

Инструкция по установке и эксплуатации ПО «ОРТQ / ОПТИКБЮ»

Программа «ОРТQ / ОПТИКБЮ» (далее Программа) представляет собой дистрибутив (файл tar.gz), внутри которого находится код, написанный на языке C++. Данная программа должна быть встроена в ПО клиента для анализа биржевых данных или торговли на Московской Бирже.

1. БАЗОВОЕ СИСТЕМНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

Системные программные средства, для которых обеспечивается эффективная работа Программы:

- операционная система для компиляции: Ubuntu 20.04
- операционная система для запуска: Ubuntu 20.04

На сервере для компиляции необходимо следующее программное обеспечение:

- Компилятор с поддержкой c++23, например clang-16
- cmake v3.19.1 или новее

2. ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ

Минимальные технические характеристики сервера для компиляции:

- процессор 2GHz;
- память 2GB
- свободное дисковое пространство 2GB

Минимальные технические характеристики сервера для запуска:

- процессор 2GHz;
- память 4GB;
- технические требования для ПО клиента

Рекомендуемые технические характеристики сервера для компиляции:

- 8-ядерный процессор 3GHz;
- память 32GB
- свободное дисковое пространство 2GB

Минимальные технические характеристики сервера для запуска:

- процессор 5GHz;
- память 16GB;
- технические требования для ПО клиента

Установка Программы на сервер для компиляции

Установка должна производиться на операционную систему Ubuntu 20.04

Перед установкой убедитесь, что на сервере установлены следующие пакеты:

- **clang-16**
- **cmake**

Для установки этих пакетов нужно добавить репозитории (см. инструкции ниже), а потом установить, используя стандартный интерфейс для установки программ в Ubuntu 20.04:

sudo apt update && sudo apt install cmake clang-16

Инструкции для добавления репозитория:

- <https://apt.llvm.org/>
- <https://apt.kitware.com/>

Для использования, распакуйте полученный дистрибутив (файл tar.gz) и разместите полученную директорию `arbitrage` внутри вашего проекта. Для подключения Программы к своему ПО, добавьте в `CMakeLists.txt` вашего проекта

```
add_subdirectory(arbitrage)
```

```
target_link_libraries(ИМЯ_ВАШЕГО_ПРИЛОЖЕНИЯ arbitrage)
```

а в `c++` файлах добавьте

```
#include "arbitrage/arbitrage.h"
```

и используйте класс

```
MicexOptqQuotePredictor
```

Данный класс имеет следующие интерфейсы:

```
MicexOptqQuotePredictor(std::string config_filename); // инициализирует настройки оценки  
профитности котирования, включая инструмент для которого осуществляется расчёт и  
параметры из json-файла конфига.
```

Примеры рекомендуемых конфигов содержатся в дистрибутиве.

```
void process(const PriceLimitSnapshot& snapshot);
```

```
void process(const TradingStatusSnapshot& snapshot);
```

```
void process(const BookSnapshot& snapshot);
```

```
void process(const TradesSnapshot& snapshot);
```

```
std::vector<QuoteEstimation> get_estimations(); // возвращает набор предсказаний выгоды  
для котировок на текущий момент.
```

Подробная информация по используемым классам предоставляется клиенту в рамках работ по первоначальной настройке Программы.

После этого ПО клиента будет доступно к сборке с подключённой Программой.

Эксплуатация скомпилированного экземпляра Программы

Использование программы заключается в вызове внутри клиентского ПО различных интерфейсов класса *MicexOptqQuotePredictor*.

Для использования ПО необходимо подготовить данные с биржевой информацией. Данная информация должна включать в себя следующее:

- Добавление, удаление и модификация заявок на покупку или продажу биржевых инструментов. На основании данной информации на стороне клиентского ПО должна формироваться книга заявок, по соответствующему инструменту. Данная книга заявок передаётся в Программу через интерфейс *void process(const BookSnapshot& snapshot);*
- Сделки, совершаемые на бирже по заданным биржевым инструментам. Данные сделки передаются в Программу через интерфейс *void process(const TradesSnapshot& snapshot);*
- Информация о статусе торгов заданными биржевыми инструментами согласно биржевой документации, опубликованной по адресу <http://ftp.moex.com/>. Статусы торгов передаются в Программу через интерфейс *void process(const TradingStatusSnapshot& snapshot);*
- Лимиты допустимых изменений цен биржевых инструментов. Лимиты передаются в Программу через интерфейс *void process(const PriceLimitSnapshot& snapshot);*

Обращаем ваше внимание на то, что биржевая информация не является публично

доступной по умолчанию! При необходимости клиенты должны самостоятельно заключать договоры с Московской Биржей на получение и использование необходимой биржевой информации. Подробнее с политикой распространения биржевой информации можно ознакомиться на сайте ПАО «Московская Биржа» по адресу <https://www.moex.com/ru/datapolicy/>

Ещё на этапе компиляции Программа должна быть проинициализирована с использованием параметров из json-файла. Данные параметры представляют собой настройки оценки профитности котирования, такие как инструмент, на котором предсказываются цены, прочие инструменты, биржевая информация о которых используется для расчёта оценок, и различные параметры учёта биржевой информации. Данная инициализация осуществляется путём вызова интерфейса *MicexOptqQuotePredictor(std::string config_filename);*

Внутри клиентского ПО проинициализированный экземпляр Программы должен получать биржевую информацию по заданным биржевым инструментам.

Для получения текущего набора предсказаний выгодности котировок в зависимости от заданных настроек прогнозирования и переданных биржевых данных требуется осуществить вызов интерфейса *std::vector<QuoteEstimation> get_estimations();*

Полученные оценки профитности котировок могут быть в дальнейшем использованы в ПО клиента для анализа биржевых данных или торговли на Московской Бирже.