**Object Oriented Programming Midterm Assessment Report**

**Program Name: Advisor Bot**

**Programmer: Fan Hao Date: 09/ Jan/ 2022**

**Description:**

**The advisor bot is a program to help a cryptocurrency investor analysis the data available on an exchange. For this project, I tried to build all functions but some of them may not work properly.**

**The help it can provide includes helping to query real-time transaction data, listing trading products, calculating transaction means, predicting product movements, and more. As long as the corresponding instructions are entered, the functions will be presented to the user one by one and the corresponding command will be executed.**

**If the terminator in VScode was used for testing, then please input the code’ .\a.exe ‘ into the terminator so that the program can run properly. Otherwise it will ask the user to do so for running the code.**

**Function:**

|  |  |
| --- | --- |
| **Command Name** | **Implementation Status**  **(Yes/ No)** |
| **C1: help** | **YES** |
| **C2 :help cmd** | **YES** |
| **C3: Prod** | **YES** |
| **C4: Min** | **YES** |
| **C5: Max** | **YES** |
| **C6: Avg** | **YES** |
| **C7: Predict** | **YES** |
| **C8: Time** | **YES** |
| **C9: Step** | **YES** |
| **Command Parsing Code** | **YES** |
| **Implement my own code** | **YES (the Common cmd)** |
| **Optimize the exchange code** | **YES, but a little** |

**For command parsing code,** I used the Tokenise function so that I can separate a whole command into different part which is also like modularization. This is because token is like a unique key to each function and with giving same token to a bunch of certain command, I can bring them to a certain brunch and modify them together. Besides, each command in this brunch can have its own input type, most of which are string type.

For input data, users may input a number to check the menu though the help menu does not contain any number. So I set a part of code to cast int type into String type so that the terminator can understand what user is asking.

The parsing category is depends on the size of token as input from user. There are 5 categories:

1. Single token command

These are – help, prod, time and step.

1. 2 tokens command

These are- help<cmd>.commands, which formed by two tokens for the required output

1. 3 tokens command

These are:

min prod bid/ ask;

max prod bid/ask;

1st token is for job , 2nd token is for product and 3rd token is for bid/ask

1. 4 tokens command

These are :

avg prod ask/bid timestep;

predict prod ask/bid timestep;

1st token is for job , 2nd token is for product, 3rd token is for bid/ask and 4th token is the number of timestep require in calculation

1. Wrong command

As the token has been set a upper bound of 3, the input, which token number is greater than 3, will not be recognized or executed. The program will print ‘Wrong command’ on the screen and inform the user to input a valid command.

文本

中度可信度描述已自动生成表格

描述已自动生成

**Custom command:**

Since there is command for asking Maximum data and Minimum data, then I try to create another command which called ‘Common’. This command is to help user find out the common price among those data, To achieve this, I create a ‘void:: getCommon’ in the advisorBotMain.cpp and called it. Just like max command and min command, common command is also search among the data, depends on user’s asking for ‘ask’ or ‘bid’, and ick out all the same number and print out.

**Optimize the exchange code:**

I set up a command that the program can recognizes and makes response regardless of whether you enter upper case for ,‘ask’ and ‘bid’, or not. This makes the program easier to use to some extent.

Since most of the commands are timestamp based, every time you go to a new timestamp, you need to calculate max, min, trend, with the use of Tokenise method, it reduces the time required for calculations and makes it much less difficult to modify the code, because the calculation command has been separated into different part and may not affect each other and reduced the size of the program