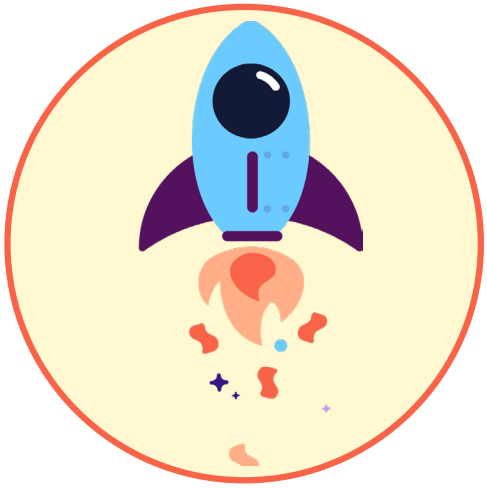
MOONHACK 2025



Astrolock

Program an escape room. Will you get out of the locked space shuttle before it takes off to the moon?

INTRODUCTION

Write a program to create an escape room. This project has three steps that progressively build in difficulty. Stop at the end of step 1, 2 or complete the whole project.

What you will need

HARDWARE

A computer

SOFTWARE - (ANY PYTHON SOFTWARE)

Raspberry Pi Editor (online)

or

Mu (download)

DOWNLOADS

To download Mu follow the instructions on this page

What you will learn

- Getting user **input**
- **Printing text** to screen
- **while loops**

Additional notes for educators

Check out our blog post for this project with tips, curriculum, and supporting material at **medium.com/@codeclubau**

Check out the code along video on our YouTube channel.

STEP 1 - A SIMPLE ESCAPE ROOM

Create a new program.

Start your program by importing the time tool. This will be used to slow down the text on screen, making it feel like a story is being told.

main.py

```
import time
```

Create a function called slow_print and define it. This will slow the speed of the text on screen.

main.py

```
import time

def slow_print(text):
    print(text)
    time.sleep(1.5)
```

To start the game, we need to bring the user into the story. Use slow print to introduce the game and let them know the 3 places they can look for a key. The emojis are optional. To add these use the keyboard shortcut, Windows key and full stop. Let's test the program so far. Add a start game function to enable testing but remember that this will be the final line of the program.

```
    print(text)
    time.sleep(1.5)

def start_game():
    slow_print("🔒You wake up locked inside the Space Shuttle!")
    slow_print("You have to escape before the doors are ready for take off to the Moon.")
    slow_print("\nWhere do you want to look?")
    slow_print("1.  Utility Wall")
    slow_print("2. Fuselage")
    slow_print("3. Compartment")

start_game()
```

Now the user needs to choose which location they will look in. Create a variable called choice1 that will store their choice.

```
    slow_print("\nWhere do you want to look?")
    slow_print("1. Utility Wall")
    slow_print("2. Fuselage")
    slow_print("3. Compartment")

    choice1 = input("What will you check first? (1, 2, or 3): ")
```

Let's use an if/elif/else to check for which room the user selects. If they type something other than a number it will restart the choice.

```
    slow_print("3. Compartment")

    choice1 = input("What will you check first? (1, 2, or 3): ")

    if choice1 == "1":
        utility()
    elif choice1 == "2":
        fuselage()
    elif choice1 == "3":
        compartment()
    else:
        slow_print("That's not a valid choice. Try again.")
        start_game()
```

Now we will define what happens for each of these choices. Let's start with the utility wall. This is where the key is hidden. Add text to tell the story and then set has_key to true so the program remembers we have found the key.

```
        start_game()

def utility():
    slow_print("\n You pull back the utility wall and see a small locked box.")
    slow_print("You find a note: 'Look behind the whiteboard.'")
    slow_print("You go to the whiteboard and find a small key taped behind it.")
    slow_print("This might be useful later.")
    has_key = True
    next_room(has_key)
```

Next we will define fuselage. This room does not have the key so next_room is false.

```
    key_found = True
    next_room(key_found)

def fuselage():
    slow_print("\n You try to open the fuselage.")
    slow_print("Looks like you'll need a key.")
    next_room(False)
```

Now define the compartment. Also, no key here, but an added bit of fun with a glitter bomb. Add a start game function so we can test the game.

```
def compartment():
    slow_print("\n You open the compartment.")
    slow_print("Uh oh! A glitter bomb goes off! ")
    slow_print("You cough and laugh, but there's nothing useful inside.")
    next_room(False)

start_game()
```

Test your program. You should see the story told and then you can input a choice of where to look. The corresponding text should then print.

