



Документация к торговому роботу  
«*Arbitrage robot*»

# Содержание

<b>История изменений</b>	<b>3</b>
<b>Руководство пользователя</b>	<b>8</b>
Соглашение об ответственности	8
Архитектура	8
Настройка подключения	8
Настройка портфелей и торговые функции	10
Редактор формул	11
Сброс статусов заявок робота	12
Неторговые функции	12
Настройка ключей подключений	13
Добавление/удаление подключений к биржам криптовалют	14
Отображение позиции на биржах криптовалют	14
Обновление программы и документация	14
Подсчет скорости транзакций	15
Настройка telegram-бота	16
Использование приказа "переместить заявку"	16
Особенности использования торговых стаканов инструментов	16
Возможные проблемы и их решение	17
Наиболее распространенные ошибки, возникающие при работе программы, и способы их устранения	18
<b>Описание параметров программы</b>	<b>21</b>
Введение	21
Параметры портфеля (редактируемые)	21
Параметры портфеля (отображаемые)	27
Параметры инструментов портфеля (всегда редактируемые, если не указано обратное)	28
Параметры уведомлений портфеля (редактируемые)	31
Порядок выставления заявок и механизм выравнивания позиции	33
Правила перемещения <i>Lim_Sell</i> и <i>Lim_Buy</i>	33
Неочевидные моменты, связанные с работой робота	34
Пример работы программы с заданием основных параметров	35
<b>Написание формул на языке программирования C++</b>	<b>37</b>
Общие сведения	37
Класс <i>security</i>	37
Класс <i>portfolio_row</i>	37
Класс <i>portfolio</i>	38
Класс <i>trade</i>	38
Дополнительные функции и константы	38
Примеры доступа к параметрам портфеля, инструмента, сделки	41
Примеры использования функций	41
Примеры написания <i>Ratio buy/sell formula</i>	42

## История изменений

Версия 1.3.1, 02.03.2016

1. Добавлено *FinRes fall notification* в [параметры уведомлений портфеля](#)
2. Добавлен раздел [Примеры написания \*Ratio formula\*](#)

Версия 1.3.2, 21.04.2016

1. Добавлено *Type price* в параметры портфеля
2. Добавлены *Sell volume OB*, *Buy volume OB* в параметры инструментов портфеля
3. Изменено описание *Percent of quantity*, *Market volume*
4. Добавлен раздел, описывающий правила использования *Ratio formula* (*доступно НЕ во всех версиях программы*)

Версия 1.3.3, 24.06.2016

1. Добавлено автоматическое хеджирование позиции (параметр *Hedge*) в параметры портфеля
2. В таблицу с раздвижками добавлена возможность расчета средних цен раздвижек на покупку и продажу для заданного портфеля
3. Изменен смысл переменной *curpos* в формулах для подвижки лимитов, теперь это НЕ позиция портфеля, а позиция по *Is first* бумаге портфеля. Это очень важное исправление, оно сказывается только на портфелях, у которых *Count* по *Is first* бумаге не равен 1.

4. Изменена формула для переменной *k3* в формулах для подвижки лимитов, было:

$$k3 = (|Lim\_Sell_0 - Lim\_Buy_0| - TP - K) \times \frac{V}{curpos} ,$$

стало:

$$k3 = (|Lim\_Sell_0 - Lim\_Buy_0 - TP| - K) \times \frac{V}{curpos} .$$

Данное изменение связано с существовавшей до сих пор сменой знака переменной *k3* в случае когда  $|Lim\_Sell_0 - Lim\_Buy_0| < TP$  (хотя, при правильных настройках такой ситуации не должно быть).

5. Добавлены уведомления о сильном изменении *Lim\_Sell* и *Lim\_Buy*

Версия 1.3.4, 21.07.2016

1. Добавлена возможность выбора кода клиента (*Client code*) с которого выставить и снять заявки по заданной бумаге
2. Изменена формула для переменной *k3* в формулах для подвижки лимитов, формула приняла свой первоначальный вид:

$$k3 = (|Lim\_Sell_0 - Lim\_Buy_0| - TP - K) \times \frac{V}{curpos} .$$

Версия 1.3.5, 10.09.2016

1. Добавлен telegram бот и описание его подключения
2. Добавлен параметр *Close* в настройки расписания
3. Добавлен параметр *To market* в настройки расписания
4. Добавлено *Too much not hedged notification* в [параметры уведомлений портфеля](#)
5. Переработано [поведение робота при срабатывании стоп-лосса и таймера](#)
6. Добавлен раздел [Подсчет скорости транзакций](#)
7. Добавлена параметр *Type trade*
8. Добавлена режим торговли *Option hedge* в параметр *Type*

Версия 1.3.6, 29.11.2016

1. Убран параметр *Log level*
2. Добавлен параметр *Count type*
3. Добавлен параметр *Count formula*

Версия 1.3.7, 20.12.2016

1. Добавлен неочевидный момент работы робота
2. *Decimals* бумаги портфеля теперь отвечает за число знаков после десятичной точки в сделках по данной бумаге в таблице раздвижек
3. *Decimals* не просто влияет на число отображаемых знаков после десятичной точки в таблице, но и округляет все редактируемые дробные числа строки таблицы до заданного числа знаков
4. Добавлен *To0* в расписание

Версия 1.3.8, 20.01.2017

1. Добавлено ограничение на количество портфелей и финансовых инструментов
2. Добавлен *FUT move limits* в параметры инструментов портфеля
3. Добавлен *SPOT move limits* в параметры инструментов портфеля

Версия 1.4.0, 05.02.2017

1. Параметр *Ratio formula* заменен на два параметра: *Ratio sell formula* и *Ratio buy formula*
2. Добавлены параметры *Custom trade* и *Trade formula*
3. Добавлен параметр *Overlay*

4. Все формулы теперь пишутся на языке программирования C++ с соответствующим API
5. Добавлен параметр *MM* в параметры бумаг портфеля
6. *K*, *K1*, *K2* и *Avg opened* добавлены в основную таблицу
7. Параметр *Wait hedge* заменен на *Max not hedged*, если вы хотите получить поведение, аналогичное *Wait hedge*, необходимо задать *Max not hedged* равным 1
8. Добавлен раздел с описанием редактора формул
9. Добавлен раздел с описанием работы приказа "переместить заявку"

Версия 1.4.2, 05.04.2017

1. Добавлен параметр *In formulas*
2. Добавлен параметр *Simply first*
3. Добавлена всплывающая подсказка для *Avg. sell*, *Avg. buy* с ценами бумаг
4. Добавлен параметр *Save/load trades* в настройки приложения, позволяющий сохранять/-загружать таблицу сделок при закрытии/открытии приложения
5. Добавлен экспорт таблиц в *CSV* файл
6. Ограничен размер таблиц лога и сделок до 100000 строк, при переполнении "старые" записи будут автоматически удаляться
7. Добавлен столбец *Color*
8. Изменены возможные значения параметра *Type price*
9. Добавлен параметр бумаги портфеля *Calc price OB*
10. Добавлен параметр бумаги портфеля *Trading price OB*
11. Добавлен параметр бумаги портфеля *Depth OB*
12. Удален параметр бумаги портфеля *Buy volume OB*
13. Удален параметр бумаги портфеля *Sell volume OB*

Версия 1.4.3, 01.06.2017

1. Добавлен раздел с описанием особенностей использования торговых стаканов инструментов

Версия 1.4.4, 11.08.2017

1. Добавлена торговля "в файл": добавлено значение *Client code*, равное *To file*
2. Добавлен фильтр некоторых сообщений в логе, такие сообщения будут отображаться в логе 1 раз в 10 секунд и будут отмечены в конце сообщения символом  $xN$ , где  $N$  – количество непоказанных сообщений

Версия 1.4.5, 13.09.2017

1. Добавлен *Type Find fastest*
2. Добавлен *Client code Use fastest/\**

Версия 1.4.8, 5.12.2017

1. Добавлен параметр *Virtual 0 pos*
2. Добавлен параметр *Timetable only stop* в окно настройки расписания
3. Добавлен менеджер ключей подключений к биржам

Версия 1.4.9, 27.12.2017

1. Добавлен параметр *BitMEX level to0* в параметры бумаги портфеля, подробнее в документации
2. Добавлен параметр *BitMEX level close* в параметры бумаги портфеля, подробнее в документации
3. Добавлен параметр *BitMEX only maker* в параметры бумаги портфеля, подробнее в документации

Версия 1.4.10, 24.01.2018

1. Для повышения безопасности соединения изменен формат файла сертификата в настройках подключения

Версия 1.4.11, 15.03.2018

1. Добавлена возможность добавления/удаления подключений к биржам криптовалют
2. Добавлен информационный канал в мессенджере Telegram [https://t.me/fkvikings\\_info](https://t.me/fkvikings_info)
3. Изменено поведение параметра *Price check*
4. Параметр *BitMEX only maker* переименован в *Only maker* и теперь работает для BitMEX и Bitfinex

Версия 1.4.12, 20.03.2018

1. В менеджер ключей добавлена возможность экспорта ключа

Версия 1.5.0, 02.04.2018

1. Добавлена возможность добавления подключения к Deribit
2. Обновлен раздел с наиболее распространенными ошибками, возникающими при работе программы, и способами их устранения (а именно торговые ошибки)

Версия 1.5.1, 16.04.2018

1. *BitMEX level to0* переименован в *Level to0* и теперь доступен еще и для Deribit
2. *BitMEX level close* переименован в *Level close* и теперь доступен еще и для Deribit
3. *Only maker* теперь работает для Deribit
4. Изменена совместная работа параметров *Simply first* и *Only maker*

Версия 1.5.2, 20.05.2018

1. Добавлены параметры портфеля *Extra formulas*, *Extra field#1* и *Extra field#2*

Версия 1.5.3, 20.06.2018

1. [Добавлено отображение позиции с некоторых криптовалютных бирж](#)

Версия 1.5.4, 10.08.2018

1. Добавлена возможность добавления подключения к Cryptofacilities
2. *Level to0* теперь доступен еще и для Cryptofacilities
3. *Level close* теперь доступен еще и для Cryptofacilities
4. *Only maker* теперь доступен еще и для Cryptofacilities

Версия 1.5.5, 29.08.2018

1. Добавлена возможность добавления подключения к KuCoin
2. [Обновлен раздел с возможными проблемами и их решениями](#)

Версия 1.5.6, 15.10.2018

1. *Client code* не может быть пустым для бумаг с *Count* отличным от нуля
2. В окно отображение позиции добавлено отображение баланса по нескольким валютам
3. Добавлена возможность получения значений индексов с CSIIndex

Версия 1.5.7, 25.10.2018

1. Добавлена возможность добавления подключения к CEX.IO
2. Добавлена возможность сохранять значения между вызовами формул в поле *data* (которое является словарем) портфеля

Версия 1.5.8, 12.12.2018

1. Удалены *Find fastest* и *Use fastest*
2. Изменен принцип работы *Round robin*

Версия 1.5.9, 25.02.2019

1. Добавлена возможность добавления подключения к Huobi Russia

## Руководство пользователя

### Соглашение об ответственности

«Финансовая Компания «Викинг» не несет ответственности за умышленные попытки пользователя дестабилизировать работу торгового робота.

### Архитектура

Торговый робот «*Arbitrage robot*» состоит из двух частей: серверной и клиентской. Серверная часть робота – это то место, где непосредственно выполняется сам торговый алгоритм. Клиентская часть робота – это графический интерфейс пользователя (GUI). Отделение графического интерфейса от торгового алгоритма позволяет разнести по разным физическим устройствам торговую часть и интерфейс пользователя, так торговый алгоритм может быть запущен на физическом сервере или виртуальной машине под управлением ОС семейства Linux в зоне колокации Московской биржи, в то время как графический интерфейс пользователя устанавливается на персональный компьютер клиента.

Принцип совместной работы двух частей торгового робота (серверной и клиентской) состоит в следующем: все настройки всегда хранятся на клиентской части, если вы запускаете серверную часть то она запускается не инициализированной настройками (т.е. торговых портфелей в ней нет и торговать она не будет). При первом подключении клиентской части робота к серверной, серверная часть получает настройки торговых портфелей и хранит их до своего (серверной части) выключения.

*Следует четко понимать следующие моменты, связанные с работой робота:*



- *если вы отключаете клиентскую часть от серверной, но не выключаете при этом серверную часть, то все настройки в ней остаются и если торговля не остановлена, то она будет продолжаться при выключенной клиентской части;*
- *если вы подключаетесь клиентской частью к серверной, которая не перезапускалась и уже инициализирована настройками, то настройки загрузятся уже из серверной части в клиентскую, именно поэтому все настройки портфелей клиентской части рекомендуется проводить будучи подключенным к серверной части робота.*

Далее речь пойдет только о настройке клиентской части, при этом подразумевается, что серверная часть запущена, и подключение ко всем требуемым торговым площадкам установлено.

*Важно:* ЗАПРЕЩЕНО два и более одновременных подключения клиентской части к серверной, одновременное подключение приведет к потере данных из-за конфликта графических частей при подключении к серверной.

### Настройка подключения

Для начала работы с торговым роботом необходимо запустить клиентскую часть. Открывшийся графический интерфейс будет иметь вид, как показано на рисунке 1.

Если это первый запуск GUI, то необходимо настроить подключение графической части к роботу, для этого нужно выбрать следующий  пункт главного меню , в открывшемся окне указать требуемые данные (Рис. 2). Обычно это *Host* и *Robot port*. Если вы подключаетесь с использованием SSL (например, вам выдали файл, содержащий все необходимые ключи и SSL сертификаты, т.е. файл с расширением *.xml*) то поставьте галку *Use SSL* и укажите путь к полученному файлу в поле *Certs path*. Далее следует нажать кнопку *Apply* для применения изменений. При последующих запусках робота настройку соединения выполнять не требуется.



