|  |  |
| --- | --- |
| **CM2010 Software design and development** | |
|  | |
| **Student Name:** | Owen Lee Wei Hern |
| **Date Submitted:** | 16th December 2022 |
| **Degree Title:** | Computer Science |
| **Local Institution:** | Singapore Institute of Management |
| **Student ID:** | 220218799 |

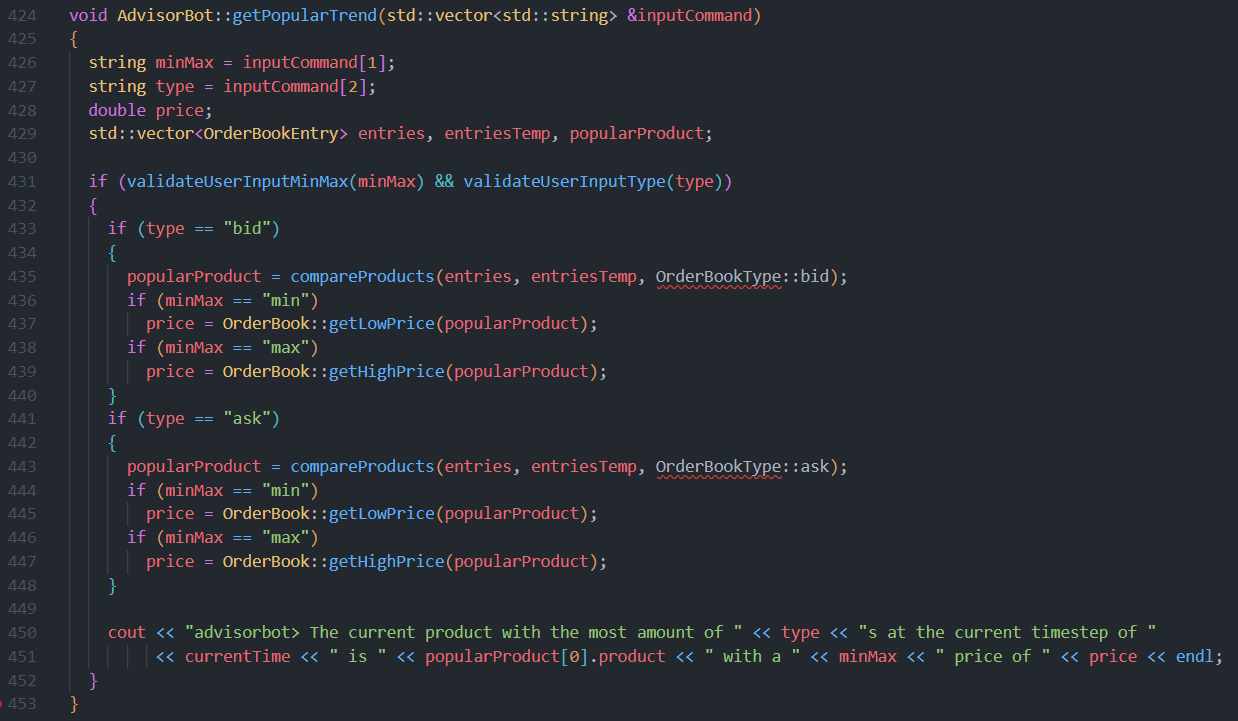
**Part 1**

For this task, I will be taking a look at the merkelsim program I developed during my Object-Oriented Programming module.

**Description of the purpose of the program and the technology it uses**

The purpose of this program was to teach us how to work with classes and code using an Object-oriented programming paradigm using a real-world example, in this case, the stock/crypto market. We create a bot that can perform functions on the cryptocurrency market for us, functions like fetching the time, the high and low prices and asking for predictions. This program was coded using C++ as the language and the technologies that helped me achieve this were Visual Studio Code, GitHub and the GNU Compiler.

**Module coupling example 1: Common Environment Coupling**



In the code extract above, this is a function within the AdvisorBot class there is common environment coupling between 3 classes. The OrderBookEntry and OrderBook classes are contained within the AdvisorBot class. This makes the AdvisorBot class the global environment and it is able to access the functions located in OrderBook as can be seen in lines 437, 439, 445 and 447 and the OrderBookEntry’s type in line 429. However, the OrderBook class contains its own functions and data is being shared within its own environment.

