1. A charged jump, the longer the player holds the jump button down, the higher they jump.

2. Plan:

- Implement a jump function so that when the player can use the spacebar to jump.
    - Alter this implementation so that the player does jump not on button press, but when button is released.
- Create a variable jump to keep track of how long the player holds down the spacebar to jump.
- Use the variable jump and multiply it by some other variable to measure how high the player should jump, setting an upper limit on how high the player can jump.

3. My plan modified with the help of ChatGpt:

1. Create variable to track:
    - Minimum jump force (When player just taps jump)
    - Maximum jump force (Max jump height)
    - Charge time (How long space bar was held down for)
    - Max charge time variable
2. When the player begins holding down the spacebar, start recording the time in the ChargeTime variable.
3. Calculate the force of the charged jump and multiply it by the jump force.
4. Execute the jump when the player releases the spacebar and reset the ChargeTime variable back to 0 so the next jump starts fresh.
5. Limit the player from double jumping by making sure they can only jump while they are on the ground.

4.

```
using UnityEngine;
```

```
public class ChargedJump : MonoBehaviour
{
    public float minJumpForce = 5f;
    public float maxJumpForce = 15f;
    public float maxChargeTime = 2f;
    private float chargeTime = 0f;

    void Start()
    {

    }

    void Update()
    {
        // Start charging while holding space
        if (Input.GetKey(KeyCode.Space))
        {
            // Only increase chargeTime while grounded
            if (IsGrounded())
            {
                chargeTime += Time.deltaTime;
                chargeTime = Mathf.Min(chargeTime, maxChargeTime); // Clamp to max
            }
        }

        // When releasing space, perform jump
        if (Input.GetKeyUp(KeyCode.Space) && IsGrounded())
        {
            float t = chargeTime / maxChargeTime; // Normalized charge
            float jumpForce = Mathf.Lerp(minJumpForce, maxJumpForce, t);
            rb.AddForce(Vector2.up * jumpForce, ForceMode2D.Impulse);

            chargeTime = 0f; // Reset charge
        }
    }

    // Simple ground check using raycast
    bool IsGrounded()
 {
        // Assumes ground is on layer "Ground" and object has collider at feel
        float extraHeight = 0.1f;
        RaycastHit2D hit = Physics2D.Raycast(transform.position,
Vector2.down,extraHeight);
```

```
        return hit.collider != null;
    }
}
```

5. After running my prompt through ChatGPT, my plan was pretty similar to the AI generated plan. I did not really think about implementing some feature to make sure the player was on the ground, because with my plan the player would have been able to jump forever, which is not something that I intended. Also, I did not think about what would happen if the player simply tapped the jump button instead of holding it down for a length of time. ChatGPT gave me the idea of implementing a minimum jump force to fix this issue, so there would be a set jump height if the player only tapped jump. The suggestions from the AI provided my idea with some really good ideas, because I am still very new to programming this kind of feature, it was greatly appreciated to receive its advice.

After creating a test scene and using the code provided by ChatGPT, I was able to successfully make a simple game frame where the player could perform the charged jump. I did not go into depth with creating a playable character but made a little circle that could do so. I did have to mess with the code a little bit, it took some trial and error but for the most part the code provided by ChatGPT was reliable. I already have implemented AI into my first project, since this project was the first time I have ever attempted to make a game, there are many things I do not know how to do and ChatGPT supplies an answer very quickly when I am trying to figure out how to do something. I feel like as I continue to learn how to do things I will be relying on it less and less, but at this moment when everything is still very fresh I believe AI is a powerful tool to help you learn if you know how to use it. Obviously AI is not perfect and will not provide flawless lines of code every time, but finding how to fix the code it provides gives me valuable learning experience in how to troubleshoot my own code in the future.