input
    else:
        print_pause("SHOPKEEPER: I don't understand, "
                    "Specify the name of the item clearly")
        shop()
    # After trading, player has to get back to the village
    print_pause("\nYou took care of the business in Traven, as of now "
                "and head back to the village")
```



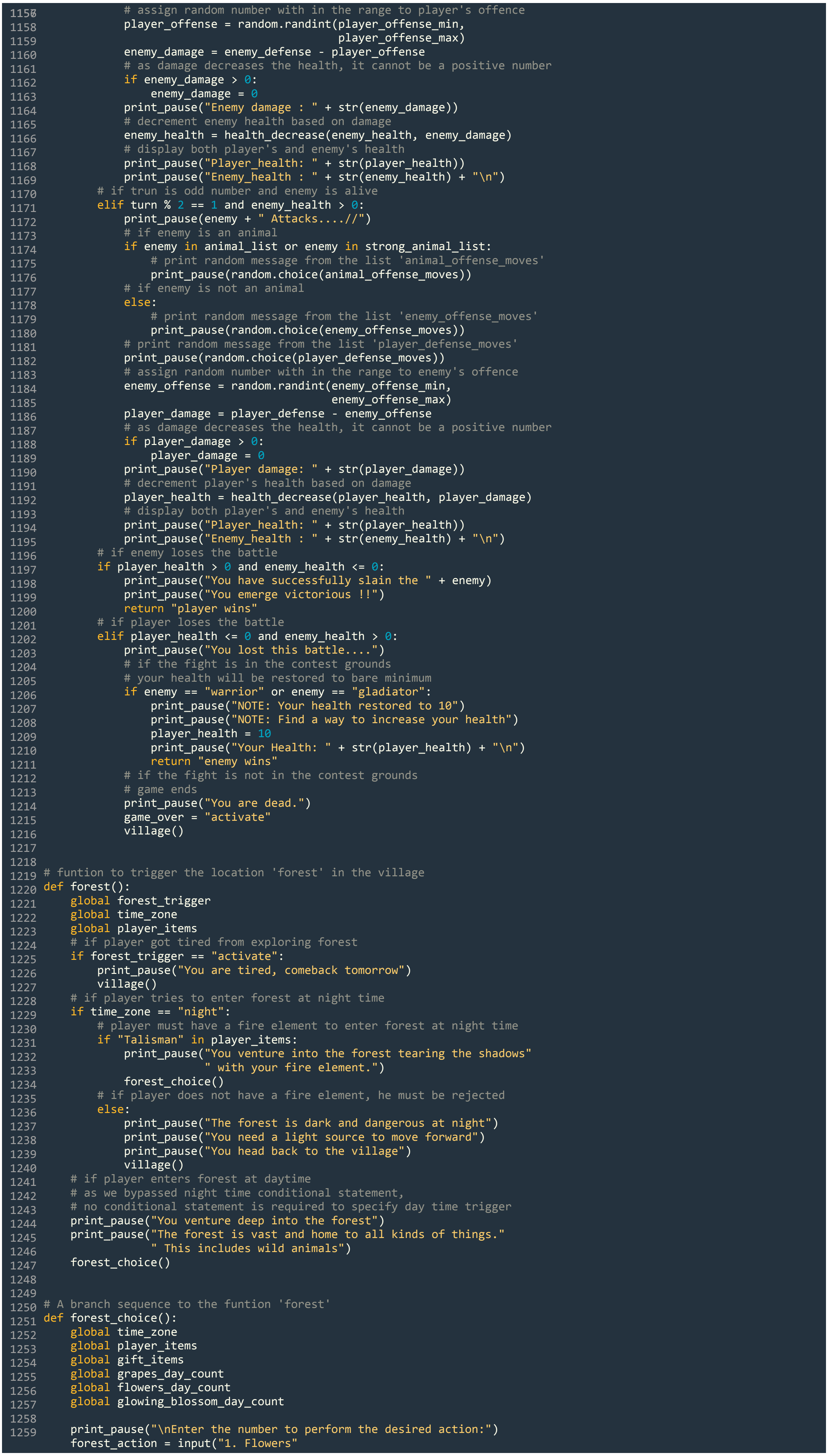
```python
    village()


# a function to run 'inn' sequence
def inn():
    print_pause("This is a place to rest & refresh your body and soul")
    inner_inn()


# A branch sequence to the funtion 'inn'
def inner_inn():
    # player's choice
    print_pause("\nWhat would you like to do:")
    inn_choice = input("1. Go to the Bar"
                       "\n2. Rent a Room"
                       "\n3. Village\n").lower()
    # if player chooses to enter the bar
    if "bar" in inn_choice:
        bar()
    # if player chooses to enter the room
    elif "room" in inn_choice:
        room()
    # if player chooses to enter the village
    elif "village" in inn_choice:
        village()
    # condition to deal with unrecognized input
    else:
        print_pause("Enter a valid choice: Bar? or Room? or village?")
        inner_inn()


# A branch sequence to the funtion 'inner_inn'
def bar():
    global bar_count
    global info_count
    global player_items
    global grapes_available
    global gold
    global time_zone
    print_pause("")
    print_pause("BARTENDER: What do you like to have?")
    bar_choice = input("1. Have a drink "
                       "\n2. Information\n").lower()
    if "drink" in bar_choice:
        bar_count += 1
        # if player has sufficient money and timezone is day, he gets drunk
        if gold >= 10 and time_zone == "day":
            if bar_count == 1:
                print_pause("BARTENDER: Here's the drink you ordered")
                # wine/grapes in bartender's possession will be diminished
                grapes_available = "no"
                drunk()
            elif bar_count > 1 and bar_count < 5:
                # if you have grapes and bartender dorsn't, you get free drink
                if "Grapes" in player_items and grapes_available == "no":
                    print_pause("\nNOTE: Grapes are given to the Bartender")
                    print_pause("BARTENDER: As promised next round is on"
                                " the house")
                    player_items = remove_item(player_items, "Grapes")
                    # grapes in your possession will be ransfered to bartender
                    grapes_available = "yes"
                    drunk()
                elif grapes_available == "yes":
                    # if bartender has grapes, you get drunk
                    drunk()
                else:
                    # if both player and bartender doen't not have grapes
                    # he asks you to go fetch some from the forest
                    print_pause("BARTENDER: My apologies, we are out of wine.")
                    print_pause("BARTENDER: If you are desparate, "
                                "you can run an errand for me.")
                    print_pause("BARTENDER: There are some wild grapes "
                                "growing in the forest")
                    print_pause("BAETENDER: It is dangerous for me to venture "
                                "into forest by myself")
                    print_pause("BARTENDER: Fetch some for me and next round "
                                "will be on the house")
            # After every 5 counts, grapes in bartenders possession will be-
            # diminished and count is reset to 0
            elif bar_count >= 5:
                print_pause("BARTENDER: We are out of wine again!!")
                print_pause("BARTENDER: I'm afraid you gonna have to "
                            "bring some more from the forest")
                grapes_available = "no"
                bar_count = 0
        # at night time, drinks are not served
        elif time_zone == "night":
            print_pause("Bartender is not serving drinks at night")
            print_pause("But you can hear extra gossips from him at night.")
        # if player does not have enough cash
        elif gold < 10:
            print_pause("Bartender: I'm afraid you don't have enough gold")
            print_pause("BARTENDER: Try to earn some gold in contest grounds")
    # if player wants info from the bartender
    elif "info" in bar_choice:
        info_count += 1
        # after getting info for more than 2 times,
        # player is sent back to the room to take rest
        if info_count > 2:
            info_count = 0
            print_pause("BARTENDER: Don't exert yourself, come back later")
            print_pause("BARTENDER: I suggest you take a room and "
                        "rest for the day")
            room()
```



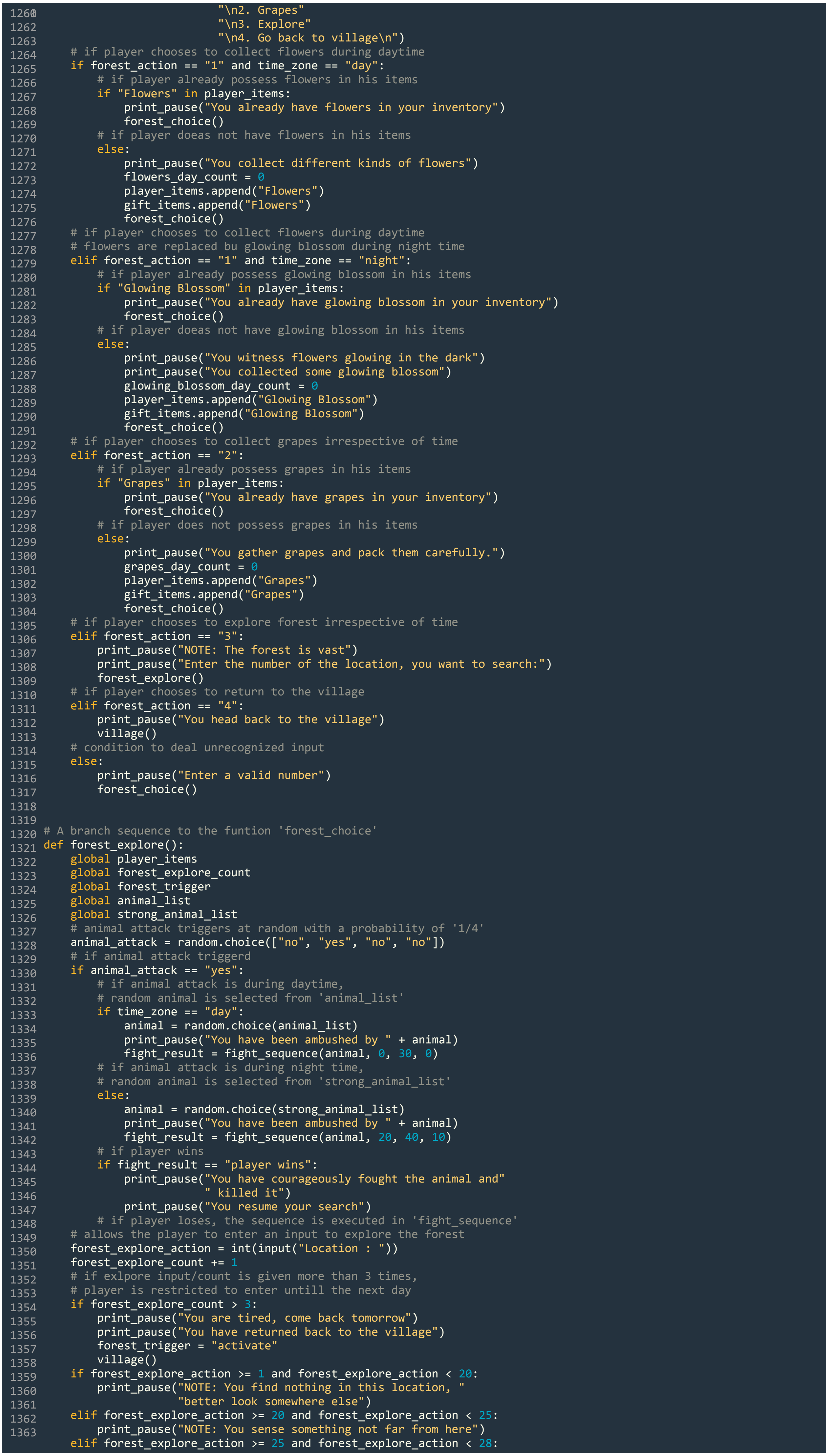
```
        # random info is displayed during daytime
        elif time_zone == "day":
            print_pause(random.choice(bar_info_day))
            bar()
        # random info different from daytime is displayed during night time
        elif time_zone == "night":
            print_pause(random.choice(bar_info_night))
            bar()
    # code to deal with unrecognized input
    else:
        print_pause("BARTENDER: I'm sorry, I don't understand your language")
        bar()
    inner_inn()


# A branch sequence to the funtion 'bar'
def drunk():
    global gold
    print_pause("You get drunk again")
    print_pause("Gold: " + str(gold) + ", -10")
    gold -= 10
    print_pause("Your Gold: " + str(gold))
    print_pause("You have no choice but to take a room\n")
    room()


# A branch sequence to the funtion 'inner_inn'
def room():
    global gold
    print_pause("\nYou enterd the room all tired and depraved of sleep")
    print_pause("You were charged 20 gold for the room")
    print_pause("Gold: " + str(gold) + ", -20")
    gold -= 20
    print_pause("Your Gold: " + str(gold))
    room_sequense()


# A branch sequence to the funtion 'room'
def room_sequense():
    global player_items
    global trade_contract_sequense
    print_pause("\nWhat would yo do now?")
    room_action = input("1. Crash on the bed"
                        "\n2. Examine the room\n").lower()
    # if player wants to sleep on bed
    if "bed" in room_action:
        print_pause("\nIt is time to rest your body")
        room_choice()
    # if player wants to examine the room
    elif "examine" in room_action:
        print_pause("\nYou examine the room throughly for any "
                    "possible anamoly")
        # if trade contract sequence initiated by the shopkeeper
        if trade_contract_sequense == "initiated":
            print_pause("You find an old article eating dust behind the "
                        "closet")
            print_pause("You read it, out of curiosity")
            print_pause("You have found some sort of 'Trade Contract'")
            player_items.append("Trade Contract")
            print_pause("\nIt is time to rest your body")
            trade_contract_sequense = "end"
            room_choice()
        # if trade contract sequence initiated by the shopkeeper
        else:
            print_pause("You find nothing abnormal")
            print_pause("It is time to rest your body")
            room_choice()
    # condition to deal with unrecognized input
    else:
        print_pause("Enter a valid action in words")
        room_sequense()


# A branch sequence to the funtion 'room_sequense'
def room_choice():
    global time_zone
    global flowers_day_count
    global grapes_day_count
    global glowing_blossom_day_count
    global keyword_count
    global forest_trigger
    global player_items
    global gift_items
    global bell

    print_pause("\nHow long do you need to sleep?")
    sleep_till = input("1. Wake up at night"
                       "\n2. Wake up at morning\n").lower()
    # if player wants to sleep till night and timezone is day
    if "night" in sleep_till and time_zone == "day":
        print_pause("You wake up at night")
        time_zone = "night"
        # increse player's health by 25 after some rest
        print_pause("Your health increased by 25")
        print_pause("Health : " + str(player_health) + ", + 25")
        health_increase(25)
        print_pause("Your Health: " + str(player_health))
        print_pause("\nsome events are different at night, go check them out")
        print_pause("You come down to the inn entrance")
        inn()
    # if player wants to sleep till night and timezone is already night
    if "night" in sleep_till and time_zone == "night":
        print_pause("Its already night time")
        room_choice()
    # if player wants to sleep till morning of next day
    elif "morning" in sleep_till:
        keyword_count = 0
        forest_trigger = "deactivate"
        flowers_day_count += 1
        grapes_day_count += 1
        bell = "deactivated"
        # if player has flowers in his possession for more than 2 days
        if flowers_day_count > 2:
            print_pause("\nNOTE: Flowers in your possession are dried up.")
            print_pause("NOTE: They are no longer useful")
            player_items = remove_item(player_items, "Flowers")
            gift_items = remove_item(player_items, "Flowers")
        # if player has glowing blossom in his possession for more than 2 days
        if glowing_blossom_day_count > 2:
            print_pause("\nNOTE: Glowing Blossom in your possession "
                        "are dried up.")
            print_pause("NOTE: They are no longer useful")
            player_items = remove_item(player_items, "Glowing Blossom")
            gift_items = remove_item(player_items, "Glowing Blossom")
        # if player has grapes in his possession for more than 2 days
        if grapes_day_count > 2:
            print_pause("\nNOTE: Grapes in your possession have gone bad")
            print_pause("NOTE: They are no longer useful.")
            player_items = remove_item(player_items, "Grapes")
            gift_items = remove_item(player_items, "Grapes")
        print_pause("\nYou wake Up in the morning fresh to resume "
                    "your journey")
        time_zone = "day"
        # increse player's health by 50 after plenty of rest
        print_pause("Your health increased by 50")
        print_pause("Health : " + str(player_health) + ", + 50")
        health_increase(50)
        print_pause("Your Health: " + str(player_health))
        print_pause("\nYou head back to the Inn entrance")
        inn()
    # condition to deal with unrecognized input
    else:
        print_pause("Enter a valid time zone. Night? or Morning?")
        room_choice()


