C++ Paradigm

Explanation of C++ and Object Oriented programming

C++ is a language that uses the Object Oriented paradigm. Built in the 1980s, C++ was designed as the next iteration of C, with the key addition of being able to use objects. Objects are instances of a class. They bundle variables and functions related to that object. Objects are modular and can be used throughout a program in many ways, such as inheritance and encapsulation.

How my code conforms to the Object Oriented paradigm

Throughout my code, strong object oriented practices are implemented. I have two classes, one for the game and one for the buildings. It is sound to separate these as doing so adheres to the concept of singular responsibility. The building class can take care of all responsibilities related to the player buildings while the game class carries out more logic for the overall game flow. The game class doesn't need to know the inner workings of everything in the building class, but may want to call upon it for some tasks and info. A prime example is encapsulation and accessing members which I use extensively. The game class hosts two building objects and these can be seen to call functions that lie in the building class.

Why I chose C++

C++ is very popular for game development due to its performance. While speed in my game isn't needed for something of this simplicity, the benefits of Object Oriented programming that the language is also very beneficial. Being able to have a building class is a cleaner way of coding. As someone with no prior C++ experience and as someone who's interested in game development, I imagined the challenge would be exciting and instrumental.