Московский Государственный Университет
имени М.В.Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра алгоритмических языков

Система управления инвестиционным портфелем

Лебедев Андрей Алексеевич
Группа 424

Москва, 2025

Содержание

1 Постановка задачи

Уточнение постановки задачи

- Разработать программную систему, осуществляющую имитационное моделирование процесса или явления и визуализирующую этот процесс или явление.

- Использовать для создания системы один из объектно-ориентированных языков программирования (в данном случае Python).

- Провести с помощью разработанной системы исследование поведения моделируемого процесса, задавая для этого различные значения параметров, от которых зависит этот процесс.

Основные требования к системе

- Система должна быть спроектирована на основе методологии объектно-ориентированного программирования, т.е. должна быть представлена в виде совокупности взаимодействующих друг с другом объектов, причём каждый объект является экземпляром определённого класса, а классы образуют иерархию. В ходе объектно-ориентированного проектирования необходимо определить и зафиксировать логическую структуру (классы и объекты) и файловую (модульную) структуру системы.

- Система должна предоставлять удобный и понятный пользовательский интерфейс, предусматривающий проведение экспериментов по моделированию и отображение необходимой информации.

- Для проведения экспериментов по моделированию перед началом каждого эксперимента пользователь должен иметь возможность устанавливать нужные значения параметров, от которых зависит этот процесс или явление. Такие параметры называются параметрами моделирования, в их числе – шаг моделирования, т.е. отрезок времени, измеряемый в тех или иных единицах времени (секундах, минутах, днях и пр.), и/или число шагов моделирования.

- Поскольку в большинстве вариантов задания моделируемый процесс или явление зависит от нескольких неопределённых факторов, следует моделировать такие факторы статистически — на основе одного

из законов вероятностного распределения (равномерного, нормального и др.).

Вариант: Система управления инвестиционным портфелем

- Разрабатываемая система реализует экономическую игру, участник которой — менеджер, управляющий работой инвестиционного фонда. Фонд осуществляет различные вложения собранных денежных средств с целью получения прибыли. Возможны вложения в:
  - срочные депозиты банков (валютные и рублёвые);
  - драгоценные металлы (золотые слитки и др.);
  - государственные облигации;
  - акции предприятий.

  Все эти виды вложений различаются доходностью и риском (обычно доход пропорционален риску).

- В начале игры устанавливается общий капитал фонда (например, 560 тыс. у.е.), и определяется его портфель — какая часть капитала куда будет вложена. В портфеле не обязательно присутствуют все виды вложений; допускается несколько вложений одного типа.

- Также в начале игры задаётся внешняя конъюнктура: известны доступные виды вложений и их условия (процент по депозиту, цена акций и т.п.).

- Игра моделирует работу фонда в течение M месяцев ($12 \leq M \leq 30$). Шаг моделирования — один месяц. В конце каждого месяца выполняются следующие действия:
  1. Расчёт доходности по всем элементам портфеля, определение прибыли и процента доходности.
  2. Выплата налога на прибыль фонда (например, 17% от суммы прибыли).
  3. Учёт новых поступивших денежных средств (например, от продажи паёв).
  4. Учёт расходов (например, возврат паёв держателями).

5. Реструктуризация портфеля с учётом изменённого капитала и новой внешней конъюнктуры.

- Операции (1) и (2) выполняются автоматически, операция (5) — игроком, а (3) и (4) могут выполняться как автоматически, так и вручную. Прибыль фонда влияет на количество вложений в него: при высокой доходности — приток капитала, при отрицательной — отток.

- Доходность по вкладам и облигациям известна заранее, доходность по акциям и металлам зависит от внешней конъюнктуры, которая моделируется случайным образом по законам распределения.

- Цель моделирования — выявление стратегий и структуры портфеля, позволяющих устойчиво увеличивать капитал фонда.

- Параметры, задаваемые пользователем: число месяцев M, исходный капитал, начальная структура портфеля, налоговая ставка, параметры волатильности рыночной конъюнктуры.

- На каждом шаге игроку доступны текущие данные по фонду: суммарный капитал, доходность по каждому активу, обновлённые условия внешней среды. По окончании игры отображается статистика за весь период (например, динамика капитала, структура доходов).