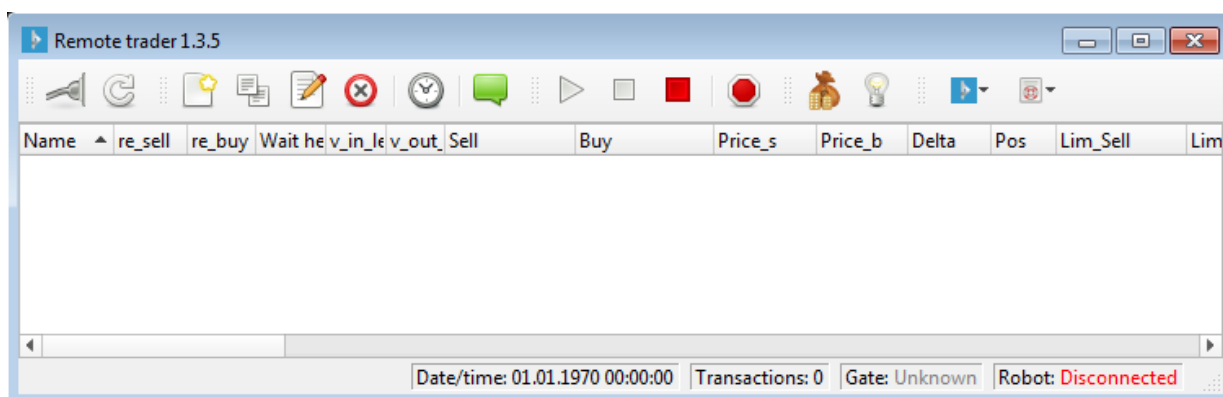


Рис. 1: Графический интерфейс пользователя до подключения

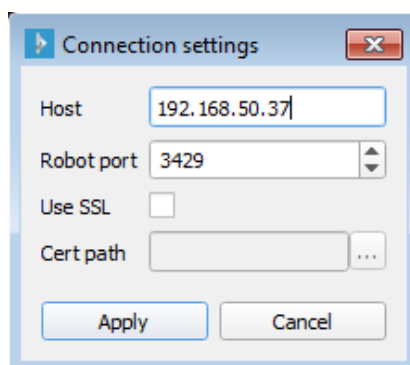


Рис. 2: Настройка подключения к серверной части робота

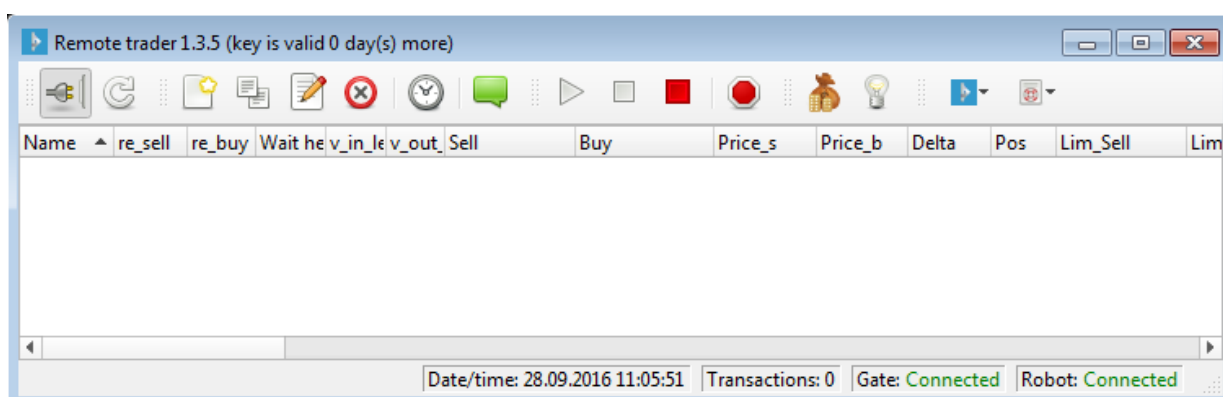


Рис. 3: Графический интерфейс пользователя, соединение установлено

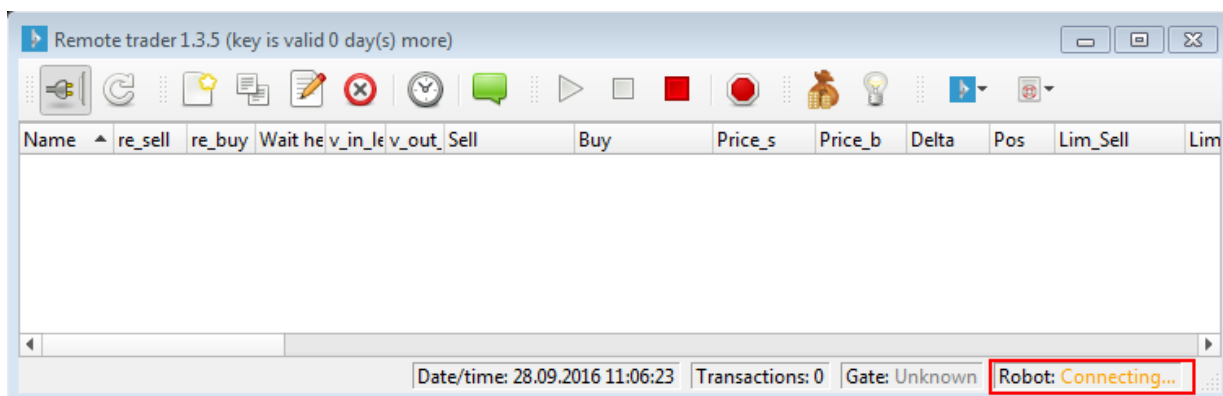












Рис. 4: Графический интерфейс пользователя, попытка установки соединения


Для подключения к серверной части робота необходимо нажать на кнопку подключения , после чего GUI примет вид как представлено на рисунке 3, если вместо этого GUI имеет вид как представлено на рисунке 4, то следует проверить указанные настройки подключения (Рис. 2), если настройки верны, необходимо обратиться к контактному лицу. Если подключение выполнено успешно, то следует выполнить обновление списка финансовых инструментов, нажав кнопку .

***Важно:** Изменение всех торговых настроек следует проводить ТОЛЬКО при наличии подключения к серверной части робота.*






## Настройка портфелей и торговые функции

После того, как выполнено подключение к серверной части робота, и робот имеет вид как показано на рисунке 3, можно приступить к настройке нового торгового портфеля. Если это не первый запуск робота, то в таблице будут отображены уже существующие портфели. Для этого необходимо нажать кнопку добавления портфеля , в открывшемся окне нужно указать необходимые настройки (Рис. 5). В каждом торговом портфеле может быть несколько инструментов, один из которых является главным – по нему рассчитывается позиция всего портфеля. Кроме создания нового портфеля доступны функции редактирования существующего портфеля, создания его копии, а так же удаления портфеля, для этого служат кнопки ,  и  соответственно. Чтобы отредактировать портфель, создать его копию или удалить его, необходимо чтобы нужный портфель был выбран в таблице. Так же существует возможность экспортировать настройки портфелей ( в главном меню ) и импортировать настройки портфелей из имеющегося файла настроек ( в главном меню ). При импорте будут добавлены портфели с уникальными именами, имеющиеся портфели изменены или удалены НЕ будут.

Когда у бумаги до даты экспирации остается 3 и менее дней, если у этого инструмента есть дата экспирации, робот оповестит вас об этом перечеркнув имя бумаги.

Для просмотра сделок, совершенных роботом, предусмотрена кнопка , в открывшейся таблице отображаются раздвижки по совершенным сделкам (если строк в таблице становится более 100000 "старые" строки будут автоматически удаляться).

При закрытии GUI все настройки сохраняются, и при следующем включении будут доступны все созданные портфели. После добавления портфелей основной экран робота будет выглядеть как показано на рисунке 6. Изменение всех торговых настроек следует проводить только при наличии подключения к серверной части робота.

Для включения и остановки торговли по всем портфелям одновременно служат кнопки  и  соответственно. Если необходимо задать расписание работы робота, т.е. отрезки времени, когда будет производиться торговля, то нажатием на кнопку  нужно вызвать окно настройки расписания. В расписании следует указывать время для того же часового пояса, который выбран на машине, где запущена серверная часть торгового робота (при торговле на Московской бирже, предполагается использование московского времени), в не зависимости от того, какое время задано на клиентской машине. Кнопка  выключает торговлю всех портфелей, причем выключает и расписание. Для настройки уведомлений по портфелям при резком изменении значений некоторых параметров используйте кнопку .

В строке состояния в нижней части главного окна программы расположен суммарный счетчик транзакций (то есть попыток выставления и снятия заявок) по всем транзакционным подключениям к бирже (*Transactions*). Также в строке состояния расположен индикатор состояния

Рис. 5: Добавление нового портфеля

Name	re_sell	re_buy	Wait he	v_in	le	v_out	Sell	Buy	Price_s	Price_b	Delta	Pos	Lim_Sell	Li
1 siz6	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	1		1	65 182.0000	65 194.0000	0.0000	0.0000	0.0000	0	0.0000	0

Рис. 6: Графический интерфейс пользователя

подключения всех подключений торгового робота к бирже (*Gate*), при наведении курсора мыши на индикатор появится всплывающая подсказка с подробной расшифровкой состояния всех подключений.

*Важно:* По умолчанию вы можете добавить не более 50-ти портфелей и при этом не более 100-а финансовых инструментов суммарно во все портфели, при превышении указанных чисел, превысивший ограничения портфель будет автоматически отключен.

## Редактор формул

В окне настроек портфеля расположена кнопка *Edit formulas*, которая открывает редактор формул данного портфеля (Рис. 7). В левом столбце в редакторе расположены только те имена параметров портфеля, которые являются формулами и используются, исходя из текущих настроек портфеля (те параметры, которые сейчас используются не как формулы НЕ будут отображены).

ражены). Чтобы открыть редактор для параметра из левого столбца, надо сделать двойной клик левой кнопки мыши на имени соответствующего параметра, новый редактор откроется ниже всех, уже открытых редакторов. Используя пункты меню: *Securities*, *Portfolios*, *Deals*, – вы можете добавлять параметры финансовых инструментов, портфелей и сделок, не набирая соответствующий код (подходит для не опытных пользователей). Формулы пишутся на языке программирования C++, вы пишете только тело функции и должны вернуть значение типа *double*. Более подробно API, доступное в редакторе формул, будет описано в отдельном разделе.

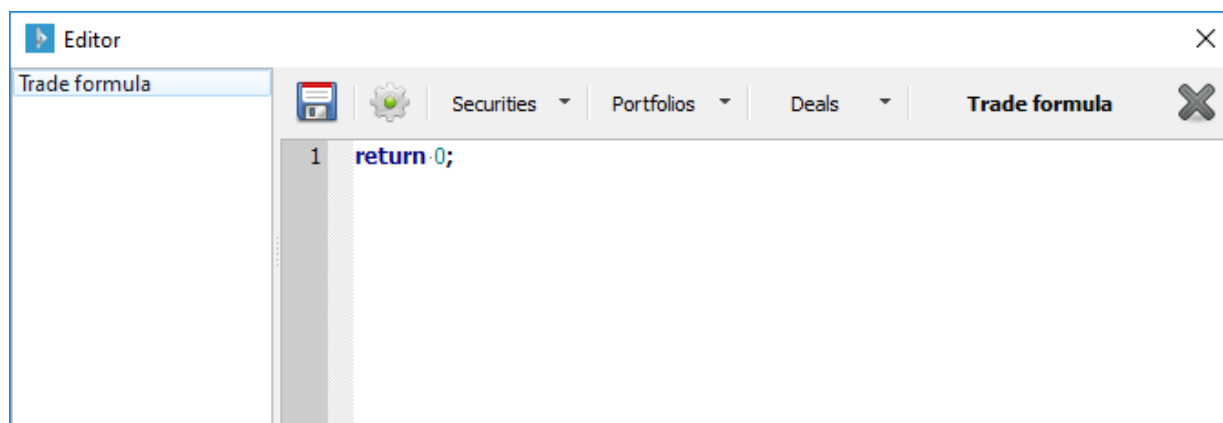






Рис. 7: Редактор формул портфеля

## Сброс статусов заявок робота

В торговом роботе есть "механизм отслеживания заявок", бывают ситуации, например, когда происходят сбои в работе биржи, что заявка выставляется роботом, а потом снимается биржей, никак не информируя робот о снятии. В таком случае заявка "зависает" в своем текущем, внутреннем для робота, статусе. Для сброса статусов всех заявок портфеля есть кнопка . Пользоваться этой кнопкой можно ТОЛЬКО в КРАЙНИХ случаях, когда торговля по портфелю отключена и Вы уверены что нет ни одной активной заявки по данному портфелю, в противном случае робот потеряет активные заявки, что приведет к неправильной позиции по бумагам в роботе.

## Неторговые функции

Во время работы робота могут возникать различные ошибки при выставлении и удалении заявок, такие ситуации не являются нештатной работой робота и могут быть вызваны причинами, не связанными с некорректной работой торгового робота, например, отсутствием денег на счете клиента. Для настройки уведомлений об ошибках, информационных сообщениях и поведения главного окна программы выберите следующий  пункт главного меню . Все ошибки и еще некоторую информацию, связанную с работой программы, можно посмотреть в логге, нажав на кнопку  (если строк в таблице становится более 100000 "старые" строки будут автоматически удаляться). Некоторые слишком часто приходящие сообщения будут отображаться в логге не все, а 1 раз в 10 секунд и будут отмечены в конце сообщения символом  $xN$ , где  $N$  – количество непоказанных сообщений. Время в логге – это время прихода сообщения из серверной части в клиентскую, это локальное время машины, на которой запущена клиентская часть. По этому времени НЕЛЬЗЯ судить о скорости работы робота.

Стоит заметить, что все таблицы робота настраиваемые. Можно менять местами столбцы, а так же отключать ненужные. Для перетаскивания столбца достаточно зажать левую кнопку

мышью на его заголовке и перетащить столбец в нужное место таблицы. Для отключения лишних столбцов нужно сделать правый клик мышью на заголовке таблицы, после этого откроется контекстное меню, как показано на рисунке 8. В этом же контекстном меню есть пункты применения фильтра к таблице и копирования/экспорта таблицы в *csv* формате. Для таблицы сделок, совершенных роботом, и таблицы лога в контекстном меню есть возможность очистить таблицу.

