```python
# Robert Blanco Integration Project COP 1500 CRN 80597
# Last edit as of 25SEP2020
# This is a simple 'Choose Your Own Adventure' style game meant to demonstrate SE j

"""COP1500 Integration Project Displaying Key Competencies"""
__author__ = "Rob Blanco"

import time
import random
import sys

# The first keyword, 'import', is used to import a module. The next word, 'time' is
# According to w3schools.com, a module is a file containing a set of functions.
# The idea to utilize this module for suspense building, and aid in reading, was ir
# https://www.youtube.com/watch?v=miuHrP207Jw&t=622s

# Importing random module in order to randomly select possible room choices for the
# Importing sys module in order to implement a sys.exit function to end the program

print("Welcome and thank you for your participation!")
# Print is a function within Python that prints a message, within those parentheses
# is run.
# The text within the parentheses and double quotes (those quotes can also be singl
# string literal.
# This is the first line of text that the user will see, so I'm greeting them. And
# participation.

time.sleep(1)
# This line utilizes the time module to suspend execution for a given number of sec
# specified within the parentheses of 'time.sleep()'.

print("This is an early version of a text based 'Choose Your Own Adventure' game. S
      "of....currently unspecified size.")
time.sleep(1)
print("First, let's pick a name and title for your character")
time.sleep(1)

game_start = True
# I'm assigning the variable 'game_start' the boolean value of 'True'
# This will be in order to start a 'while loop' which will keep returning to an inp
# below.

while game_start:
    # This starts the loop command of 'while' which means that while 'game_start' i
    # continue 'returning' here.
    # Until the user completes an action which results in 'game_start' being assigr
    # https://www.w3schools.com/python/python_while_loops.asp
    # https://sites.google.com/site/profvanselow/programming/languages/python/loops

    introduction_choice = input("Ready to get started? Enter 1 for 'Yes' or 2 for '
    # The '=' symbol is an assignment operator that assigns the variable of 'introc
    # inputs through the function input().
    # The function 'input()', contains a prompt. Which is a string representing a c
    # Basically, instructions for the user.

    if introduction_choice != "1" and introduction_choice != "2":
        # AND is a python logical/boolean operator. Here, the IF conditional statem
        # whether 'introduction_choice' is NOT equal-
        # -to BOTH (due to the AND) "1" and "2"
        print("")
        print("Sorry! That was an invalid entry. Just enter a 1 or 2. Nothing else
        time.sleep(1)
        print("Let's try again.")
        time.sleep(1)
        print("")
```

```
    # This is an 'if' logical condition statement that tests conditions based arour
    # != is a comparison operator that compares whether two values are NOT equal to
    # Additionally, the logical operator of 'and' is testing whether both condition
    # So, from the top. While game_start remains True, which it still is, since not
    # These lines of code are going to run these print statements and loop back.
    # Because it's 'True' that introduction_choice is not assigned 1 and 2. Meaning
    # and the code will loop.

    else:
        # 'Else' is often paired with 'if' conditional statements, to catch any oth
        # by a prior 'if' or 'elif' statement.
        # So provided the user enters a 1 or 2....

        if introduction_choice == "1":
            # Relational/conditional operator. Used in conjunction with the IF cond
            # if the user's input is equal to "1"
            print("Great! Let's get started...")
            time.sleep(1)
            game_start = False
            # The 'if' conditional statement here gets a 'True' when it evaluates i
            # Since a prior assignment operation assigned a 1 to introduction_choic
            # It now runs the next three lines of code from top to bottom. Right to
            # Most importantly, it assigns game_start to 'False'
            # And now the 'game_start' loop is broken and the rest of the code can

        else:
            if introduction_choice == "2":
                print("Understood! No worries, come back any time. Should you chang
                time.sleep(1)
            quit()
            # Here, the user decides to not play.
            # I thank them for their time and the function quit() runs.
            # https://www.geeksforgeeks.org/python-exit-commands-quit-exit-sys-exit
            # There appear to be SystemExit concerns with using this function, but
            # it a further date when it becomes a concern.
            # https://stackoverflow.com/questions/73663/how-to-terminate-a-python-s

user_name = ''
rank = ''

name_choice = True
while name_choice:
    user_name = input("\nPlease type a fictional last name for your character. Such
                      "\nType it here: ")
    # Nothing new here, except that I utilize the escape character  \n to create a
    # And \" to allow for double quotes within a string, which would cause an error

    answer_verify = input(
        "You selected \"" + user_name + "\" as your character's last name. Is this
                                        " No: ")
    # Within the above input function, I conduct a string operation.
    # I combine the literal string of 'You selected' with the variable 'user_name'
    # the second portion of the text.
    # An input function already produces a string, so there is no need to format 'u
    # More on that later.

    if answer_verify == "1":
        print("Great!")
        time.sleep(1)
        print("\"" + user_name + "\"" + " it is, then.")
        time.sleep(1)
        name_choice = False
        # Again, same 'while' to 'if' loop format as above. The user accepts their
        # breaks.
        # 'user_name' can now be used throughout the program.
```

```
    else:
        if answer_verify == "2":
            print("Understood, let's try again.")
            time.sleep(1)
            # Python sees that name_choice still is assigned 'True' and loops back
        else:
            print("I'm sorry, that wasn't an input the program understood. Let's tr
            time.sleep(1)
            # Following the tree of decision making, this would be printed if the u
            # than 1 or 2.
            # ' name_choice ' is still true, and Python loops back.
            # Ideally, the program would loop back without the user having to re-ir
            # Will be implemented in a future Sprint.

rank_choice = True
print("Moving on, you are playing a pilot in space.")
time.sleep(2)
print("So let's decide on a pseudo-military rank, or title.")
time.sleep(2)
# Nothing new introduced here. Just more variable assignments and print functions.

while rank_choice:
    answer_verify = input(
        "Which sounds good?\n1. Commander\n2. Chief Warrant Officer (addressed as '
        "selection here: ")
    if answer_verify != "1" and answer_verify != "2" and answer_verify != "3":
        print("")
        time.sleep(1)
        print("Sorry, that wasn't a valid input. Enter only a 1, 2, or 3 to select
        print("")
    else:
        if answer_verify == "1":
            rank = "Commander"
            answer_verify = input(
                "Alright, " + rank + " " + user_name + ". Does that sound good? Ent
            if answer_verify != "1" and answer_verify != "2":
                print("Sorry, that wasn't a valid input. Enter just the number 1 to
                print(rank + ", or 2 to start again with rank selection.")
                print("")
                time.sleep(1)
            else:
                if answer_verify == "2":
                    print("Alright, let's go over the rank choices again.")
                    print("")
                    time.sleep(1)
                else:
                    if answer_verify == "1":
                        print("Great! " + rank + " " + user_name + " it is.")
                        rank_choice = False
                        print("")
                        time.sleep(1)

        else:
            if answer_verify == "2":
                rank = "Chief"
                answer_verify = input(
                    "Alright, " + rank + " " + user_name + ". Does that sound good?
                if answer_verify != "1" and answer_verify != "2":
                    print("Sorry, that wasn't a valid input. Enter just the number
                    print(", or 2 to start again with rank selection.")
                    print("")
                    time.sleep(1)
                else:
                    if answer_verify == "2":
                        print("Alright, let's go over the rank choices again.")
```

```
                    print("")
                    time.sleep(1)
                else:
                    if answer_verify == "1":
                        print("Great! " + rank + " " + user_name + " it is.")
                        rank_choice = False
                        print("")
                        time.sleep(1)
            else:
                if answer_verify == "3":
                    rank = "Captain"
                    answer_verify = input(
                        "Alright, " + rank + " " + user_name + ". Does that sound g
                    if answer_verify != "1" and answer_verify != "2":
                        print(
                            "Sorry, that wasn't a valid input. Enter just the numbe
                            ", or 2 to start again with rank selection.")
                        print("")
                        time.sleep(1)
                    else:
                        if answer_verify == "2":
                            print("Alright, let's go over the rank choices again.")