2 Диаграмма классов

В данном разделе представлены три диаграммы, отражающие архитектуру разработанной системы. Каждая из них отображает логически обособленную часть модели, что позволяет сделать представление более наглядным.

2.1 Архитектура игровой модели

На данной диаграмме показано взаимодействие основных компонентов симуляции: игровых объектов *(Game, Player, Fund, Portfolio)* и структуры инвестиций. Отражена связь между решениями игрока *(PlayerPurchase, Purchase)* и реальными инвестициями, входящими в портфель.

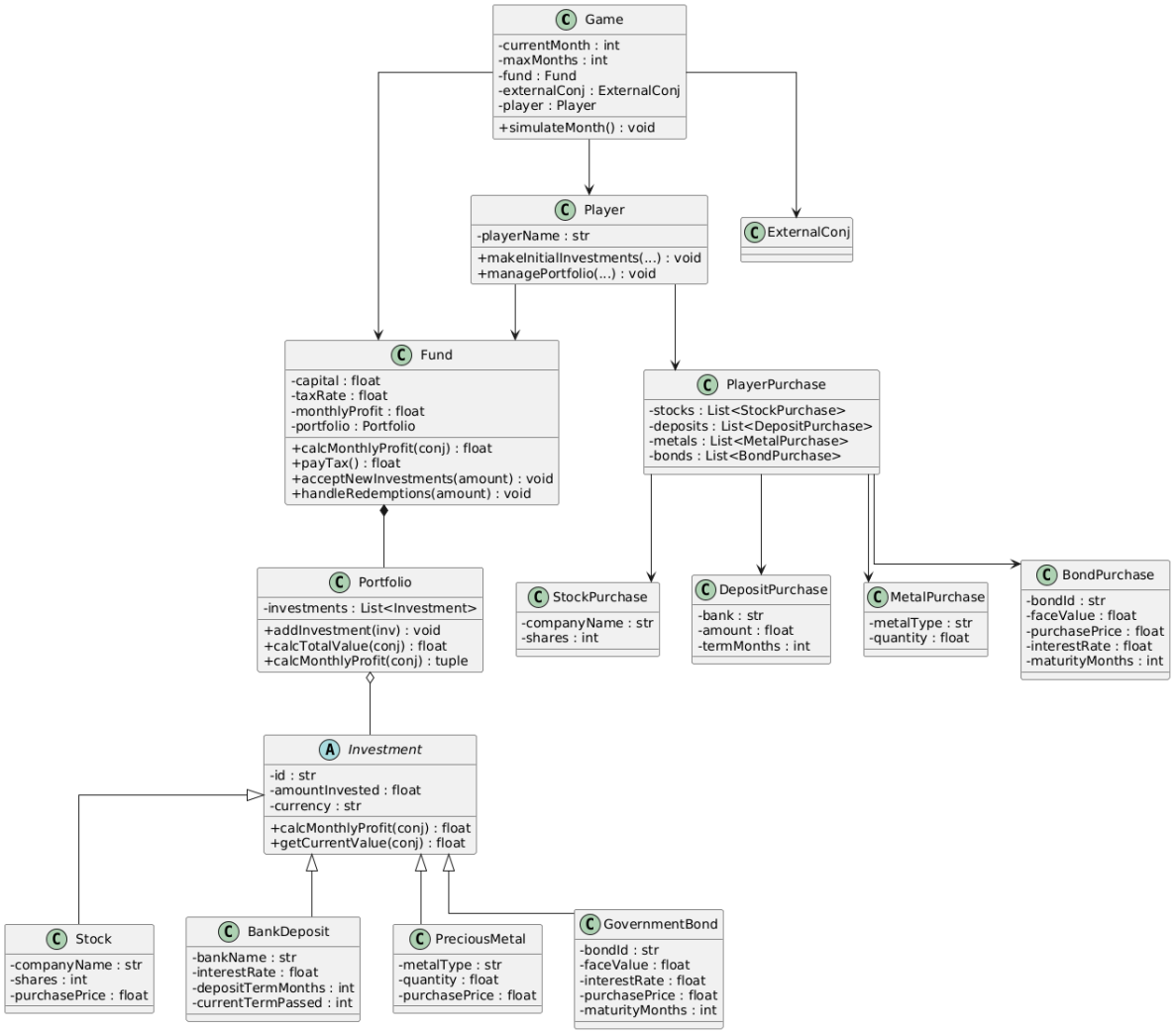


Рис. 1: Диаграмма классов: архитектура игровой модели

2.2 Рыночная среда и внешняя конъюнктура

В данной диаграмме представлено, как формируется и обновляется рыночная информация в процессе моделирования. Схема охватывает классы конфигурации *(MarketConfig, Config)*, структуру рыночных данных и механизм обновления внешней конъюнктуры *(ExternalConj, ExternalConjData)*.