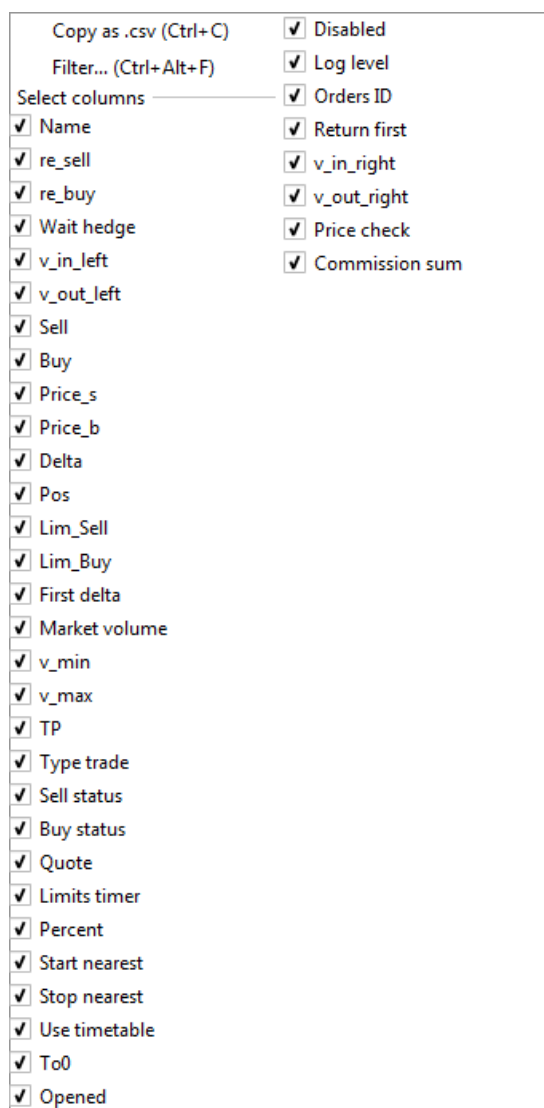







Рис. 8: Отключение столбцов таблицы

## Настройка ключей подключений


Для того, чтобы добавить в робот ключи, сгенерированные при помощи программы [CryptoKeyGen](#), нужно выбрать следующий  пункт главного меню  , в открывшемся окне поставить флаг *Use keys*, чтобы включить использование ключей, и добавить ключи из соответствующих файлов (*\*\_key\_for\_trader.key*) используя кнопку *Add key*. Содержимое столбца таблицы *Comment* – произвольный пользовательский комментарий к ключу, содержимое столбца *Key* – непосредственно сам ключ.

## Добавление/удаление подключений к биржам криптовалют

Для того, чтобы добавить/удалить в робот подключение к криптовалютной бирже, нужно выбрать следующий  пункт главного меню . В верхней половине открывшегося окна расположен список *market data* подключений, которые можно включить или выключить, используя флаг *Disabled*, в нижней половине открывшегося окна расположен список транзакционных подключений, кнопки для удаления/добавления транзакционных подключений и кнопка обновления списка текущих подключений. При добавлении нового транзакционного подключения необходимо выбрать тип подключения (биржу) и заполнить список необходимых параметров, при наведении на иконку  отображается необходимая информация о каждом из параметров. Двойной клик по ячейке в столбце статуса соединения позволяет переподключить данное подключение.




*Важно:* Нельзя удалить транзакционное подключение на котором есть активные заявки; подключения, которые запрещены на серверной части, не будут добавлены; по умолчанию нельзя добавить больше трех транзакционных подключений. Ключи подключений будут автоматически добавлены в менеджер ключей, описанный в [предыдущем](#) разделе


## Отображение позиции на биржах криптовалют

Для того чтобы посмотреть текущие позиции на бирже (поддерживаются не все биржи, список поддерживаемых бирж обновляется) используйте кнопку . В открывшемся окне вы можете выбрать интересующее подключение и посмотреть позиции по бумагам и/или баланс по деньгам в соответствующих таблицах (отображаемые значения зависят от конкретной биржи).

*Важно:* Настоятельно не рекомендуется все время держать данное окно открытым, т.к. это довольно сильно увеличивает количество передаваемой информации от серверной части робота к графической (особенно если у вас много открытых позиций по выбранному подключению), все равно позиции по всем подключениям одновременно вы не увидите

## Обновление программы и документация

Клиентская часть робота автоматически проверяет наличие обновлений при запуске и далее 1 раз в час (убедитесь, что адрес <http://fkvikings.ru/downloads> НЕ добавлен в черный список вашего фаервола). При наличии обновлений на панели появится иконка  (вместо иконки меню помощи ), нажав на которую вы сможете открыть меню помощи и, выбрав соответствующий пункт меню, скачать инсталлятор новой версии программы. Для проверки наличия обновлений программы в ручном режиме необходимо выбрать следующий  пункт меню помощи.

Для того чтобы открыть файл с документацией, используйте следующий  пункт меню помощи, документация откроется в просмотрщике *pdf* файлов, установленном в системе по умолчанию.

*Важно:* При наличии обновлений клиентской части рекомендуется ВСЕГДА обновлять программу, во избежании конфликта версий и невозможности подключения "старой" клиентской части к более новой (обновляемой без Вашего участия) серверной части. Серверная часть робота обновляется независимо от клиентской и обычно ее обновляют ночью, после окончания торговых сессий на всех рынках, поэтому рекомендуется НЕ оставлять активных заявок после окончания торгов, так как если серверная часть будет перезапущена, то она уже не "увидит" оставленных заявок, и эти заявки будут потеряны роботом и не будут корректно

обработаны, что приведет к несоответствию позиции в работе и реальной позиции на счете (если эти заявки исполнятся).



## Подсчет скорости транзакций

При наведении курсора мыши на надпись, отображающую число транзакций, совершенных роботом (*Transactions*), появится всплывающая подсказка, в которой отображена статистика скорости транзакций для каждого транзакционного подключения. Данные отображены в таблице, строки таблицы соответствуют имеющимся в работе транзакционным подключениям, столбцы таблицы:

1. *connect* – название подключения;
2. *all* – суммарное количество транзакций по подключению;
3. *> 0.5* – количество транзакций с round-trip-ом больше 0.5 мс по подключению;
4. *> 1* – количество транзакций с round-trip-ом больше 1 мс по подключению;
5. *> 2* – количество транзакций с round-trip-ом больше 2 мс по подключению;
6. *> 4* – количество транзакций с round-trip-ом больше 4 мс по подключению;
7. *> 8* – количество транзакций с round-trip-ом больше 8 мс по подключению;
8. *> 16* – количество транзакций с round-trip-ом больше 16 мс по подключению;
9. *adds* – суммарное количество успешных транзакций на выставление заявки по подключению;
10. *dels* – суммарное количество успешных транзакций на удаление заявки по подключению;
11. *moves* – суммарное количество успешных транзакций на перемещение заявки по подключению;
12. *rejects* – суммарное количество отвергнутых транзакций по подключению;
13. *avg* – среднее время round-trip-а транзакций (в микросекундах) по подключению;
14. *avga* – среднее время round-trip-а транзакций на выставление заявки (в микросекундах) по подключению;
15. *avgd* – среднее время round-trip-а транзакций на удаление заявки (в микросекундах) по подключению;
16. *avgm* – среднее время round-trip-а транзакций на перемещение заявки (в микросекундах) по подключению;
17. *avgr* – среднее время round-trip-а отвергнутых транзакций (в микросекундах) по подключению.

Важно: Время замеряется именно для round-trip-а, т.е. это время между моментом отправки транзакции и моментом прихода ответного сообщения на данную транзакцию.

## Настройка telegram-бота

Telegram-бот доступен по адресу <https://telegram.me/FKVikingBot>. Для настройки telegram-бота необходимо подключиться клиентской частью робота к серверной, далее выбрать следующий  пункт главного меню  и запросить ключ вашего робота, нажав в открывшемся окне на кнопку "Generate new key", после этого скопировать полученный ключ (он выделен жирным шрифтом) в чат telegram-бота в диалог добавления нового робота (ключ действителен 3 минуты, если не успели, еще раз нажмите на кнопку "Generate new key"). Если вы все сделали правильно, то в списке роботов в telegram-боте появится добавленный вами робот.

*Важно: Если вы случайно или специально перегенерируете ключ робота, робот останется "привязан" к telegram-боту. "Отвязать" робота от telegram-бота можно только выбрав пункт "Remove robot" в меню бота.*

## Использование приказа "переместить заявку"

На некоторых подключениях реализована отправка приказа "переместить заявку" (иногда эта команда также называется изменением заявки). Использование данного приказа позволяет сократить количество транзакций и увеличить отношение количества сделок к количеству транзакций, особенно при котировании, а так же увеличить время нахождения заявок в рынке. Так как особенности использования данной команды отличаются на разных рынках и типах подключений, то, в зависимости от подключения, данный приказ может применяться только для первой ноги портфеля или же для инструментов обеих ног. На данный момент отправка такой команды реализована для FIX-подключений фондового и валютного рынков Московской биржи, а так же plaza2 и TWIME подключений срочного рынка Московской биржи. Для FIX-протокола срочного рынка Московской биржи использование данной команды не поддерживается.

Использование приказа "переместить заявку" для поддерживаемых подключений не является обязательным. Эта возможность включается и отключается на серверной стороне. Если необходимо включить или отключить эту функциональность, следует обратиться в поддержку. В графической части никаких изменений при этом не требуется.

## Особенности использования торговых стаканов инструментов

В большинстве подключений к торговым площадкам стаканы по всем бумагам или полный лог заявок (далее всегда будем называть это просто стаканом в независимости от реально используемого потока) приходят в одном потоке, т.е. нельзя получать данные по конкретным бумагам, получать вы всегда будете все данные, а использовать можете только для нужных бумаг. Робот НЕ строит стакан сразу по всем бумагам, строит только по используемым в портфелях бумагам. Для добавления новой бумаги в список бумаг, по которым строится стакан, необходимо переоткрыть поток с текущим слепком стакана, а потом накатывать на него обновления из потока инкрементальных обновлений. В момент получения потока со слепком стакана стакан в роботе временно перестает обновляться по всем используемым бумагам, он начнет обновляться только после того как будет закрыт поток со слепком стакана (это может занять некоторое время, зависящее от общего количества бумаг, приходящих в данном потоке, и размера стаканов) и начнут применяться инкрементальные обновления. Соответственно, в момент переоткрытия стакана вы можете наблюдать отсутствие цен по бумагам.

Стакан в роботе всегда переоткрывается в следующих случаях:

1. создание нового портфеля;
2. добавление бумаги в портфель;



3. снятие флага *Disabled* с портфеля;
4. пропуски в номерах сообщений инкрементального потока обновлений стакана в UDP подключениях к биржам (чем больше у вас портфелей и бумаг, тем выше вероятность возникновения этих пропусков).

## Возможные проблемы и их решение

Описание наиболее распространенных ошибок в работе робота и способов их устранения.

### ● Проблема

При подключении к серверной части меняются настройки торговых портфелей.

#### Решение

Все правильно, так и должно быть. Все изменения настроек портфелей рекомендуется производить только когда выполнено подключение к серверной части. В противном случае после подключения загрузятся те настройки, которые были переданы на сервер ранее.

### ● Проблема

Торговля включается и выключается самопроизвольно.


#### Решение

Проверьте, не задано ли расписание торговли.

### ● Проблема

Отсутствует нужный инструмент в списке инструментов.

#### Решение


Список бумаг в роботе обновляется каждое утро в 9:25 по Москве, чтобы выгрузить список бумаг из робота в графическую часть необходимо нажать кнопку .

*Важно: время на серверах с крипто роботами — 3 часа от Москвы. Если вы не видите какой-то бумаги в списке бумаг (при этом вы обновили список), а эта бумага уже есть на бирже, то либо дождитесь указанного выше времени и бумага добавится сама, либо переподключите дата подключение (это можно сделать из менеджера подключений, доступно НЕ для всех подключений) и после этого обновите список бумаг в графической части робота.*

### ● Проблема

Робот ставит заявки по второй ноге в "непонятном", неправильном объеме.

#### Решение

Данная ситуация может возникать после какого-либо сбоя на бирже, когда биржа не прислала необходимую информацию по заявкам робота и в роботе "зависли" внутренние статусы заявок. В данной ситуации необходимо остановить торговлю по проблемному портфелю, убедиться что по данному портфелю нет активных заявок в рынке и сбросить статусы заявок, нажав кнопку .

## Наиболее распространенные ошибки, возникающие при работе программы, и способы их устранения

Важно: большинство возникающих ошибок в работе робота, это не ошибки самого робота, а проблемы со связью, между клиентской и серверной частями, вызванные плохим качеством интернет соединения, и ошибки выставления заявок, например, из-за нехватки денег на счете.

Торговые ошибки:

- **Ошибка**

*Order adding error on \*, error: REASON\_NO\_MONEY*

или

*order adding error on \*, order exceeds limit*

### Описание и способ устранения

Нехватка денег на счете. Проверьте хватает ли у вас денег на счете для выставления заявки по данной бумаге данного объема. Торговля будет остановлена автоматически.

Важно: С биржи приходит много разных сообщений об ошибке выставления заявки, в работе они сгруппированы и под ошибку REASON\_NO\_MONEY попадает несколько разных сообщений от биржи, например: "Account has insufficient Available Balance" и "Value of position and orders exceeds position Risk Limit". Эти сообщения присылает биржа в ответ на выставление заявки, робот сам никогда не возвращает ошибку REASON\_NO\_MONEY, это только сообщение с биржи. Почему приходит данное сообщение спрашивайте у соответствующей биржи. Сообщение, непосредственно пришедшее с биржи, вы можете найти в логе графической части робота.

