3rd Person Controller (no jump)

Packages needed:
Cinemachine

1. First make an environment for your player to move around. Add a plane and some cubes to the scene. Make a new game object and call it environment', dragging everything you added to the scene underneath it. Then give them all a new layer called 'Ground'. This layer will be used in the script for it to be easily identifiable.
2. Drag a cylinder into the scene to be the player and scale the size to your desire. Remove the capsule collider and give it the 'player' tag. This tag will make it easier to tell objects how to interact with it and how it should interact with the code.
3. Then, right click the cylinder you just made and add a cube to indicate which direction the player is facing and remove its box collider.
4. Make an empty game object by right clicking in the hierarchy and name it 'ThirdPersonPlayer' (TPP), reset its position and make it parent your cylinder player object. This is so the whole player is together as one object.
5. Go to Window, Packet manager and search for cinemachine in the Unity registry. Once downloaded, go to the cinemachine tab and add a 'free look' camera to the project. Rename it to 'ThirdPersonCamera'.(TPC). A free look camera automatically provides a third person view.
6. Go down to extensions and add a cinemachine collider. Make it collide against the 'Ground' layer, Ignore the player tag and change the strategy to pull the camera forward. This is so the camera doesn't go past the ground surface, blocking the view of the player. The closer the mouse is to the player the closer the camera view will be to the player.
7. Drag the 'ThirdPersonPlayer' game object into the 'Follow' and 'Look at' boxes in the 'ThirdPersonCamera' inspector; so the main focus of the camera view is the player.
8. Then find the lens section in the TPC and reduce the size of the near clip plane to 0.01 so the player can get closer to an object without the camera going through it.
9. Find the orbits section in the TPC, in the BottomRig height and radius, reduce the height so that the bottom line is almost touching the ground and increase the size of the radius to 4. Increase the height of the MiddleRig and TopRig and then the radius of the MiddleRig to 5.5 and the radius of the TopRIg the same as the BottomRig. You can change these values to position the camera how you want it.
10. On your TPP object, add a character controller component, make sure to adjust the radius and height to the size of the player cylinder. You add a character controller to allow the player to move easily constrained by collisions, without needing a rigid body because it isn't affected by force.Then add a new script, calling it 'Movement'.
11. Open the movement script and delete the void start section because you don't need it, but leave the void update. Make 4 public variables, one being a character controller named 'controller' another being a float called 'speed', next one being a transform called ' cam', another float called 'turnSmoothTime' and finally a private float called 'turnSmoothVeloctiy'. We make these public so we can reference them in the Unity inspector. Controller is to get the character controller on the TPP, speed is so we can adjust how fast we want the player to move, cam is to get reference to the

TPC position and turnSmoothVelocity is the speed at which the player can turn left or right.

12. In the void update, make a float called ‘h’ for and make it equal to Input.GetAxisRaw open brackets with quotation marks type “Horizontal”. This means when we press the A, D or the left and right arrow keys the player will move left or right. On the next line do the same, but instead of the float being called ‘h’, call it ‘v’, and instead of “Horizontal”, type “Vertical”. This means when we press W, S or the up and down arrow keys the player will move forwards or backwards.
13. Make a new Vector3 called direction and make it equal to a new Vector3(h, 0f, v).normalized; This new Vector direction will calculate the direction the player should move in depending on the keys being pressed, but we do not want the player moving up and down the Y axis. Normalized just means if we are holding 2 keys it wont move faster.
14. Make a new if statement for if the direction and magnitude is greater than 0.1 and the player is moving (f meaning float and a float represents fraction numbers or numbers that are too big or too small to be an integer). If (direction.magnitude >= 0.1f), inside make a new float called ‘targetAngle’ and make it equal to Mathf.Atan2(direction.x, direction.y) * Mathf.Rad2Deg + cam.eulerAngles.y; This just means the player will move in the direction they are facing based on their rotation. Atan is a mathematical function that gives an angle on the X axis. We use Rad2Deg to convert radians to degrees.
15. Next, make another float called ‘angle’ and make it equal to Mathf.SmoothDampAngle(transform.eulerAngles.y, targetAngle, ref turnSmoothVelocity, turnSmoothTime); This is to set the rotation of the player by getting the current angle of the player and the target angle. turnSmoothTime and tunrSmoothVelocity will smooth the angles in Unity.
16. Then type transform.rotation = Quaternion.Euler(0f, targetAngle, 0f); to make the rotation of the player smooth using the smoothed angles.
17. After this, make a new Vector3 called ‘moveDir’ for move direction and make it equal to Quaternion.Euler(0f, targetAngle, 0f) * Vector3.forward; To make it move in the direction of the angle the player is rotated at and facing.
18. Finally type controller.Move(direction*speed*Time.deltaTime); To make the player controller move the player using the direction Vector3 made, speed value that's in Unity units and delaTime to make the movement the same on all devices running the code.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

Unity Script (1 asset reference) | 0 references
public class Movement : MonoBehaviour
{
    public CharacterController controller;
    public float speed;
    public Transform cam;

    public float turnSmoothTime;
    float turnSmoothVelocity;

    // Update is called once per frame
    Unity Message | 0 references
    void Update()
    {
        float h = Input.GetAxisRaw("Horizontal");
        float v = Input.GetAxisRaw("Vertical");
        Vector3 direction = new Vector3(h, 0f, v).normalized;

        if (direction.magnitude >=0.1f)
        {
            float targetAngle = Mathf.Atan2(direction.x, direction.y) * Mathf.Rad2Deg + cam.eulerAngles.y;
            float angle = Mathf.SmoothDampAngle(transform.eulerAngles.y, targetAngle, ref turnSmoothVelocity, turnSmoothTime);
            transform.rotation = Quaternion.Euler(0f, targetAngle, 0f);

            Vector3 moveDir = Quaternion.Euler(0f, targetAngle, 0f) * Vector3.forward;
            controller.Move(moveDir.normalized * speed * Time.deltaTime);
        }
    }
}
```