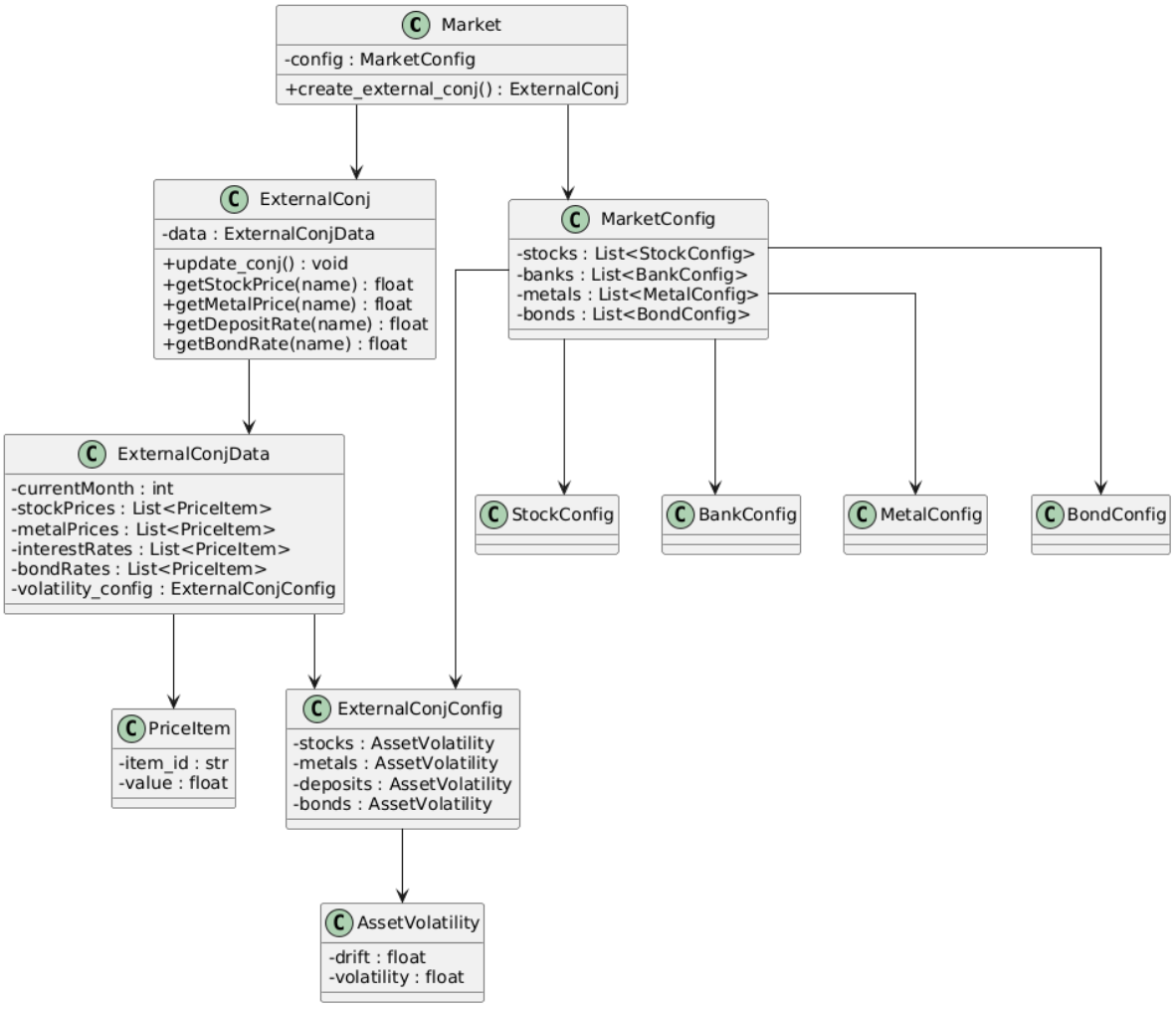


Рис. 2: Диаграмма классов: рыночная среда и внешняя конъюнктура

3 Текстовые спецификации основных классов

Ниже представлены фрагменты конструкторов классов, реализующих основную игровую логику.