- **Ошибка**

*order adding error on \*, continue trading, error: REASON\_FLOOD*

### Описание и способ устранения

Флуд-контроль, превышен лимит максимального разрешенного количества транзакций (удалений/снятий заявок) в секунду для данного логина или биржа прислала сообщение в духе "Биржа перегружена, попробуйте позже" или некоторые другие сообщения от биржи, которые говорят о том, что биржа сейчас испытывает некоторые "трудности", но в принципе работает. Для данного робота 30 транзакций в секунду (единица транзакционной активности Московской биржи, принятая на текущий момент) – это элементарно.

- **Ошибка**

*order adding error on \*, continue trading, error: REASON\_UNDEFINED*

### Описание и способ устранения

Вы получите эту ошибку в том случае, если ошибку выставления заявки, которую присылает биржа, робот видит впервые (в документации бирж не все сообщения описывают почему-то). В случае данной ошибки в лог графической части приходит сообщение, которое прислала биржа, читайте его внимательнее, абсолютно не факт что что-то не так с роботом и нужно писать в поддержку, и у бирж бывают проблемы. Если же вы понимаете что данное сообщение должно быть как-то обработано роботом, например, как REASON\_FLOOD или REASON\_NO\_MONEY, то пишите в поддержку.

- **Ошибка**

*order adding error on \*, continue trading, error: REASON\_PRICE\_OUT\_OF\_LIMIT*

**Описание и способ устранения**

Данная ошибка означает, что цена заявки вне лимита, и это нормально, так может быть, т.к. цены бумаг в некоторых случаях рассчитываются и зависят от ваших настроек.

- **Ошибка**

*order adding error on \*, continue trading, error: REASON\_CROSS*

**Описание и способ устранения**

Данная ошибка возникает когда вы пытаетесь торговать сами с собой. Когда происходит ошибка робот первую ногу не трогает, а вторую старается выставить на один шаг в сторону улучшения цены.

Неторговые ошибки:

- **Ошибка**

*Error on sendMsg (<class 'ConnectionRefusedError'>, ConnectionRefusedError(10061, 'Подключение не установлено, т.к. конечный компьютер отверг запрос на подключение', None, 10061, None))*

или

*Error on sendMsg (<class 'ConnectionRefusedError'>, ConnectionRefusedError(111, 'Connection refused'))*

**Описание и способ устранения**

Нет соединения с роботом, скорее всего робот просто не запущен в данный момент. Проверьте запущена ли серверная часть робота, если у вас нет на это прав, то обратитесь к администратору за информацией о состоянии робота. Если данная ошибка продолжалась порядка нескольких минут и после этого клиентская часть снова подключилась к серверной, то, возможно, серверная часть по какой-то причине была перезапущена.

- **Ошибка**

*Error on sendMsg (<class 'socket.timeout'>, timeout('timed out',))*

или

*Error on sendMsg (<class 'OSError'>, OSError(113, 'No route to host'))*

или

*Error on sendMsg (<class 'OSError'>, OSError(101, 'Network is unreachable'))*

**Описание и способ устранения**

Нет соединения с компьютером на котором запущена серверная часть. Вероятнее всего указан не верный IP-адрес серверной части в настройках подключения, либо на удаленной машине запущен фаервол, который запрещает входящие подключения, либо отсутствует соединение с интернетом (или с локальной сетью) на машине где запущена клиентская часть, либо отсутствует соединение с интернетом (или с локальной сетью) на машине где запущена серверная часть. Если отсутствует соединения с сетью на машине, на которой запущена серверная часть, то сообщение всегда *socket.timeout*.

- **Ошибка**

*Can't parse message from server (<class 'xml.etree.ElementTree.ParseError'>, ParseError('no element found: line 1, column 0',))*

**Описание и способ устранения**

Данная ошибка сопутствует предыдущей ошибке, так как программа не получила валидное сообщение от серверной части и не смогла его корректно обработать.

- **Ошибка**

*Multiple client connection is not allowed!!!*

**Описание и способ устранения**

Запрещено одновременное подключение нескольких клиентских частей к серверной. Убедитесь что у Вас нигде не запущена вторая клиентская часть уже подключенная к серверной части. Подождите 10 секунд, и попробуйте подключиться снова.

- **Ошибка**

*Error on sendMsg (<class 'OSError'>, OSError(9, 'Bad file descriptor'))*

или

*Error on sendMsg (<class 'OSError'>, OSError(10038, 'Сделана попытка выполнить операцию на объекте, не являющемся сокетом', None, 10038, None))*

**Описание и способ устранения**

Попытка подключения используя уже закрытый сокет. Скорее всего сокет был принудительно закрыт серверной частью из-за попытки подключиться одновременно второй клиентской частью

- **Ошибка**

*Error reading application version file "\*.version"from site, during update check: (<class 'urllib.error.URLError'>, URLError( TimeoutError(10060, 'Попытка установить соединение была безуспешной, т.к. от другого компьютера за требуемое время не получен нужный отклик, или было разорвано уже установленное соединение из-за неверного отклика уже подключенного компьютера', None, 10060, None))))*

или

*Error reading application version file "\*.version"from site, during update check: (<class 'urllib.error.URLError'>, URLError( gaierror(-5, 'No address associated with hostname')))*

**Описание и способ устранения**

Программа не смогла проверить обновление на сервере из-за проблем со связью.

## Описание параметров программы

### Введение

Робот производит торговлю несколькими портфелями одновременно, параметры портфелей приведены ниже. Параметры делятся на редактируемые (т.е. собственно настройки) и отображаемые или расчетные (например, финансовый результат). Параметры портфеля – это параметры отображаемые в таблице на главном окне робота и в окне редактирования настроек портфеля НЕ в таблице, параметры инструментов портфеля – это параметры отображаемые в таблице в окне редактирования настроек портфеля. Параметры уведомлений портфеля – это редактируемые параметры, отображаемые в окне настроек уведомлений портфеля.

### Параметры портфеля (редактируемые)

1. **Name** – имя портфеля (всегда НЕ редактируемое, т.к. является уникальным ключом);
2. **re\_sell** – разрешена продажа (при взведении флага включает робот на продажу);
3. **re\_buy** – разрешена покупка (при взведении флага включает робот на покупку);
4. **v\_in\_left**<sup>0</sup> – минимальный разрешенный объем для входа в позицию (в штуках портфелей);
5. **v\_in\_right**<sup>0</sup> – максимальный разрешенный объем для входа в позицию (в штуках портфелей);
6. **v\_out\_left**<sup>0</sup> – минимальный разрешенный объем для выхода из позиции (в штуках портфелей), разрешено выставление меньшим объемом в случае сведения позиции к нулю данной заявкой;
7. **v\_out\_right**<sup>0</sup> – максимальный разрешенный объем для выхода из позиции (в штуках портфелей);
8. **Virtual 0 pos**<sup>0</sup> – котирующая заявка по *Is first* бумаге с направлением в сторону закрытия позиции может не только сводить позицию к нулю, но и сразу открывать новую позицию с противоположным направлением, кроме того объем заявки никогда не может быть меньше *v\_in\_left* или *v\_out\_left*;
9. **Delta** – минимальное отклонение *Price\_s* и *Price\_b* от цены выставленной заявки на продажу или покупку, соответственно, при превышении которого необходимо переставить котируемую заявку, то есть заявку по *Is first* инструменту (используется только при включенном режиме *Quote*);
10. **Pos**<sup>1</sup> – текущая позиция портфеля (в штуках портфелей), вычисляется по формуле  $Pos = [Curpos_{first}/Count_{first}]$ , где *Curpos<sub>first</sub>* и *Count<sub>first</sub>* – это параметры *Curpos* и *Count* для инструмента портфеля с взведенным флагом *Is first*;
11. **Lim\_Sell** – сигнальная цена на продажу, заявка по *Is first* бумаге выставляется если  $Sell \geq Lim\_Sell$  в не зависимости от того включен или нет режим *Quote*;
12. **Lim\_Buy** – сигнальная цена на покупку, заявка по *Is first* бумаге выставляется если  $Buy \leq Lim\_Buy$  в не зависимости от того включен или нет режим *Quote*;

13. **First delta** – задается в процентах (%), котируемая заявка переставляется если ее текущий не исполненный объем меньше, чем *First delta* от первоначального выставленного объема (используется только при включенном режиме *Quote*);
14. **Market volume** – максимальное количество лотов по лучшей цене на продажу или покупку (или суммарное количество лотов в стакане до "нашей" предполагаемой цены если *Type price = Orderbook* или *Type price = Orderbook + filter*), соответственно, при котором выставляется заявка по *Is first* бумаге, при условии, что заявка выставляется не в спред, если цена заявки попадает в спред то значение данного параметра не используется;
15. **Price check** – если предполагаемая цена выставления заявки по *Is first* бумаге попадает в стакан глубже, чем на *Price check* пунктов, то не выставляться (то есть если  $offer + Price\ check < Price\_s$ , где *offer* – лучшая цена на продажу *Is first* инструмента, то не выставляться, для покупки аналогично);
16. **v\_min**<sup>1</sup> – минимальная разрешенная позиция для главного инструмента портфеля (в лотах);
17. **v\_max**<sup>1</sup> – максимальная разрешенная позиция для главного инструмента портфеля (в лотах);
18. **Quote** – котировать инструмент, если включено, то заявка всегда держится в стакане, если выключено, то заявка на продажу выставляется когда  $Sell \geq Lim\_Sell$ , заявка на покупку выставляется когда  $Buy \leq Lim\_Buy$ ;
19. **Limits timer** – время таймера, таймер включается если торговля включена и проходит сигнал на покупку/продажу, но торговля запрещена из-за того, что позиция (*Curpos*) по главной бумаге равна *v\_max* или *v\_min*, соответственно;
20. **Percent** – используется при срабатывании *Limits timer*, если сигнал на торговлю продержался *Percent* процентов времени *Limits timer* то *Lim\_Sell* и *Lim\_Buy* передвигаются на *K*, не смотря на отсутствие сделок по *Is first* бумаге;
21. **Use timetable** – использовать торговлю по расписанию, расписание представляет собой список групп параметров:
  - (a) **Begin** – начало торгового периода;
  - (b) **End** – окончание торгового периода;
  - (c) **Close** – попытаться закрыть позицию сразу после окончания торгового периода;
  - (d) **To market** – попытаться выровнять позицию сразу после окончания торгового периода;
  - (e) **To0** – использовать *To0* в данный период;

если текущее локальное время на компьютере (*curTime*) попадает в один из периодов, т.е.  $Begin_i \leq curTime \leq End_i$ , то *re\_sell* и *re\_buy* включаются автоматически (кроме режима *Timetable only stop*), в противном случае (если текущее время не попадает ни в один из периодов) *re\_sell* и *re\_buy* выключаются автоматически;

22. **Timetable only stop** – автоматически выключать торговлю если текущее время не попадает в периоды расписания, но при этом автоматически не включать торговлю если время попадает в один из периодов расписания;

23. **To0<sup>0</sup>** – если флаг взведен, то не открывать позицию, а только закрывать и не открываться из нулевой позиции;
24. **Opened** – параметр, используемый для подсчета финансового результата, вычисляется по формуле:

$$Opened = - \left( \sum_{i \in bought} tradePrice_i \times tradeAmount_i \times Mult_i \right) + \left( \sum_{i \in sold} tradePrice_i \times tradeAmount_i \times Mult_i \right),$$

где  $tradePrice_i$  – цена сделки, а  $tradeAmount_i$  – количество лотов в сделке,  $bought$  – список сделок на покупку,  $sold$  – список сделок на продажу,  $Mult$  – *Fin res multiplier* инструмента портфеля;

25. **Commission sum** – сумма комиссии по всем сделкам портфеля, используется для подсчета финансового результата;
26. **Decimals** – параметр, определяющий сколько знаков после десятичной точки отображается в параметрах, для которых значение является дробным числом;
27. **In formulas** – флаг, если взведен, то портфель и его инструменты могут использоваться в формулах на C++ (рекомендуется снимать данный флаг для ненужных в формулах портфелей, это ускоряет работу робота);
28. **Disabled** – полностью выключить портфель из всех расчетов и торговли не удаляя его;
29. **Sell clicker** – ”кликер”, выставить заявку на продажу заданного в количестве портфелей объема;
30. **Buy clicker** – ”кликер”, выставить заявку на покупку заданного в количестве портфелей объема;
31. **To market** – ”кликер”, снять все заявки и выставить их по рыночным ценам (с учетом параметра  $k\_sl$ ) и выровнять позицию;
32. **K<sup>2</sup>** – параметр, используемый для смещения значений  $Lim\_Sell$  и  $Lim\_Buy$  (пример использования описан ниже);
33. **K1<sup>2</sup>** – параметр, используемый для смещения значений  $Lim\_Sell$  и  $Lim\_Buy$  (пример использования описан ниже);
34. **K2** – параметр, используемый для смещения значений  $Lim\_Sell$  и  $Lim\_Buy$  (пример использования описан ниже);
35. **TP** – параметр, используемый для смещения значений  $Lim\_Sell$  и  $Lim\_Buy$  (пример использования описан ниже);
36. **Order ID** – идентификатор всех заявок портфеля, так же определяет приоритет вызова торгового алгоритма портфеля. Рассмотрим пример, пусть есть несколько портфелей и во всех этих портфелях есть один и тот же инструмент. При изменении цены или объема лучшей цены на покупку/продажу по данному инструменту вызывается алгоритм торговли по всем портфелям, в которых этот инструмент присутствует. Торговый алгоритм портфелей будет вызываться в порядке возрастания *Order ID*, т.е. самым первым вызовется

торговый алгоритм портфеля с  $Order ID = a$ , а самым последним – с  $Order ID = z$ , при совпадении  $Order ID$  у разных портфелей порядок вызова торгового алгоритма портфелей с одинаковым  $Order ID$  НЕ определен и будет произвольным;