# funtion to trigger the location 'contest grounds' in the village
def contest_grounds():
    global time_zone
    global contest_day_count
    global contest_night_count
    # if player visits 'contest grounds' during daytime
    if time_zone == "day":
        contest_day_count += 1
        # intro to be used only once in the game
        if contest_day_count == 1:
            print_pause("surpricingly, this country helds contests "
                        "in gambling")
            print_pause("People of this country prefers peace over voilance")
        # if player visits more than once
        if contest_day_count >= 1:
            print_pause("Would you like to take part in the gambling?")
            contest_choice_day = input("1. Yes"
                                       "\n2. No\n").lower()
            if "yes" in contest_choice_day:
                game_day()
            elif "no" in contest_choice_day:
                print_pause("HOST: Farewell, stranger.")
                print_pause("HOST: Do come back another time")
                village()
            # condition to handle unrecognized inputs from the player
            else:
                print_pause("HOST: Make up your mind stranger. 'yes' "
                            "or 'no' ?")
                contest_grounds()
    # if player visits 'contest grounds' during night time
    elif time_zone == "night":
        contest_night_count += 1
        # intro to be used only once in the game
        if contest_night_count == 1:
            print_pause("Enen thought voilance is prohibited in face "
                        "the of public")
            print_pause("Under the clock of shadows, the Arena conducts"
                        " battles")
            print_pause("Would you dare to confront the person "
                        "reponsible for the fights at night time?")
        # if player visits more than once
        if contest_night_count > 1:
            print_pause("Would you dare to take part in the fights?")
        # take input from the player
        contest_choice_night = input("1. Yes"
                                     "\n2. No\n").lower()
        if "yes" in contest_choice_night:
            game_night()
        elif "no" in contest_choice_night:
            print_pause("Perhaps you are not ready yet")
            print_pause("Do come back another time....")
            village()
        # condition to handle unrecognized inputs from the player
        else:
            print_pause("HOST: Make up your mind stranger. 'yes' or 'no' ?")
            contest_grounds()


# A branch sequence to the funtion 'contest_grounds'
def game_day():
    global gold
    global contest_items
    global rewards_day
    global contest_items
    global player_items
    global player_status
    # a trigger if the player is wanted
    if player_status == "wanted":
        print_pause("Leave stranger, you are gonna get me in trouble")
        print_pause("You are forced to leave the contest grounds")
        print_pause("You head back to the village")
        village()
    # choice to play different games
    print_pause("\nHOST: welcome stranger, what would you like to play?")
    contest_day_choice = input("1. Evens & Odds"
                               "\n2. Double or nothing"
                               "\n3. View rewards\n").lower()
    # if player choose 'evens & odds' game
    if "even" in contest_day_choice:
        if gold >= 10:
            print_pause("HOST: Deposit 10 gold to play the game")
            print_pause("Your Gold: " + str(gold) + " , -10")
            gold -= 10
            print_pause("Your Gold: " + str(gold))
            value1, value2 = roll_dice()
            if (value1+value2) % 2 == 0:
                value = "even"
            else:
                value = "odd"

            print_pause(str(value1) + "+" + str(value2) + " "
                        "= " + str(value1 + value2) + ", " + value)

            if value == "even":
                print_pause("HOST: Evens it is..., Congratulations stranger")
                print_pause("HOST: You have won 50 gold")
                print_pause("Your Gold: " + str(gold) + " , +50")
                gold += 50
                print_pause("Your Gold: " + str(gold))
            else:
                print_pause("HOST: Odds are against you stranger.., "
                            "you lost your deposit")
        # if player does not have enough gold
        else:
            low_gold_message()
            village()
    # if player choose 'double or nothing' game
    elif "double" in contest_day_choice:
        print_pause("\nHOST: Deposit gold as you desire")
        print_pause("HOST: If both dices are evens or odd , you win")
        print_pause("HOST: If you win , you will have double the deposit")
        print_pause("HOST: Lose, I keep the gold.")
        # if player has not win the special prize already
        if "Unknown Key" in contest_items:
            print_pause("Also I have a special prize waiting to be won.")
        # take a desired amount of bet from the player's input
        gold_deposited = int(input("How much would you like to bet:"))
        print_pause("Your Gold: " + str(gold) + " , - " + str(gold_deposited))
        gold -= gold_deposited
        if gold >= 0:
            print_pause("Your Gold= " + str(gold))
            value1, value2 = roll_dice()
            print_pause("Dices : " + str(value1) + "," + str(value2))
            # if player wins
            if (value1 % 2) == (value2 % 2):
                # if player has not win the special prize already
                if "Unknown Key" in contest_items:
                    print_pause("HOST: Marvelous stranger!, it's your"
                                " lucky day")
                    print_pause("HOST: I have just the rare item as the prize")
                    print_pause("Note: You have received an 'Unknown Key'.")
                    contest_items = remove_item(contest_items, "Unknown Key")
                    player_items.append("Unknown Key")
                    rewards_day = remove_item(rewards_day,
                                              "Special Prize: Unknown Key")
                    print_pause("Host: I'll keep the deposit just this once")
                    print_pause("HOST: consider it an even trade for the "
                                "rare item")
                # if player has already won the special prize
                else:
                    print_pause("HOST: Congratulation, stranger! you won "
                                "this game")
                    gold_deposited = 2*gold_deposited
                    print_pause("Your Gold: " + str(gold) + " , "
                                "+ " + str(gold_deposited))
                    gold += gold_deposited
                    print_pause("Your Gold: " + str(gold))
            # if player loses
            else:
                print_pause("Host: It is not as easy as it sounds stranger!!")
                print_pause("HOST: I get to keep the gold")
        # if player does not have enough gold
        else:
            low_gold_message()
            gold += gold_deposited
            print_pause("Your Gold= " + str(gold))
            village()
    # if player wants to check the rewards before playing the game
    elif "reward" in contest_day_choice:
        print_pause(list_serial(rewards_day))
        game_day()
    # condition to deal unrecognized input
    else:
        print_pause("HOST: Say the correct words stranger")
        game_day()
    # choice to play the game again
```



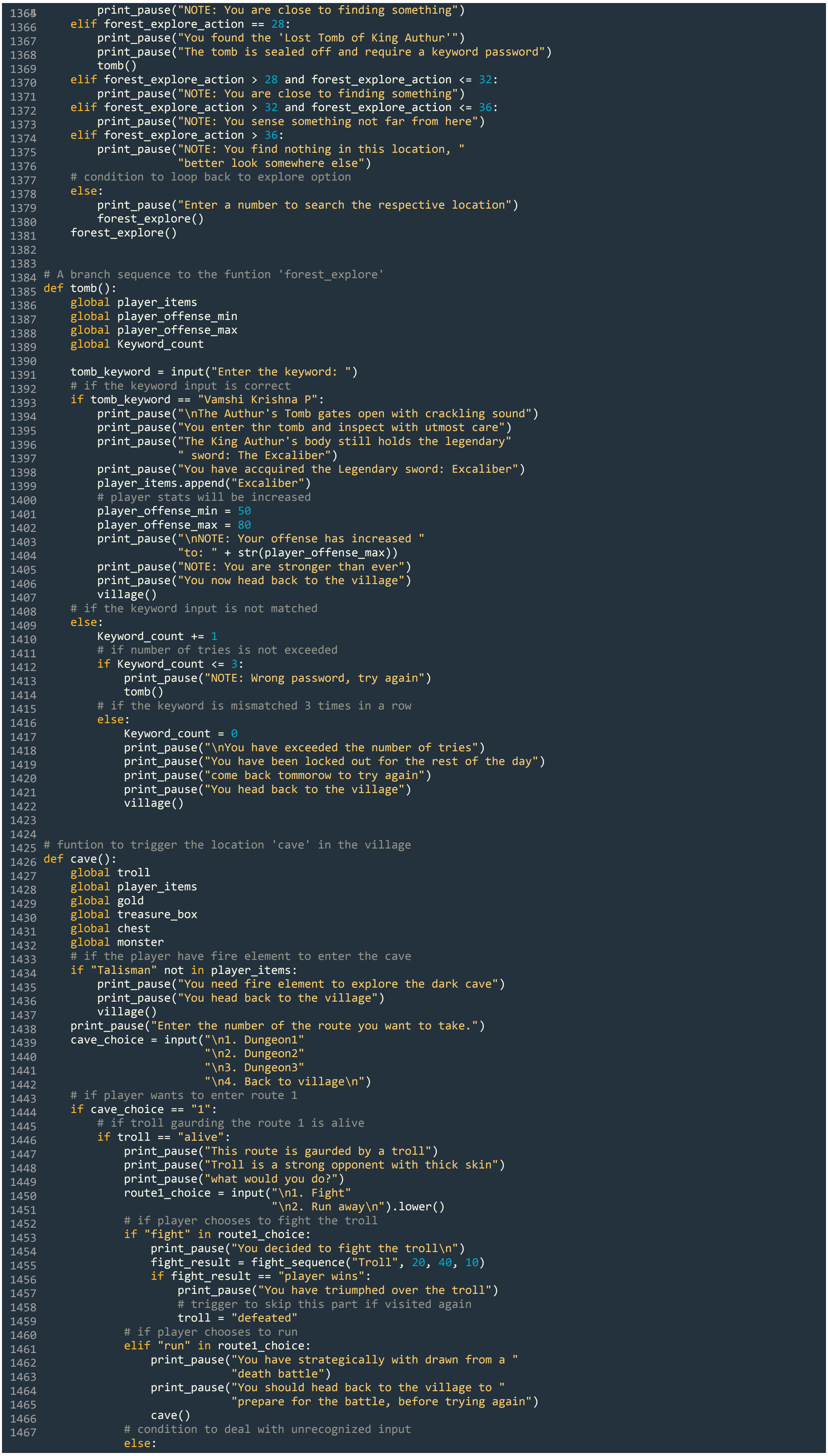
```
    print_pause("\nHOST: would you like to play again stranger ?")
    play_again = input("1.yes\n2.no\n").lower()
    if "yes" in play_again:
        game_day()
    # if player wants to leave the contest grounds
    else:
        print_pause("HOST: Good bye, stranger")
        village()


# A branch sequence to the funtion 'game_day'
def roll_dice():
    print_pause("\nRolling Dice:")
    value1 = random.randint(1, 6)
    print_pause("Dice 1: " + str(value1))
    value2 = random.randint(1, 6)
    print_pause("Dice 2: " + str(value2))
    # returns a random number from each dice
    return value1, value2


# A branch sequence to the funtion 'contest_grounds'
def game_night():
    global game_night_count
    game_night_count += 1
    print_pause("\nHOST: Welcome Stranger")
    # an intro to be used only once in the game
    if game_night_count == 1:
        print_pause("HOST: What brings you here at this hour of time?")
        input("Respond in your own words:\n")
        # irrespective of player's input game moves forward
        print_pause("HOST: I see you knew about our little secret contests")
        print_pause("HOST: Can you handle the voilance, stranger?")
        game_forward = input("1. Yes"
                             "\n2. No\n").lower()
        if "yes" in game_forward:
            game_night_contest()
        elif "no" in game_forward:
            village()
        else:
            print_pause("HOST: Hmm., I don't understand. Better meet later")
            village()
    # if player visits more than once
    elif game_night_count > 1:
        game_night_contest()


# A branch sequence to the funtion 'game_night'
def game_night_contest():
    global gold
    global contest_items
    global player_items
    global player_status
    global rewards_day
    global rewards_night
    # a trigger is player is wanted for a crime
    if player_status == "wanted":
        print_pause("HOST: I see you got yourself in some trouble stranger !!")
        print_pause("HOST: Fear not, I am already operating an illegal "
                    "activity myself")
        print_pause("HOST: I'm not gonna turn you in, feel free to come at "
                    "night time")
    # game choice to fight the desired enemy
    print_pause("what would you like to to stranger?")
    contest_night_choice = input("1. Fight Gladiator\n2. Fight a warrior\n"
                                 "3. View rewards\n").lower()
    # if player wants to fight gladiator
    if "gladiator" in contest_night_choice:
        print_pause("\nHOST: Are you sure stranger?")
        print_pause("HOST: He is the best fighter around "
                    "all the nearby villages")
        print_pause("HOST: His strength rivals a chariot")
        fight_choice = input("1.yes"
                             "\n2.no\n").lower()
        # if player chooses to fight the gladiator
        if "yes" in fight_choice:
            print_pause("\nHOST: I hope you don't regret your choice "
                        "stranger!!")
            print_pause("HOST: Deposit gold as you desire")
            print_pause("HOST: If you win , you will have double the deposit")
            print_pause("HOST: Lose, I keep the gold.")
            # input to bet desired amount of gold
            gold_deposited = int(input("How much would you like to bet:"))
            print_pause("Your Gold: " + str(gold) + " , -"
                        " " + str(gold_deposited))
            gold -= gold_deposited
            print_pause("Your Gold= " + str(gold))
            # if player does not have enough gold
            if gold < 0:
                low_gold_message()
                print_pause("You are sent back to the village")
                village()
            # if player has enough gold, game resumes
            print_pause("\n The battle begins ..... \n")
            fight_result = fight_sequence("gladiator", 20, 50, 20)
            # if player wins
            if fight_result == "player wins":
                print_pause("You have emerged victorious in this battle")
                print_pause("The Gladiator kneels down before you\n")
                print_pause("HOST: A combat to remember by stranger!!")
                # if player has not won the special night prize before
                if "Mystery Box" in contest_items:
                    print_pause("HOST: You are the first to bring down my "
                                "gladiator, stranger!!")
                    print_pause("HOST: Let me reward you with the best "
                                "prize in my possession")
                    contest_items = remove_item(contest_items, "Mystery Box")
                    player_items.append("Mystery Box")
                    rewards_day = remove_item(rewards_day, "Special Prize"
                                              ": Mystery Box")
                    print_pause("Note: You have received a 'Mystery Box'.")
                # if player has won the special night prize already
                else:
                    print_pause("HOST: Congratulation, stranger! you won "
                                "this battle")
                    print_pause("HOST: You earn the right to call yourself"
                                "- 'The strongest among the village'")
                    gold_deposited = 2*gold_deposited
                    print_pause("Your Gold: " + str(gold) + " , +"
                                " " + str(gold_deposited))
                    gold += gold_deposited
                    print_pause("Your Gold: " + str(gold))
            # if player loses
            else:
                print_pause("Host: It is no simple task to defeat "
                            " the gladiator stranger!!")
        # if player chooses to fight the gladiator
        elif "no" in fight_choice:
            print_pause("HOST: There is no shame in retreat, stranger")
            game_night_contest()
        # condition to deal unrecognized input
        else:
            print_pause("HOST: I don't understand your words stranger")
            game_night_contest()
    # if player wants to fight the warrior
    elif "warrior" in contest_night_choice:
        # a special situation for the player to raise gold if broken completely
        print_pause("\nHOST: You don't have to Deposit any gold to "
                    "play the game")
        print_pause("HOST: You get 50 gold on win and nothing to lose")
        print_pause("Your Gold: " + str(gold))
        print_pause("\n The battle begins ..... \n")
        fight_result = fight_sequence("warrior", 10, 30, 10)
        # if player wins
        if fight_result == "player wins":
            print_pause("HOST: Well done stranger!")
            print_pause("HOST: You are tougher than you look")
            print_pause("HOST: Perhaps you might have a chance to "
                        "defeat our gladiator")
            print_pause("HOST: You have won 50 gold")
            print_pause("Your Gold: " + str(gold) + " , +50")
            gold += 50
            print_pause("Your Gold: " + str(gold))
        # if player loses
        else:
            print_pause("HOST: Arena is merciless stranger.., "
                        "you lost your deposit")
    # if player wants to check the reward before fighting
    elif "reward" in contest_night_choice:
        print_pause(list_serial(rewards_night))
        game_night_contest()
    # condition to deal unrecognized input
    else:
        print_pause("HOST: Say the correct words stranger")
        game_night_contest()
    # trigger to let the player compete again
    print_pause("HOST: would you like to fight again stranger ?")
    play_again = input("1.yes\n2.no\n").lower()
    if "yes" in play_again:
        game_night_contest()
    else:
        print_pause("HOST: Good bye, stranger")
        village()