                            print("")
                            time.sleep(1)
                        else:
                            if answer_verify == "1":
                                print("Great! " + rank + " " + user_name + " it is.
                                rank_choice = False
                                print("")
                                time.sleep(1)

# This entire block of code is essentially a second 'while loop'. Similar to the fi
# Also, I try to utilize trailing "else" statements, so I catch all possibilities o
# In the hopes of preventing the program from crashing due to an invalid input.

print("And with that, " + rank + " " + user_name + ", let's take a brief step forwa
# It bears mentioning that if the variables in the above print function contained i
# function to format them into strings.
# As only strings can be concatonated through a string operation of '+'.

time.sleep(2)
print(
    "The silence of space and the calm din of the cockpit surrounds you, as you're
    "'bridge' of the Endeavor.")
time.sleep(2)
print(
    "The amount of human intellectual, and financial capital, that it took to bring
    "ancestors to finally crest Alpha Centauri can't be understated.")
time.sleep(2)
print(
    "Even as you rub your eyes and shrug off any lingering drowsiness from your sle
    "of travel and data collection....")
time.sleep(2)
print("Even if some of the first daily tasks aren't exactly thrilling....")
time.sleep(2)
print("A screen in front of you lights up, as the ship's AI 'Abby' greets you.")
# A future sprint might allow the user to specify whether 'Abby' could be an 'Arthu
time.sleep(2)
print("A calm feminine voice reaches out to you from a nearby display panel...")
print("")
time.sleep(2)
print("Abby: Good morning, and welcome back to the bridge, " + rank,
      user_name + ". The daily SITREP is due to the ISS~")
# Here I utilize a 'comma' to space the variables of 'rank' and 'user_name'
time.sleep(1)
```

```
# This stack of print functions and time.sleep functions act as a brief introductic
print("")
print("")

dialogue_1 = True

while dialogue_1:
    dialogue_1_answer = input(
        "Please select a dialogue option:\n1. Sure!\n2. Uh.....what's a SITREP?\n3.
        "5 more minutes before you start bossing me around?\n")
    if dialogue_1_answer == "1":
        print("")
        time.sleep(1)
        print(
            "Abby: That's the spirit " + rank + " " + user_name + "! Positivity is
                                                                  "space isolation!
        dialogue_1 = False
        print("")
        time.sleep(2)
    elif dialogue_1_answer == "2":
        print("")
        time.sleep(1)
        print(
            "Abby: Oh very funny, " + rank + "....you know all too well that SITREP
                                             "No procrastinating! Let's get to it~"
        dialogue_1 = False
        print("")
        time.sleep(2)
    elif dialogue_1_answer == "3":
        print("")
        time.sleep(1)
        print(
            "Abby: Nope! I waited until the completion of your morning coffee, as p
            ". Now....the SITREP will only take a few moments....")
        dialogue_1 = False
        print("")
        time.sleep(2)
    else:
        print("")
        print("Sorry! Invalid input! Enter only a 1, 2, or 3")
        print("")
        time.sleep(2)

# The above section of code is a much simpler version of the prior two 'while' loop
# Here, this combines IF, ELIF, and ELSE conditional statements together. IF has al
# ELIF is another conditional statement. And is short of ELSE IF. ELIF, when used t
# - conditional statement, is essentially the bridge that connects their logic. The
# - first premise. The ELIF's have all sequential arguments/conditions. Ideally, th
# - remaining outlying possibilities.


# I utilized the 'Elif' keyword which is Pythons way of saying "if the previous cor
# then try this condition.
# I finish it off with an 'Else' statement, which isn't required. But since Else st
# the logical condition branch,
# I put it there because it's easier for me to proofread/debug. Knowing that this i


print("You rotate your chair to a nearby computer console and log into it.....")
time.sleep(2)
print("")

# The below variable assignments are important for the next function that the user

days_abroad = 256
```

```
# Simply an assignment for the ship's period of 'days abroad. As of yesterday, in t

days_abroad_n_plus_one = days_abroad + 1
# An assignment which performs an addition of '1' via the addition operator '+' whi
# as of today, in the storyline.

current_julian_date = 210
current_julian_date += 1
# Similar as the prior addition operation, but demonstrates the += addition shortcu
# Certain organizations utilize julian dates, particularly in the government sector
# implementing it's usage.

prior_day_fuel = 490320
prior_day_fuel_usage = 320
current_day_fuel = prior_day_fuel - prior_day_fuel_usage
# Two lines of assignment into a subtraction operation via the subtraction operator
# This is intended to display today's current fuel level (490000) after 'prior_day_
# from prior_day_fuel level '490320'.

fuel_percent_usage = ((current_day_fuel - prior_day_fuel) / prior_day_fuel)
# This is a calculation for percentage change in fuel levels via a subtraction oper
# There is also a division operator '/', or back slash, used in this line of code.
# The formula is (new amount - old amount) / old amount. Don't forget the parenthes
# The same rules of order of operations, as within mathematics, apply here.

days_fuel_remaining = current_day_fuel // prior_day_fuel_usage
# Here, I use a floor division arithmetic operator '//' as I only want to know the
# division.
# Not the remainder
# Basically, we're not interested in knowing a number like 30.2345 days of fuel rem
# We just want to know there's 30 days of fuel remaining, for the sake of brevity.

prior_day_expendables_level = 12984
prior_day_expendables_level_usage = 984
current_expendables_level = prior_day_expendables_level - prior_day_expendables_lev
organic_storage_percentage_change = \
    (current_expendables_level - prior_day_expendables_level) / prior_day_expendabl
approx_expendables_remaining = current_expendables_level // prior_day_expendables_l
# Similar to the prior block of variables, assignments, and arithmetic operators.
# This does the same except for expendables. Or a generic term for food.


current_year = 2116
current_solar_battery_levels = 10
gamma_ray_shielding = 25
# Assignments which are utilized to show the modulus and multiplication operator
# More on that further in the code


SITREP = True
while SITREP:
    print("")
    print("The panel reads as follows....")
    print("")
    time.sleep(2)
    print("Reconnaissance and Science Vessel Endeavor - Logistics and Communicatior
    print("Main Menu:")
    time.sleep(2)
    print("1. Send automated SITREP and recharge daily defensive measures.")
    time.sleep(2)
    print("2. Display a brief summary of yesterday's expendable metrics.")
    time.sleep(2)
    print("3. Log out.")
    time.sleep(2)
    # All of this is my attempt to help the player visualize what the computer pane
```

```
log_computer_input = input("\nPlease select an option: ")
if log_computer_input != "1" and log_computer_input != "2" and log_computer_inp
    # An example of a relational/comparison operator. "!=" means not equal to.
    # anything other than input that's NOT EQUAL to-
    # - "1" "2" or "3", there are given a response to re-enter a valid input.
    print("The console gives an error tone and asks for a valid input....")
    # A more "narrative driven" user input validation. The character in game re
    # subsequently the player.
    time.sleep(1)
    print("")
    # input validation
else:
    if log_computer_input == "3":
        print("")
        time.sleep(1)
        print("As you try to log out of the computer, you hear Abby chime in...
        time.sleep(2)
        print(
            "Abby: " + rank + ", I've already automated the metrics for the SIT
                              "the first option....please?")
        time.sleep(2)
        print("")
        # Basically, the ship's AI is nagging you back into sending the SITREP.
    else:
        if log_computer_input == "2":
            print("")
            print("Now compiling summary....")