37. **Hedge (sec)** – интервал времени в секундах по прошествии которого автоматически хеджировать незахеджированную позицию (значение  $-1$  – не использовать данный параметр);
38. **Type** – тип торгового алгоритма портфеля:
- *Arbitrage* – обычная арбитражная торговля с использованием всех заданных параметров;
  - *Option hedge* – режим хеджирования, *Count* всех инструментов кроме *Is first* равен 1, *Count* для *Is first* инструмента задается как  $\frac{1}{delta}$ , где *delta* – ”грек”, вычисляемый с использованием модели Блека-Шоулза, *Is first* инструмент в таком режиме должен быть опционом;
  - *Test algo* – тестовый режим для проверки скорости работы ”движков” подключений на получение рыночных данных и на выставление заявок, использовать данный режим без обращения к технической поддержке НЕ рекомендуется, прибыли он НЕ приносит;
  - *TP algo* – режим работы с выставление *take profit* заявки по главному инструменту (доступно НЕ во всех версиях программы);
  - *TP algo 2* – режим работы с выставление *take profit* заявки по главному инструменту после каждой сделки (не по этой *take profit* заявке), у выставленной заявки есть *Timer* и *SL* (доступно НЕ во всех версиях программы);
39. **Type trade** – тип торговли, используется при расчете цен *Sell*, *Buy*, *Price\_s*, *Price\_b*:
- *Price* – режим торговли с расчетом вышеперечисленных цен по ценам инструментов на покупку и продажу;
  - *IV* – режим торговли с расчетом вышеперечисленных цен по *implied volatility* инструментов с использованием модели Блека-Шоулза;
40. **Overlay** – хеджировать только если разница (в портфелях) между *Is first* бумагой и остальными инструментами портфеля больше или равна значению данного параметра (в штуках портфелей, то есть в той же размерности, что и *v\_in\_left* и *v\_in\_right*);
41. **Type price** – тип определения цены инструмента (доступно НЕ во всех версиях программы):
- *Bid/offer* – использовать лучшую цену на покупку и лучшую цену на продажу;
  - *Orderbook* – искать цену в стакане, таким образом, чтобы набрать необходимый объем  $Count \times Percent\ of\ quantity \times 0.01 \times \begin{cases} v\_in\_left, & \text{if } open\ pos \\ v\_out\_left, & \text{if } close\ pos \end{cases}$ , набирать цены на покупку и продажу среди цен в стакане, начиная от лучшей цены в нужном направлении и далее вглубь стакана;
  - *Orderbook+filter* – аналогично *Orderbook*, но вычитать из набираемого объема цены своих заявок (если на соответствующих ценах присутствуют свои заявки);
42. **Custom trade** – использовать *Trade formula* для подсчета раздвижки в таблице с финансовыми результатами (доступно НЕ во всех версиях программы);



43. **Trade formula** – формула на языке программирования C++ для подсчета раздвижки в таблице с финансовыми результатами, вы пишете только тело функции и должны вернуть значение типа *double* (*доступно НЕ во всех версиях программы*);
44. **Simply first** – если флаг взведен, то если цены *Price\_s* и *Price\_b* попадают в спред или на противоположную сторону стакана, то они всегда будут выставляться не глубже, чем на один шаг цены в спред:

$$Price\_s_1 = \max(Price\_s_0, offer - step),$$

$$Price\_b_1 = \min(Price\_b_0, bid + step),$$

где *bid*, *offer*, *step* – это бид, оффер и шаг цены по *Is first* бумаге, нижний индекс 0 означает текущее значение параметра, нижний индекс 1 означает новое значение параметра; если у *Is first* бумаги взведен флаг *Maker* и текущий спред в стакане равен одному шагу цены, то заявка на продажу будет выставлена по цене *offer*, а на покупку – по цене *bid* (в противном случае заявка просто не смогла бы выставиться и ”спамила” бы биржу, что не хорошо) (*доступно НЕ во всех версиях программы*);

Важно: если наша заявка становится бидом или оффером, при этом является единственной заявкой на данном ценовом уровне и цена следующего уровня стакана отличается от цены нашей заявки более чем на один шаг цены, то заявка снимается и далее выставляется по ценам, описанным выше.

45. **Always timer** – всегда использовать *Limits timer*, даже тогда, когда можно торговать и происходит выставление заявки (*доступно НЕ во всех версиях программы*);
46. **Equal prices** – выставлять вторую ногу по такой цене, чтобы *Sell = Lim\_Sell* и *Buy = Lim\_Buy* (работает только для портфелей с двумя бумагами) (*доступно НЕ во всех версиях программы*);
47. **Max not hedged** – максимальное количество не захеджированных открытий по *Is first* инструменту (т.е. когда по любому из не *Is first* инструментов ”висит” в рынке не менее, чем *Max not hedged* активных заявок), после которого торговля по *Is first* инструменту будет остановлена до тех пор пока хотя бы одна из незахеджированных позиций не захеджируется;
- Важно: даже если *Max not hedged* = 1 и по какой-то причине не выставились заявки по не *Is first* бумагам, то позиция будет автоматически выравнена при следующей сделке по *Is first* бумаге (если снова не будет ошибок выставления) и, соответственно, торговля будет разрешена при НЕ ровной позиции портфеля (до тех пор пока позиция снова не станет ровной). То есть разрешено торговать при НЕ ровной позиции, НО только в случае ошибок выставления не *Is first* инструментов.
48. **Extra formulas** – флаг, включает расчет *Extra field#1* и *Extra field#2* (*доступно НЕ во всех версиях программы*);
49. **Extra field#1** – формула на языке программирования C++, вы пишете только тело функции и должны вернуть значение типа *double* (нигде в алгоритме никак не используется, чтобы активировать, используйте флаг *Extra formulas*) (*доступно НЕ во всех версиях программы*);
50. **Extra field#2** – формула на языке программирования C++, вы пишете только тело функции и должны вернуть значение типа *double* (нигде в алгоритме никак не используется, чтобы активировать, используйте флаг *Extra formulas*) (*доступно НЕ во всех версиях программы*);

51. **Color** – цвет ячейки таблицы, используемый для более удобной "идентификации" портфеля.

*Замечание 0:* значение параметра *right* должно быть больше или равно значения параметра *left* (иначе просто торговать не будет), это условие НЕ проверяется в клиентской части. Для всех не *Is first* инструментов вычисляется количество портфелей которое возможно выставить по каждому из этих инструментов, по формуле:

$$v_i = \frac{amount_i \times 100}{Percent\ of\ quantity_i \times Count_i},$$

где  $amount_i$  – объем лучшей цены на покупку/продажу в зависимости от направления торговли. После этого вычисляется  $v = \min_i v_i$ , это объем заявки в штуках портфелей, на который хватает бумаг в рынке. Далее вычисляется  $q$  – итоговый выставляемый объем заявки в штуках портфелей существует несколько случаев:

- вычисляем объем заявки на покупку

- текущая позиция больше или равна нулю:

если включен режим *To0*, то  $q = 0$ , иначе если  $v \geq v\_in\_left$  или  $count_{first} \times v \geq v\_max - pos_{first}$ , то  $q = \min(count_{first} \times \min(v\_in\_r, v), v\_max - pos_{first}, |pos_{first}|)$ , если включен режим *virtual\_0\_pos* и  $q < count_{first} \times v\_in\_l$ , то  $q = 0$ ; иначе  $q = 0$

- текущая позиция меньше нуля:

если  $v \geq v\_out\_l$  или  $count_{first} \times v \geq v\_max - curpos_{first}$  или  $count_{first} \times v \geq |pos_{first}|$ , то  $q = \min(count_{first} \times \min(v\_out\_right, v), v\_max - pos_{first}, |pos_{first}|)$ , если включен режим *virtual\_0\_pos* и  $q < count_{first} \times v\_out\_l$ , то если  $v \geq v\_out\_left$  и  $0 \leq pos_{first} + count_{first} \times v\_out\_left \leq v\_max$ , то  $q = count_{first} \times v\_out\_l$  иначе  $q = 0$ ; иначе  $q = 0$

- вычисляем объем заявки на продажу

- текущая позиция меньше или равна нулю:

если включен режим *To0*, то  $q = 0$ , иначе если  $v \geq v\_in\_left$  или  $count_{first} \times v \geq -v\_min + pos_{first}$ , то  $q = \min(count_{first} \times \min(v\_in\_r, v), -v\_min + pos_{first}, |pos_{first}|)$ , если включен режим *virtual\_0\_pos* и  $q < count_{first} \times v\_in\_l$ , то  $q = 0$ ; иначе  $q = 0$

- текущая позиция больше нуля:

если  $v \geq v\_out\_l$  или  $count_{first} \times v \geq -v\_min + curpos_{first}$  или  $count_{first} \times v \geq |pos_{first}|$ , то  $q = \min(count_{first} \times \min(v\_out\_right, v), -v\_min + pos_{first}, |pos_{first}|)$ , если включен режим *virtual\_0\_pos* и  $q < count_{first} \times v\_out\_l$ , то если  $v \geq v\_out\_left$  и  $v\_min \leq pos_{first} - count_{first} \times v\_out\_left \leq 0$ , то  $q = count_{first} \times v\_out\_l$  иначе  $q = 0$ ; иначе  $q = 0$

Где  $count_{first}$  – Count *Is first* бумаги,  $pos_{first}$  – Curpos *Is first* бумаги.

*Замечание 1:*  $v\_min$  и  $v\_max$  задаются НЕ в той же размерности, что позиция портфеля (*Pos*).

*Замечание 2:* рекомендуется, чтобы значение параметра  $K$  было больше, чем значение параметра  $K1$  ( $K > K1$ ), в противном случае, исходя из правил изменения цен *Lim\_Sell* и *Lim\_Buy*, вы можете получить  $Lim\_Buy > Lim\_Sell$ , и робот будет продавать дешевле, чем покупает.

## Параметры портфеля (отображаемые)

1. **Sell** – расчетная цена на продажу;
2. **Buy** – расчетная цена на покупку;
3. **Price\_s** – цена выставления заявки на продажу по *Is first* бумаге, вычисляется как обратная функция для *Sell*, где цена *Sell* заменяется на *Lim\_Sell*;
4. **Price\_b** – цена выставления заявки на покупку по *Is first* бумаге, вычисляется как обратная функция для *Buy*, где цена *Buy* заменяется на *Lim\_Buy*;
5. **Sell status** – статус заявки на продажу по *Is first* бумаге (не используется при выставлении заявки с помощью "кликеров"). Для того чтобы освободить "зависшую" заявку, необходимо сделать двойной клик на ячейке таблицы. Ручная смена статуса может привести к потере заявки роботом, данную операцию рекомендуется делать только в крайних случаях;
6. **Buy status** – статус заявки на покупку по *Is first* бумаге (не используется при выставлении заявки с помощью "кликеров"). Для того чтобы освободить "зависшую" заявку, необходимо сделать двойной клик на ячейке таблицы. Ручная смена статуса может привести к потере заявки роботом, данную операцию рекомендуется делать только в крайних случаях;
7. **Start nearest** – ближайшее по расписанию время автоматического включения портфеля;
8. **Stop nearest** – ближайшее по расписанию время автоматического выключения портфеля;
9. **Avg opened** – вычисляется по формуле  $Avg\ opened = Opened / Pos$ ;
10. **Return first** – оборот по *Is first* бумаге, вычисляется с момента старта серверной части работа как сумма модулей количеств лотов в сделках по *Is first* инструменту;
11. **Fin res** – предполагаемый финансовый результат портфеля, вычисляется по формуле:

$$Fin\ res = Opened + Commission\ sum + \sum_{i \in secs} Curpos_i \times Mult_i \times \begin{cases} secBid_i, & \text{if } Curpos_i > 0 \\ secOffer_i, & \text{if } Curpos_i < 0 \end{cases},$$

где  $secBid_i$  – лучшая цена на покупку инструмента портфеля,  $secOffer_i$  – лучшая цена на продажу инструмента портфеля,  $Curpos_i$  – текущая позиция инструмента портфеля,  $Mult$  – *Fin res multiplier* инструмента портфеля,  $secs$  – список инструментов портфеля. Двойной клик по ячейке в таблице переключает отображение финансового результата с учетом комиссии или без ее учета (финансовый результат без учета комиссии выделен жирным шрифтом);

12. **Extra field#1** – просто отображаемое значение, рассчитываемое по соответствующей пользовательской формуле (*доступно НЕ во всех версиях программы*);
13. **Extra field#2** – просто отображаемое значение, рассчитываемое по соответствующей пользовательской формуле (*доступно НЕ во всех версиях программы*).

## Параметры инструментов портфеля (всегда редактируемые, если не указано обратное)

1. **SecKey** – уникальный идентификатор инструмента портфеля (не редактируемый);
2. **SecBoard** – режим инструмента портфеля (не редактируемый);
3. **SecCode** – код инструмента портфеля (не редактируемый);
4. **SecType** – тип инструмента (например, *CURR* для валютного рынка) (не редактируемый);
5. **Curpos** – текущая позиция робота по данной бумаге в лотах;
6. **Count** – количество лотов инструмента в одном портфеле;
7. **Count type** – использовать *Count* или *Count formula* (доступно *НЕ* во всех версиях программы);
8. **Count formula** – количество лотов инструмента в одном портфеле, задается как код на языке программирования C++, вы пишете только тело функции, и должны вернуть значение типа *double* (доступно *НЕ* во всех версиях программы);

*Важно:* значения *Count* или *Count formula* определяют соотношение именно между позициями инструментов портфеля (соотношение в конкретной сделке может отличаться). По этой же причине значение *Count formula* не зависит от направления выставляемой заявки.

*Важно:* настоятельно рекомендуется для *Is first* бумаги никогда не возвращать значение 0, если вы хотите не торговать, используйте *Ratio formula* и задавайте необходимые значения для раздвижки. Если вы все-таки получили *Count* равный 0 для *Is first* бумаги, то портфель не будет торговать ни одной бумагой и для подсчета позиции портфеля в том месте где необходимо поделить на *Count Is first* бумаги (который в вашем случае равен 0) будет делиться на 1.

