Object-Oriented Programming

Mid Term Project - advisorBot

Ng Xuan Min

Table of Contents

[Introduction 2](#_Toc92740705)

[Commands implemented 2](#_Toc92740706)

[Command parsing and Validating user’s input 3](#_Toc92740707)

[Command Parsing 3](#_Toc92740708)

[Validating User’s Input 3](#_Toc92740709)

[Implementation of custom command 5](#_Toc92740710)

[New command 5](#_Toc92740711)

[Optimization of the exchange code 5](#_Toc92740712)

[Conclusion 6](#_Toc92740713)

# Introduction

In this assignment I created a program called advisorbot. Advisorbot is a

command line program that can carry out various tasks to help a cryptocurrency

investor analyse the data available on an exchange.

With the commands available in my program, it will greatly help cryptocurrency investors, including new investors to analyse and make smart decisions simply/

# Commands implemented

I was able to implement all the commands required in this assignment, as well as an extra command.

|  |  |  |
| --- | --- | --- |
| Command | Implemented | Function |
| help | Yes | Print all available commands |
| help <cmd> | Yes | Print each command’s purpose and how to use it |
| prod | Yes | Print all the products available |
| min | Yes | Print the minimum of the product’s ask/bid on the current timestamp. |
| max | Yes | Print the maximum of the product’s ask/bid on the current timestamp. |
| avg | Yes | Print the average of the product’s ask/bid depending on the timestep that the user have keyed in.  Example: avg ETH/BTC ask 4  Function: Takes the current timestep average and takes the 3 previous timestep’s average to calculate the final average. |
| predict | Yes | Print the prediction of the next product’s min/max ask/bid using the last 5 previous timestep value, including the the current timestep value |
| time | Yes | Print the current timestamp |
| step | Yes | Goes to the next timestamp. |
| how <hcmd> | Yes | Users can see how to use the program. Example if user key in how hmin, Example: min ETH/BTC ask will be printed. |

# Command parsing and Validating user’s input

## Command Parsing

Graphical user interface, website

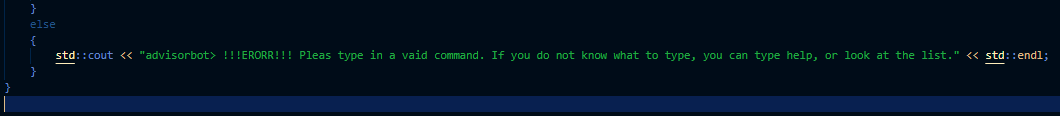
Description automatically generated

**A screenshot of a computer

Description automatically generated with medium confidence**

Functions for minimum, maximum, average and predict, I used the tokenise function which was taught in lecture to parse the user input to a vector of strings, allowing the program to taken in the user’s input. The first word that will be type by the user, should be any of the functions which is tokens[0]. The different functions have different tokens.size() , depending on the user’s input needed.

## Validating User’s Input



Text

Description automatically generated

As for validating the user’s input, I created a message that will appear if incorrect commands is entered.

Text

Description automatically generated

The first layer of validating user’s input is in the processUserOption (std::string userOption) function. If the user types in an unrecognizable command, the error message appear.

Text

Description automatically generated

(printMarketStats function to print the minimum or maximum of the product)

# Implementation of custom command

### New command

A screenshot of a computer

Description automatically generated with medium confidence

This command allows users to be able to understand how to use the program.

A screenshot of a computer

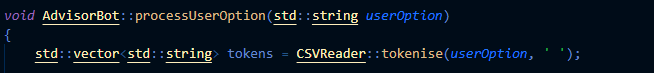
Description automatically generated with medium confidence

If users type how hmin for example, Example: min ETH/BTC ask will be printed.

Text

Description automatically generated

# Optimization of the exchange code



I’ve tokenised the user input into vectors of string using space as a character separation so I can manipulate it easily.

Text

Description automatically generated

After tokenizing the user’s input, order book types and products will be dynamically filled in and output the user’s desired product and order book type. This method is applied for average, predict.

# Conclusion

In conclusion, I’ve implemented all the commands required, and added extra command how.