# A branch sequence to the funtion 'game_night_contest'
def fight_sequence(enemy, enemy_offense_min, enemy_offense_max, enemy_defense):
    global player_health
    global player_offense_min
    global player_offense_max
    global player_defense
    global player_offense_moves
    global player_defense_moves
    global enemy_offense_moves
    global enemy_defense_moves
    global animal_offense_moves
    global animal_defense_moves
    global rewards_night
    global animal_list
    global strong_animal_list
    global game_over

    enemy_health = 100
    # a loop to let player and enemy deal random damage in each turn
    for turn in range(40):
        # if trun is even number and player is alive
        if turn % 2 == 0 and player_health > 0:
            print_pause("Player Attacks....//")
            # print random message from the list 'player_offense_moves'
            print_pause(random.choice(player_offense_moves))
            # if enemy is an animal
            if enemy in animal_list or enemy in strong_animal_list:
                # print random message from the list 'animal_defense_moves'
                print_pause(random.choice(animal_defense_moves))
            # if enemy is not an animal
            else:
                # print random message from the list 'enemy_defense_moves'
                print_pause(random.choice(enemy_defense_moves))
```



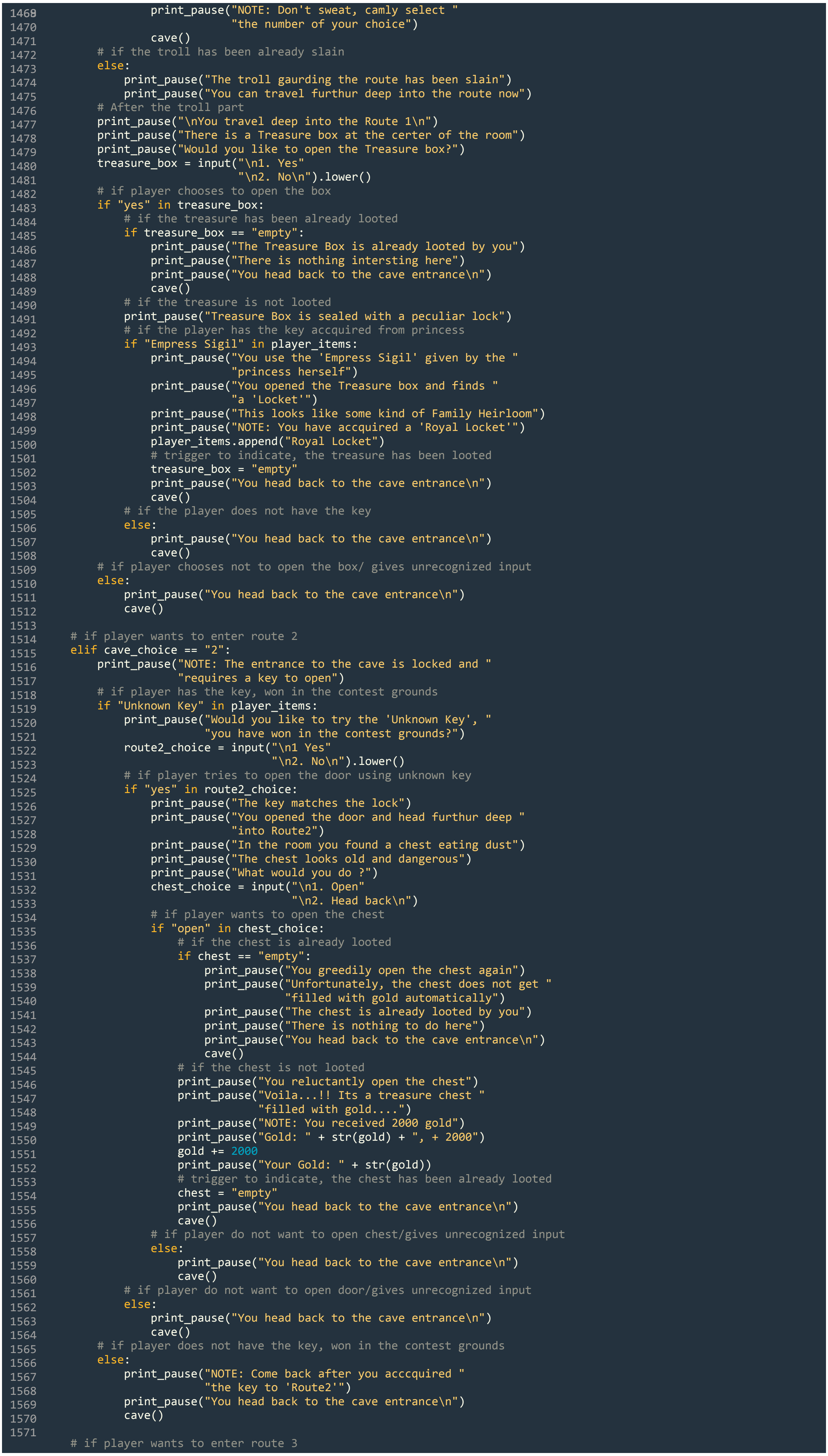
```
            # assign random number with in the range to player's offence
            player_offense = random.randint(player_offense_min,
                                            player_offense_max)
            enemy_damage = enemy_defense - player_offense
            # as damage decreases the health, it cannot be a positive number
            if enemy_damage > 0:
                enemy_damage = 0
            print_pause("Enemy damage : " + str(enemy_damage))
            # decrement enemy health based on damage
            enemy_health = health_decrease(enemy_health, enemy_damage)
            # display both player's and enemy's health
            print_pause("Player health: " + str(player_health))
            print_pause("Enemy health : " + str(enemy_health) + "\n")
        # if trun is odd number and enemy is alive
        elif turn % 2 == 1 and enemy_health > 0:
            print_pause(enemy + " Attacks....//")
            # if enemy is an animal
            if enemy in animal_list or enemy in strong_animal_list:
                # print random message from the list 'animal_offense_moves'
                print_pause(random.choice(animal_offense_moves))
            # if enemy is not an animal
            else:
                # print random message from the list 'enemy_offense_moves'
                print_pause(random.choice(enemy_offense_moves))
            # print random message from the list 'player_defense_moves'
            print_pause(random.choice(player_defense_moves))
            # assign random number with in the range to enemy's offence
            enemy_offense = random.randint(enemy_offense_min,
                                           enemy_offense_max)
            player_damage = player_defense - enemy_offense
            # as damage decreases the health, it cannot be a positive number
            if player_damage > 0:
                player_damage = 0
            print_pause("Player damage: " + str(player_damage))
            # decrement player's health based on damage
            player_health = health_decrease(player_health, player_damage)
            # display both player's and enemy's health
            print_pause("Player health: " + str(player_health))
            print_pause("Enemy health : " + str(enemy_health) + "\n")
        # if enemy loses the battle
        if player_health > 0 and enemy_health <= 0:
            print_pause("You have successfully slain the " + enemy)
            print_pause("You emerge victorious !!")
            return "player wins"
        # if player loses the battle
        elif player_health <= 0 and enemy_health > 0:
            print_pause("You lost this battle....")
            # if the fight is in the contest grounds
            # your health will be restored to bare minimum
            if enemy == "warrior" or enemy == "gladiator":
                print_pause("NOTE: Your health restored to 10")
                print_pause("NOTE: Find a way to increase your health")
                player_health = 10
                print_pause("Your Health: " + str(player_health) + "\n")
                return "enemy wins"
            # if the fight is not in the contest grounds
            # game ends
            print_pause("You are dead.")
            game_over = "activate"
            village()


# funtion to trigger the location 'forest' in the village
def forest():
    global forest_trigger
    global time_zone
    global player_items
    # if player got tired from exploring forest
    if forest_trigger == "activate":
        print_pause("You are tired, comeback tomorrow")
        village()
    # if player tries to enter forest at night time
    if time_zone == "night":
        # player must have a fire element to enter forest at night time
        if "Talisman" in player_items:
            print_pause("You venture into the forest tearing the shadows"
                        " with your fire element.")
            forest_choice()
        # if player does not have a fire element, he must be rejected
        else:
            print_pause("The forest is dark and dangerous at night")
            print_pause("You need a light source to move forward")
            print_pause("You head back to the village")
            village()
    # if player enters forest at daytime
    # as we bypassed night time conditional statement,
    # no conditional statement is required to specify day time trigger
    print_pause("You venture deep into the forest")
    print_pause("The forest is vast and home to all kinds of things."
                " This includes wild animals")
    forest_choice()


# A branch sequence to the funtion 'forest'
def forest_choice():
    global time_zone
    global player_items
    global gift_items
    global grapes_day_count
    global flowers_day_count
    global glowing_blossom_day_count

    print_pause("\nEnter the number to perform the desired action:")
    forest_action = input("1. Flowers"
```



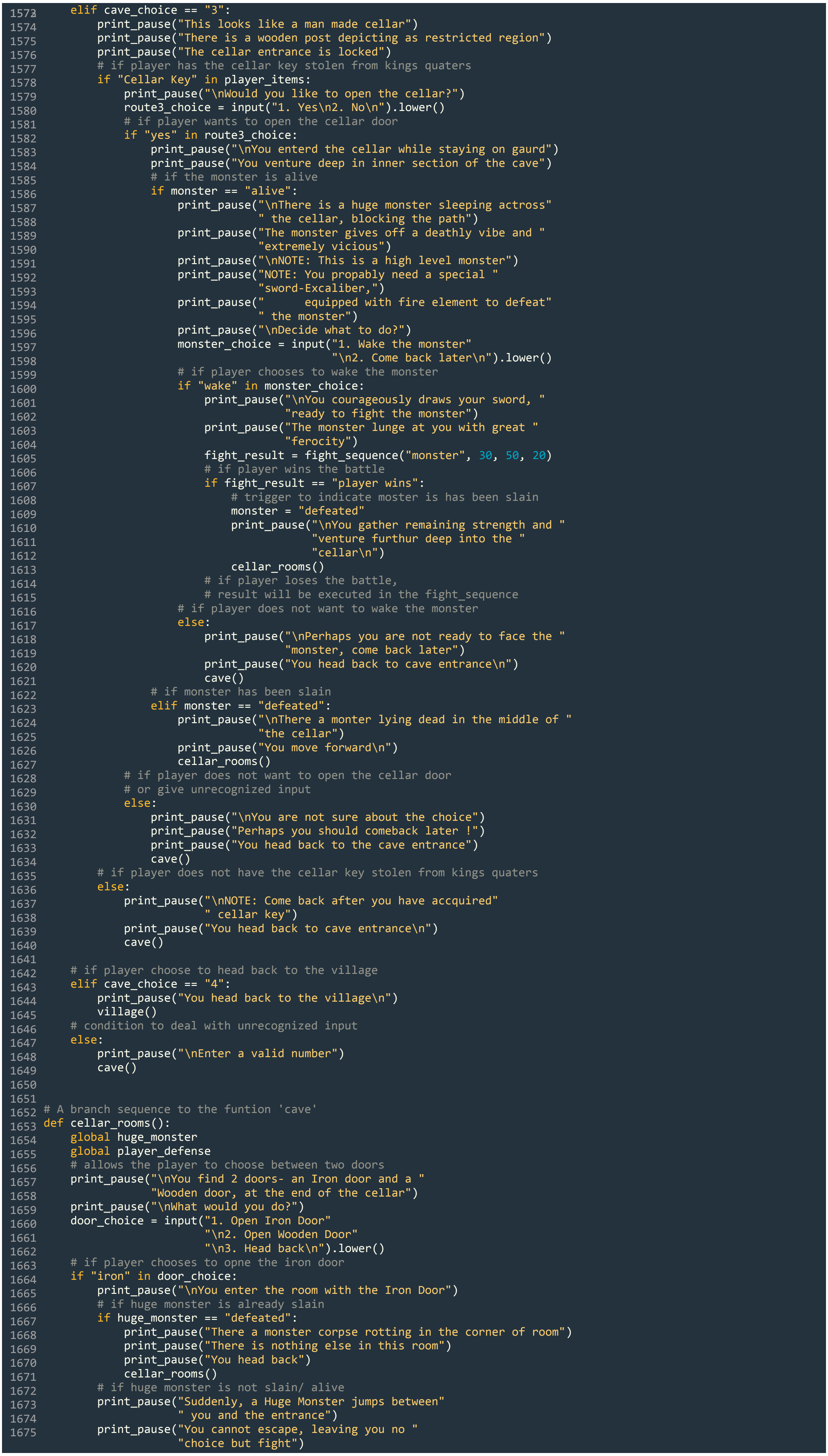
```
                          "\n2. Grapes"
                          "\n3. Explore"
                          "\n4. Go back to village\n")
    # if player chooses to collect flowers during daytime
    if forest_action == "1" and time_zone == "day":
        # if player already possess flowers in his items
        if "Flowers" in player_items:
            print_pause("You already have flowers in your inventory")
            forest_choice()
        # if player doeas not have flowers in his items
        else:
            print_pause("You collect different kinds of flowers")
            flowers_day_count = 0
            player_items.append("Flowers")
            gift_items.append("Flowers")
            forest_choice()
    # if player chooses to collect flowers during daytime
    # flowers are replaced bu glowing blossom during night time
    elif forest_action == "1" and time_zone == "night":
        # if player already possess glowing blossom in his items
        if "Glowing Blossom" in player_items:
            print_pause("You already have glowing blossom in your inventory")
            forest_choice()
        # if player doeas not have glowing blossom in his items
        else:
            print_pause("You witness flowers glowing in the dark")
            print_pause("You collected some glowing blossom")
            glowing_blossom_day_count = 0
            player_items.append("Glowing Blossom")
            gift_items.append("Glowing Blossom")
            forest_choice()
    # if player chooses to collect grapes irrespective of time
    elif forest_action == "2":
        # if player already possess grapes in his items
        if "Grapes" in player_items:
            print_pause("You already have grapes in your inventory")
            forest_choice()
        # if player does not possess grapes in his items
        else:
            print_pause("You gather grapes and pack them carefully.")
            grapes_day_count = 0
            player_items.append("Grapes")
            gift_items.append("Grapes")
            forest_choice()
    # if player chooses to explore forest irrespective of time
    elif forest_action == "3":
        print_pause("NOTE: The forest is vast")
        print_pause("Enter the number of the location, you want to search:")
        forest_explore()
    # if player chooses to return to the village
    elif forest_action == "4":
        print_pause("You head back to the village")
        village()
    # condition to deal unrecognized input
    else:
        print_pause("Enter a valid number")
        forest_choice()