**Module coupling example 2: Control Coupling**





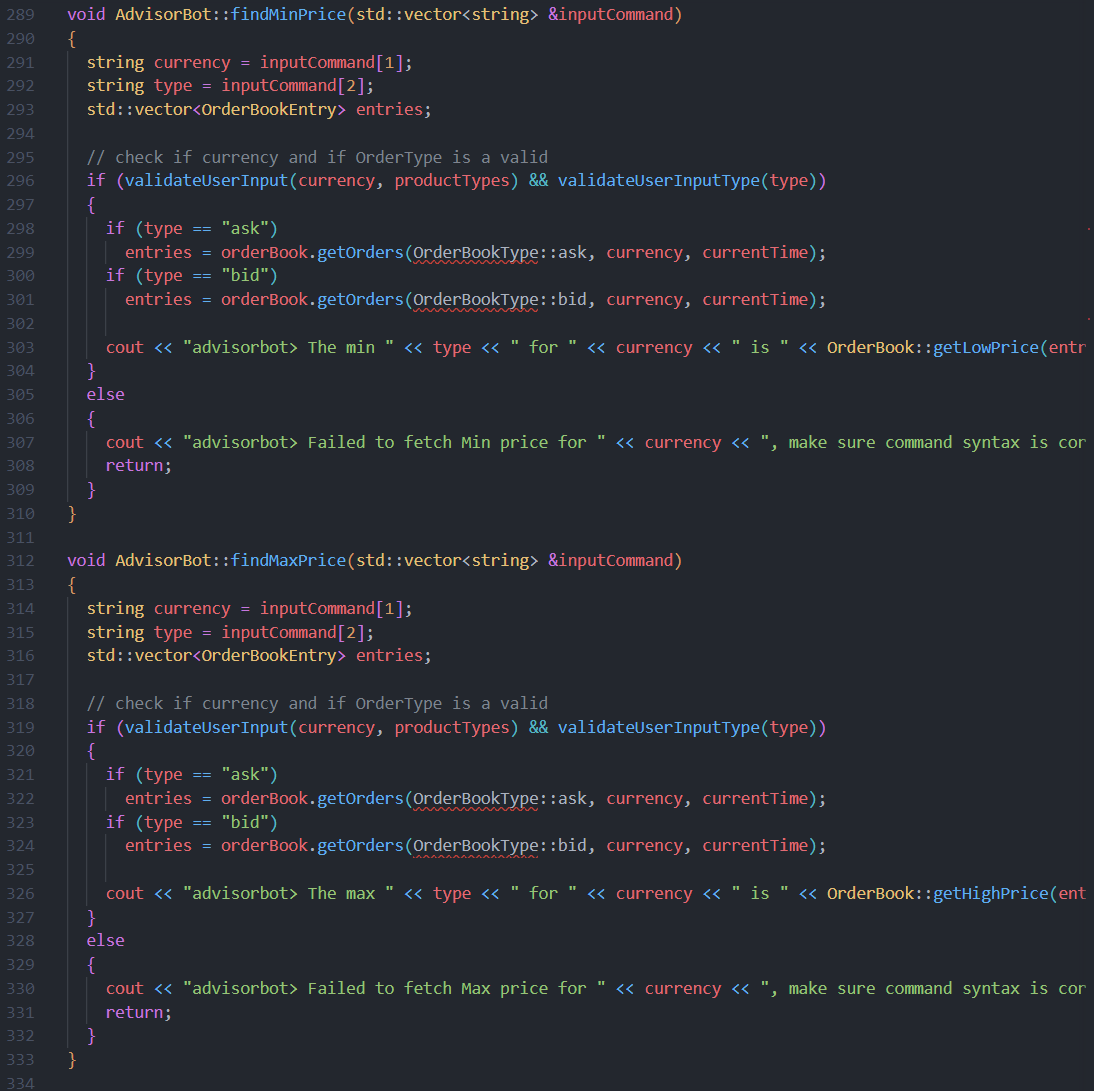
This is an extract of the processUserInput function in AdvisorBot.cpp, what it does is it takes in an inputCommand, which is a string returned from the promptUserInput function that parses the key typed from the command line. Depending on the cmd keyed in, AdvisorBot will perform one of its many functions like listing the commands it knows, listing the available currency on the market, fetching the minimum price of a currency etc. This can be considered control coupling as the user keyed in command is the flag that will dictate which pathway the processUserInput function will take.

**Module cohesion example 1: Functional Cohesion**



The OrderBook class is a good example of functional cohesion. The functions contained within all manipulate the list of orderbook entries in one way or another. For example, in lines 29 – 32, they all return price-related data pulled from the list of entries.

**Module cohesion example 2: Communicational Cohesion**



An example of communicational cohesion can be found in the findMinPrice and findMaxPrice functions. Although both functions perform different tasks, min finding the minimum price of a currency and max finding the maximum price of a currency. They are communicationally cohesive as they take in the same parameters, being the user’s input command and output a similar message.