            time.sleep(2)
            print("Days Abroad: " + str(days_abroad_n_plus_one) + "\nCurrent Fu
                current_day_fuel) + "\nCurrent Food Storage Amount: " + str(cur
            print("")
            time.sleep(2)
            # Just a much simpler version of the report
        else:
            if log_computer_input == "1":
                print("SITREP " * 5)
                time.sleep(2)
                # An example of string multiplication. The literal string 'SITR
                # The scrolling SITREP is meant to simulate a report banner.
                print("Current Julian Date: " + str(current_julian_date))
                # I have to format the integer value of 'current_julian_date' t
                # before the string operation of '+'
                print("Days Abroad: " + str(days_abroad_n_plus_one) + "\nCurrer
                    current_day_fuel))
                print("Current Fissile Percentage Change From Last Report: ", f
                      "%")
                # Here, I utilize the format function to force the percentage v
                # places
                time.sleep(2)
                print("Days of fuel remaining: " + str(days_fuel_remaining))
                print("Current Food Expendable Levels: " + str(current_expendab
                print("Current Expendable Percentage Change From Last Report: "
                      format(organic_storage_percentage_change, '4f'), "%")
                print("Diverting solar energy capacitance. Total energy reserve
                    current_solar_battery_levels ** 2))
                # An example of the exponential multiplication arithmetic opera
                time.sleep(2)
                print("Gamma Ray Shielding Levels: " + str(gamma_ray_shielding
                # An example of a multiplication arithmetic operator '*'.
                print("Current year is: " + str(current_year))
                print("Current year divided by 4 has a remainder of...." + str(
                # An example of a modulus arithmetic operator '%' in use.
                # When the program is executed, Python will divide the current
                # There are no logical conditions evaluating this.
                # This is simply meant to read, when the program is executed.
```

```
                    # That the current year, divided by 4, has no remainder. And wi
                    print("Thus, ISS, congratulations on making it to a Leap Year!'
                    # This is just the main character putting a random bit of flair
                    # A more creative and novel use of modulus will be used in a fu
                    time.sleep(2)
                    print("Nothing else significant to report.")
                    print(rank + " " + user_name + " out.")
                    time.sleep(1)
                    print("Now returning to main menu...")
                    time.sleep(1)
                    SITREP = False


def communications_array_old(num):
    """Function is a narrative mechanic to print a list of values from a text file'
    """Paramenter being passed is whether a certain interger indicates whether the
    access_card = num
    leave_room = False
    print("You step into the communication's array")
    while not leave_room:
        # The usage of the "Not" operator means the while loop will continue while
        user_choice = int(input(
            "Please make a selection:\n1.Start pinging the drones, as Abby requeste
        if access_card == 1 and user_choice == 1:
            list_elements = open('elements.txt')
            for index in range(1, 7):
                # From left to right, in this statement. The FOR loop will iterate
                # which is a list called 'elements.txt'
                # The IN statement specifies that the FOR loop pertains to the rang
                # list ends at entry six
                # The range() function will return a sequence of numbers, within th
                # Here, it will start at 1. And run until 7.
                list_off = list_elements.readline()
                print(str(index) + ". ", list_off)
                leave_room = True

        elif access_card == 2 and user_choice == 1:
            print("Wait, you don't have the access card. Head back to the cockpit a

        elif user_choice == 2:
            leave_room = True
            cockpit_old()
        else:
            print("Invalid input")


def cockpit_old():
    """Another narrative focused function which demonstrates parameter passing. Sp

    leave_room = True
    obtain_card = False
    while leave_room:
        user_choice = int(
            input("Make a selection:\n1. Head to the communications array\n2. Grab
        time.sleep(1)

        if user_choice < 1 or user_choice > 2:
            # The '>' and '<' comparison operators are greater than and less than,
            # Both those conditions are identifying any user input that isn't "1" o
            # The Boolean operator "or" is used to state that either of those prior
            # while loop.
            print("That isn't an option. Please select either 1 or 2")
        elif user_choice == 2 and obtain_card is False:
            print("Obtained common access card!")
            time.sleep(1)
```

```
            obtain_card = True
        elif user_choice == 2 and obtain_card is True:
            print("You already did that!")
            time.sleep(1)
        elif user_choice == 1 and obtain_card is True:
            print("You head to the communication's room")
            leave_room = False
            communications_array_old(1)
            time.sleep(1)
        elif user_choice == 1 and obtain_card is False:
            print("You head to the communication's room")
            leave_room = False
            communications_array_old(2)
            time.sleep(1)
        else:
            print("Invalid input. Please try again")


print("Abby: Great job! Alright " + rank + " " + user_name)
time.sleep(1)
print("Let's move on. Today's an exciting day! We're finally within communication's
time.sleep(1)
print(" that was fired to Proxima B almost 20 years ago!")
time.sleep(1)
print("Make sure to grab your Common Access Card and head to the communications arr
time.sleep(1)
print("And with that, the AI disappeared from your display.")
time.sleep(1)
print("Most likely to check some other systems while you head to the communications
time.sleep(1)
print("Select your next choice:")

cockpit_old()

time.sleep(1)
print(".........a few hours pass, and more tasks go by. This planet mirrors the atm
time.sleep(1)
print("An initial jubilation is cut off by a sudden sharp jarring that rocks your e
time.sleep(1)
print("You start to fade to black as Abby starts to yell warnings about a nearby su
time.sleep(1)
print("You barely are awake as her voice starts slurring and errors start plaguing


################################################################################
# The below function, 'monster movement' attempts to mimic a patrolling threat, thr
# The start of the function defines the possible 'rooms' that the 'monster' can mov
# This is explained again further down in the program but,
# Imagine a 3x3 square
# Each of those squares is named as followed, from top left to bottom right
# Storage Room - Engine Room (victory condition) - Mess Hall
# Battery Room - Reactor - Communications
# Solar Control Room - Cockpit - Oxygen Unit
# The below list defines the possibilities which the monster can move to.
# So, the bottom left room is "Solar Control Room". The only two rooms the 'monster
# Battery Room and Cockpit
# For the sake of brevity, the monster will not remain in a room and will continue
# This function will be set to execute whenever the user decides to move rooms and


def monster_movement(location1, location2):
    """Determines a monster's movement based around the room it is in and the rooms
    room_repeat = True
    monster_next_location = ""
    monster_last_location = location1
```

```
    monster_previous_to_last_location = location2
    # The above variables explained, in layman's terms:
    # monster_next_location is the initialization of a variable which will store wh
    # this function executes
    # monster_last_location is where the threat currently IS NOW, BEFORE this funct
    # monster_previous_to_last_location is where the monster was one move ago
    solar_list = ["battery", "cockpit"]
    cockpit_list = ["solar", "reactor", "oxygen"]
    oxygen_list = ["cockpit", "communication"]
    battery_list = ["solar", "reactor", "storage"]
    reactor_list = ["cockpit", "battery", "engine", "communication"]
    communication_list = ["oxygen", "reactor", "mess"]
    storage_list = ["engine", "battery"]
    engine_list = ["reactor", "mess", "storage"]
    mess_list = ["engine", "communication"]
    # The above are lists, which are 4 built in data types within python.
    # Lists items can be strings, integers, or booleans data types. Here, I just cr
    # to represent rooms. If the threat is in "solar", then the only possible rooms
    # and "cockpit"

    if monster_last_location == "solar":
        while room_repeat is True:
            monster_next_location = random.choice(solar_list)
            if monster_previous_to_last_location == monster_next_location:
                monster_next_location = random.choice(solar_list)
            else:
                room_repeat = False
                return monster_next_location
            # The choice() method function returns a randomly selected element from
            # random.choice() function would return a random string from the list l
            # In the above instance, randomly select a string from the list called
            # In order to prevent the threat from returning to a previous room, and
            # I included a 'while' loop that would keep cycling through possible ro
            # NOT the monster's immediately previous last location.

    elif monster_last_location == "cockpit":
        while room_repeat is True:
            monster_next_location = random.choice(cockpit_list)
            if monster_next_location == monster_previous_to_last_location:
                monster_next_location = random.choice(cockpit_list)
            else:
                room_repeat = False

                return monster_next_location

    elif monster_last_location == "oxygen":
        while room_repeat is True:
            monster_next_location = random.choice(oxygen_list)
            if monster_next_location == monster_previous_to_last_location:
                monster_next_location = random.choice(oxygen_list)
            else:
                room_repeat = False

                return monster_next_location

    elif monster_last_location == "battery":
        while room_repeat is True:
            monster_next_location = random.choice(battery_list)
            if monster_next_location == monster_previous_to_last_location:
                monster_next_location = random.choice(battery_list)
            else:
                room_repeat = False

                return monster_next_location
```

```
    elif monster_last_location == "reactor":
        while room_repeat is True:
            monster_next_location = random.choice(reactor_list)
            if monster_next_location == monster_previous_to_last_location:
                monster_next_location = random.choice(reactor_list)
            else:
                room_repeat = False

                return monster_next_location

    elif monster_last_location == "communication":
        while room_repeat is True:
            monster_next_location = random.choice(communication_list)
            if monster_next_location == monster_previous_to_last_location:
                monster_next_location = random.choice(communication_list)
            else:
                room_repeat = False

                return monster_next_location

    elif monster_last_location == "storage":
        while room_repeat is True:
            monster_next_location = random.choice(storage_list)
            if monster_next_location == monster_previous_to_last_location:
                monster_next_location = random.choice(storage_list)
            else:
                room_repeat = False

                return monster_next_location

    elif monster_last_location == "engine":
        while room_repeat is True:
            monster_next_location = random.choice(engine_list)
            if monster_next_location == monster_previous_to_last_location:
                monster_next_location = random.choice(engine_list)
            else:
                room_repeat = False

                return monster_next_location

    elif monster_last_location == "mess":
        while room_repeat is True:
            monster_next_location = random.choice(mess_list)
            if monster_next_location == monster_previous_to_last_location:
                monster_next_location = random.choice(mess_list)
            else:
                room_repeat = False

                return monster_next_location

    # All subsequent lines of code mirror the first explanation. Checking the room
    # the threat moved to.


##########################################################################################
# Using the logic from the prior IF statements explained in this function, the belo
# the monster against where the player is. The arrangement of the rooms should be m
# is in an adjacent room
def monster_location_check(location1, location2):
    """Checks proximity of the user to the monster to give a textual warning that t
    monster_location = location1
    player_location = location2
    if player_location == "cockpit" and (monster_location == "reactor" or monster_l
                                         monster_location == "oxygen"):
        return print("You hear something approaching...something grinding along the
```

```
    elif player_location == "solar" and (monster_location == "battery" or monster_l
        return print("You hear something approaching...something grinding along the
    elif player_location == "oxygen" and (monster_location == "cockpit" or monster_
        return print("You hear something approaching...something grinding along the
    elif player_location == "reactor" and (monster_location == "cockpit" or monster
                                           monster_location == "engine" or monster_
        return print("You hear something approaching...something grinding along the
    elif player_location == "battery" and (monster_location == "reactor" or monster
                                           monster_location == "solar"):
        return print("You hear something approaching...something grinding along the
    elif player_location == "communication" and (monster_location == "oxygen" or mo
                                                 monster_location == "mess"):
        return print("You hear something approaching...something grinding along the
    elif player_location == "mess" and (monster_location == "communication" or mons
        return print("You hear something approaching...something grinding along the
    elif player_location == "engine" and (monster_location == "mess" or monster_loc
                                          monster_location == "storage"):
        return print("You hear something approaching...something grinding along the
    elif player_location == "storage" and (monster_location == "engine" or monster_
        return print("You hear something approaching...something grinding along the
    else:
        return print("Whatever that thing is....it isn't in a nearby room. At least
    # The layout of the map/ship was kept simple. It goes:
    # Storage Room - Engine Room (victory condition) - Mess Hall
    # Battery Room - Reactor - Communications
    # Solar Control Room - Cockpit - Oxygen Unit
    # So, place each of those names in a 3x3 square and that's the 'map' or ship.
    # All of the lines above simply take the location of the monster and check it t
    # If the 'monster' is in a nearby room, they get a warning.
    # If the monster is more than a room away, then they are told so. However, not

    # This next function defines the initial room. And is named differently than th
    # i.e. the cockpit, because I didn't want there to be a chance of the player st


##############################################################################################
# This next function is the unfortunate instance of the user entering the same room
# Or passing by the threat. So, they both enter opposite rooms.
# And the user fails the 'hiding game'
# It's fairly straightforward, with an initially grim but hopefully encouraging sta
# user to enjoy the experience and not feel like they were tricked.
# The function is a 'game over' print statement with a while input validation loop.

def game_over():
    """Gives the player the option of starting the chase sequence over"""
    print("The face of the metallic hulk locks onto you...suddenly motionless. You
          " you. Moments of anxious silence pass as you weigh the option of turning
          " You wheel about, right"
          " at the same time you hear the churning behemoth suddenly come up behind
    player_exit = True
    print(".....or at least. That's one way this could have happened! Want to give
    while player_exit:
        player_choice = input("Press 1 for Yes or 2 for No:\n")
        if player_choice == "1":
            player_exit = False
            initial_room("engine", "engine")
        elif player_choice == "2":
            print("Thank you for playing!")

            sys.exit()
            # The sys module allows for the exit function. With no argument inputte
            # the default value is zero and is considered a successful termination.
    else:
        print("That wasn't a valid entry. Please try again.")
```

```
##############################################################################
# Barring more complicated code, this is a simple game to test the user's input.
# This definitely will need to be updated to something significantly more engaging,
# to fully code something like a quick time event.
# My original intent was to have a count down and the user must input a randomly sp
def hiding_game(location1, location2):
    """A input validation mini game that gives you one opportunity to enter the cor
    """Inputs where the user and monster are, and feeds it back into original room'
    monster_location = location1
    player_location = location2
    print("You step into the next room, and freeze up as you hear mechanical shambl
    time.sleep(1)
    print("You and that hulk are about to share the same room, as your eyes dart fo
    time.sleep(1)
    print("Press 1 to hide!")
    time.sleep(1)
    player_choice = input(":")
    if player_choice != "1":
        game_over()
    elif player_choice == "1":
        monster_location = hiding_status_danger(monster_location, player_location)
        return monster_location
    else:
        game_over()


##############################################################################
# This next function occurs usually after the hiding_game function. Or if the playe
# room.
# The function gets passed the location of the threat and player, does a few print
# monster_movement function to have it move away.
def hiding_status_danger(location1, location2):
    """A function which checks to see if the player is hiding and the threat/monste
    monster_location = location1
    player_location = location2
    print("Terrifying grinding noises fill the room as the AI's hulk lumbers in...'
    time.sleep(1)
    print("You remain hidden....and hope that the bots whirring visual sensors don'
    time.sleep(1)
    print("Eventually...it lumbers away....")
    monster_location = monster_movement(monster_location, player_location)
    hiding_status(monster_location, player_location)


##############################################################################
# Same as the hiding_status_danger function but no indication of the threat enterir
# Monster_movement function is called and the threat moves to a new room. So pullir
# performs actions cautiously, they can easily reach the engine room.

def hiding_status(location1, location2):
    """A function which gives a hiding player an indication that the monster is not
    monster_location = location1
    player_location = location2
    monster_location_check(monster_location, player_location)
    print("You remain hiding in the meanwhile....")
    return monster_location


##############################################################################
# Everything prior to this point was, more or less, getting the player into the mir
# simple 'Choose your own Adventure game'. Now, they've hopefully realized this is
# they're going to be hunted down by the ship's AI.
# I define an initial room function, separate from where the player starts. Which i
# makes things a little easier from a narrative point of view. Since the player wil
```

```
# opposed to waking up.

def initial_room(location1, location2):
    """A function where the 'chase game' starts in earnest. This the starting room
    player_left_room = False
    monster_current_room = location1
    monster_previous_room = location2
    player_current_room = "cockpit"
    print("-----")
    time.sleep(1)
    print("You awake on the floor, the ship dim with the exception of red emergency
    time.sleep(1)
    print("You go to rise to your feet but instinct suggests you remain low. And hi
    time.sleep(1)
    print(" motors and the grate of a metallic hulk lurching and dragging itself al
    time.sleep(1)
    print(" the ship's wall. It lurches its way past you. Garbled and disturbing te
    time.sleep(1)
    print(" mangled face. It's arms, normally sleek and appealing, are damaged with
    time.sleep(1)
    print(" Even given your condition, your 'gut' tells you that something is wrong
    time.sleep(1)
    print(" hulk and it's logic unstable. Normally calming, it's clear there's some
    time.sleep(1)
    print(" automations. The normally soothing voice of 'Abby' replaced with gitter
    time.sleep(1)
    print(" inefficiencies'. The six foot tall metal hulk scrapes it's way past you
    time.sleep(1)
    print(" After a few minutes of hiding, you hear nothing. And can assume the hul
    time.sleep(1)
    print(" ship. Inside the engine room. Where Abby's AI control unit is housed...