# A branch sequence to the funtion 'forest_choice'
def forest_explore():
    global player_items
    global forest_explore_count
    global forest_trigger
    global animal_list
    global strong_animal_list
    # animal attack triggers at random with a probability of '1/4'
    animal_attack = random.choice(["no", "yes", "no", "no"])
    # if animal attack triggerd
    if animal_attack == "yes":
        # if animal attack is during daytime,
        # random animal is selected from 'animal_list'
        if time_zone == "day":
            animal = random.choice(animal_list)
            print_pause("You have been ambushed by " + animal)
            fight_result = fight_sequence(animal, 0, 30, 0)
        # if animal attack is during night time,
        # random animal is selected from 'strong_animal_list'
        else:
            animal = random.choice(strong_animal_list)
            print_pause("You have been ambushed by " + animal)
            fight_result = fight_sequence(animal, 20, 40, 10)
        # if player wins
        if fight_result == "player wins":
            print_pause("You have courageously fought the animal and"
                        " killed it")
            print_pause("You resume your search")
        # if player loses, the sequence is executed in 'fight_sequence'
    # allows the player to enter an input to explore the forest
    forest_explore_action = int(input("Location : "))
    forest_explore_count += 1
    # if exlpore input/count is given more than 3 times,
    # player is restricted to enter untill the next day
    if forest_explore_count > 3:
        print_pause("You are tired, come back tomorrow")
        print_pause("You have returned back to the village")
        forest_trigger = "activate"
        village()
    if forest_explore_action >= 1 and forest_explore_action < 20:
        print_pause("NOTE: You find nothing in this location, "
                    "better look somewhere else")
    elif forest_explore_action >= 20 and forest_explore_action < 25:
        print_pause("NOTE: You sense something not far from here")
    elif forest_explore_action >= 25 and forest_explore_action < 28:
```



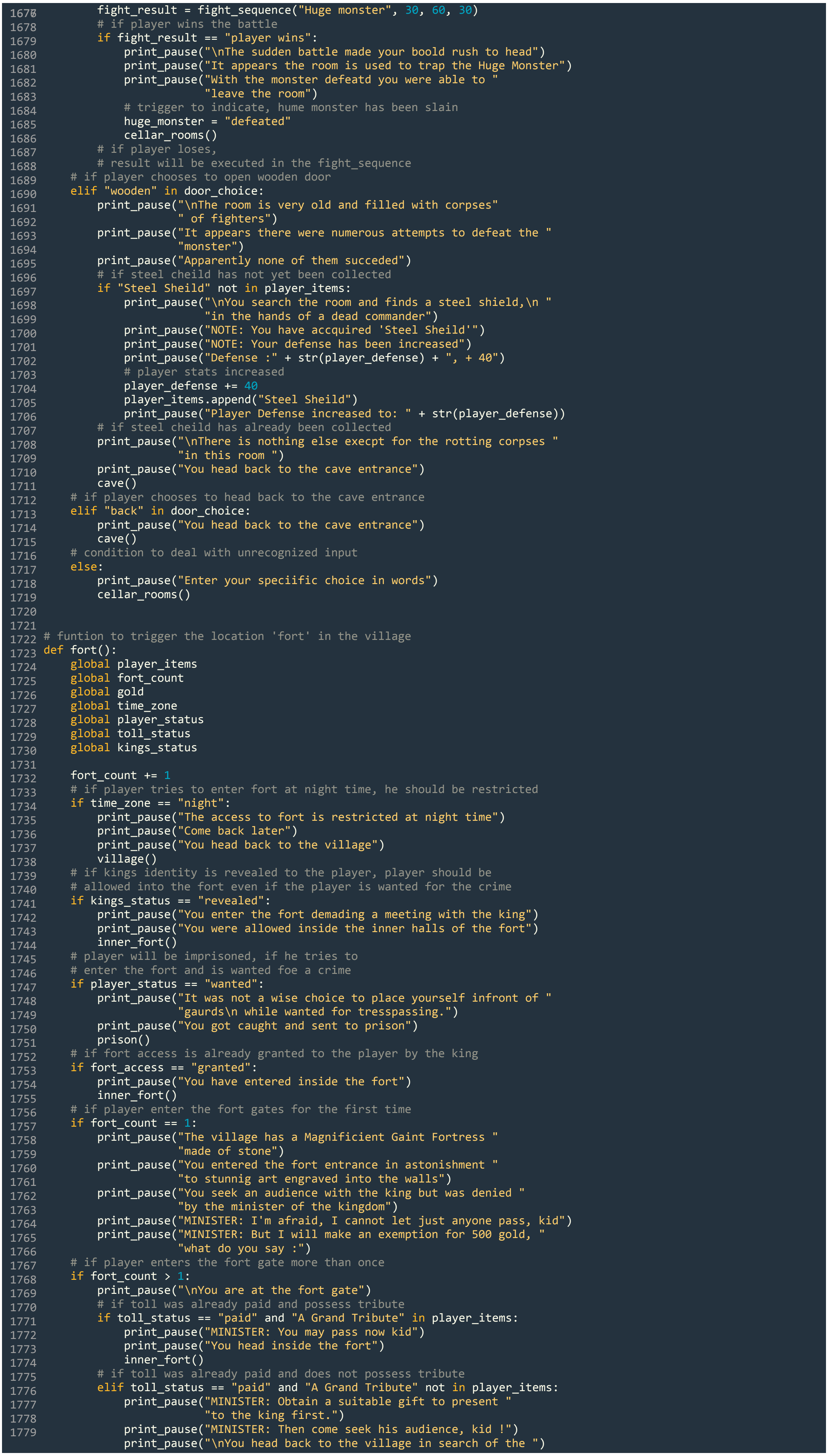
```
        print_pause("NOTE: You are close to finding something")
    elif forest_explore_action == 28:
        print_pause("You found the 'Lost Tomb of King Authur'")
        print_pause("The tomb is sealed off and require a keyword password")
        tomb()
    elif forest_explore_action > 28 and forest_explore_action <= 32:
        print_pause("NOTE: You are close to finding something")
    elif forest_explore_action > 32 and forest_explore_action <= 36:
        print_pause("NOTE: You sense something not far from here")
    elif forest_explore_action > 36:
        print_pause("NOTE: You find nothing in this location, "
                    "better look somewhere else")
    # condition to loop back to explore option
    else:
        print_pause("Enter a number to search the respective location")
        forest_explore()
    forest_explore()


# A branch sequence to the funtion 'forest_explore'
def tomb():
    global player_items
    global player_offense_min
    global player_offense_max
    global Keyword_count

    tomb_keyword = input("Enter the keyword: ")
    # if the keyword input is correct
    if tomb_keyword == "Vamshi Krishna P":
        print_pause("\nThe Authur's Tomb gates open with crackling sound")
        print_pause("You enter thr tomb and inspect with utmost care")
        print_pause("The King Authur's body still holds the legendary"
                    " sword: The Excaliber")
        print_pause("You have accquired the Legendary sword: Excaliber")
        player_items.append("Excaliber")
        # player stats will be increased
        player_offense_min = 50
        player_offense_max = 80
        print_pause("\nNOTE: Your offense has increased "
                    "to: " + str(player_offense_max))
        print_pause("NOTE: You are stronger than ever")
        print_pause("You now head back to the village")
        village()
    # if the keyword input is not matched
    else:
        Keyword_count += 1
        # if number of tries is not exceeded
        if Keyword_count <= 3:
            print_pause("NOTE: Wrong password, try again")
            tomb()
        # if the keyword is mismatched 3 times in a row
        else:
            Keyword_count = 0
            print_pause("\nYou have exceeded the number of tries")
            print_pause("You have been locked out for the rest of the day")
            print_pause("come back tommorow to try again")
            print_pause("You head back to the village")
            village()


# funtion to trigger the location 'cave' in the village
def cave():
    global troll
    global player_items
    global gold
    global treasure_box
    global chest
    global monster
    # if the player have fire element to enter the cave
    if "Talisman" not in player_items:
        print_pause("You need fire element to explore the dark cave")
        print_pause("You head back to the village")
        village()
    print_pause("Enter the number of the route you want to take.")
    cave_choice = input("\n1. Dungeon1"
                        "\n2. Dungeon2"
                        "\n3. Dungeon3"
                        "\n4. Back to village\n")
    # if player wants to enter route 1
    if cave_choice == "1":
        # if troll gaurding the route 1 is alive
        if troll == "alive":
            print_pause("This route is gaurded by a troll")
            print_pause("Troll is a strong opponent with thick skin")
            print_pause("what would you do?")
            route1_choice = input("\n1. Fight"
                                  "\n2. Run away\n").lower()
            # if player chooses to fight the troll
            if "fight" in route1_choice:
                print_pause("You decided to fight the troll\n")
                fight_result = fight_sequence("Troll", 20, 40, 10)
                if fight_result == "player wins":
                    print_pause("You have triumphed over the troll")
                    # trigger to skip this part if visited again
                    troll = "defeated"
            # if player chooses to run
            elif "run" in route1_choice:
                print_pause("You have strategically with drawn from a "
                            "death battle")
                print_pause("You should head back to the village to "
                            "prepare for the battle, before trying again")
                cave()
            # condition to deal with unrecognized input
            else:
```



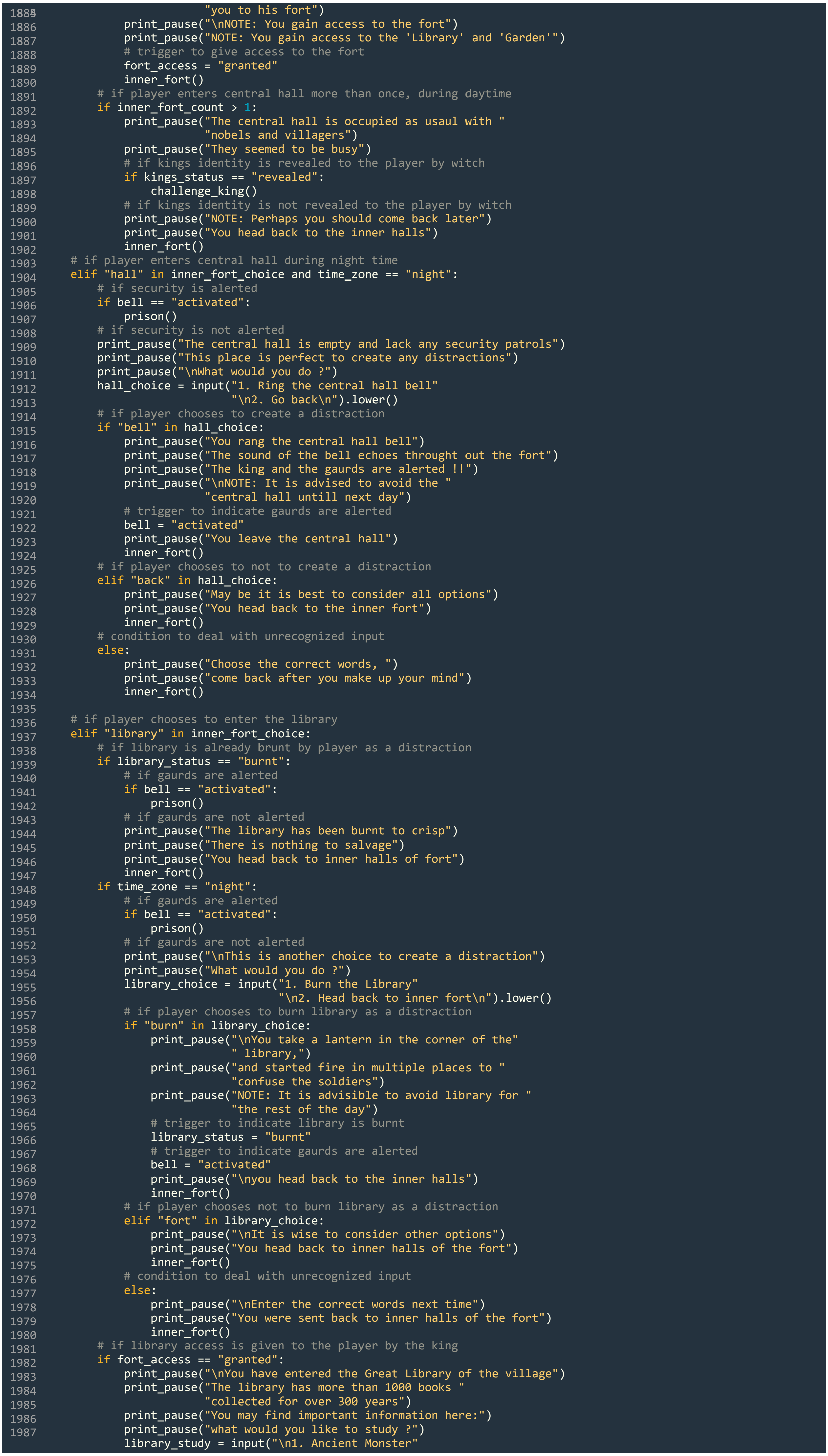
```
                print_pause("NOTE: Don't sweat, camly select "
                            "the number of your choice")
                cave()
        # if the troll has been already slain
        else:
            print_pause("The troll gaurding the route has been slain")
            print_pause("You can travel furthur deep into the route now")
        # After the troll part
        print_pause("\nYou travel deep into the Route 1\n")
        print_pause("There is a Treasure box at the certer of the room")
        print_pause("Would you like to open the Treasure box?")
        treasure_box = input("\n1. Yes"
                             "\n2. No\n").lower()
        # if player chooses to open the box
        if "yes" in treasure_box:
            # if the treasure has been already looted
            if treasure_box == "empty":
                print_pause("The Treasure Box is already looted by you")
                print_pause("There is nothing intersting here")
                print_pause("You head back to the cave entrance\n")
                cave()
            # if the treasure is not looted
            print_pause("Treasure Box is sealed with a peculiar lock")
            # if the player has the key accquired from princess
            if "Empress Sigil" in player_items:
                print_pause("You use the 'Empress Sigil' given by the "
                            "princess herself")
                print_pause("You opened the Treasure box and finds "
                            "a 'Locket'")
                print_pause("This looks like some kind of Family Heirloom")
                print_pause("NOTE: You have accquired a 'Royal Locket'")
                player_items.append("Royal Locket")
                # trigger to indicate, the treasure has been looted
                treasure_box = "empty"
                print_pause("You head back to the cave entrance\n")
                cave()
            # if the player does not have the key
            else:
                print_pause("You head back to the cave entrance\n")
                cave()
        # if player chooses not to open the box/ gives unrecognized input
        else:
            print_pause("You head back to the cave entrance\n")
            cave()