Next, we need to set the parameters to get off the shuttle, or not.

If the player has found the key (they chose the Utility room), we need them to have an option to use it and escape the shuttle. **Continue your program above the start_game function

```
main.py
def compartment():
    slow_print("\n You open the compartment.")
    slow_print("Uh oh! A glitter bomb goes off! ")
    slow_print("You cough and laugh, but there's nothing useful inside.")
    next_room(False)

def next_room(has_key):
    slow_print("\nYou spot a door in the back of the space shuttle. It might be your way out.")
    action = input("Do you want to try opening the door? (yes or no): ")
```

Let's define what action is to be taken according to the player's answer. If they say yes to escape the shuttle, there are two paths - one if they found the key and one if they didn't.

```
def next_room(has_key):
    slow_print("\nYou spot a door in the back of the space shuttle. It might be your way out.")
    action = input("Do you want to try opening the door? (yes or no): ")

    if action.lower() == "yes":
        if has_key:
            slow_print("\n You use the key you found.")
            slow_print("The door clicks open! You're free! ")
            slow_print(" YOU ESCAPED THE SPACE SHUTTLE! ")
        else:
            slow_print("\n The door is locked. You need a key.")
            slow_print("You couldn't escape in time and the space shuttle takes off. GAME OVER.")
```

If they answered no to opening the door, or a weird answer, let's program what will happen.

```
        else:
            slow_print("\n The door is locked. You need a key.")
            slow_print("You couldn't escape in time and the space shuttle takes off. GAME OVER.")
    elif action.lower() == "no":
        slow_print("\nYou decide to stay... and visit the moon? ")
        slow_print("Well, that's one way to end the game.")
    else:
        slow_print("That's not a valid answer.")
        next_room(has_key)
```

Your start_game function should now be on line 65 and the game is complete.

```
            slow_print("You couldn't escape in time and the space shuttle takes off. GAME OVER.")
    elif action.lower() == "no":
        slow_print("\nYou decide to stay... and visit the moon? ")
        slow_print("Well, that's one way to end the game.")
    else:
        slow_print("That's not a valid answer.")
        next_room(has_key)

start_game()
```

Run your program. You should be able to play the whole escape room from start to end. Test multiple different pathways - find the key and escape, find the key and don't escape, don't find the key and try the door etc.

STEP 2 - ADDING MULTIPLE ROOM CHECKS

Return to line 9. Create a variable called has_key and set it to false (they do not have the key). Create another variable called explored_rooms and set it to create a list of rooms that the user has visited.

```
6
7  def start_game():
8      slow_print("🔒 You wake up locked inside the Space Shuttle!")
9      slow_print("You have to escape before the doors are locked for take off to the M
10
11     has_key = False
12     explored_rooms = set()
```

Add a while True loop above the next 4 lines of print text. We need to ensure that from line 15 -30 everything is moved one indentation to the right, so that it stays within the loop.

```
11     has_key = False
12     explored_rooms = set()
13
14     while True:
15         slow_print("\nWhere do you want to look?")
16         slow_print("1. Utility Wall")
17         slow_print("2. Fuselage")
```

We need to add code to each of the room choices (1, 2, or 3) that the user makes so that once they have seen the outcome of their choice, they can search another room. We do this by storing their choices in the explored_rooms variable. The Utility Room is first.

```
22         if choice == "1":
23             if "utility" not in explored_rooms:
24                 has_key = utility()
25                 explored_rooms.add("utility")
26             else:
27                 slow_print("You already searched the Utility Wall.")
28         elif choice == "2":
```

Now make the changes for the fuselage.

```
29             if "fuselage" not in explored_rooms:
30                 fuselage()
31                 explored_rooms.add("fuselage")
32             else:
33                 slow_print("You already searched the Fuselage.")
34         elif choice == "3":
```