Листинг 1: Класс Game

```
class Game:
    def __init__(self, max_months: int, fund: Fund, external_conj:
    ExternalConj, player: Player) -> None:
        self.current_month = 1  # Текущий месяц моделирования
        self.max_months = max_months  # Общее число месяцев игры
        self.fund = fund  # Объект инвестиционного фонда
        self.external_conj = external_conj  # Рыночные условия
        self.player = player  # Игрок-менеджер, управляющий портфелем
        self.statistics = []  # Список для накопления статистики по ходу игры
```

Листинг 2: Класс Fund

```
class Fund:
    def __init__(self, capital: float, tax_rate: float, portfolio: Portfolio)
    -> None:
        self.capital = capital  # Текущий капитал фонда
        self.tax_rate = tax_rate  # Налоговая ставка на доходы фонда
        self.portfolio = portfolio  # Портфель, содержащий инвестиции
        self.monthly_profit = 0.0  # Прибыль за текущий месяц
```

Листинг 3: Класс Player

```
class Player:
    def __init__(self, player_name: str) -> None:
        self.player_name = player_name  # Имя игрока, управляющего фондом
```

Листинг 4: Класс Portfolio

```
class Portfolio:
    def __init__(self) -> None:
        self.investments = []  # Список объектов Investment (акции, облигации,
    металлы, депозиты)
```

Листинг 5: Класс Market

```
class Market:
    def __init__(self, config_path: str):
        self.config = self.load_market_config(config_path)  # Загрузка рыночно
    й конфигурации из файла
        self.stocks = self.config.stocks  # Список доступных акций
        self.banks = self.config.banks  # Список доступных банк. депозитов
        self.metals = self.config.metals  # Список доступных драг. металлов
        self.bonds = self.config.bonds  # Список доступных облигаций
```