    # if player wants to enter route 2
    elif cave_choice == "2":
        print_pause("NOTE: The entrance to the cave is locked and "
                    "requires a key to open")
        # if player has the key, won in the contest grounds
        if "Unknown Key" in player_items:
            print_pause("Would you like to try the 'Unknown Key', "
                        "you have won in the contest grounds?")
            route2_choice = input("\n1 Yes"
                                  "\n2. No\n").lower()
            # if player tries to open the door using unknown key
            if "yes" in route2_choice:
                print_pause("The key matches the lock")
                print_pause("You opened the door and head furthur deep "
                            "into Route2")
                print_pause("In the room you found a chest eating dust")
                print_pause("The chest looks old and dangerous")
                print_pause("What would you do ?")
                chest_choice = input("\n1. Open"
                                     "\n2. Head back\n")
                # if player wants to open the chest
                if "open" in chest_choice:
                    # if the chest is already looted
                    if chest == "empty":
                        print_pause("You greedily open the chest again")
                        print_pause("Unfortunately, the chest does not get "
                                    "filled with gold automatically")
                        print_pause("The chest is already looted by you")
                        print_pause("There is nothing to do here")
                        print_pause("You head back to the cave entrance\n")
                        cave()
                    # if the chest is not looted
                    print_pause("You reluctantly open the chest")
                    print_pause("Voila...!! Its a treasure chest "
                                "filled with gold....")
                    print_pause("NOTE: You received 2000 gold")
                    print_pause("Gold: " + str(gold) + ", + 2000")
                    gold += 2000
                    print_pause("Your Gold: " + str(gold))
                    # trigger to indicate, the chest has been already looted
                    chest = "empty"
                    print_pause("You head back to the cave entrance\n")
                    cave()
                # if player do not want to open chest/gives unrecognized input
                else:
                    print_pause("You head back to the cave entrance\n")
                    cave()
            # if player do not want to open door/gives unrecognized input
            else:
                print_pause("You head back to the cave entrance\n")
                cave()
        # if player does not have the key, won in the contest grounds
        else:
            print_pause("NOTE: Come back after you acccquired "
                        "the key to 'Route2'")
            print_pause("You head back to the cave entrance\n")
            cave()

    # if player wants to enter route 3
```



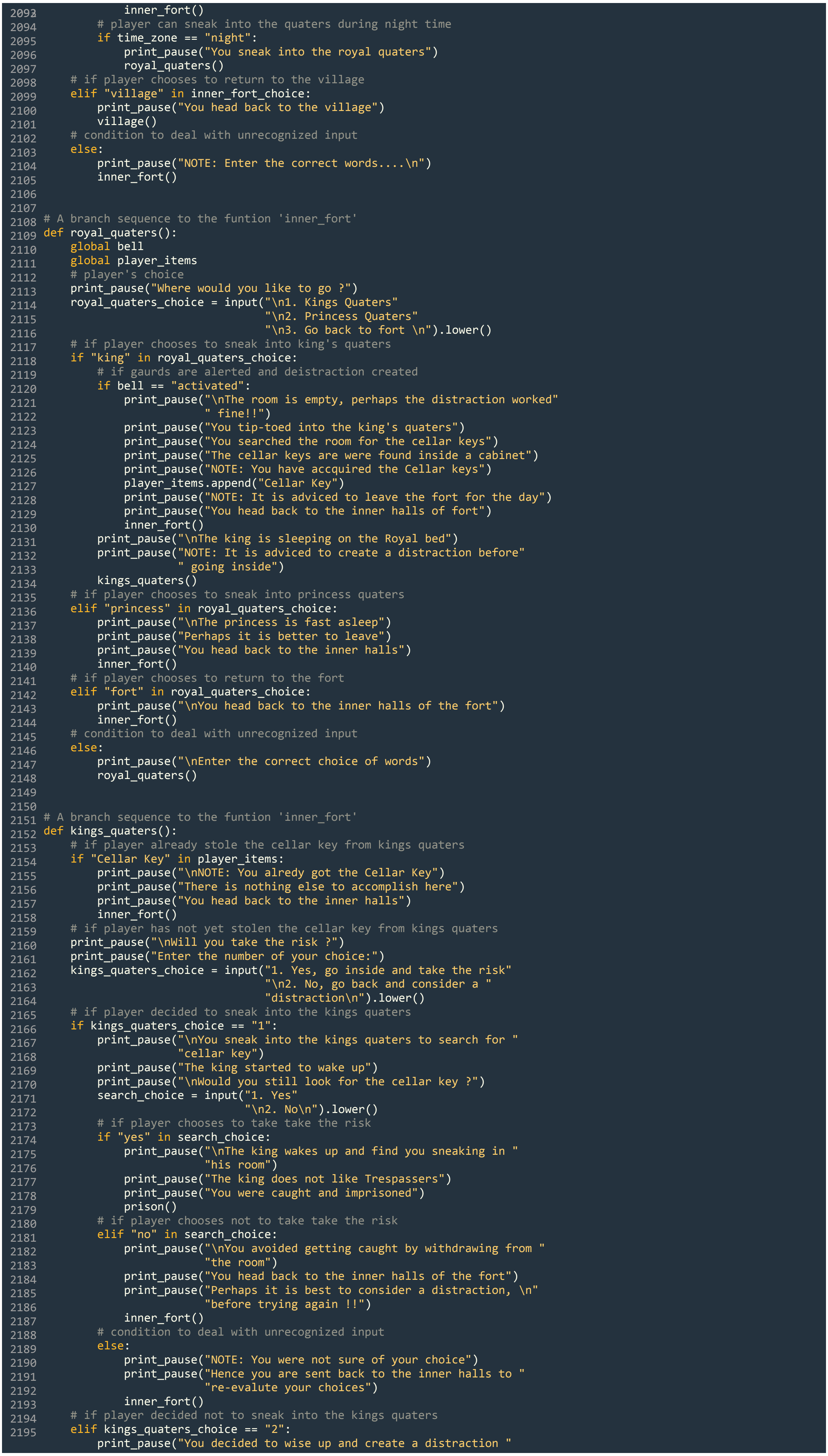
```
    elif cave_choice == "3":
        print_pause("This looks like a man made cellar")
        print_pause("There is a wooden post depicting as restricted region")
        print_pause("The cellar entrance is locked")
        # if player has the cellar key stolen from kings quaters
        if "Cellar Key" in player_items:
            print_pause("\nWould you like to open the cellar?")
            route3_choice = input("1. Yes\n2. No\n").lower()
            # if player wants to open the cellar door
            if "yes" in route3_choice:
                print_pause("\nYou enterd the cellar while staying on gaurd")
                print_pause("You venture deep in inner section of the cave")
                # if the monster is alive
                if monster == "alive":
                    print_pause("\nThere is a huge monster sleeping actross"
                                " the cellar, blocking the path")
                    print_pause("The monster gives off a deathly vibe and "
                                "extremely vicious")
                    print_pause("\nNOTE: This is a high level monster")
                    print_pause("NOTE: You propably need a special "
                                "sword-Excaliber,")
                    print_pause("      equipped with fire element to defeat"
                                " the monster")
                    print_pause("\nDecide what to do?")
                    monster_choice = input("1. Wake the monster"
                                           "\n2. Come back later\n").lower()
                    # if player chooses to wake the monster
                    if "wake" in monster_choice:
                        print_pause("\nYou courageously draws your sword, "
                                    "ready to fight the monster")
                        print_pause("The monster lunge at you with great "
                                    "ferocity")
                        fight_result = fight_sequence("monster", 30, 50, 20)
                        # if player wins the battle
                        if fight_result == "player wins":
                            # trigger to indicate moster is has been slain
                            monster = "defeated"
                            print_pause("\nYou gather remaining strength and "
                                        "venture furthur deep into the "
                                        "cellar\n")
                            cellar_rooms()
                        # if player loses the battle,
                        # result will be executed in the fight_sequence
                    # if player does not want to wake the monster
                    else:
                        print_pause("\nPerhaps you are not ready to face the "
                                    "monster, come back later")
                        print_pause("You head back to cave entrance\n")
                        cave()
                # if monster has been slain
                elif monster == "defeated":
                    print_pause("\nThere a monter lying dead in the middle of "
                                "the cellar")
                    print_pause("You move forward\n")
                    cellar_rooms()
            # if player does not want to open the cellar door
            # or give unrecognized input
            else:
                print_pause("\nYou are not sure about the choice")
                print_pause("Perhaps you should comeback later !")
                print_pause("You head back to the cave entrance")
                cave()
        # if player does not have the cellar key stolen from kings quaters
        else:
            print_pause("\nNOTE: Come back after you have accquired"
                        " cellar key")
            print_pause("You head back to cave entrance\n")
            cave()

    # if player choose to head back to the village
    elif cave_choice == "4":
        print_pause("You head back to the village\n")
        village()
    # condition to deal with unrecognized input
    else:
        print_pause("\nEnter a valid number")
        cave()


# A branch sequence to the funtion 'cave'
def cellar_rooms():
    global huge_monster
    global player_defense
    # allows the player to choose between two doors
    print_pause("\nYou find 2 doors- an Iron door and a "
                "Wooden door, at the end of the cellar")
    print_pause("\nWhat would you do?")
    door_choice = input("1. Open Iron Door"
                        "\n2. Open Wooden Door"
                        "\n3. Head back\n").lower()
    # if player chooses to opne the iron door
    if "iron" in door_choice:
        print_pause("\nYou enter the room with the Iron Door")
        # if huge monster is already slain
        if huge_monster == "defeated":
            print_pause("There a monster corpse rotting in the corner of room")
            print_pause("There is nothing else in this room")
            print_pause("You head back")
            cellar_rooms()
        # if huge monster is not slain/ alive
        print_pause("Suddenly, a Huge Monster jumps between"
                    " you and the entrance")
        print_pause("You cannot escape, leaving you no "
                    "choice but fight")
```



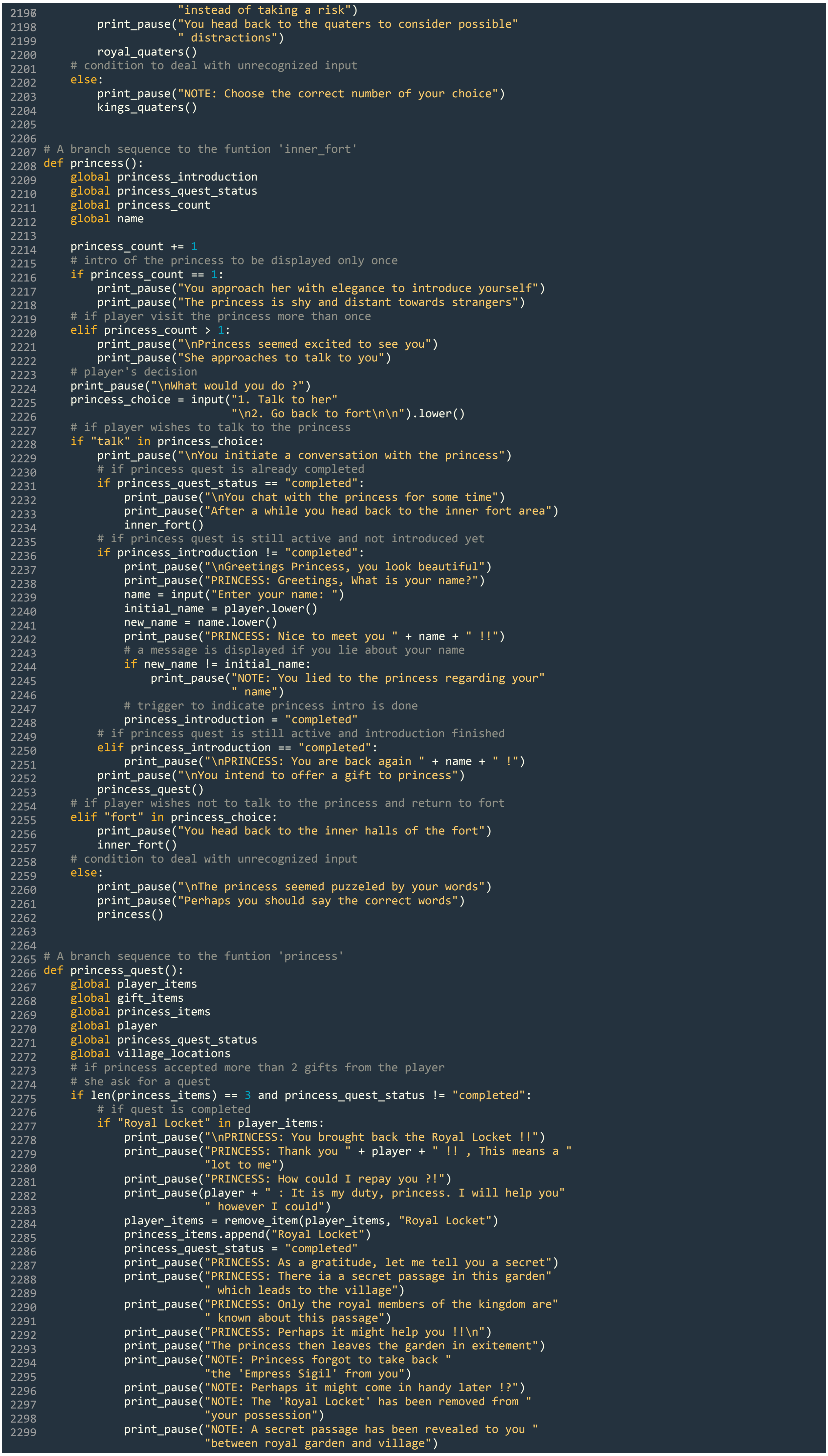
```
        fight_result = fight_sequence("Huge monster", 30, 60, 30)
        # if player wins the battle
        if fight_result == "player wins":
            print_pause("\nThe sudden battle made your boold rush to head")
            print_pause("It appears the room is used to trap the Huge Monster")
            print_pause("With the monster defeatd you were able to "
                        "leave the room")
            # trigger to indicate, hume monster has been slain
            huge_monster = "defeated"
            cellar_rooms()
        # if player loses,
        # result will be executed in the fight_sequence
    # if player chooses to open wooden door
    elif "wooden" in door_choice:
        print_pause("\nThe room is very old and filled with corpses"
                    " of fighters")
        print_pause("It appears there were numerous attempts to defeat the "
                    "monster")
        print_pause("Apparently none of them succeded")
        # if steel cheild has not yet been collected
        if "Steel Sheild" not in player_items:
            print_pause("\nYou search the room and finds a steel shield,\n "
                        "in the hands of a dead commander")
            print_pause("NOTE: You have accquired 'Steel Sheild'")
            print_pause("NOTE: Your defense has been increased")
            print_pause("Defense :" + str(player_defense) + ", + 40")
            # player stats increased
            player_defense += 40
            player_items.append("Steel Sheild")
            print_pause("Player Defense increased to: " + str(player_defense))
        # if steel cheild has already been collected
        print_pause("\nThere is nothing else execpt for the rotting corpses "
                    "in this room ")
        print_pause("You head back to the cave entrance")
        cave()
    # if player chooses to head back to the cave entrance
    elif "back" in door_choice:
        print_pause("You head back to the cave entrance")
        cave()
    # condition to deal with unrecognized input
    else:
        print_pause("Enter your speciific choice in words")
        cellar_rooms()


