C++ Paradigm

When developing a solution in C++ I decided to use an object-orientated approach. OOP has a large focus on abstraction, encapsulation, polymorphism, and inheritance. When developing in C++ I ensured that many of the features in the code were converted into classes, such as the board becoming its class, as well as each boat. Instances of each could then be made and manipulated. Each class acted as a blueprint, and then when creating each instance, the boats could be manipulated and formed to the requirements. In addition, by coding each class and creating instances it greatly reduces duplication of code. Encapsulation wraps up variables and methods in classes. Having a display board method, meant whenever the console board needed to be updated, the method could just be called instead of worrying about including loops nested inside each other, making the code extremely confusing. In addition, it also meant that when generating boats and placing them on the grid, there was also less repetition of code. This also meant that the main method could be kept extremely clean and have a lot less code. This also meant that error handling and finding were a lot easier as you could easily isolate a class and test for errors.