    time.sleep(1)
    print(" the ship stands a chance of repair....")
    time.sleep(1)
    monster_location_check(monster_current_room, player_current_room)
    # Monster location checks the monster's location against the player's location
    # As for now, the monster is far off in the engine room. However, once monster_
    # times, my hope is that the player will run into it. And have to hide or take
    time.sleep(1)
    while player_left_room is False:
        # Start of a while loop to ensure proper input validation
        print("What do you want to do?\n1. Examine the room\n2. Go left, to the sol
              " to the oxygen control unit\n4. Go forward into the reactor\n5. Hide
        time.sleep(2)
        player_choice = input("Make your selection here:")
        if player_choice == "1":
            print("The cockpit is mostly undamaged. However, dark with only the hue
            time.sleep(2)
            print("-----")
        elif player_choice == "2":
            monster_current_room = monster_movement(monster_current_room, monster_p

            # The player has opted to move to a different room, and so the 'threat'
            # function is called. And that is stored as the monster's current room.
            if monster_current_room == "solar":
                # A check to determine if the monster has moved into the same room
                # given a textual cue in that instance and will have to react in th
                player_left_room = True
                print("You head off to the West, into the solar control unit....")
                time.sleep(2)
                print("-----")
                monster_current_room = hiding_game("solar", player_current_room)
                # After a series of function calls, starting with 'hiding_game' if
                # then the threat moves by to a new room which is stored as its cur
                solar(monster_current_room)
```

```
        elif monster_current_room == "cockpit":
            # Also, if the monster literally moves into the room that the playe
            # to pass a hide challenge.
            player_left_room = True
            print("You head off to the West, into the solar control unit....")
            time.sleep(2)
            print("-----")
            monster_current_room = hiding_game("solar", player_current_room)
            solar(monster_current_room)
        else:
            player_left_room = True
            print("You head off to the West, into the solar control unit....")
            time.sleep(2)
            print("-----")
            solar(monster_current_room)
        # This block of elif and else statements is meant to test whether the p
        # threat in the next room. If the threat is going to move into the same
        # player and the threat are going to move by each other. If so, hiding_
        # And the  player/monster locations are passed to it.

    elif player_choice == "3":
        monster_current_room = monster_movement("engine", player_current_room)
        if monster_current_room == "oxygen":
            player_left_room = True
            print("You head off the East, into the oxygen control unit....")
            time.sleep(2)
            print("-----")
            monster_current_room = hiding_game("oxygen", player_current_room)
            oxygen(monster_current_room)
        elif monster_current_room == "cockpit":
            player_left_room = True
            print("You head off the East, into the oxygen control unit....")
            time.sleep(2)
            print("-----")
            monster_current_room = hiding_game("solar", player_current_room)
            oxygen(monster_current_room)
        else:
            player_left_room = True
            print("You head off the East, into the oxygen control unit....")
            time.sleep(2)
            print("-----")
            oxygen(monster_current_room)

    elif player_choice == "4":
        monster_current_room = monster_movement("engine", player_current_room)
        if monster_current_room == "reactor":
            player_left_room = True
            print("You head off the North, into the Reactor holding unit....")
            time.sleep(2)
            print("-----")
            monster_current_room = hiding_game("reactor", player_current_room)
            reactor(monster_current_room)
        elif monster_current_room == "cockpit":
            player_left_room = True
            print("You head off the North, into the Reactor holding unit....")
            time.sleep(2)
            print("-----")
            monster_current_room = hiding_game("solar", player_current_room)
            reactor(monster_current_room)
        else:
            player_left_room = True
            print("You head off the North, into the Reactor holding unit....")
            time.sleep(2)
            print("-----")
            reactor(monster_current_room)
```

```
        elif player_choice == "5":
            print("You look around for a locker, some debris, or otherwise somethir
            time.sleep(2)
            print("-----")
            monster_current_room = monster_movement("engine", player_current_room)
            if monster_current_room == "cockpit":
                hiding_status_danger(monster_current_room, player_current_room)
            else:
                hiding_status(monster_current_room, player_current_room)
        else:
            print("Not a valid input. Please try again.")
            time.sleep(2)
            print("-----")

    # From top to bottom, a while loop leading to a else statement ensures input va
    # room after their desired room and the monsters new room are compared. From th
    # is presented or the player just moves into the next room. The monster's 'curr
    # room/function.


################################################################################
# This is the Solar room function, mostly identical to the initial room. The 'game'
# point. The only real difference is that the threat's location will be passed alor
# opposed to being concretely defined as the engine room.

def solar(location1):
    """Another room focused function that gives options for subsequent rooms to mov
    player_left_room = False
    monster_current_room = location1
    player_current_room = "solar"
    print("-----")
    time.sleep(2)
    print("You've entered the solar control room...undamaged for the most part. Red
    time.sleep(2)
    print("The engine room is further to the North....the ship can't manage like th
    time.sleep(2)
    monster_location_check(monster_current_room, player_current_room)
    time.sleep(2)
    while player_left_room is False:
        print("What do you want to do?\n1. Examine the room\n2. Head north to the E
              "3. Head East to the Cockpit\n4. Hide and wait...")
        time.sleep(2)
        player_choice = input("Make your selection here:")
        if player_choice == "1":
            print("The solar control room doesn't look much better than the rest of
            time.sleep(2)
            print("-----")
        elif player_choice == "2":
            monster_current_room = monster_movement(monster_current_room, player_cu
            if monster_current_room == "battery":
                player_left_room = True
                print("You head off to the North, into the battery control unit....
                time.sleep(2)
                print("-----")
                monster_current_room = hiding_game(monster_current_room, player_cur
                battery(monster_current_room)
            elif monster_current_room == "solar":
                player_left_room = True
                print("You head off to the North, into the battery control unit....
                time.sleep(2)
                print("-----")
                monster_current_room = hiding_game(monster_current_room, player_cur
                battery(monster_current_room)
            else:
```

```
                player_left_room = True
                print("You head off to the North, into the battery control unit....
                time.sleep(2)
                print("-----")
                battery(monster_current_room)

        elif player_choice == "3":
            monster_current_room = monster_movement(monster_current_room, player_cu
            if monster_current_room == "cockpit":
                player_left_room = True
                print("You head off to the East, into the cockpit area again....")
                time.sleep(2)
                print("-----")
                monster_current_room = hiding_game(monster_current_room, player_cur
                cockpit(monster_current_room)
            elif monster_current_room == "solar":
                player_left_room = True
                print("You head off to the East, into the cockpit area again....")
                time.sleep(2)
                print("-----")
                monster_current_room = hiding_game(monster_current_room, player_cur
                cockpit(monster_current_room)
            else:
                player_left_room = True
                print("You head off to the East, into the cockpit area again....")
                time.sleep(2)
                print("-----")
                cockpit(monster_current_room)

        elif player_choice == "4":
            print("You look around for a locker, some debris, or otherwise somethir
            time.sleep(2)
            print("-----")
            monster_current_room = monster_movement(monster_current_room, player_cu
            if monster_current_room == "cockpit":
                hiding_status_danger(monster_current_room, player_current_room)
            else:
                hiding_status(monster_current_room, player_current_room)
        else:
            print("Not a valid input. Please try again.")
            time.sleep(2)
            print("-----")


##############################################################################
# Again, just a function definition for the cockpit room

def cockpit(location1):
    """Another room focused function that gives options for subsequent rooms to mov
    player_left_room = False
    monster_current_room = location1
    player_current_room = "cockpit"
    print("-----")
    time.sleep(2)
    print("You've re-entered the cockpit...")