Now make the changes for the compartment.

```
34         elif choice == "3":
35             if "compartment" not in explored_rooms:
36                 compartment()
37                 explored_rooms.add("compartment")
38             else:
39                 slow_print("You already searched the Compartment.")
40         else:
41             slow_print("That's not a valid choice.")
42             continue
```

Replace the start_game with continue, as we don't want them to go back to the start of the game. If the player found the key let's give them the option to exit the game by breaking the loop.

```
            if 'compartment' not in explored_rooms:
                has_key = compartment()
                explored_rooms.add("compartment")
            else:
                slow_print("You already searched the compartment.")
        else:
            slow_print("That's not a valid choice.")
            continue
        if has_key:
            break
```

If they haven't found the key give them the option to exit the game or check another room.

```
        if has_key:
            break

        retry = input("\nDo you want to try the exit or check another room? (exit/check):").lower()
        if retry == "exit":
            break

    next_room(has_key)

```

For each of the rooms that are checked we need to adapt the code to the new conditions we've set. Delete the functions next_room and key_found.

```
    slow_print("\n📦 You pull back the utility wall and see a small locked box.")
    slow_print("You find a note: 'Look behind the whiteboard.'")
    slow_print("You go to the whiteboard and find a small key taped behind it.")
    slow_print("This might be useful later.")
    return True

def fuselage():
    slow print("\n🚪 You try to open the fuselage.")
```

```

def fuselage():
    slow_print("\n🚪 You try to open the fuselage.")
    slow_print("Looks like you'll need a key.")
    return False

```

```

def compartment():
    slow_print("\n🎁 You open the compartment.")
    slow_print("Uh oh! A glitter bomb goes off! 🎆")
    slow_print("You cough and laugh, but there's nothing useful inside.")
    return False
```

Run your program. You should be able to play the whole escape room from start to end, having the option to try all of the rooms for clues. Test multiple different pathways - find the key and escape, find the key and don't escape, don't find the key and try the door etc.

Code not working? Check your indentations for each line of code. When adding in the explore_rooms function, everything underneath while True has moved one indentation to the right.

STEP 3 - ADDING RIDDLES FOR EACH ROOM

This iteration of the code will randomise where the key is hidden, meaning a change in some of our existing code. At the start of our program import the random tool.

main.py

```
1  import time
2  import random
```

Delete has_key at line 12 and add new code. Set the key_location variable to be global, meaning it is used across all parts of the program. Then set up the 3 locations that can be randomly selected from.

```
6      time.sleep(1.5)
7
8  def start_game():
9      slow_print("🔒 You wake up locked inside the Space Shuttle!")
10     slow_print("You have to escape before the doors are locked for take off to the Moon.")
11
12     global key_location
13     key_location = random.choice(["utility wall", "fuselage", "compartment"])
14     explored_room()
15
```

Let's define the function explored_room. Keep the has_key function. Change line 18 to be checked_spots as a variable.

```
11
12     global key_location
13     key_location = random.choice(["utility wall", "fuselage", "compartment"])
14     explored_room()
15
16 def explored_room():
17     has_key = False
18     checked_spots = set()
```

On line 20 change the code to while not has_key as the check in point.

```
17     has_key = False
18     checked_spots = set()
19
20     while not has_key:
21         slow_print("\nWhere do you want to look?")
22         slow print("1. Utility Wall")
```

When the user makes a choice of room, the program needs to check two things at once. Delete the current if else code for the first room, and replace it with code to check if this is the room with the key and to add the choice to check_spots.

```
23         slow_print("2. Fuselage")
24         slow_print("3. Compartment")
25
26         choice1 = input("What will you check? (1, 2, or 3): ")
27
28         if choice1 == "1" and "utility wall" not in checked_spots:
29             has_key = utility()
30             checked_spots.add("utility wall")
```

Complete similar programming for rooms 2 and 3.

```
30            checked_spots.add("utility wall")
31        elif choice1 == "2" and "fuselage" not in checked_spots:
32            has_key = fuselage()
33            checked_spots.add("fuselage")
```

```
32            has_key = fuselage()
33            checked_spots.add("fuselage")
34        elif choice1 == "3" and "compartment" not in checked_spots:
35            has_key = compartment()
36            checked_spots.add("compartment")
```

