Directions

- **Complete Chapter 8 "Scripting Game Mechanics"** in the Learning C# textbook by Ferrone.
- You are expected to do everything in the chapter for this submission--and beware: the book doesn't always number the steps!
- The summary below is to help you know what to show in your recordings. It's NOT a substitute to reading the chapter.

Chapter 8 Summary in Brief

1. Update the **PlayerBehavior** script so hitting the J key causes **jump**.
    a. *Discrepancy*: the text says it's the spacebar that causes a jump, but it's the **J key**.
    b. *Note*: when the instructions tell you to make an IF structure in FixedUpdate() that says `if(IsGrounded() && _isJumping)`, it REPLACES the `if(_isJumping)` structure you had earlier.
2. Select **Environment** set its Layer = **Ground** (Yes, change children). Alter PlayerBehavior to have an **IsGrounded()** method to check **`public LayerMask GroundLayer`** so Player can't jump while in the air.
    a. *Note*: be sure the Raised Platform and Ramps are assigned to be on the Ground layer, too.
3. Prefab - **Bullet**: Sphere, Scale 0.15, yellow material, Rigidbody, deleted from Hierarchy.
4. Update PlayerBehavior script so **FixedUpdate() Instantiates a Bullet** when **left mouse button** pressed.
    a. *Discrepancy*: the text mistakenly says it's the left mouse button that fires a bullet, but it's the **spacebar** key. You may wish to decrease bullet speed from 100 to 15 if it's too fast to see.
5. Bullet has **BulletBehavior** script, **deleting** itself after 3 sec.
6. Empty game object **Game_Manager** has **GameBehavior.cs** script that counts **_itemsCollected** and **_playerHP**, but private so they need public variables **Items** and **HP** with getters and setters.
7. Add to the **ItemBehavior** script on the Pickup, so collision **adds to Items** var.
8. Creating a GUI: UI > "Text – TextMeshPro" and name it Health. (When prompted, choose "Import TMP Essentials.") Anchor = Top Left, Rect Transform position to x110 y–35 to position it in the upper-left [*Typo*: The book says "right"] corner. Set Text = "Health", choose black for the Vertex Color [and I had to lessen its Font Size].
9. UI > "Text – TextMeshPro" and name it **Items**. Anchor = Top Left, Rect Transform position to x 110, y –85, Text = "Items " [and choose black for the Vertex Color and lessen its Font Size]
10. UI > "Text – TextMeshPro" and name it **Progress**. Anchor = Top Left, Rect Transform position to x0 y15, Width=435, Text = "Collect all items to win!" [and choose black for the Vertex Color and lessen its Font Size]
11. Update GameBehavior to collect an item and display Progress text when Items or HP change (with each pickup and when all are picked up). Select Game_Manager in the Hierarchy and drag over our three text objects into their corresponding GameBehavior script fields in the Inspector.

12. **Win condition**: Alter GameBehavior to **make a "WinButton" setActive** once all items are found. Set Max Items to 1 for now.

    a. *Clarification*: Where you add the code for the Button, you still need the rest of the code you had before (the public TMP_Text properties, the public int HP property, etc). ALL you have to add are these lines, in their proper places:

```
using UnityEngine.UI;
public Button WinButton;
WinButton.gameObject.SetActive(true);
```

13. UI > "**Button** -TextMeshPro" named **Win Condition**, position x0 y0, w 225 h115, Text="You won!", un-check to hide it at first.

    a. *Typo*: Where the instructions step 4 say "Select the Win Condition parent object again and click the checkmark icon in the upper right of the Inspector:" it means upper-left, to the left of the name "Win Condition."

14. **Game restarts when you pick up the Item:** Add **using** UnityEngine.SceneManagement to GameBehavior and manipulate **Time.timeScale** to enable pause and restart when win occurs. Select Win Condition, in Inspector's **OnClick** section assign the Game_Manager and its GameBehavior | **RestartScene ().** Go to Window > Rendering > Lighting, select Generate Lighting at the bottom, and ensure Auto Generate is not selected. (This step is necessary to address a Unity issue that reloads scenes without any lighting.)