9. **k** – используется для определения цены выставления заявки по алгоритму (отступ от рыночной цены (в пунктах), т.е. при покупке цена выставления  $offer + k$ , при продаже цена выставления  $bid - k$ , где *bid* и *offer* – лучшие цены на продажу и покупку, соответственно);
10. **On buy** – определяет покупаем мы или продаем инструмент при срабатывании сигнала на покупку по главному инструменту портфеля;
11. **Is first** – определяет главный инструмент портфеля, этот инструмент выставляется первым и позиция портфеля считается по нему;
12. **SLE<sup>3</sup>** – включить/выключить функцию переставления по стоп-лоссу;
13. **SL<sup>3</sup>** – значение стоп-лосса (в пунктах), при достижении которого необходимо снимать заявку, если она не прошла до этого момента и бросать снова по новой рыночной цене (стоп-лосс откладывается от первоначальной цены выставления заявки);
14. **k\_sl<sup>3</sup>** – аналог параметра *k* для работы по стоп-лоссу и по таймеру;
15. **TE<sup>3</sup>** – включить/выключить функцию переставления по таймеру;
16. **Timer<sup>3</sup>** – параметр, определяющий через сколько времени снимать заявку, если она не прошла до этого момента и бросать снова по новой рыночной цене;

17. **Percent of quantity** – если на бирже в объеме лучшей цены на продажу или покупку (или в найденном объеме в стакане если *Type price = Orderbook* или *Type price = Orderbook + filter*), соответственно, есть нужное количество процентов (%) от объема заявки инструмента не являющегося *Is first* и это условие выполняется для всех не *Is first* инструментов, то можно выставляться по *Is first* инструменту;
18. **Ratio sign** – знак используемый перед параметром *Ratio* при расчете цен *Sell* и *Buy*, "+" или "×";
19. **Ratio** – параметр, используемый при расчете цен *Sell* и *Buy*;
20. **Fin res multiplier** – множитель, используется для подсчета финансового результата, чтобы привести все цены к одной размерности;
21. **Commission type** – параметр, определяющий тип расчета комиссии;
22. **Commission** – комиссия по инструменту, если *Comission type = "%"*, то комиссия указывается в процентах от цены сделки, если *Comission type = "pt"*, то комиссия указывается в той же размерности, в которой считается финансовый результат по портфелю (например, для акции Сбербанка комиссия указывается в процентах и для большинства брокеров она равна 0.01%, а для фьючерса на акцию Сбербанка комиссия указывается в пунктах и равна 0.25 пункта для скальперских сделок);
23. **Client code**<sup>5</sup> – код клиента с которого надо выставлять заявку по данной бумаге или пустая строка если необходимо выставлять с "кода по-умолчанию", отображает все коды, которые прописаны в настройках робота, код *To file* означает торговлю "в файл"; код, начинающийся с *Round robin*, означает выставление и снятие заявок поочередно через все подключения к бирже с заданным клиентским кодом, порядок подключений в очереди зависит от скорости выставления заявок через эти подключения (кто в данный момент быстрее, тот первый в очереди, порядок подключений в очереди меняется раз в секунду, кроме того каждую секунду движение по очередит начинается заново, т.о. загрузка подключений НЕ равномерная);  
*Важно:* код клиента не может быть пустым для бумаг с *Count* отличным от нуля. Скорость параметра *Round robin* считается только по важным заявкам. Важными заявками являются: *is first* заявка, выставленная по алгоритму первой ноги (т.е не кликером, не стопом и прочими способами), либо по второй ноге брошенная по алгоритму после сделки по 1 ноге. Раз в час сбрасывается значение скоростей для того, чтобы проверить скорость выставления заявок всех подключений.
24. **MM** – флаг, если взведен, то все заявки по инструменту выставляются с признаком "заявка маркет-мейкера" (доступно не для всех подключений);
25. **TP** – величина тейк-профит, используется при *Type* равном *TP algo*, откладывается от цены сделки по заявке *Is first* инструмента (доступно НЕ во всех версиях программы);
26. **Ratio type** – использовать *Ratio* или *Ratio formula* (доступно НЕ во всех версиях программы);
27. **Ratio sell formula** – параметр, используемый при расчете цены *Sell*, задается как код на языке программирования C++, вы пишете только тело функции, и должны вернуть значение типа *double* (доступно НЕ во всех версиях программы);

28. **Ratio buy formula** – параметр, используемый при расчете цены *Buy*, задается как код на языке программирования C++, вы пишете только тело функции, и должны вернуть значение типа *double* (*доступно НЕ во всех версиях программы*);

29. **FUT move limits**<sup>4</sup> – флаг, если взведен, то при каждой смене дня будет осуществляться автоматическая "подвижка" лимитов по формулам:

$$\begin{aligned} Lim\_Sell_1 &= Lim\_Sell_0 - \frac{(Lim\_Sell_0 + Lim\_Buy_0) \times days\_to\_expiry_{SPOT}}{2 \times days\_to\_expiry}, \\ Lim\_Buy_1 &= Lim\_Buy_0 - \frac{(Lim\_Sell_0 + Lim\_Buy_0) \times days\_to\_expiry_{SPOT}}{2 \times days\_to\_expiry}, \end{aligned}$$

где *days\_to\_expiry* – целое количество дней до экспирации данной бумаги, *days\_to\_expiry<sub>SPOT</sub>* – целое количество дней до экспирации бумаги, отмеченной флагом *SPOT move limits*, или 1, если такая бумага не указана, нижний индекс 0 означает текущее значение параметра, нижний индекс 1 означает новое значение параметра;

30. **SPOT move limits**<sup>4</sup> – флаг, если взведен, то данная бумага используется в формулах для *FUT move limits*;

31. **Depth OB** – максимальный уровень глубины стакана (в штуках шагов цены, считая от биды/оффера) до которого включительно вычислять цены и объемы, доступен только для не *Is first* бумаг, используется только в режимах *Type price = Orderbook* и *Type price = Orderbook + filter* (*доступно НЕ во всех версиях программы*);

32. **Calc price OB** – тип цены, используемой для расчета *Sell*, *Buy*, *Price\_s*, *Price\_b*, доступен только для не *Is first* бумаг, используется только в режимах *Type price = Orderbook* и *Type price = Orderbook + filter* (*доступно НЕ во всех версиях программы*):

- *Deepest* – цена того уровня в стакане, на котором набрали искомый объем;
- *Weighted avg.* – средневзвешенная цена до того уровня в стакане включительно, на котором набрали искомый объем;

33. **Trading price OB** – тип цены, используемой при торговле, доступен только для не *Is first* бумаг, используется только в режимах *Type price = Orderbook* и *Type price = Orderbook + filter* (*доступно НЕ во всех версиях программы*):

- *Deepest* – цена того уровня в стакане, на котором набрали искомый объем;
- *Weighted avg.* – средневзвешенная цена до того уровня в стакане включительно, на котором набрали искомый объем;

34. **Level to0** – если хотя бы для одной бумаги портфеля модуль разности *Mark price* и *Liquidity price* строго меньше данного значения, то взвести флаг *To0* и НЕ давать его снять пока условие выполняется, когда условие перестанет выполняться – снять флаг *To0* (*доступно НЕ во всех версиях программы, имеет смысл только для бумаг с BitMEX, Deribit, Cryptofacilities*);

35. **Level close** – если хотя бы для одной бумаги портфеля модуль разности *Mark price* и *Liquidity price* строго меньше данного значения, то взвести флаг *To0* и НЕ давать его снять пока условие выполняется (когда условие перестанет выполняться – снять флаг *To0*), раз в 5 секунд выставлять заявку в направлении закрытия позиции в объеме *v\_out\_left* портфелей до тех пор пока описанное выше условие не перестанет выполняться или позиция по портфелю не станет равн 0 (*доступно НЕ во всех версиях программы, имеет смысл только для бумаг с BitMEX, Deribit, Cryptofacilities*);

36. **Only maker** – выставлять котирующие (т.е. при включенном режиме *Quote*) заявки по *Is first* бумаге с признаком "снять если заявка будет taker" (это параметр заявки, не работает на московской и питерской бирже так как они данный параметр не поддерживают; имеет смысл только для бумаг с BitMEX, Bitfinex, Deribit, Cryptofacilities);

37. **Decimals** – параметр, определяющий сколько знаков после десятичной точки отображается в параметрах, для которых значение является дробным числом.

*Важно:* данный параметр так же отвечает за число знаков после десятичной точки в ценах сделок по данной бумаге в таблице раздвижек. При смене значения, число знаков у уже добавленных в таблицу сделок НЕ изменится.


Замечание 3: параметры *TE*, *Timer*, *SLE*, *SL*, *k\_sl* не используются для бумаги *Is first* при торговле по алгоритму. Используются только при торговле с использованием "кликеров" (*Sell clicker*, *Buy clicker*).

Замечание 4: алгоритм смещения *Lim\_Sell/Lim\_Buy* на размер ставки свопа по заданной па-  
ре, данная опция позволяет учитывать смещение *Sell/Buy* на размер овернайт.

Замечание 5: торговля "в файл" реализована по аналогии с тестером: выставленные заявки проходят только в том случае, если, для покупки, цена заявки больше или равна текущей цене на продажу по данной бумаге и есть достаточное число бумаг на рынке, для продажи, если цена заявки меньше или равна текущей цене на покупку по данной бумаге и есть достаточное число бумаг на рынке (также учитывается параметр, отвечающий за процент прохождения заявок, описанный ниже), причем ценами прохождения заявок ВСЕГДА являются лучшая цена на покупку и лучшая цена на продажу (или цены глубже в стакане, при наличии стакана) для данной бумаги (т.е. идеино это работает как будто мы всегда "бьем" по стоящим в стакане ценам).

Замечание 6: в параметрах портфеля и в параметрах инструментов есть параметры с одинаковыми или очень похожими названиями, это РАЗНЫЕ параметры.

## Параметры уведомлений портфеля (редактируемые)

Если возле названия уведомления присутствует иконка , то это обновление будет приходить в telegram-бота.

1. **Trades frequency notifications** – уведомлять о большом количестве сделок ( $\geq Count$ ) по портфелю за *Time* секунд:

(a) **Trades freq type** – тип расчета сигнального значения:

- *Trades count* – считать количество сделок;
- *Trades quantity* – считать суммарный объем сделок;

(b) **Time (sec)** – период времени, за который считать сигнальное количество сделок (откладывается назад от текущего времени);

(c) **Count** – сигнальное количество сделок или сигнальный суммарный объем сделок (в зависимости от *Trades freq type*);

2. **FinRes fall notifications** – уведомлять о падении финансового результата по портфелю за *Time* секунд, то есть когда текущий финансовый результат  $> \max(\text{Min fall}, \text{Fin res}_{\text{saved}} - |\text{Fin res}_{\text{saved}} \times \text{Fall} \times 0.01|)$ , где *Fin res<sub>saved</sub>* – "эталонный" финансовый результат сохраняемый (обновляемый) каждые *Time* секунд:

- (a) **Time (sec)** – период времени в секундах, откладываемый от момента включения робота или редактирования параметров, влияющих на вычисление финансового результата, с указанной периодичностью будет обновляться "эталонный" финансовый результат;
  - (b) **Min fall (pt)** – если процентное изменение финансового результата меньше данной величины, то не уведомлять;
  - (c) **Fall (%)** – процентное изменение "эталонного" финансового результата о достижении которого необходимо уведомить пользователя;
  - (d) **Stop trading** – вместе с уведомлением выключить торговлю по портфелю (расписание также будет выключено);
3. **Lim\_Sell change notifications** – уведомлять об изменении *Lim\_Sell* портфеля за *Time* секунд больше, чем на *Value*:
- (a) **Time (sec)** – период времени в секундах, не чаще, чем с указанной периодичностью, будет обновляться "эталонный" *Lim\_Sell*;
  - (b) **Value** – сигнальное значение изменения *Lim\_Sell*;
  - (c) **Stop trading** – вместе с уведомлением выключить торговлю по портфелю (расписание также будет выключено);
4. **Lim\_Buy change notifications** – уведомлять об изменении *Lim\_Buy* портфеля за *Time* секунд больше, чем на *Value*:
- (a) **Time (sec)** – период времени в секундах, не чаще, чем с указанной периодичностью, будет обновляться "эталонный" *Lim\_Buy*;
  - (b) **Value** – сигнальное значение изменения *Lim\_Buy*;
  - (c) **Stop trading** – вместе с уведомлением выключить торговлю по портфелю (расписание также будет выключено);
5. **Severe sell change notification** – уведомлять о "резком" изменении *Sell* портфеля ( $\geq$  *Value*) за *Time* секунд (только когда портфель включен):
- (a) **Time (sec)** – период времени, за который считать изменение *Sell* (откладывается назад от текущего времени) как разницу между текущим *Sell* и *Sell Time* секунд назад;
  - (b) **Value** – сигнальное значение изменения *Sell*;
6. **Severe buy change notification** – уведомлять о "резком" изменении *Buy* портфеля ( $\geq$  *Value*) за *Time* секунд (только когда портфель включен):
- (a) **Time (sec)** – период времени, за который считать изменение *Buy* (откладывается назад от текущего времени) как разницу между текущим *Buy* и *Buy Time* секунд назад;
  - (b) **Value** – сигнальное значение изменения *Buy*;
7. **Severe pos change notification** – уведомлять о "резком" изменении позиции (*Pos*) по портфелю ( $\geq$  *Value*) за *Time* секунд:
- (a) **Time (sec)** – период времени, за который считать изменение позиции (откладывается назад от текущего времени) как разницу между текущей позицией и позицией *Time* секунд назад;



(b) **Value** – сигнальное значение изменения позиции;

8. **Too much running orders notification** – уведомлять о слишком большом количестве активных заявок по не *Is first* инструментам портфеля ( $max$  (число активных заявок по каждому из не *Is first* инструментов портфеля)  $\geq Percent \times Max\ not\ hedged \times 0.01$ , где *Max not hedged* по-умолчанию равен 30):

(a) **Percent (%)** – сигнальный процент;

9. **Too much not hedged notification** – уведомлять о слишком большой не захеджированной позиции по *Is first* инструменту портфеля:

(a) **Limit portfolios** – сигнальное значение не захеджированной позиции (вычисляется в штуках портфелей).