    time.sleep(2)
    monster_location_check(monster_current_room, player_current_room)
    time.sleep(2)
    while player_left_room is False:
        print("What do you want to do?\n1. Examine the room\n2. Go left, to the sol
              " to the oxygen control unit\n4. Go forward into the reactor\n5. Hide
        time.sleep(2)
        player_choice = input("Make your selection here:")
        if player_choice == "1":
            print("The cockpit is mostly undamaged. However, dark with only the hue
```

```
            time.sleep(2)
            print("-----")
        elif player_choice == "2":
            monster_current_room = monster_movement(monster_current_room, player_cu
            if monster_current_room == "solar":
                player_left_room = True
                print("You head off to the West, into the solar control unit....")
                time.sleep(2)
                print("-----")
                monster_current_room = hiding_game(monster_current_room, player_cur
                solar(monster_current_room)
            elif monster_current_room == "cockpit":
                player_left_room = True
                print("You head off to the West, into the solar control unit....")
                time.sleep(2)
                print("-----")
                monster_current_room = hiding_game(monster_current_room, player_cur
                solar(monster_current_room)
            else:
                player_left_room = True
                print("You head off to the West, into the solar control unit....")
                time.sleep(2)
                print("-----")
                solar(monster_current_room)

        elif player_choice == "3":
            monster_current_room = monster_movement(monster_current_room, player_cu
            if monster_current_room == "oxygen":
                player_left_room = True
                print("You head off the East, into the oxygen control unit....")
                time.sleep(2)
                print("-----")
                monster_current_room = hiding_game(monster_current_room, player_cur
                oxygen(monster_current_room)
            elif monster_current_room == "cockpit":
                player_left_room = True
                print("You head off the East, into the oxygen control unit....")
                time.sleep(2)
                print("-----")
                monster_current_room = hiding_game(monster_current_room, player_cur
                oxygen(monster_current_room)
            else:
                player_left_room = True
                print("You head off the East, into the oxygen control unit....")
                time.sleep(2)
                print("-----")
                oxygen(monster_current_room)

        elif player_choice == "4":
            monster_current_room = monster_movement(monster_current_room, player_cu
            if monster_current_room == "reactor":
                player_left_room = True
                print("You head off the North, into the Reactor holding unit....")
                time.sleep(2)
                print("-----")
                monster_current_room = hiding_game(monster_current_room, player_cur
                reactor(monster_current_room)
            elif monster_current_room == "cockpit":
                player_left_room = True
                print("You head off the North, into the Reactor holding unit....")
                time.sleep(2)
                print("-----")
                monster_current_room = hiding_game(monster_current_room, player_cur
                reactor(monster_current_room)
            else:
```

```
                player_left_room = True
                print("You head off the North, into the Reactor holding unit....")
                time.sleep(2)
                print("-----")
                reactor(monster_current_room)

        elif player_choice == "5":
            print("You look around for a locker, some debris, or otherwise somethir
            time.sleep(2)
            print("-----")
            monster_current_room = monster_movement(monster_current_room, player_cu
            if monster_current_room == "cockpit":
                hiding_status_danger(monster_current_room, player_current_room)
            else:
                hiding_status(monster_current_room, player_current_room)
        else:
            print("Not a valid input. Please try again.")
            time.sleep(2)
            print("-----")


#############################################################################
# Function definition for the reactor room, in the very center of the map

def reactor(location1):
    """Another room focused function that gives options for subsequent rooms to mov
    player_left_room = False
    monster_current_room = location1
    player_current_room = "reactor"
    print("-----")
    time.sleep(2)
    print("The reactor room is normally very quiet. Now, deathly so.")
    time.sleep(2)
    monster_location_check(monster_current_room, player_current_room)
    time.sleep(2)
    while player_left_room is False:
        print("What do you want to do?\n1. Examine the room\n2. Go West, to the bat
        print(" Engine Room.\n4. Go East, to the Communication Control Unit\n5. Go
              "6. Hide and wait")
        time.sleep(2)
        player_choice = input("Make your selection here:")
        if player_choice == "1":
            print("You stare at the multi-billion dollar nuclear reactor core, and
            time.sleep(2)
            print(" robot running around the ship. Typical Monday.")
            time.sleep(2)
            print("-----")
        elif player_choice == "2":
            monster_current_room = monster_movement(monster_current_room, player_cu
            if monster_current_room == "battery":
                player_left_room = True
                print("You head off to the West, into the battery control unit....'
                time.sleep(2)
                print("-----")
                monster_current_room = hiding_game(monster_current_room, player_cur
                battery(monster_current_room)
            elif monster_current_room == "reactor":
                player_left_room = True
                print("You head off to the West, into the battery control unit....'
                time.sleep(2)
                print("-----")
                monster_current_room = hiding_game(monster_current_room, player_cur
                battery(monster_current_room)
            else:
                player_left_room = True
```

```
            print("You head off to the West, into the battery control unit....'
            time.sleep(2)
            print("-----")
            battery(monster_current_room)

    elif player_choice == "3":
        monster_current_room = monster_movement(monster_current_room, player_cu
        if monster_current_room == "engine":
            player_left_room = True
            print("You head off to the North, into the engine room....")
            time.sleep(2)
            print("-----")
            monster_current_room = hiding_game(monster_current_room, player_cur
            engine(monster_current_room)
        elif monster_current_room == "reactor":
            player_left_room = True
            print("You head off to the North, into the engine room....")
            time.sleep(2)
            print("-----")
            monster_current_room = hiding_game(monster_current_room, player_cur
            engine(monster_current_room)
        else:
            player_left_room = True
            print("You head off to the North, into the engine room....")
            time.sleep(2)
            print("-----")
            engine(monster_current_room)

    elif player_choice == "4":
        monster_current_room = monster_movement(monster_current_room, player_cu
        if monster_current_room == "communication":
            player_left_room = True
            print("You head off to the East, into the Communication Array....")
            time.sleep(2)
            print("-----")
            monster_current_room = hiding_game(monster_current_room, player_cur
            communication(monster_current_room)
        elif monster_current_room == "reactor":
            player_left_room = True
            print("You head off to the East, into the Communication Array....")
            time.sleep(2)
            print("-----")
            monster_current_room = hiding_game(monster_current_room, player_cur
            communication(monster_current_room)
        else:
            player_left_room = True
            print("You head off to the East, into the Communication Array....")
            time.sleep(2)
            print("-----")
            communication(monster_current_room)

    elif player_choice == "5":
        monster_current_room = monster_movement(monster_current_room, player_cu
        if monster_current_room == "cockpit":
            player_left_room = True
            print("You head off to the South, into the cockpit....")
            time.sleep(2)
            print("-----")
            monster_current_room = hiding_game(monster_current_room, player_cur
            cockpit(monster_current_room)
        elif monster_current_room == "reactor":
            player_left_room = True
            print("You head off to the South, into the cockpit....")
            time.sleep(2)
            print("-----")
```

```
                monster_current_room = hiding_game(monster_current_room, player_cur
                cockpit(monster_current_room)
            else:
                player_left_room = True
                print("You head off to the South, into the cockpit....")
                time.sleep(2)
                print("-----")
                cockpit(monster_current_room)

        elif player_choice == "6":
            print("You look around for a locker, some debris, or otherwise somethir
            time.sleep(2)
            print("-----")
            monster_current_room = monster_movement(monster_current_room, player_cu
            if monster_current_room == "reactor":
                hiding_status_danger(monster_current_room, player_current_room)
            else:
                hiding_status(monster_current_room, player_current_room)

        else:
            print("Not a valid input. Please try again.")
            time.sleep(2)
            print("-----")


################################################################################
# Function definition for the oxygen room, at the bottom right  of the map

def oxygen(location1):
    """Another room focused function that gives options for subsequent rooms to mov
    player_left_room = False
    monster_current_room = location1
    player_current_room = "oxygen"
    print("-----")
    time.sleep(2)
    print("The Oxygen Control Unit is dim but functioning.")
    time.sleep(2)
    print("For how long given the ship's current condition? Who knows.")