Add in another else if statement that if the user selects a room they have already been to, it will tell them so.

```
34        elif choice1 == "3" and "compartment" not in checked_spots:
35            has_key = compartment()
36            checked_spots.add("compartment")
37        elif choice1 in ["1", "2", "3"]:
38            slow_print("You've already checked that. Try somewhere else!")
39        else:
40            slow_print("That's not a valid choice.")
41
```

Delete the code in rows 41 - 47 as this no longer suits our program. The next line of code should now read next_room(has_key)

```
37        elif choice1 in ["1", "2", "3"]:
38            slow_print("You've already checked that. Try somewhere else!")
39        else:
40            slow_print("That's not a valid choice.")
41
42    next_room(has_key)
43
44 def utility():
```

Run your program. You should be able to play the whole escape room from start to end, but it will still run with the key in the utility wall, as our definitions for each room haven't changed. Let's work on them now. Each room will have a puzzle to solve as well.

The utility room will have a number puzzle to solve. Delete lines 46 - 49 Add in code to ask the user the riddle.

```
41
42    next_room(has_key)
43
44 def utility():
45    slow_print("\n📦 You pull back the utility wall and see a small locked box.")
46    slow_print("It has a 3-digit code lock. A sticky note on top says:")
47    slow_print('"What number do you get if you multiply all the numbers on a phone keypad?" 🤔')
48
49    code = input("Enter the 3-digit code to unlock the box: ")
50
```

The answer is 000. Program what happens if they enter the correct answer, and check this against the location of the key. Also add in what happens if they answer incorrectly.

```
    if code == "000":
        if key_location == "utility wall":
            slow_print("Correct! The box pops open")
            slow_print("Inside is a small key taped under the lid. You got it!")
            return True
        else:
            slow_print("\nCorrect, but the box is empty.  Just a joke inside.")
            return False
    else:
        slow_print("\nWrong code.  The box stays locked")
        return False
```

The fuselage is going to have a word riddle. Change the words that are printed to provide the riddle.

```

def fuselage():
    slow_print("\n  You try to open the fuselage. It has a word puzzle:")
    slow_print('"I have hands but no arms, and a face but no eyes. What am I?"')

    answer = input("Enter your answer: ").strip().lower()

```

The answer is clock. Program what happens if the correct answer is entered, and check this against the location of the key. Also add in what happens if they answer incorrectly.

```
    answer = input("Enter your answer").strip().lower()

    if answer == "clock":
        if key_location == 'fuselage':
            slow_print("\nCorrect! The fuselage unlocks")
            slow_print("You find a key taped behind the door.")
            return True
        else:
            slow_print("\n Correct - but there's only a cap, no key.")
            return False
    else:
        slow_print("\nWrong answer.  The fuselage stays locked.")
        return False
```

The final puzzle is to do with colours.

```

def compartment():
    slow_print("\n🌈 You open the compartment and find a colourful card with a riddle:")
    slow_print('"red + blue = ?"')

    answer = input("What colour do you get? ").strip().lower()

```

The answer is purple. Program what happens if they answer the questions correctly or not.

```
    answer = input("What colour do you get? ").strip().lower()

    if answer == "purple":
        if key_location == "compartment":
            slow_print("\n✅ Correct! Hidden under the card is a key! 🔑")
            return True
        else:
            slow_print("\n✅ Correct, but nothing else inside... except some glitter. ✨")
            return False
    else:
        slow_print("\n❌ That's not it. Suddenly — 💥 a glitter bomb goes off!")
        return False
```

The program is complete! Run your program. Can you find the key and escape?

Challenges:

Different puzzles and compartments

The escape room is quite short with only 3 locations to look in. Can you add more choices and different puzzles to make the game longer and more in depth?

Hints

A person playing this game may find some of the puzzles hard. How can you give them hints to help them figure out answers? Or perhaps they can find clues while they are looking around that will help them answer the questions.

More levels

Imagine that the door at the end of this game only leads to the next segment of the Space Shuttle, not the actual exit. Can you build more levels into the game?

Congratulations you're a Moonhack changemaker! You can try one of our other projects or try one of the challenges.

Don't forget to talk to an adult about registering your participation and downloading your certificate at moonhack.com