## Порядок выставления заявок и механизм выравнивания позиции

Робот выставляет заявку по *Is first* бумаге при торговле в режиме *Quote* в зависимости от настроек, но независимо от сигналов на продажу/покупку, по *Lim\_Sell/Lim\_Buy* или по сигналам *Lim\_Sell/Lim\_Buy* (условие для выставления было описано ранее) при выключенном режиме *Quote*. При прохождении сделки по *Is first* бумаге выставляются заявки по остальным инструментам портфеля, направление заявок определяется направлением сделки по *Is first* бумаге а также значением параметра *On buy Is first* и не *Is first* инструмента. Объем для выставления каждого не *Is first* инструмента вычисляется исходя из текущей позиции в портфеле по *Is first* бумаге и *Count* как по *Is first* бумаге так и по текущей, таким образом чтобы отношение "новой" (которая будет после прохождения сделки по еще невыставленной, но выставляемой в данный момент заявке) позиции текущей бумаги к позиции по *Is first* бумаге было равно отношению *Count* не *Is first* бумаги к *Count Is first* бумаги. Причем если по данному инструменту в данном портфеле есть неисполненные заявки, то они учитываются как исполненные при определении "новой" позиции по данной не *Is first* бумаге.

Важно: Робот всегда выставляет лимитные котировочные заявки. Для *Is first* инструмента ценами выставления являются:  $Price\_s - k$  на продажу и  $Price\_b + k$  на покупку. Для не *Is first* инструментов ценами выставления являются (для основной ветки алгоритма):  $bid - k$  на продажу и  $offer + k$  на покупку.

## Правила перемещения *Lim\_Sell* и *Lim\_Buy*

Сигнальные цены *Lim\_Sell* и *Lim\_Buy* перемещаются только при прохождении сделок по *Is first* инструменту портфеля.

Правила перемещения сигнальных цен можно разделить на две части: произошла продажа по *Is first* бумаге и произошла покупка по *Is first* бумаге. Внутри каждой из этих частей алгоритм делится еще на две части: позиция портфеля до прохождения данной сделки была равна нулю и была не равна нулю.

Введем следующие обозначения: *diffpos* – знаковое количество лотов в сделке по *Is first* бумаге, *V* – это  $v\_in\_left \times Count$  или  $v\_out\_left \times Count$  в зависимости от того открываем мы позицию или закрываем данной заявкой, *Count* – это *Count Is first* бумаги, *curpos* – текущая позиция по *Is first* бумаге портфеля (т.е. прошедшая только что сделка еще НЕ учтена), нижний индекс 0 – предыдущее значение параметра, 1 – новое значение параметра. В данных обозначениях алгоритм перемещения сигнальных цен примет вид:

- прошла продажа (соответственно, в количестве *diffpos*):

■  $curpos \neq 0$ :

$$k3 = (|Lim\_Sell_0 - Lim\_Buy_0| - TP - K) \times \frac{V}{curpos},$$

$$k4 = \begin{cases} k3 + K2, & \text{if } Lim\_Sell_0 - Lim\_Buy_0 \geq 0 \\ -k3 + K2, & \text{if } Lim\_Sell_0 - Lim\_Buy_0 < 0 \end{cases},$$

$$Lim\_Buy_1 = Lim\_Buy_0 + \frac{|diffpos|}{V} \times \begin{cases} k4, & \text{if } curpos > 0 \\ K1, & \text{if } curpos < 0 \end{cases},$$

$$Lim\_Sell_1 = Lim\_Sell_0 + \frac{|diffpos|}{V} \times \begin{cases} K2, & \text{if } curpos > 0 \\ K, & \text{if } curpos < 0 \end{cases},$$

■  $curpos = 0$ :

$$Lim\_Buy_1 = Lim\_Sell_0 - TP,$$

$$Lim\_Sell_1 = Lim\_Sell_0 + \frac{|diffpos|}{V} \times K,$$

- прошла покупка (соответственно, в количестве  $diffpos$ ):

■  $curpos \neq 0$ :

$$k3 = (|Lim\_Sell_0 - Lim\_Buy_0| - TP - K) \times \frac{V}{curpos},$$

$$k4 = \begin{cases} -k3 + K2, & \text{if } Lim\_Sell_0 - Lim\_Buy_0 \geq 0 \\ k3 + K2, & \text{if } Lim\_Sell_0 - Lim\_Buy_0 < 0 \end{cases},$$

$$Lim\_Sell_1 = Lim\_Sell_0 - \frac{|diffpos|}{V} \times \begin{cases} k4, & \text{if } curpos < 0 \\ K1, & \text{if } curpos > 0 \end{cases},$$

$$Lim\_Buy_1 = Lim\_Buy_0 - \frac{|diffpos|}{V} \times \begin{cases} K2, & \text{if } curpos < 0 \\ K, & \text{if } curpos > 0 \end{cases},$$

■  $curpos = 0$ :

$$Lim\_Sell_1 = Lim\_Buy_0 + TP,$$

$$Lim\_Buy_1 = Lim\_Buy_0 - \frac{|diffpos|}{V} \times K.$$

Также перемещение сигнальных цен происходит когда заявка не может быть выставлена из-за ограничений по  $v\_min$ ,  $v\_max$ ,  $To0$ . Если робот не может купить из-за ограничений по  $v\_max$ , то в соответствии с параметрами портфеля *Limits timer* и *Percent* цены  $Lim\_Sell$  и  $Lim\_Buy$  уменьшаются на величину параметра портфеля  $K$ , если же робот не может продать из-за ограничений по  $v\_min$ , то в соответствии с параметрами портфеля *Limits timer* и *Percent* цены  $Lim\_Sell$  и  $Lim\_Buy$  увеличиваются на величину параметра портфеля  $K$ .

## Неочевидные моменты, связанные с работой робота

1. Если при срабатывании *SL* или *Timer* заявка не проходит в течение 1 секунды, то она будет автоматически переставлена по цене  $bid - k\_sl$  на продажу или  $offer + k\_sl$  на покупку, в независимости от значений *SL* и *Timer* и в не зависимости от того включен ли *Timer* вообще.
2. Заявки, выставляемые при закрытии или выравнивании позиции (это либо настройка в расписании, либо "кликер" *To market*) всегда выставляются с включенным таймером и значение параметра *Timer* для таких заявок всегда равно 1.
3. Финансовый результат (в смысле просто число) считается по сделкам и никаких "экзотических" случаев, связанных с его подсчетом нет. НО есть случаи когда не будет раздвижки

в таблице финансовых результатов. Нужно запомнить главное правило, чтобы была раздвижка, должна быть сделка по главной бумаге, если ее нет, то и раздвижки нет. То есть, если у вас по какой-то причине "кривая" позиция и вы ее "подравняли", нажав на кнопку *To market*, то вы не получите нормальной раздвижки в данной таблице. У вас будет только одна "кривая" раздвижка, в которой фигурирует только главная бумага (как пример, флуд контроль, проходят сделки по первой ноге, а по второй не дают выставиться, у вас будут одноногие "кривые" раздвижки с первой ногой, а после нажатия *To market*, когда пройдут сделки, т.к. они были только по второй ноге, то раздвижки в таблице не будет, но с финансовым результатом все будет в порядке).

И еще один вариант, это когда *Count* первой ноги больше *Count*-а второй. Т.е. пусть вы торгуете долларом валюты против доллара на срочном рынке, но валюта первая нога. Т.е. у вас стоит *Count* у валюты 100 (ну или 1000, сколько там должно быть?, думаю, вы меня поняли), а у срочки 1. Т.е. вы перекрываете на срочке каждые 100 контрактов валютки. Вот вы выставили 100 контрактов на валютке, у вас взяли 60, раздвижки в таблице не будет, т.к. она получится заведомо "кривой", вторую ногу же не кидали, потом прошло еще 50, и вы снова не увидите раздвижки. Да, вы кините одну бумагу на срочку, она пройдет (в финансовом результате все нормально учтется). Но опять же не совсем ясно к каким сделкам ее привязывать, если привязать как обычно к последней (т.е. к 50), то будет заведомо "кривая" раздвижка, а искать какие-то предыдущие сделки уже не вариант, т.к. все могло быть не так просто, как в описанном примере.

4. В момент выставлении заявки по *is first* инструменту запоминаются текущие цены по не *is first* бумагам. Таким образом в момент совершения сделки по *is first* инструменту все остальные бумаги в портфеле выставятся по ценам которые робот запомнил во время выставления заявки по *is first* инструменту.

## Пример работы программы с заданием основных параметров

Пусть заданы следующие параметры:

- $Lim\_Sell = 700$ ,
- $Lim\_Buy = 600$ ,
- $TP = 50$ ,
- $K = 10$ ,
- $K1 = 5$ ,
- $K2 = 3$ ,
- $v\_in\_left = 100$ ,
- $v\_in\_right = 100$ ,
- $v\_out\_left = 100$ ,
- $v\_out\_right = 100$ .

Если раздвижка дает "нагружаться" по 700 и выше (т.е.  $Sell \geq Lim\_Sell$ ), то будут проходить следующие сделки:

1. продажа 100 портфелей по 700,

2. продажа 100 портфелей по 710,
3. продажа 100 портфелей по 720,
- ... и т.д., увеличивая уровень входа на 10 (значение  $K$ ),
7. продажа 100 портфелей по 760.

$Lim\_Buy$  к моменту седьмой продажи будет подтягиваться каждый раз на 5 пунктов (значение  $K1$ ), начиная со второй продажи. После первой продажи  $Lim\_Buy$  примет значение 650 ( $Lim\_Sell - TP$ ). В итоге ПОСЛЕ седьмого входа уровни на вход/выход примут следующий вид:

$$Lim\_Sell = 760 + K = 770, \quad Lim\_Buy = 700 - 50 + 5 + 5 + 5 + 5 + 5 + 5 = 680,$$

где 760 – уровень последнего входа.

Для "разгрузки" используется коэффициент  $K2$ . При выходе из позиции по  $Lim\_B$  сделки будут проходить на следующих уровнях:

1. покупка 100 портфелей по 680,
2. покупка 100 портфелей по 677,
3. покупка 100 портфелей по 674,
- ... и т.д., уменьшая уровень выхода на 3 (значение  $K2$ ),
7. покупка 100 портфелей по  $662 = 680 - 3 - 3 - 3 - 3 - 3 - 3$ .

$Lim\_Sell$  при "разгрузке" будет плавно подтягиваться к  $Lim\_Buy$  с учетом уже закрытых позиций таким образом, что когда позиция станет равна нулю,  $Lim\_Sell$  примет значение  $Lim\_Buy + TP$ .

Уровни  $Lim\_Sell$  и  $Lim\_Buy$  будут двигаться не на строго заданные значения ( $K$ ,  $K1$ ,  $K2$ ) если объем проходит частями, например, если прошел не весь  $v\_in\_left = v\_in\_right$ , а только половина, тогда уровень  $Lim\_Sell$  и  $Lim\_Buy$  сдвинется только на половину того значения, которое задано коэффициентами  $K$ ,  $K1$ ,  $K2$ .

Для того, чтобы "разгрузиться" сразу на одном уровне необходимо задать коэффициент  $K2 = 0$  (в таком случае подвижки  $Lim\_Buy$  происходить не будет). Для того, чтобы "отгрузить", например, 233 контракта на уровне 680, 233 на 650 и 233 на 620, необходимо коэффициенту  $K2$  присвоить значение 8.57 ( $8.57 = (680 - 620)/7$ ). В действительности, робот не будет держать твердые уровни 680, 650 и 620, а будет плавно сдвигаться на 8.57, в конечном итоге, достигая нужного уровня.

## Написание формул на языке программирования C++

Важно: функционал, описываемый в данном разделе, доступен НЕ во всех версиях робота.

### Общие сведения

При написании формул на языке программирования C++ на код накладываются некоторые ограничения:

- вы пишете только тело соответствующих функций, все функции должны возвращать значение типа *double*;
- запрещено использование некоторых символов и слов: "\001", "#nl;", "#tab;";
- при создании портфеля автоматически создаются объекты класса *security*, они помещаются в словарь (словарь называется *S*) с ключом равным *SecKey* инструментов портфеля (это инструменты с биржи со своими ценами) и объект класса *portfolio* с именем соответствующим имени портфеля (он помещается в словарь *P*), который содержит в себе словарь, объектов класса *portfolio\_row* с ключами соответствующими *SecKey* инструментов портфеля (это позиции по инструментам портфеля и их направления торговли), также у портфеля есть еще некоторые поля (далее данные классы будут описаны более подробно);
- редактирование любых полей бумаг и портфелей НЕ приведет к изменению соответствующих полей в настройках робота, вы редактируете копии, НО, исходя из того как создаются и используются данные копии, при изменении полей объекта из одного портфеля в другой портфель может попасть измененный объект;
- в портфеле есть специальное поле *data*, которое является словарем и в нем можно сохранять значения между вызовами формул;
- если в роботе есть возможность подсчета раздвижки по сделкам, то для написания формулы с раздвижкой доступен словарь *D*, в котором лежат объекты класса *trade* (т.е. сделки по данному портфелю, если по какой-то бумаге портфеля прошло несколько сделок, то в словаре все-равно будет лежать только одна сделка по данной бумаге, но со средневзвешенной ценой и с суммарным количеством лотов) по ключу *SecKey* бумаги.