    time.sleep(2)
    monster_location_check(monster_current_room, player_current_room)
    while player_left_room is False:
        print("What do you want to do?\n1. Examine the room\n2. Go West, to the coc
        print(" Communication Array.\n4. Hide and wait.")
        time.sleep(2)
        player_choice = input("Make your selection here:")
        if player_choice == "1":
            print("Scattered tough boxes and the sound of air hissing through filte
            time.sleep(2)
            print("Nothing else of note can be seen.")
            time.sleep(2)
            print("-----")
        elif player_choice == "2":
            monster_current_room = monster_movement(monster_current_room, player_cu
            if monster_current_room == "cockpit":
                player_left_room = True
                print("You head off to the West, into the cockpit....")
                time.sleep(2)
                print("-----")
                monster_current_room = hiding_game(monster_current_room, player_cur
                cockpit(monster_current_room)
            elif monster_current_room == "oxygen":
                player_left_room = True
                print("You head off to the West, into the cockpit....")
                time.sleep(2)
                print("-----")
```

```
                monster_current_room = hiding_game(monster_current_room, player_cur
                cockpit(monster_current_room)
            else:
                player_left_room = True
                print("You head off to the West, into the cockpit....")
                time.sleep(2)
                print("-----")
                cockpit(monster_current_room)

        elif player_choice == "3":
            monster_current_room = monster_movement(monster_current_room, player_cu
            if monster_current_room == "communication":
                player_left_room = True
                print("You head off to the North, into the Communication Array....'
                time.sleep(2)
                print("-----")
                monster_current_room = hiding_game(monster_current_room, player_cur
                communication(monster_current_room)
            elif monster_current_room == "oxygen":
                player_left_room = True
                print("You head off to the North, into the Communication Array....'
                time.sleep(2)
                print("-----")
                monster_current_room = hiding_game(monster_current_room, player_cur
                communication(monster_current_room)
            else:
                player_left_room = True
                print("You head off to the North, into the Communication Array....'
                time.sleep(2)
                print("-----")
                communication(monster_current_room)

        elif player_choice == "4":
            print("You look around for a locker, some debris, or otherwise somethir
            time.sleep(2)
            print("-----")
            monster_current_room = monster_movement(monster_current_room, player_cu
            if monster_current_room == "oxygen":
                hiding_status_danger(monster_current_room, player_current_room)
            else:
                hiding_status(monster_current_room, player_current_room)
        else:
            print("Not a valid input. Please try again.")
            time.sleep(2)
            print("-----")


################################################################################
# Function definition for the battery room, in the mid left portion of the map

def battery(location1):
    """Another room focused function that gives options for subsequent rooms to mov
    player_left_room = False
    monster_current_room = location1
    player_current_room = "battery"
    print("-----")
    time.sleep(2)
    print("The battery unit hums with the drone of thousands of ohms of current. Th
    time.sleep(2)
    print("is reassuring, regardless of the hulking menace somewhere on the ship...
    time.sleep(2)
    monster_location_check(monster_current_room, player_current_room)
    time.sleep(2)
    while player_left_room is False:
        print("What do you want to do?\n1. Examine the room\n2. Go North, to the st
```

```
print("3. Go East, to the Reactor.\n4. Go South, to the Solar Control Unit\
time.sleep(2)
player_choice = input("Make your selection here:")
if player_choice == "1":
    print("Giant capacitors line the walls, with ample opportunities to hic
    time.sleep(2)
    print("-----")
elif player_choice == "2":
    monster_current_room = monster_movement(monster_current_room, player_cu
    if monster_current_room == "storage":
        player_left_room = True
        print("You head off to the North, into the storage unit....")
        time.sleep(2)
        print("-----")
        monster_current_room = hiding_game(monster_current_room, player_cur
        storage(monster_current_room)
    elif monster_current_room == "battery":
        player_left_room = True
        print("You head off to the North, into the storage unit....")
        time.sleep(2)
        print("-----")
        monster_current_room = hiding_game(monster_current_room, player_cur
        storage(monster_current_room)
    else:
        player_left_room = True
        print("You head off to the North, into the battery control unit....
        time.sleep(2)
        print("-----")
        storage(monster_current_room)

elif player_choice == "3":
    monster_current_room = monster_movement(monster_current_room, player_cu
    if monster_current_room == "reactor":
        player_left_room = True
        print("You head off to the East, into the reactor....")
        time.sleep(2)
        print("-----")
        monster_current_room = hiding_game(monster_current_room, player_cur
        reactor(monster_current_room)
    elif monster_current_room == "battery":
        player_left_room = True
        print("You head off to the East, into the reactor....")
        time.sleep(2)
        print("-----")
        monster_current_room = hiding_game(monster_current_room, player_cur
        reactor(monster_current_room)
    else:
        player_left_room = True
        print("You head off to the East, into the reactor....")
        time.sleep(2)
        print("-----")
        reactor(monster_current_room)

elif player_choice == "4":
    monster_current_room = monster_movement(monster_current_room, player_cu
    if monster_current_room == "solar":
        player_left_room = True
        print("You head off to the South, into the Solar Array....")
        time.sleep(2)
        print("-----")
        monster_current_room = hiding_game(monster_current_room, player_cur
        solar(monster_current_room)
    elif monster_current_room == "battery":
        player_left_room = True
        print("You head off to the South, into the Solar Array....")
```

```
                time.sleep(2)
                print("-----")
                monster_current_room = hiding_game(monster_current_room, player_cur
                solar(monster_current_room)
            else:
                player_left_room = True
                print("You head off to the South, into the Solar Array....")
                time.sleep(2)
                print("-----")
                solar(monster_current_room)

        elif player_choice == "5":
            print("You look around for a locker, some debris, or otherwise somethir
            time.sleep(2)
            print("-----")
            monster_current_room = monster_movement(monster_current_room, player_cu
            if monster_current_room == "battery":
                hiding_status_danger(monster_current_room, player_current_room)
            else:
                hiding_status(monster_current_room, player_current_room)
        else:
            print("Not a valid input. Please try again.")
            time.sleep(2)
            print("-----")


################################################################################
# Function definition for the communication room, in the mid right portion of the m

def communication(location1):
    """Another room focused function that gives options for subsequent rooms to mov
    player_left_room = False
    monster_current_room = location1
    player_current_room = "communication"
    print("-----")
    time.sleep(2)
    print("Normally the communication array is alive with lit panels showing a stea
    time.sleep(2)
    print(". However, those same panels are plastered with error messages and excep
    time.sleep(2)
    print(" the ship to malfunction this badly....")
    time.sleep(2)
    monster_location_check(monster_current_room, player_current_room)
    time.sleep(2)
    while player_left_room is False:
        print("What do you want to do?\n1. Examine the room\n2. Go South, to the Ox
        print(" Reactor.\n4. Go North, to the Mess Hall\n5. Hide and wait.")
        time.sleep(2)
        player_choice = input("Make your selection here:")
        if player_choice == "1":
            print("You take note of the grim errors lining the panels...not many pl
            time.sleep(2)
            print("-----")
        elif player_choice == "2":
            monster_current_room = monster_movement(monster_current_room, player_cu
            if monster_current_room == "oxygen":
                player_left_room = True
                print("You head off to the South, to the oxygen unit....")
                time.sleep(2)
                print("-----")
                monster_current_room = hiding_game(monster_current_room, player_cur
                oxygen(monster_current_room)
            elif monster_current_room == "communication":
                player_left_room = True
                print("You head off to the South, to the oxygen unit....")