# funtion to trigger the location 'fort' in the village
def fort():
    global player_items
    global fort_count
    global gold
    global time_zone
    global player_status
    global toll_status
    global kings_status

    fort_count += 1
    # if player tries to enter fort at night time, he should be restricted
    if time_zone == "night":
        print_pause("The access to fort is restricted at night time")
        print_pause("Come back later")
        print_pause("You head back to the village")
        village()
    # if kings identity is revealed to the player, player should be
    # allowed into the fort even if the player is wanted for the crime
    if kings_status == "revealed":
        print_pause("You enter the fort demading a meeting with the king")
        print_pause("You were allowed inside the inner halls of the fort")
        inner_fort()
    # player will be imprisoned, if he tries to
    # enter the fort and is wanted foe a crime
    if player_status == "wanted":
        print_pause("It was not a wise choice to place yourself infront of "
                    "gaurds\n while wanted for tresspassing.")
        print_pause("You got caught and sent to prison")
        prison()
    # if fort access is already granted to the player by the king
    if fort_access == "granted":
        print_pause("You have entered inside the fort")
        inner_fort()
    # if player enter the fort gates for the first time
    if fort_count == 1:
        print_pause("The village has a Magnificient Gaint Fortress "
                    "made of stone")
        print_pause("You entered the fort entrance in astonishment "
                    "to stunnig art engraved into the walls")
        print_pause("You seek an audience with the king but was denied "
                    "by the minister of the kingdom")
        print_pause("MINISTER: I'm afraid, I cannot let just anyone pass, kid")
        print_pause("MINISTER: But I will make an exemption for 500 gold, "
                    "what do you say :")
    # if player enters the fort gate more than once
    if fort_count > 1:
        print_pause("\nYou are at the fort gate")
        # if toll was already paid and possess tribute
        if toll_status == "paid" and "A Grand Tribute" in player_items:
            print_pause("MINISTER: You may pass now kid")
            print_pause("You head inside the fort")
            inner_fort()
        # if toll was already paid and does not possess tribute
        elif toll_status == "paid" and "A Grand Tribute" not in player_items:
            print_pause("MINISTER: Obtain a suitable gift to present "
                        "to the king first.")
            print_pause("MINISTER: Then come seek his audience, kid !")
            print_pause("\nYou head back to the village in search of the ")
            print_pause("worthy present to offer the king")
            village()
        # if toll is not paid
        print_pause("MINISTER: You have to pay the Toll kid")

    # prompted to pay the toll on first visit to the fort gate
    print_pause("\nNOTE: Will you pay the TOll ? what do you choose to do ?")
    toll = input("1. Pay 500 Gold"
                 "\n2. Deny"
                 "\n3. Persuade\n").lower()
    # if player chooses to pay the toll
    if "gold" in toll:
        # if player does not have enough gold to pay the toll
        if gold < 500:
            print_pause("\nNOTE: Low in gold")
            print_pause("MINISTER: Perhaps you should come back after "
                        "earning the gold, kid")
            print_pause("You are sent back to the village")
            village()
        # if player has enough gold to pay the toll
        print_pause("\nYou pay the toll")
        print_pause("Gold: " + str(gold) + ", -500")
        gold -= 500
        print_pause("Your Gold: " + str(gold))
        # trigger to indicate toll has been paid
        toll_status = "paid"
        print_pause("\nMINISTER: Good choice kid., but I cannot just "
                    "let you pass yet")
        print_pause("MINISTER: Do you really think I'll let you pass for "
                    "500 gold alone ?")
        print_pause("MINISTER: Bring a gift to present to the king")
        print_pause("MINISTER: Can't really put you in his presense, "
                    "empty handed")
        # if player already has tribute in his possession on first visit
        if "A Grand Tribute" in player_items:
            print_pause("Luckily you have the right gift in your possession "
                        "to present to the king")
            print_pause("You were allowed inside the fort")
            inner_fort()
        # if player does not have tribute in his possession on first visit
        else:
            print_pause("MINISTER: Come back after you have the anything "
                        "worthy to present to a king")
            print_pause("You were sent back to the village")
            village()
    # if player chooses not to pay the toll
    elif "deny" in toll:
        print_pause("\nMINISTER: Then I won't let you pass kid")
        print_pause("You were sent back to the village")
        village()
    # if player chooses to convince without paying the toll
    elif "persuade" in toll:
        # player must have tribute in his possession to convince
        if "A Grand Tribute" in player_items:
            print_pause(player + ": I came from far lands bearing gifts "
                        "to the king")
            print_pause(player + ": Do you think it is a good idea to stop "
                        "me at the fort gate?")
            print_pause("The minister stay silent for a minute and "
                        "lets you pass")
            inner_fort()
        # if player does not have tribute in his possession to convince
        else:
            print_pause("\nYou have nothing to bargain with the access "
                        "to the fort")
            print_pause("NOTE: You can pay the Toll or accquire certain "
                        "item to persuade")
            print_pause("You head back to the village")
            village()
    # condition to deal with unrecognized input
    else:
        print_pause("MINISTER: Say the correct words kid")
        fort()


# A branch sequence to the funtion 'fort'
def inner_fort():
    global player_items
    global time_zone
    global inner_fort_count
    global bell
    global fort_access
    global library_status
    global kings_status

    print_pause("\nYou are inside the fort")
    print_pause("What do you choose :")
    # displays list of locations in a serial order, during daytime
    if time_zone == "day":
        inner_fort_choice = input(list_serial(fort_locations_day)).lower()
    # displays different list of locations in a serial, during night time
    elif time_zone == "night":
        inner_fort_choice = input(list_serial(fort_locations_night)).lower()
    # if player enters central hall during daytime
    if "hall" in inner_fort_choice and time_zone == "day":
        inner_fort_count += 1
        # intro to be triggered only once
        if inner_fort_count == 1:
            print_pause("The central hall is filled with many "
                        "dignitaries and villagers")
            print_pause("In the middle is the king occupying the grand throne")
            print_pause("You present yourself to the king bearing "
                        "the 'Grand Tribute' as the gift to his highness")
            print_pause("The king is pleased to the gift and welcomes "
```



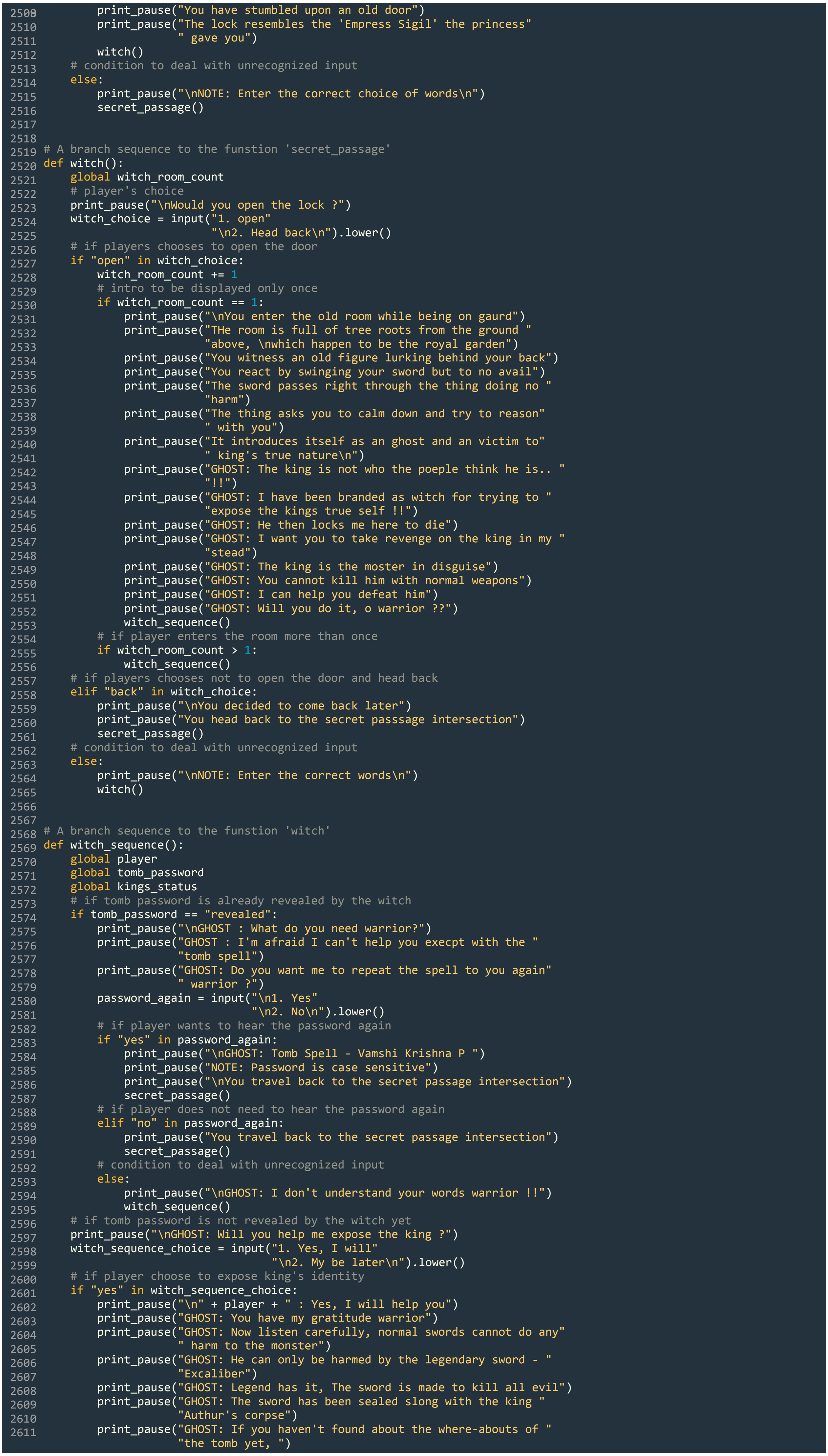
```
                        "you to his fort")
            print_pause("\nNOTE: You gain access to the fort")
            print_pause("NOTE: You gain access to the 'Library' and 'Garden'")
            # trigger to give access to the fort
            fort_access = "granted"
            inner_fort()
        # if player enters central hall more than once, during daytime
        if inner_fort_count > 1:
            print_pause("The central hall is occupied as usaul with "
                        "nobels and villagers")
            print_pause("They seemed to be busy")
            # if kings identity is revealed to the player by witch
            if kings_status == "revealed":
                challenge_king()
            # if kings identity is not revealed to the player by witch
            print_pause("NOTE: Perhaps you should come back later")
            print_pause("You head back to the inner halls")
            inner_fort()
    # if player enters central hall during night time
    elif "hall" in inner_fort_choice and time_zone == "night":
        # if security is alerted
        if bell == "activated":
            prison()
        # if security is not alerted
        print_pause("The central hall is empty and lack any security patrols")
        print_pause("This place is perfect to create any distractions")
        print_pause("\nWhat would you do ?")
        hall_choice = input("1. Ring the central hall bell"
                            "\n2. Go back\n").lower()
        # if player chooses to create a distraction
        if "bell" in hall_choice:
            print_pause("You rang the central hall bell")
            print_pause("The sound of the bell echoes throught out the fort")
            print_pause("The king and the gaurds are alerted !!")
            print_pause("\nNOTE: It is advised to avoid the "
                        "central hall untill next day")
            # trigger to indicate gaurds are alerted
            bell = "activated"
            print_pause("You leave the central hall")
            inner_fort()
        # if player chooses to not to create a distraction
        elif "back" in hall_choice:
            print_pause("May be it is best to consider all options")
            print_pause("You head back to the inner fort")
            inner_fort()
        # condition to deal with unrecognized input
        else:
            print_pause("Choose the correct words, ")
            print_pause("come back after you make up your mind")
            inner_fort()

    # if player chooses to enter the library
    elif "library" in inner_fort_choice:
        # if library is already brunt by player as a distraction
        if library_status == "burnt":
            # if gaurds are alerted
            if bell == "activated":
                prison()
            # if gaurds are not alerted
            print_pause("The library has been burnt to crisp")
            print_pause("There is nothing to salvage")
            print_pause("You head back to inner halls of fort")
            inner_fort()
        if time_zone == "night":
            # if gaurds are alerted
            if bell == "activated":
                prison()
            # if gaurds are not alerted
            print_pause("\nThis is another choice to create a distraction")
            print_pause("What would you do ?")
            library_choice = input("1. Burn the Library"
                                   "\n2. Head back to inner fort\n").lower()
            # if player chooses to burn library as a distraction
            if "burn" in library_choice:
                print_pause("\nYou take a lantern in the corner of the"
                            " library,")
                print_pause("and started fire in multiple places to "
                            "confuse the soldiers")
                print_pause("NOTE: It is advisible to avoid library for "
                            "the rest of the day")
                # trigger to indicate library is burnt
                library_status = "burnt"
                # trigger to indicate gaurds are alerted
                bell = "activated"
                print_pause("\nyou head back to the inner halls")
                inner_fort()
            # if player chooses not to burn library as a distraction
            elif "fort" in library_choice:
                print_pause("\nIt is wise to consider other options")
                print_pause("You head back to inner halls of the fort")
                inner_fort()
            # condition to deal with unrecognized input
            else:
                print_pause("\nEnter the correct words next time")
                print_pause("You were sent back to inner halls of the fort")
                inner_fort()
        # if library access is given to the player by the king
        if fort_access == "granted":
            print_pause("\nYou have entered the Great Library of the village")
            print_pause("The library has more than 1000 books "
                        "collected for over 300 years")
            print_pause("You may find important information here:")
            print_pause("what would you like to study ?")
            library_study = input("\n1. Ancient Monster"
                                  "\n2. Village"
                                  "\n3. Fort"
                                  "\n4. Kingdom History\n").lower()
            # if player wants to study about the monster
            if "monster" in library_study:
                print_pause("\nThe monster is old and witty")
                print_pause("It is said to possess the strength greater than "
                            "any creature known")
                print_pause("The monster has the ability to disguise itself "
                            "as human being")
                print_pause("It can also manipulate other humans according "
                            "to its will")
                print_pause("The monster's skin is thick enough to deflect any"
                            " weapon's attacks")
            # if player wants to study about the village
            elif "village" in library_study:
                print_pause("\nThe village is formed 40 years after the "
                            "construction of fort")
                print_pause("The early settelrs of the villagers were "
                            "Immigrants of other kingdom")
                print_pause("Due to famine, most of the people in the "
                            "neighbouring kingdoms,")
                print_pause("immigrated to this kingdom in hoope of new life")
                print_pause("The king was kind enough to provide food and "
                            "shelter to every one")
                print_pause("Eventually the village prospered and became an "
                            "important trade center")
            # if player wants to study about the fort
            elif "fort" in library_study:
                print_pause("\nThe fort is old and aged over 200 years old")
                print_pause("It has been built as a military post to overwatch"
                            " any enemy activities")
                print_pause("The fort has withstood many seizes and "
                            "weather conditions")
                print_pause("Over the time, only few minor renovations are "
                            "made to the fort")
            # if player wants to study about the kingdom
            elif "kingdom" in library_study:
                print_pause("\nThe kingdom has a history of over 600 years")
                print_pause("Many kings have ruled the kingdom over the time")
                print_pause("The kingdom has been ruled by the 'Rafel "
                            "Dynasty' and has never been fallen")
                print_pause("The kingdom eventually expanded and divided "
                            "among many kings for effective ruling")
            # condition to deal with unrecognized input
            else:
                print_pause("\nThere are no records of the word you mentioned")
                print_pause("You were sent back to inner halls of the fort")
                inner_fort()
            # after each study player is sent back to the 'inn'
            print_pause("\nYou are tired from all the reading")
            print_pause("You have no choice but to rest till morning")
            print_pause("You head back to the INN to rent a room")
            room()
        # if fort access is not given to the player
        elif fort_access != "granted":
            print_pause("\nLibrary is not accessable to everyone !")
            print_pause("You need permit from the king himself")
            print_pause("You were sent back to the inner halls of the fort")
            inner_fort()