### Класс *security*

Атрибуты:

Название	Тип	Описание
strike	double	цена страйк, есть только для опционов
exp_date	double	дата экспирации, в формате <i>epoch</i>
bid	double	лучшая цена на покупку
offer	double	лучшая цена на продаж
bid_depth	int	объем бида в лотах
offer_depth	int	объем оффера в лотах
yield_buy	double	доходность, рассчитанная по биду
yield_sell	double	доходность, рассчитанная по офферу
theor_price	double	расчетная цена, есть только для опционов
put	int	1 – опцион Put, 0 – опцион Call, –1 – иначе

### Класс *portfolio\_row*

Атрибуты:

Название	Тип	Описание
on_buy	int	определяет покупаем мы или продаем инструмент при срабатывании сигнала на покупку по главному инструменту портфеля (возможные значения: 1 – покупка, 2 – продажа)
pos	int	позиция по инструменту в портфеле, в лотах
count	int	вес бумаги в одном портфеле, в лотах

## Класс *portfolio*

Атрибуты:

Название	Тип	Описание
lim_sell	double	сигнал на продажу
lim_buy	double	сигнал на покупку
v_min	int	минимальная разрешенная позиция по главной бумаге портфеля
v_max	int	максимальная разрешенная позиция по главной бумаге портфеля
v_in_left	int	минимальное число портфелей в заявках на открытие позиции
v_out_left	int	минимальное число портфелей в заявках на закрытие позиции
pos	int	позиция портфеля
rows	map<string, portfolio_row>	словарь бумаг портфеля
data	map<string, double>	словарь, сохраняемый между вызовами формул

Методы:

Сигнатура	Описание
portfolio_row& operator[] (string sec_key)	получить portfolio_row по имени бумаги
portfolio_row& at(string sec_key)	получить portfolio_row по имени бумаги

## Класс *trade*

Атрибуты:

Название	Тип	Описание
price	double	средневзвешенная цена сделки
quantity	int	суммарный объем заявки, в лотах
dir	int	направление сделки: 1 – покупка, 2 – продажа

## Дополнительные функции и константы

Константы:

SELL	BUY
------	-----

Функции:

delta()	gamma()	vega()
theta()	iv()	price()
c()	p()	cdelta()
pdelta()	cgamma()	pgamma()
cvega()	pvega()	ctheta()
ptheta()	civ()	piv()

## SELL

константа типа *int*, ее значение равно 2

## BUY

константа типа *int*, ее значение равно 1

**double delta(security& sec, double rr=0)**

**double delta(portfolio& p, double rr=0)**

вычисляет дельту финансового инструмента или портфеля со ставкой рефинансирования *rr* (указывается в процентах)

**double gamma(securit& sec, double rr=0)**

**double gamma(portfolio& p, double rr=0)**

вычисляет гамму финансового инструмента или портфеля со ставкой рефинансирования *rr* (указывается в процентах)

**double vega(securit& sec, double rr=0)**

**double vega(portfolio& p, double rr=0)**

вычисляет вегу финансового инструмента или портфеля со ставкой рефинансирования *rr* (указывается в процентах)

**double theta(securit& sec, double rr=0)**

**double theta(portfolio& p, double rr=0)**

вычисляет тету финансового инструмента или портфеля со ставкой рефинансирования *rr* (указывается в процентах)

**double iv(securit& sec, double rr=0)**

**double iv(portfolio& p, double rr=0)**

вычисляет ожидаемую волатильность опциона или портфеля со ставкой рефинансирования *rr* (указывается в процентах)

**double price(security& sec, double rr=0)**

вычисляет справедливую цену опциона со ставкой рефинансирования *rr* (указывается в процентах)

**double c(double futPrice, double strike, double expDate, double iv, double rr=0)**

вычисляет справедливую цену опциона *call*

Аргументы:

Название	Описание
futPrice	цена базового актива
strike	цена страйк опциона
expData	дата экспирации опциона в формате <i>epoch</i>
iv	ожидаемая волатильность
rr	ставка рефинансирования в процентах

**double p(double futPrice, double strike, double expDate, double iv, double rr=0)**

вычисляет справедливую цену опциона *put*, аргументы такие же, как для *c()*

**double cdelta(double futPrice, double strike, double expDate, double iv, double rr=0)**

вычисляет дельту опциона *call*, аргументы такие же, как для *c()*

**double pdelta(double futPrice, double strike, double expDate, double iv, double rr=0)**

вычисляет дельту опциона *put*, аргументы такие же, как для *c()*

**double cgamma(double futPrice, double strike, double expDate, double iv, double rr=0)**

вычисляет гамму опциона *call*, аргументы такие же, как для *c()*

**double pgamma(double futPrice, double strike, double expDate, double iv, double rr=0)**

вычисляет гамму опциона *put*, аргументы такие же, как для *c()*

**double cvega(double futPrice, double strike, double expDate, double iv, double rr=0)**

вычисляет вегу опциона *call*, аргументы такие же, как для *c()*

**double pvega(double futPrice, double strike, double expDate, double iv, double rr=0)**

вычисляет вегу опциона *put*, аргументы такие же, как для *c()*

**double ctheta(double futPrice, double strike, double expDate, double iv, double rr=0)**

вычисляет тету опциона *call*, аргументы такие же, как для *c()*

**double ptheta(double futPrice, double strike, double expDate, double iv, double rr=0)**

вычисляет тету опциона *put*, аргументы такие же, как для *c()*

**double civ(double futPrice, double strike, double expDate, double call, double rr=0)**

вычисляет справедливую цену опциона *call*

Аргументы:

Название	Описание
futPrice	цена базового актива
strike	цена страйк опциона
expData	дата экспирации опциона в формате <i>epoch</i>
call	цена опциона
rr	ставка рефинансирования в процентах

**double piv(double futPrice, double strike, double expDate, double put, double rr=0)**

вычисляет справедливую цену опциона *put*

Аргументы:



Название	Описание
futPrice	цена базового актива
strike	цена страйк опциона
expData	дата экспирации опциона в формате <i>epoch</i>
put	цена опциона
rr	ставка рефинансирования в процентах

## Примеры доступа к параметрам портфеля, инструмента, сделки

Пусть имеется портфель с именем "si" и в этом портфеле есть один инструмент – фьючерс на доллар "SiH6". Для того, чтобы получить, например, бид и объем биды по бумаге портфеля и сложить эти значения в переменные, надо написать следующий код:

```
double bid = S["SiH6"].bid;
int amount_bid = S["SiH6"].bid_depth;
```

Чтобы, получить, например, позицию и сигнал на покупку портфеля надо написать:

```
double lim_buy = P["si"].lim_buy;
int pos = P["si"].pos;
```

Позицию портфеля можно посчитать самому, разделив позицию главной бумаги портфеля на ее вес в портфеле, вот так:

```
int pos = P["si"]["SiH6"].pos / P["si"]["SiH6"].count;
```

Для подсчета раздвижки по сделкам портфеля можно использовать формулу:

```
return D["SiH6"].price;
```

## Примеры использования функций

Пусть имеется портфель с именем "test" и в этом портфеле есть два инструмента: фьючерс на индекс РТС "RIH6" и его опцион *call* на страйк 70000 "RI70000BB6", позиция по обоим бумагам в портфеле равна 1, а направление торговли у обоих *On buy = Buy*.

Рассчитаем дельту одного из инструментов портфеля, например, "RIH6". Для этого воспользуемся функцией *delta* из модуля *options*. Дельта для фьючерса всегда равна 1, проверим, для этого надо написать:

```
return delta(S["RIH6"]);
```

и выполнить этот код, в результате получим 1.

Если хотим рассчитать дельту с учетом ставки рефинансирования, то укажем необязательный параметр *rr*, например, равный 6.25%:

```
return delta(S["RIH6"], 6.25);
```

для фьючерса результат так и останется 1.

Если мы хотим рассчитать дельту всего портфеля, то надо написать:

```
return delta(P["test"]);
```

здесь "test" – имя портфеля (для портфелей с другими именами нужно писать их имена). Дельта всего портфеля будет равна сумме дельт его инструментов с учетом позиции и направления торговли:

$$\Delta_{portfolio} = \sum_{i \in portfolio} \Delta_i \times pos_i \times \begin{cases} 1, & \text{if } On\ buy_i = Buy \\ -1, & \text{if } On\ buy_i = Sell \end{cases}$$

где  $i$  –  $i$ -ый инструмент портфеля,  $\Delta_i$  – дельта  $i$ -го инструмента портфеля,  $pos_i$  – позиция  $i$ -го инструмента портфеля,  $On\ buy_i$  –  $On\ buy$   $i$ -го инструмента портфеля. Величины гамма, вега, тета и ожидаемая волатильность для портфеля рассчитываются аналогично. Функции для их расчета вызываются аналогично примеру с дельтой. Функция *price* так же вызывается аналогично *delta*, но НЕ может быть вызвана для портфеля, может быть вызвана только для финансового инструмента.

Теперь рассмотрим примеры использования функций, которые позволяют более "гибко" рассчитывать необходимые величины, так как им на вход можно передать более широкий набор аргументов, но эти функции работают исключительно для опционов и при использовании этих функций необходимо четко знать для опциона *call* или для опциона *put* вы хотите рассчитать значение. Например, рассчитаем ожидаемую волатильность опциона "RI70000BB6", в качестве цен опциона и базового актива возьмем их лучшие цены на покупку:

---

```
return civ(S["RIH6"].bid, S["RI70000BB6"].strike, S["RI70000BB6"].expDate, S["RI70000BB6"].bid);
```

---

аналогично можно взять их лучшие цены на продажу или любые комбинации или просто константы:

---

```
return civ(S["RIH6"].offer, S["RI70000BB6"].strike, S["RI70000BB6"].expDate, S["RI70000BB6"].offer);
```

---

или

---

```
return civ((S["RIH6"].offer + S["RIH6"].bid) * 0.5, S["RI70000BB6"].strike, S["RI70000BB6"].expDate, S["RI70000BB6"].offer);
```

---

или

---

```
return civ((S["RIH6"].offer + S["RIH6"].bid) * 0.5, 70000, S["RI70000BB6"].expDate, S["RI70000BB6"].bid);
```

---

## Примеры написания *Ratio buy/sell formula*

При написании формул можно использовать все те инструменты ([security](#)), которые используются в любом из портфелей, также можно использовать доступные значения других портфелей ([portfolio](#)), например, их позиции по бумагам.

Для того, чтобы использовать поле *Ratio buy/sell formula* необходимо для выбранного инструмента портфеля выбрать *Ratio type = Ratio formula*. После этого двойным кликом войти в редактор и написать необходимое значение.

Пусть имеется портфель с именем "si" и в этом портфеле есть один инструмент – фьючерс на доллар "SiH6", направление торговли этого инструмента *On buy = Buy*.

Если *Ratio sign = "x"*, то ничего особо-то интересного с формулами не придумаешь, разве что какой-то хитрый множитель (и для покупки и для продажи), например, такой:

---

```
return sqrt(S["SiH6"].bid);
```

---

в таком случае для расчета *Buy* и *Sell* будет использован один и тот же множитель, если же вы хотите использовать разные множители надо вписать разные значения в *Ratio buy formula* и *Ratio sell formula*, например, так:

---

```
return sqrt(S["SiH6"].bid);
```

---

и

---

```
return sqrt(S["SiH6"].offer);
```

---

для покупки и продажи, соответственно.

в таком случае для расчета *Buy* будет использован квадратный корень из бида, а для расчета *Sell* будет использован квадратный корень из оффера.

Если же *Ratio sign* = "+", то вы можете полностью изменить формулу расчета *Buy* и *Sell*, для этого надо для начала вычесть те значения, которые используются в данный момент, тем самым обнулив *Buy* и *Sell*:

---

```
return -S["SiH6"].offer;
```

---

и

---

```
return -S["SiH6"].bid;
```

---

для покупки и продажи, соответственно, а после этого прибавить к *Buy* и *Sell* новое значение, например, так:

---

```
double price = S["SiH6"].offer * 3 + 5;
return -S["SiH6"].offer + price;
```

---

и

---

```
double price = S["SiH6"].bid * 3 + 5;
return -S["SiH6"].bid + price;
```

---

теперь значения переменной *price* при расчете каждого из параметров будут новыми значениями для *Buy* и *Sell*. Хочется отметить, что без использования *Ratio formula* такое "хитрое" значение получить бы не удалось.

Рассмотрим еще один пример. Пусть имеется портфель с именем "*test*" и в этом портфеле есть два инструмента: фьючерс на доллар "*SiH6*", направление торговли этого инструмента *On buy* = *Buy* и он является *Is first* и фьючерс на индекс РТС "*RIH6*", направление торговли этого инструмента тоже *On buy* = *Buy* (для примера направление не *Is first* инструмента значения не имеет). Для того чтобы использовать эти два инструмента в одном портфеле нужно привести их цены в пунктах к одной размерности, как известно, доллар торгуется в рублях ( $1\ pt = 1\ rub$ ), а вот индекс торгуется не в рублях, для него  $1\ pt = 0.02 \times \$_{price}\ rub$  (где  $\$_{price}$  – это курс доллара в рублях, но это не константа а динамически изменяющаяся величина). Есть два варианта решения поставленной задачи, оба реализуемы только с использованием *Ratio formula* и оба приводят к абсолютно одинаковому результату. Вот они:

1. для доллара просто зададим *Ratio* = 1, а вот для индекса РТС выберем *Ratio sign* = "×", а в *Ratio buy formula* напомним следующее:

---

```
return (0.02 * S["SiH6"].offer * 0.001);
```

---

в *Ratio sell formula* напомним следующее:

---

```
return (0.02 * S["SiH6"].bid * 0.001);
```

---

таким образом при расчете *Buy* мы будем использовать бид доллара, а при расчете *Sell* – его оффер, и величину получим в рублях

2. для доллара просто зададим  $Ratio = 1$ , а вот для индекса РТС выберем  $Ratio\ sign = "+"$ , а в *Ratio buy formula* напишем следующее (вначале обнулим значение, как в предыдущем примере, а затем зададим новое):

---

```
double price = S["RIH6"].offer * 0.02 * S["SiH6"].offer * 0.001;  
return -S["RIH6"].offer + price;
```

---

в *Ratio sell formula* напишем следующее:

---

```
double price = S["RIH6"].bid * 0.02 * S["SiH6"].bid * 0.001;  
return -S["RIH6"].bid + price;
```

---

теперь значения переменной *price* и будут новыми значениями (так сказать, со стороны индекса РТС), используемыми для расчета *Buy* и *Sell*, соответственно.