```
/*
        Author's Name:              Kenneth Larot Yamat

        Purpose of Program:         To create a program that automatically creates
                                    trading tickets for a security, for example, buy
                                    and sell orders for shares of an exchange traded fund.

        Date Due:                   11:59 PM on March 4th, 2024
*/

Project Proposal:       To create a program that automatically creates
                        trading tickets for a security, for example, buy
                        and sell orders for shares of an exchange traded fund.

                        A user would only manually enter the first order, either to
                        buy or sell a security, the program would populate and submit
                        a new ticket based on the fulfillment of the previous ticket,
                        the chain of tickets would continue until the user decided to
                        cancel the chain.

A) Background and the needs:

                        This program is needed because there are many securities
                        that are difficult to trade because they are illiquid as
                        a result of large bid and ask spreads, or because they lack
                        volume.

                        The goal is to reduce spreads while increasing volume.

                        Another need is due to the fact that manually performing this task
                        is laborious and prone to error.


B) Function list:

                        getSecurityPrice
                        setSecurityPrice

                        setPurchasePrice
                        getPurchasePrice

                        setLiquidationPrice
                        getLiquidationPrice

                        setAvarageTrueRange
                        getAverageTrueRange

                        setBollingerBandWidth
                        getBollingerBandWidth

                        setAverageDirectionalIndex
                        getAverageDirectionalIndex

C) User interface (UI) design:

    Step 1

```

```

                Trade Ticket
                ============

                Security:           [User Input Element]
                Buy or Sell:        [User Input Element]
                Limit:              [User Input Element]
                Quantity:           [User Input Element]
                ATR:                [User Input Element]
                BBW:                [User Input Element]
                ADX:                [User Input Element]

    Step 2


                Your initial [Buy/Sell] Trade ticket for [Security] has been
                submitted at the following price [Limit Price] and quantity [Quantity].

                Subsequent orders will be automatically generated and submitted contingent
                upon the fulfillment of the previous order, with buy limits and sell limits
                based on the Average True Range, Bollinger Band Width, and Average Directional
                Index entered on the initializing ticket.

                Sell orders will be generated with a limit of   [Calculated Amount] above the previously filled ticket
                Buy  orders will be generated with a limit of   [Calculated Amount] below the previously filled ticket

                [   User Input Element   [Accept and Submit]   [Override and Submit]   [Start Over]   ]

    Step 3

                [Ticker Symbol]     [Buy/Sell]          [Order Quantity]            [Limit Price]               [Ticket Status]

                [   HFH.P                   Buy                     1                           86.86                   Open          ]

    Step 4
                [Ticker Symbol]     [Buy/Sell]          [Order Quantity]            [Limit Price]               [Ticket Status]

                [   HFH.P                   Buy                     1                           86.86                   Filled        ]

    Step 5

                [Ticker Symbol]     [Buy/Sell]          [Order Quantity]            [Limit Price]               [Ticket Status]

                [   HFH.P                   Sell                    1                           86.89                   Open          ]

    Step 6

                [Ticker Symbol]     [Buy/Sell]          [Order Quantity]            [Limit Price]               [Ticket Status]

                [   HFH.P                   Sell                    1                           86.89                   Filled        ]

    Step 7

                [Ticker Symbol]     [Buy/Sell]          [Order Quantity]            [Limit Price]               [Ticket Status]

                [   HFH.P                   Buy                     1                           86.87                   Open          ]

```

Step 8

[Ticker Symbol]	[Buy/Sell]	[Order Quantity]	[Limit Price]	[Ticket Status]
[HFH.P	Buy	1	86.87	Filled]

Notes: this sequence is based on + 00.03 to Sell orders and - 00.02 to Buy orders for first issue preferred shares for the security HFH

D) Class diagram

[AverageTrueRange] [Ticket] [AutoTicket]
extends from Ticket

E) File and database design:

[Data Dictionary for Database Tables and Non-Database Files]

[File and Database Design]

data will be instantiated as an Arraylist and printed initially and stored as .txt files, a program will be created to convert these .txt files into .xml and .csv files where and when appropriate.

[Data Dictionary]

the data dictionary will define the columns ticker symbol, buy/sell, order quantity, limit price, and ticket status.
the data dictionary will also contain the methods and classes that modify or control this data.

[Database Tables]

will be organized by column headers such as date, ticker symbol, order quantity, limit price, ticket status

[Non-Database Files]

will contain the initial trade authorization, and the user inputs, authorization for the subsequent auto trades based on the other inputs, or, authorization for the automated trades based on user overridden inputs.

[Relational Database]

each row in the data file represents on ticket, all the data on that ticket is related to that particular ticket, each ticket is related to the previous ticket

[Plain Text Files]

all data will initially be created as ArrayLists and converted into plain text files.

F) Expectations of project fulfillment:

a. [Ticket] instantiates based on user input.

[AutoTicket] instantiates based on fulfillment of previous ticket.

b. [Controller Classes]

c. [GUI applications]

d. [Arraylist] Arraylist will be used to log the sequence of trades

e. [Exception handling] a user may enter alphabetical values in a field that requires an int or double, and vice versa, an invalid data message will prompt the user.

f. [Database]

g.

[Documentation] very detailed and elaborate notes will be included in every program, class, method, and attribute regarding the purpose, design, development, and miscellaneous other notes as well. JavaDoc will see extensive use.

G) Project Report

a.

b.

c.

d.

e.

f.

g.

h.