    # if player chooses to enter the garden
    elif "garden" in inner_fort_choice:
        # if player enters the garden during night time
        if time_zone == "night":
            print_pause("\nThe garden is dark without any light")
            print_pause("One can hide / sneek in the garden at night "
                        "without getting detected")
            print_pause("There is nothing interesting here")
            print_pause("\nWhat would you do ?")
            garden_choice = input("1. Go Back to Fort"
                                  "\n2. Enter Secret Passage\n").lower()
            # if player wants to enter fort
            if "fort" in garden_choice:
                print_pause("\nYou sneaks into the inner halls of fort")
                inner_fort()
            # if player wants to enter secret passage
            elif "secret" in garden_choice:
                print_pause("\nYou enter the Secret Passage")
                secret_passage()
            # condition to deal with unrecognized input
            else:
                print_pause("\nEnter the correct choice")
                print_pause("You were sent to the inner halls anyway")
                inner_fort()
        # if fort/garden access was not granted to the player
        if fort_access != "granted":
            print_pause("\nThe garden area is restricted to strangers")
            print_pause("You need permit from the king himself")
            print_pause("You were sent back to the inner halls of the fort")
            inner_fort()
        # if fort/garden access is granted to the player
        elif fort_access == "granted" and time_zone == "day":
            print_pause("\nThe garden looks beautiful under the radiant sun")
            print_pause("You find princess roaming in the garden")
            princess()

    # if player chooses to enter the quaters
    elif "quaters" in inner_fort_choice:
        # quaters are restricted during daytime
        if time_zone == "day":
            print_pause("The quaters are not accessable to anyone "
                        "but royal family")
            print_pause("You head back to the inner halls")
```



```python
            inner_fort()
        # player can sneak into the quaters during night time
        if time_zone == "night":
            print_pause("You sneak into the royal quaters")
            royal_quaters()
    # if player chooses to return to the village
    elif "village" in inner_fort_choice:
        print_pause("You head back to the village")
        village()
    # condition to deal with unrecognized input
    else:
        print_pause("NOTE: Enter the correct words....\n")
        inner_fort()


# A branch sequence to the funtion 'inner_fort'
def royal_quaters():
    global bell
    global player_items
    # player's choice
    print_pause("Where would you like to go ?")
    royal_quaters_choice = input("\n1. Kings Quaters"
                                 "\n2. Princess Quaters"
                                 "\n3. Go back to fort \n").lower()
    # if player chooses to sneak into king's quaters
    if "king" in royal_quaters_choice:
        # if gaurds are alerted and deistraction created
        if bell == "activated":
            print_pause("\nThe room is empty, perhaps the distraction worked"
                        " fine!!")
            print_pause("You tip-toed into the king's quaters")
            print_pause("You searched the room for the cellar keys")
            print_pause("The cellar keys are were found inside a cabinet")
            print_pause("NOTE: You have accquired the Cellar keys")
            player_items.append("Cellar Key")
            print_pause("NOTE: It is adviced to leave the fort for the day")
            print_pause("You head back to the inner halls of fort")
            inner_fort()
        print_pause("\nThe king is sleeping on the Royal bed")
        print_pause("NOTE: It is adviced to create a distraction before"
                    " going inside")
        kings_quaters()
    # if player chooses to sneak into princess quaters
    elif "princess" in royal_quaters_choice:
        print_pause("\nThe princess is fast asleep")
        print_pause("Perhaps it is better to leave")
        print_pause("You head back to the inner halls")
        inner_fort()
    # if player chooses to return to the fort
    elif "fort" in royal_quaters_choice:
        print_pause("\nYou head back to the inner halls of the fort")
        inner_fort()
    # condition to deal with unrecognized input
    else:
        print_pause("\nEnter the correct choice of words")
        royal_quaters()


# A branch sequence to the funtion 'inner_fort'
def kings_quaters():
    # if player already stole the cellar key from kings quaters
    if "Cellar Key" in player_items:
        print_pause("\nNOTE: You alredy got the Cellar Key")
        print_pause("There is nothing else to accomplish here")
        print_pause("You head back to the inner halls")
        inner_fort()
    # if player has not yet stolen the cellar key from kings quaters
    print_pause("\nWill you take the risk ?")
    print_pause("Enter the number of your choice:")
    kings_quaters_choice = input("1. Yes, go inside and take the risk"
                                 "\n2. No, go back and consider a "
                                 "distraction\n").lower()
    # if player decided to sneak into the kings quaters
    if kings_quaters_choice == "1":
        print_pause("\nYou sneak into the kings quaters to search for "
                    "cellar key")
        print_pause("The king started to wake up")
        print_pause("\nWould you still look for the cellar key ?")
        search_choice = input("1. Yes"
                              "\n2. No\n").lower()
        # if player chooses to take take the risk
        if "yes" in search_choice:
            print_pause("\nThe king wakes up and find you sneaking in "
                        "his room")
            print_pause("The king does not like Trespassers")
            print_pause("You were caught and imprisoned")
            prison()
        # if player chooses not to take take the risk
        elif "no" in search_choice:
            print_pause("\nYou avoided getting caught by withdrawing from "
                        "the room")
            print_pause("You head back to the inner halls of the fort")
            print_pause("Perhaps it is best to consider a distraction, \n"
                        "before trying again !!")
            inner_fort()
        # condition to deal with unrecognized input
        else:
            print_pause("NOTE: You were not sure of your choice")
            print_pause("Hence you are sent back to the inner halls to "
                        "re-evalute your choices")
            inner_fort()
    # if player decided not to sneak into the kings quaters
    elif kings_quaters_choice == "2":
        print_pause("You decided to wise up and create a distraction "
```



```
                    "instead of taking a risk")
        print_pause("You head back to the quaters to consider possible"
                    " distractions")
        royal_quaters()
    # condition to deal with unrecognized input
    else:
        print_pause("NOTE: Choose the correct number of your choice")
        kings_quaters()


# A branch sequence to the funtion 'inner_fort'
def princess():
    global princess_introduction
    global princess_quest_status
    global princess_count
    global name

    princess_count += 1
    # intro of the princess to be displayed only once
    if princess_count == 1:
        print_pause("You approach her with elegance to introduce yourself")
        print_pause("The princess is shy and distant towards strangers")
    # if player visit the princess more than once
    elif princess_count > 1:
        print_pause("\nPrincess seemed excited to see you")
        print_pause("She approaches to talk to you")
    # player's decision
    print_pause("\nWhat would you do ?")
    princess_choice = input("1. Talk to her"
                            "\n2. Go back to fort\n\n").lower()
    # if player wishes to talk to the princess
    if "talk" in princess_choice:
        print_pause("\nYou initiate a conversation with the princess")
        # if princess quest is already completed
        if princess_quest_status == "completed":
            print_pause("\nYou chat with the princess for some time")
            print_pause("After a while you head back to the inner fort area")
            inner_fort()
        # if princess quest is still active and not introduced yet
        if princess_introduction != "completed":
            print_pause("\nGreetings Princess, you look beautiful")
            print_pause("PRINCESS: Greetings, What is your name?")
            name = input("Enter your name: ")
            initial_name = player.lower()
            new_name = name.lower()
            print_pause("PRINCESS: Nice to meet you " + name + " !!")
            # a message is displayed if you lie about your name
            if new_name != initial_name:
                print_pause("NOTE: You lied to the princess regarding your"
                            " name")
            # trigger to indicate princess intro is done
            princess_introduction = "completed"
        # if princess quest is still active and introduction finished
        elif princess_introduction == "completed":
            print_pause("\nPRINCESS: You are back again " + name + " !")
        print_pause("\nYou intend to offer a gift to princess")
        princess_quest()
    # if player wishes not to talk to the princess and return to fort
    elif "fort" in princess_choice:
        print_pause("You head back to the inner halls of the fort")
        inner_fort()
    # condition to deal with unrecognized input
    else:
        print_pause("\nThe princess seemed puzzeled by your words")
        print_pause("Perhaps you should say the correct words")
        princess()


# A branch sequence to the funtion 'princess'
def princess_quest():
    global player_items
    global gift_items
    global princess_items
    global player
    global princess_quest_status
    global village_locations
    # if princess accepted more than 2 gifts from the player
    # she ask for a quest
    if len(princess_items) == 3 and princess_quest_status != "completed":
        # if quest is completed
        if "Royal Locket" in player_items:
            print_pause("\nPRINCESS: You brought back the Royal Locket !!")
            print_pause("PRINCESS: Thank you " + player + " !! , This means a "
                        "lot to me")
            print_pause("PRINCESS: How could I repay you ?!")
            print_pause(player + " : It is my duty, princess. I will help you"
                        " however I could")
            player_items = remove_item(player_items, "Royal Locket")
            princess_items.append("Royal Locket")
            princess_quest_status = "completed"
            print_pause("PRINCESS: As a gratitude, let me tell you a secret")
            print_pause("PRINCESS: There ia a secret passage in this garden"
                        " which leads to the village")
            print_pause("PRINCESS: Only the royal members of the kingdom are"
                        " known about this passage")
            print_pause("PRINCESS: Perhaps it might help you !!\n")
            print_pause("The princess then leaves the garden in exitement")
            print_pause("NOTE: Princess forgot to take back "
                        "the 'Empress Sigil' from you")
            print_pause("NOTE: Perhaps it might come in handy later !?")
            print_pause("NOTE: The 'Royal Locket' has been removed from "
                        "your possession")
            print_pause("NOTE: A secret passage has been revealed to you "
                        "between royal garden and village")
            # a secret passage will be added to the village locations
            village_locations.append("Secret Passage")
            print_pause("You head back to the inner halls of the fort")
            inner_fort()
        # if quest is not yet completed but briefed already
        elif "Empress Sigil" in player_items:
            print_pause("\nPRINCESS: Did you get the family "
                        "heirloom " + player + " ?")
            print_pause("PRINCESS: I hope you find it !!")
            print_pause("You head out to retrive the stolen item")
            village()
        # if quest is not yet completed nor briefed
        print_pause("\nPRINCESS: I have a request to make of you " + player)
        print_pause("PRINCESS: There ia a family heirloom passed down for"
                    " generations")
        print_pause("PRINCESS: My mother gave it to me as a present,")
        print_pause("          on my 16th birthday, before she passed away")
        print_pause("PRINCESS: Apparently its been lost, more like stolen")
        print_pause("PRINCESS: I heard it has been stashed in one of the cave"
                    " partitions")
        print_pause("PRINCESS: Will you bring it for me " + player + " ??")
        print_pause("PRINCESS: The heirloom is sealed in a royal box")
        print_pause("PRINCESS: You may need this royal sigil to open it")
        player_items.append("Empress Sigil")
        print_pause("NOTE: You have accquired 'Empress Sigil'.")
        print_pause("You head back to the inner halls of the fort")
        inner_fort()

    # if princesss quest is does not meet required conditions to get triggered
    # and player wants to give gift to the princess
    print_pause("\nWhat do you choose to offer ?")
    # if player does not have anything to offer to the princesss
    # / len(gift_items) == 1 / is used because / leave / option
    # is used inside gift list
    if len(gift_items) == 1:
        print_pause("\nNOTE:You have nothing to offer to the princess")
        print_pause("NOTE: Come back later")
        print_pause("You head back to the inner halls")
        inner_fort()
    # displays list of gift items in a serial list
    princess_choice = input(list_serial(gift_items)).lower()
    # if player wants to gift flowers and have them in his possession
    if "flowers" in princess_choice and "Flowers" in gift_items:
        print_pause("PRINCESS: Beautiful flowers but nothing I haven't "
                    "seen in this garden")
        player_items = remove_item(player_items, "Flowers")
        gift_items = remove_item(gift_items, "Flowers")
        print_pause("NOTE: 'Flowers' has been removed from your possession\n")
        princess_quest()
    # if player wants to gift glowing blossom and have them in his possession
    elif "glowing" in princess_choice and "Glowing Blossom" in gift_items:
        print_pause("PRINCESS: I have never seen a flower of this kind")
        print_pause("PRINCESS: Thank you " + player)
        player_items = remove_item(player_items, "Glowing Blossom")
        gift_items = remove_item(gift_items, "Glowing Blossom")
        print_pause("NOTE: 'Glowing Blossom' has been removed from your "
                    "possession\n")
        # if princess do not have them in her possession
        # she consider them as a gift item
        if "Glowing Blossom" not in princess_items:
            princess_items.append("Glowing Blossom")
            print_pause("NOTE: You relationship strengthened")
        princess_quest()
    # if player wants to gift freya dress and have them in his possession
    elif "freya" in princess_choice and "Robes of Freya" in gift_items:
        print_pause("PRINCESS: Odd dress !!, though unique it is not to "
                    "my taste")
        player_items = remove_item(player_items, "Robes of Freya")
        gift_items = remove_item(gift_items, "Robes of Freya")
        print_pause("NOTE: 'Robes of Freya' has been removed from your "
                    "possession\n")
        princess_quest()
    # if player wants to gift princess dress and have them in his possession
    elif "princess" in princess_choice and "Dress of a princess" in gift_items:
        print_pause("PRINCESS: Elegent dress, I like it " + player)
        player_items = remove_item(player_items, "Dress of a princess")
        gift_items = remove_item(gift_items, "Dress of a princess")
        print_pause("NOTE: 'Dress of a princess' has been removed from "
                    "your possession\n")
        princess_quest()
    # if player wants to gift steel armour and have them in his possession
    elif "armour" in princess_choice:
        if "Glittering steel armour" in gift_items:
            print_pause("PRINCESS: Thank you very much " + player + " !!")
            print_pause("PRINCESS: I always wanted to try one of these")
            print_pause("PRINCESS: But my father never let me")
            print_pause("PRINCESS: You have my gratitude " + player)
            player_items = remove_item(player_items, "Glittering steel armour")
            gift_items = remove_item(gift_items, "Glittering steel armour")
            print_pause("NOTE: 'Glittering steel armour' has been removed "
                        "from your possession\n")
            # if princess do not have them in her possession
            # she consider them as a gift item
            if "Glittering steel armour" not in princess_items:
                princess_items.append("Glittering steel armour")
                print_pause("NOTE: You relationship strengthened")
        else:
            print_pause("you do not have Glittering steel armour")
        princess_quest()
    # if player wants to gift grapes and have them in his possession
    elif "grapes" in princess_choice and "Grapes" in gift_items:
        print_pause("PRINCESS: I appreciate the intention but I am not "
                    "hungry now")
        player_items = remove_item(player_items, "Grapes")
        gift_items = remove_item(gift_items, "Grapes")
        print_pause("NOTE: 'Grapes' has been removed from your possession\n")
        princess_quest()
    # if player wants to gift jewellary and have them in his possession
    elif "jewellary" in princess_choice and "Jewellary" in gift_items:
        print_pause("PRINCESS: Splendid jewellary !!")
        print_pause("PRINCESS: Perhaps I will wear it for our next meeting")
        player_items = remove_item(player_items, "Jewellary")
        gift_items = remove_item(gift_items, "Jewellary")
        print_pause("NOTE: 'Jewellary' has been removed from your"
                    " possession\n")
        # if princess do not have them in her possession
        # she consider them as a gift item
        if "Jewellary" not in princess_items:
            princess_items.append("Jewellary")
            print_pause("NOTE: You relationship strengthened")
        princess_quest()
    # if player chooses to leave the garden
    elif "leave" in princess_choice:
        print_pause("You head back to the inner halls")
        inner_fort()
    # condition to deal with unrecognized input
    else:
        print_pause("Enter the correct gift....")
        princess_quest()