```

```
            time.sleep(2)
            print("-----")
            monster_current_room = hiding_game(monster_current_room, player_cur
            oxygen(monster_current_room)
        else:
            player_left_room = True
            print("You head off to the South, to the oxygen unit....")
            time.sleep(2)
            print("-----")
            oxygen(monster_current_room)

    elif player_choice == "3":
        monster_current_room = monster_movement(monster_current_room, player_cu
        if monster_current_room == "reactor":
            player_left_room = True
            print("You head off to the West, into the reactor....")
            time.sleep(2)
            print("-----")
            monster_current_room = hiding_game(monster_current_room, player_cur
            reactor(monster_current_room)
        elif monster_current_room == "communication":
            player_left_room = True
            print("You head off to the West, into the reactor....")
            time.sleep(2)
            print("-----")
            monster_current_room = hiding_game(monster_current_room, player_cur
            reactor(monster_current_room)
        else:
            player_left_room = True
            print("You head off to the West, into the reactor....")
            time.sleep(2)
            print("-----")
            reactor(monster_current_room)

    elif player_choice == "4":
        monster_current_room = monster_movement(monster_current_room, player_cu
        if monster_current_room == "mess":
            player_left_room = True
            print("You head off to the North, into the Mess Hall....")
            time.sleep(2)
            print("-----")
            monster_current_room = hiding_game(monster_current_room, player_cur
            mess(monster_current_room)
        elif monster_current_room == "communication":
            player_left_room = True
            print("You head off to the North, into the Mess Hall....")
            time.sleep(2)
            print("-----")
            monster_current_room = hiding_game(monster_current_room, player_cur
            mess(monster_current_room)
        else:
            player_left_room = True
            print("You head off to the North, into the Mess Hall....")
            time.sleep(2)
            print("-----")
            mess(monster_current_room)

    elif player_choice == "5":
        print("You look around for a locker, some debris, or otherwise somethir
        time.sleep(2)
        print("-----")
        monster_current_room = monster_movement(monster_current_room, player_cu
        if monster_current_room == "battery":
            hiding_status_danger(monster_current_room, player_current_room)
        else:
```

```
                hiding_status(monster_current_room, player_current_room)
        else:
            print("Not a valid input. Please try again.")
            time.sleep(2)
            print("-----")


################################################################################
# This is the Storage room n in the top left of the map

def storage(location1):
    """Another room focused function that gives options for subsequent rooms to mov
    player_left_room = False
    monster_current_room = location1
    player_current_room = "storage"
    print("-----")
    time.sleep(2)
    print("You've entered the storage unit...undamaged for the most part.")
    time.sleep(2)
    print("The engine room is just to the East!")
    time.sleep(2)
    monster_location_check(monster_current_room, player_current_room)
    time.sleep(2)
    while player_left_room is False:
        print("What do you want to do?\n1. Examine the room\n2. Head South to the E
              "3. Head East to the Engine\n4. Hide and wait...")
        time.sleep(2)
        player_choice = input("Make your selection here:")
        if player_choice == "1":
            print("You can practically see the Engine room from here....past all th
            time.sleep(2)
            print("-----")
        elif player_choice == "2":
            monster_current_room = monster_movement(monster_current_room, player_cu
            if monster_current_room == "battery":
                player_left_room = True
                print("You head off to the South, into the battery control unit....
                time.sleep(2)
                print("-----")
                monster_current_room = hiding_game(monster_current_room, player_cur
                battery(monster_current_room)
            elif monster_current_room == "storage":
                player_left_room = True
                print("You head off to the South, into the battery control unit....
                time.sleep(2)
                print("-----")
                monster_current_room = hiding_game(monster_current_room, player_cur
                battery(monster_current_room)
            else:
                player_left_room = True
                print("You head off to the South, into the battery control unit....
                time.sleep(2)
                print("-----")
                battery(monster_current_room)

        elif player_choice == "3":
            monster_current_room = monster_movement(monster_current_room, player_cu
            if monster_current_room == "engine":
                player_left_room = True
                print("You head off to the East, into the Engine Room....")
                time.sleep(2)
                print("-----")
                monster_current_room = hiding_game(monster_current_room, player_cur
                engine(monster_current_room)
            elif monster_current_room == "storage":
```

```
                player_left_room = True
                print("You head off to the East, into the Engine Room....")
                time.sleep(2)
                print("-----")
                monster_current_room = hiding_game(monster_current_room, player_cur
                engine(monster_current_room)
            else:
                player_left_room = True
                print("You head off to the East, into the Engine Room....")
                time.sleep(2)
                print("-----")
                engine(monster_current_room)

        elif player_choice == "4":
            print("You look around for a locker, some debris, or otherwise somethir
            time.sleep(2)
            print("-----")
            monster_current_room = monster_movement(monster_current_room, player_cu
            if monster_current_room == "cockpit":
                hiding_status_danger(monster_current_room, player_current_room)
            else:
                hiding_status(monster_current_room, player_current_room)
        else:
            print("Not a valid input. Please try again.")
            time.sleep(2)
            print("-----")


#############################################################################################
# Function for the cockpit room and the victory condition for this stage

def engine(location1):
    """The player wins should they reach this function"""
    player_left_room = False
    monster_current_room = location1
    player_current_room = "engine"
    print("-----")
    time.sleep(2)
    print("The Engine Room! An alternate power unit powering the ship's AI is acros
    time.sleep(2)
    print("You dash across the room....right as you hear the roque hulk stampeding
    time.sleep(2)
    print("You reach out, and with one massive pull, yank the J5 cable powering the
    time.sleep(2)
    print("And subsequently the hulk....you turn around in time to see a massive me
    time.sleep(2)
    print("The hulk, separated from the AI's logic, has a fail safe to lock all mot
    time.sleep(2)
    print("You've done it....now the bigger issue of getting the ship fully back or


#############################################################################################
# Function definition for the mess hall, at the top right of the map

def mess(location1):
    """Another room focused function that gives options for subsequent rooms to mov
    player_left_room = False
    monster_current_room = location1
    player_current_room = "mess"
    print("-----")
    time.sleep(2)
    print("The mess hall and living area looms in the dark...however, the Engine Rc
    time.sleep(2)
    monster_location_check(monster_current_room, player_current_room)
    while player_left_room is False:
```

```
print("What do you want to do?\n1. Examine the room\n2. Go West, to the Eng
print(" Communication Array.\n4. Hide and wait.")
time.sleep(2)
player_choice = input("Make your selection here:")
if player_choice == "1":
    print("A single wall mounted bunk, dining area, and tiny lounge can be
    time.sleep(2)
    print("-----")
elif player_choice == "2":
    monster_current_room = monster_movement(monster_current_room, player_cu
    if monster_current_room == "engine":
        player_left_room = True
        print("You head off to the West, into the Engine Room....")
        time.sleep(2)
        print("-----")
        monster_current_room = hiding_game(monster_current_room, player_cur
        engine(monster_current_room)
    elif monster_current_room == "mess":
        player_left_room = True
        print("You head off to the West, into the Engine Room....")
        time.sleep(2)
        print("-----")
        monster_current_room = hiding_game(monster_current_room, player_cur
        engine(monster_current_room)
    else:
        player_left_room = True
        print("You head off to the West, into the Engine Room....")
        time.sleep(2)
        print("-----")
        engine(monster_current_room)

elif player_choice == "3":
    monster_current_room = monster_movement(monster_current_room, player_cu
    if monster_current_room == "communication":
        player_left_room = True
        print("You head off to the South, into the Communication Array....'
        time.sleep(2)
        print("-----")
        monster_current_room = hiding_game(monster_current_room, player_cur
        communication(monster_current_room)
    elif monster_current_room == "mess":
        player_left_room = True
        print("You head off to the South, into the Communication Array....'
        time.sleep(2)
        print("-----")
        monster_current_room = hiding_game(monster_current_room, player_cur
        communication(monster_current_room)
    else:
        player_left_room = True
        print("You head off to the South, into the Communication Array....'
        time.sleep(2)
        print("-----")
        communication(monster_current_room)

elif player_choice == "4":
    print("You look around for a locker, some debris, or otherwise somethir
    time.sleep(2)
    print("-----")
    monster_current_room = monster_movement(monster_current_room, player_cu
    if monster_current_room == "oxygen":
        hiding_status_danger(monster_current_room, player_current_room)
    else:
        hiding_status(monster_current_room, player_current_room)
else:
    print("Not a valid input. Please try again.")
```

```
            time.sleep(2)
            print("-----")


initial_room("engine", "engine")
print("-----")
print("-----")
print("-----")
print("Congratulations on making it to the engine room! Thank you for playing! That
```