PPA Assignment 6

This coursework is designed to test the content from Weeks 1, 2, 3, 4, 5 and 7.

Do you think there is a problem with any of the content below? Let us know immediately at programming@kcl.ac.uk.

Read through this brief carefully before starting your attempted solution and make sure you follow it as closely as possible, paying particular attention to the names required for elements of your code. Also ensure that you comment your code, as part of the assignment documentation.

The total marks available for this assignment are 7.

For this week's assessment, consider the following scenario, and then complete the tasks that follow it:

We would like to create a simple door maze game. In this game, there are a number of rooms arranged sequentially. The player starts in the first room and must travel through each room in the sequence until they reach the final room, in which case they win the game. However, in order to go from one room to the next in the sequence, they must choose between two doors: a blue door or a red door. Note that the door which they used to enter the room is ignored, as the player cannot go back. Only one of the two doors leads to the next room, whereas the other lets a monster into the current room, which attacks the player before retreating back through the door which it came from. This results in the player losing a life. The player may pick a door again if they have at least one life left, otherwise they lose the game.

1. Model this scenario based on the following requirements:

1. A `Room` has a `name`. Also, it contains a `blueDoorRoom`, which is the room that the player will go to if they choose the blue door. Similarly, it also has a `redDoorRoom`, which is the room that the player will go to if they choose the red door. Next, it has a status `containsMonster`, specifying that either the room does, or does not, contain the monster. Lastly, it has a status `isFinalRoom`, indicating whether or not the room is the final one. (1 mark)
2. A `Player` has a `name`, a number of `lives` left and the `currentRoom` which they are in at any point in the game (1 mark)
3. A Player can `move` to a supplied `room`. This returns a result stating if the chosen room was the correct one or incorrect one to select. This works as follows:
    - The room may be the next valid room in the sequence, in which case the player moves to the room.
    - Otherwise, the room contains the monster which will attack the player, resulting in the player losing a life.

    (1 mark)

2. Create a class `DoorMazeGame`, which can be compiled and run from the command line. Use this class to do the following (in order), using the classes and methods you have created for Question 1.

    1. Create the following rooms:
        - Monster Room:
            - the variable name must be *monsterRoom*
            - the `name` as *The Monster Room*
            - the `containsMonster` status as *true*
        - Room 1:
            - the variable name must be *room1*
            - the `name` as *Chamber One*
            - the `blueDoorRoom` as *room2*
            - the `redDoorRoom` as *monsterRoom*
        - Room 2:

        - the variable name must be *room2*
        - the `name` as *Chamber Two*
        - the `blueDoorRoom` as *monsterRoom*
        - the `redDoorRoom` as *room2*
    - Room 3:
        - the variable name must be *room3*
        - the `name` as *Chamber Three*
        - the `blueDoorRoom` as *room4*
        - the `redDoorRoom` as *monsterRoom*
    - Room 4:
        - the variable name must be *room4*
        - the `name` as *Chamber Four*
        - the `blueDoorRoom` as *monsterRoom*
        - the `redDoorRoom` as *room5*
    - Room 5:
        - the variable name must be *room5*
        - the `name` as *Chamber Five*
        - the `blueDoorRoom` as *monsterRoom*
        - the `redDoorRoom` as *room6*
    - Room 6:
        - the variable name must be *room6*
        - the `name` as *Chamber Six*
        - the `isFinalRoom` status as *true*

(1 mark)

2. Create a player, give them a maximum of 2 lives and start them off in room1. (1 mark)

3. Create the main game loop using the following guidelines:

    - The game should first output an introductory message to the user, informing them of the game rules.
    - Ask the user to input their name, and update the player's name in the game accordingly.
    - Next, repeatedly present the status of the game to the user, including the player's name, lives left and details of the current room. This should also repeatedly ask the user to input their choice of door, either "blue" or "red". Make the player `move` to the selected room. Any other input from the user should result in a message being output stating the acceptable commands they can enter, along with their meaning.
    - Ensure that when the player's lives reach 0, they lose the game.
    - Ensure that when the player's reaches the final room they win the game.

  (2 mark)

Once completed, submit your assignment using the link marked `Assignment 6: Nexus Submission Link' on KEATS.

You must complete the plagiarism and collusion training before submitting this assignment.

You must also submit complete documentation of your solution. You will find a sample piece of documentation in the Support section on KEATS marked `Sample Assignment Documentation'. Submit your documentation using the link marked `Assignment 6: Documentation Submission' on KEATS.

Students who do not submit documentation along with their code, or vice-versa, will receive a mark of zero.

Any submitted code or documentation that is found to be unduly similar to the code or documentation submitted by any other student(s), will result in a penalty for those involved.

Provisional mark for your code will be released on KEATS within one week of submission. Final assignment grades will be submitted to the exam board at the end of the semester, and will take into consideration the quality of your documentation and the quality of the comments written into your code directly.

For all other queries, see the Support section on KEATS, specifically the document marked `Introduction'.