# A branch sequence to the multiple funstions
def prison():
    global bell
    global player_items
    global gold
    global time_zone
    global village_locations
    global fort_locations
    global player_status
    print_pause("\nYou were caught by the gaurds")
    print_pause("The king is not tolrent of trespassers")
    print_pause("You were sent to prison")
    print_pause("After 4 days of imprisonment, you strted to lose hope")
    print_pause("Many time sensitive items in your inventory, "
                "like grapes and flowers are dimished")
    # time is reset to night
    # gaurds are reset to not alerted as player is captured
    # player status is changed to wanted
    time_zone = "night"
    bell = "deactivated"
    player_status = "wanted"
    # any time sensitive items in player's possession will be removed
    player_items = remove_item(player_items, "Grapes")
    player_items = remove_item(player_items, "Flowers")
    player_items = remove_item(player_items, "Glowing Flowers")
    print_pause("\nOn 4th day night, when the new moon is in the sky "
                "with no light,")
    print_pause("Princess herslf comes to save you from the shackles "
                "of imprisonment")
    print_pause("You were rescued and sneaked into the inner fort")
    print_pause("You are on your own from here")
    inner_fort()


# A branch sequence to the multiple funstions
def secret_passage():
    global time_zone
    # player's choice
    print_pause("You are in the secret passage")
    print_pause("\nchoose a location to travel undetected:")
    secret_passage_choice = input("1. Royal Garden"
                                  "\n2. Village"
                                  "\n3. Explore the passage\n").lower()
    # if player chooses to enter garden during daytime
    if "garden" in secret_passage_choice and time_zone == "day":
        print_pause("\nYou head towards the garden in the fort")
        print_pause("NOTE: You might expose the secret passage in"
                    " broad-day-light")
        print_pause("\nDo you want to risk it ?")
        passage_to_garden_choice = input("1. Yes"
                                         "\n2. No\n").lower()
        # if player wants to risk it
        if "yes" in passage_to_garden_choice:
            print_pause("\nYou have entered the garden during the day time")
            print_pause("The gaurds caught you and sent you to prison")
            prison()
        # if player does not want to risk it
        else:
            print_pause("You have decided to use another route \nto avoid risk"
                        " exposing the secret path")
            secret_passage()
    # if player chooses to enter garden during night time
    elif "garden" in secret_passage_choice and time_zone == "night":
        print_pause("\nYou entered the garden area through secret passage")
        print_pause("As it is dark, you are able to sneak past gaurds")
        print_pause("You then enter the inner halls of fort")
        inner_fort()
    # if player chooses to enter village during night time
    elif "village" in secret_passage_choice:
        print_pause("You head towards the village")
        village()
    # if player chooses to explore the passage
    elif "explore" in secret_passage_choice:
        print_pause("\nYou hear peculiar sounds coming from inside "
                    "the passage")
        print_pause("You decided to investigate the passage")
        print_pause("You went furthur deep into the passage")
```



```
        print_pause("You have stumbled upon an old door")
        print_pause("The lock resembles the 'Empress Sigil' the princess"
                    " gave you")
        witch()
    # condition to deal with unrecognized input
    else:
        print_pause("\nNOTE: Enter the correct choice of words\n")
        secret_passage()


# A branch sequence to the funstion 'secret_passage'
def witch():
    global witch_room_count
    # player's choice
    print_pause("\nWould you open the lock ?")
    witch_choice = input("1. open"
                         "\n2. Head back\n").lower()
    # if players chooses to open the door
    if "open" in witch_choice:
        witch_room_count += 1
        # intro to be displayed only once
        if witch_room_count == 1:
            print_pause("\nYou enter the old room while being on gaurd")
            print_pause("THe room is full of tree roots from the ground "
                        "above, \nwhich happen to be the royal garden")
            print_pause("You witness an old figure lurking behind your back")
            print_pause("You react by swinging your sword but to no avail")
            print_pause("The sword passes right through the thing doing no "
                        "harm")
            print_pause("The thing asks you to calm down and try to reason"
                        " with you")
            print_pause("It introduces itself as an ghost and an victim to"
                        " king's true nature\n")
            print_pause("GHOST: The king is not who the poeple think he is.. "
                        "!!")
            print_pause("GHOST: I have been branded as witch for trying to "
                        "expose the kings true self !!")
            print_pause("GHOST: He then locks me here to die")
            print_pause("GHOST: I want you to take revenge on the king in my "
                        "stead")
            print_pause("GHOST: The king is the moster in disguise")
            print_pause("GHOST: You cannot kill him with normal weapons")
            print_pause("GHOST: I can help you defeat him")
            print_pause("GHOST: Will you do it, o warrior ??")
            witch_sequence()
        # if player enters the room more than once
        if witch_room_count > 1:
            witch_sequence()
    # if players chooses not to open the door and head back
    elif "back" in witch_choice:
        print_pause("\nYou decided to come back later")
        print_pause("You head back to the secret passsage intersection")
        secret_passage()
    # condition to deal with unrecognized input
    else:
        print_pause("\nNOTE: Enter the correct words\n")
        witch()


# A branch sequence to the function 'witch'
def witch_sequence():
    global player
    global tomb_password
    global kings_status
    # if tomb password is already revealed by the witch
    if tomb_password == "revealed":
        print_pause("\nGHOST : What do you need warrior?")
        print_pause("GHOST : I'm afraid I can't help you execpt with the "
                    "tomb spell")
        print_pause("GHOST: Do you want me to repeat the spell to you again"
                    " warrior ?")
        password_again = input("\n1. Yes"
                               "\n2. No\n").lower()
        # if player wants to hear the password again
        if "yes" in password_again:
            print_pause("\nGHOST: Tomb Spell - Vamshi Krishna P ")
            print_pause("NOTE: Password is case sensitive")
            print_pause("\nYou travel back to the secret passage intersection")
            secret_passage()
        # if player does not need to hear the password again
        elif "no" in password_again:
            print_pause("You travel back to the secret passage intersection")
            secret_passage()
        # condition to deal with unrecognized input
        else:
            print_pause("\nGHOST: I don't understand your words warrior !!")
            witch_sequence()
    # if tomb password is not revealed by the witch yet
    print_pause("\nGHOST: Will you help me expose the king ?")
    witch_sequence_choice = input("1. Yes, I will"
                                  "\n2. My be later\n").lower()
    # if player choose to expose king's identity
    if "yes" in witch_sequence_choice:
        print_pause("\n" + player + " : Yes, I will help you")
        print_pause("GHOST: You have my gratitude warrior")
        print_pause("GHOST: Now listen carefully, normal swords cannot do any"
                    " harm to the monster")
        print_pause("GHOST: He can only be harmed by the legendary sword - "
                    "Excaliber")
        print_pause("GHOST: Legend has it, The sword is made to kill all evil")
        print_pause("GHOST: The sword has been sealed slong with the king "
                    "Authur's corpse")
        print_pause("GHOST: If you haven't found about the where-abouts of "
                    "the tomb yet, ")
        print_pause("GHOST: It is located somewhere inside the forest")
        print_pause("\nGHOST: The spell to unbind the seal "
                    "is : " + "'Vamshi Krishna P'\n")
        print_pause("NOTE: You may enter king Authur's Tomb now")
        print_pause("GHOST: Also you might need a sturdy sheild to defend "
                    "against monster's attacks")
        print_pause("GHOST: I can't help you with the sheild though !!")
        print_pause("GHOST: I plead you warrior, bring peace to my "
                    "agonized soul")
        print_pause(player + " : I will, you may rest in peace now\n")
        # tomb password and king's identity is revealed by the witch
        tomb_password = "revealed"
        kings_status = "revealed"
        print_pause("You head back to the secret passage intersection")
        secret_passage()
    # if player chooses not to expose king's identity
    elif "later" in witch_sequence_choice:
        print_pause("\nYou were not sure of the choice")
        print_pause("You decide to come back later")
        print_pause("You head back to the secret passage intersection")
        secret_passage()
    # condition to deal with unrecognized input
    else:
        print_pause("\nGHOST: I don't understand your words warrior !!")
        witch_sequence()


# A branch sequence to the funstion 'inner_fort'
def challenge_king():
    # player's choice
    print_pause("\nWould you challenge the king to a battle ?")
    challenge = input("1. yes, challenge the king"
                      "\n2. No, not yet\n").lower()
    # if player chooses to fight the king
    if "yes" in challenge:
        print_pause("\nYou challenge the king to single combat")
        print_pause("The king did not hesitate to accept the challenge\n")
        fight_result = fight_sequence("King", 50, 70, 50)
        # if player wins
        if fight_result == "player wins":
            print_pause("The king is on his final breath")
            print_pause("You approach him with your sword drawn")
            kings_fate_scenario()
    # if player chooses not to fight the king
    elif "no" in challenge:
        print_pause("\nYou were not ready yet")
        print_pause("You decided to wait for the right time")
        print_pause("You head back to the village")
        village()
    # condition to deal with unrecognized input
    else:
        print_pause("\nNOTE: Enter the correct words\n")
        challenge_king()


# A branch sequence to the funstion 'challenge_king'
def kings_fate_scenario():
    global monster_list
    # player's choice
    print_pause("\nEnter the number of your choice :")
    kings_fate = input("1. Kill the king and finish once and for all"
                       "\n2. Somethings not right, spare his life\n")
    # if player decides to kill the king
    if kings_fate == "1":
        print_pause("\nYou deliver a powerful blow to the king ")
        print_pause("The king took the final blow with out any "
                    "resistance")
        print_pause("almost as if he wanted you to slay him !!")
        print_pause("You have slain the monster, but something feels unsettled"
                    " to you.")
        print_pause("You were branded as king's slayer and abolished "
                    "from the kingdom")
        print_pause(".")
        print_pause(".")
        print_pause(".")
        print_pause(".")
        print_pause("After 4 weeks of the incident")
        print_pause("You were drinking in an Inn in the neighbouring"
                    " town")
        print_pause("You enquire the BARTENDER for the info as a daily"
                    " routine")
        print_pause("According to the information,")
        print_pause("The princess was kidnapped by the monster and "
                    "never returned !!")
        print_pause("As suspected, the monster was not the king !!")
        print_pause("There is something behind the scenes playing the king")
        print_pause("NOTE: Play the game again for different end "
                    "scenario")
        print_pause("Time to return to the village ....//\n")
        print_pause("--------------- GAME ENDED ------------------")
        credits()
    # if player decides not to kill the king
    elif kings_fate == "2":
        print_pause("\nYou take a leap of faith, to spare the king's life")
        print_pause("You gave the king a chance to explain his actions")
        print_pause("The king is in no condition to talk to you,")
        print_pause("He appears to be be possessed")
        print_pause("\nYou sense a dark presence lurking in the shadows !!")
        print_pause("It appears to be the monster, manipulating the king all"
                    " along")
        print_pause("The king is nothing but a puppet,\n a victim to monster's"
                    " vicious nature")
        print_pause("You leap towards the monster at full throttle")
        print_pause("The monster was caught off gaurd and wonded badly")
```

```python
        # monster will be revealed as a random from a list of characters
        monster_trueform = random.choice(monster_list)
        print_pause("\nThe monster actually is the  " + monster_trueform + ", "
                    "\ndisguised itself as a human !!\n")
        print_pause("The monster is both wounded and drain from "
                    "manipulating king")
        print_pause("This is the perfect chance to end this once "
                    "and for all..\n")
        fight_result = fight_sequence("Monster", 40, 60, 40)
        if fight_result == "player wins":
            print_pause("The monster is finally slain")
            print_pause("You rescued the king and the village from the "
                        "monster's evil plan")
            print_pause("You were rewarded with gold and title in the "
                        "kingdom\n")
            print_pause("------------------ GAME ENDED -------------------")
            credits()
    # condition to deal with unrecognized input
    else:
        print_pause("\nEnter the number of your choice to decide king's fate")
        kings_fate_scenario()


# function to be displayed if player successfully complete the game
def credits():
    global player
    print_pause("\n\n         //  Congratulations on completing the game  //")
    print_pause("\n\n\n\n___________________ DISCLAIMER ____________________")
    print_pause("\n>> This game is a pure work of fiction,")
    print_pause("   resemblance to any king or Protagonist is a pure "
                "coincidence")
    print_pause(">> No monsters are actually slain / harrased in "
                "this game")
    print_pause(">> We strictly followed the 'Player-Monster-Coexist-World"
                " Peace' rule in this game\n\n")
    print_pause("___________________ CREDITS ____________________")
    print_pause("\nThis game was developed by and\nsole property of "
                "// 'Vamshi Krishna P' //\n")
    print_pause("Cast and Crew:")
    print_pause("Protagonist - " + player)
    print_pause("Princess    - princess herself")
    print_pause("king        - Acted against his will, manipulated"
                " by monster!")
    print_pause("Shopkeeper  - Shopkeeper himself")
    print_pause("Host        - remains anonymous")
    print_pause("Minister    - special appearance")
    print_pause("Bartender   - bartender himself")
    print_pause("Monster     - actually slain")
    print_pause("\n\n")
    print_pause("Thanks for playing the game, wait of the next sequel....//")
    exit()


# ------------ Game Trigger ------------


adventure_game()
```

AWESOME

✅ The `pycodestyle` tool reports zero errors and zero warnings. Nice job! 👍

▸ README.md

RETURN TO PATH

Rate this review

☆☆☆☆☆