4 Диаграмма объектов

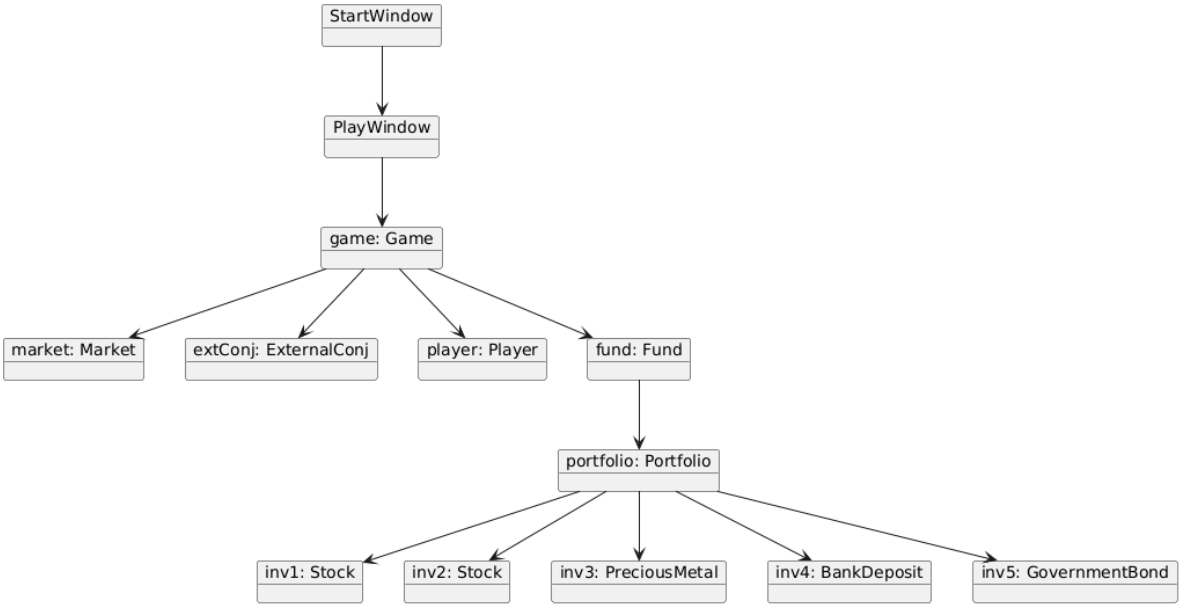


Рис. 3: Пример взаимодействия объектов в процессе моделирования

5 Инструментальные средства

- Язык программирования: **Python 3.12**
- Среда разработки (IDE): **Visual Studio Code** (версия 1.88.0)
- Система контроля версий: **Git**
- Графическая библиотека: **PyQt5** версии **5.15.11**
- Визуализация данных: **matplotlib** версии **3.10.1**
- Работа с конфигурациями: **PyYAML** версии **6.0.2**
- Валидация и модели данных: **pydantic** версии **2.10.6**
- Используемые стандартные модули Python: `random`, `io`, `sys`, `typing`

6 Файловая структура

- **main.py** — основная точка входа в приложение
- **configs**/ — конфигурации, задающие начальные параметры рынка:
  - `external_conj_config.yaml` — файл с настройками волатильности активов.
  - `market_config.yaml` — файл с исходной информацией о доступных активах: акциях, депозитах, металлах и облигациях.
- **core**/ — основной модуль с бизнес-логикой игры:
  - `external_conjuncture.py` — реализация внешней конъюнктуры (изменение цен и ставок во времени).
  - `fund.py` — класс инвестиционного фонда, обрабатывающий прибыль, налоги, инвестиции и выкуп паёв.
  - `game.py` — основной цикл моделирования. Запускает месячные итерации, рассчитывает доходность.
  - `models.py` — pydantic-модели для структуры данных: решения игрока, активы и др.
  - `player.py` — логика игрока: принятие инвестиционных решений, начальные вложения и реструктуризация.
  - `portfolio.py` — реализация портфеля фонда: набор всех инвестиций.
- **gui**/ — графический интерфейс программы:
  - `start_window.py` — стартовое окно с выбором параметров игры и первой покупкой.
  - `simulation_window.py` — главное окно симуляции: переход по месяцам, графики, финальная статистика.
  - `portfolio_graph_widget.py` — график стоимости портфеля по месяцам.
  - `assets_graph_widgets.py` — отдельные графики по категориям активов: акции, металлы, облигации, депозиты.
- **investments**/ — конкретные реализации инвестиционных инструментов:
  - `investment.py` — базовый абстрактный класс Investment.

  - `stock.py` — модель акции с параметрами.
  - `bank_deposit.py` — модель банковского депозита.
  - `government_bond.py` — модель облигации.
  - `precious_metal.py` — модель металла.

- **trading_market**/ — система рынка, которая определяет доступные активы в каждый момент:
  - `market.py` — базовая логика изменения рынка: генерация цен, ставок, доступных активов.
  - `market_config.py` — загрузка рыночных данных из YAML-файла.
  - `available_assets.py` — сборка всех доступных на данный момент активов для передачи игроку.

7 Пользовательский интерфейс

Разработанная система обладает графическим пользовательским интерфейсом, реализованным с использованием библиотеки PyQt5. Интерфейс разделён на три основных окна: окно настройки игры, окно с меню покупки и вкладкой визуализации портфеля.

7.1 Окно настройки игры

На стартовом экране пользователь задаёт ключевые параметры игры: стартовый капитал, налоговую ставку, длительность моделирования.

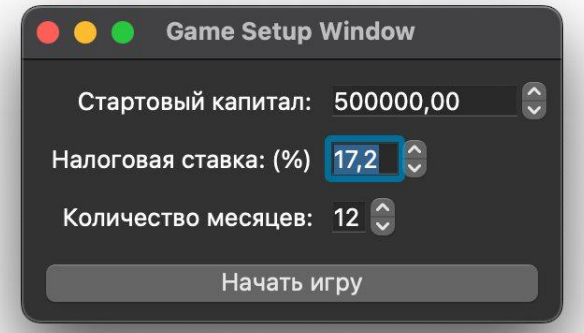


Рис. 4: Интерфейс: окно настройки игры

7.2 Меню управления портфелем

После запуска игры открывается главное окно управления инвестициями. Пользователь может перейти на вкладку *Меню покупки*, где доступно изменение структуры портфеля: покупка или продажа активов по текущим рыночным условиям.

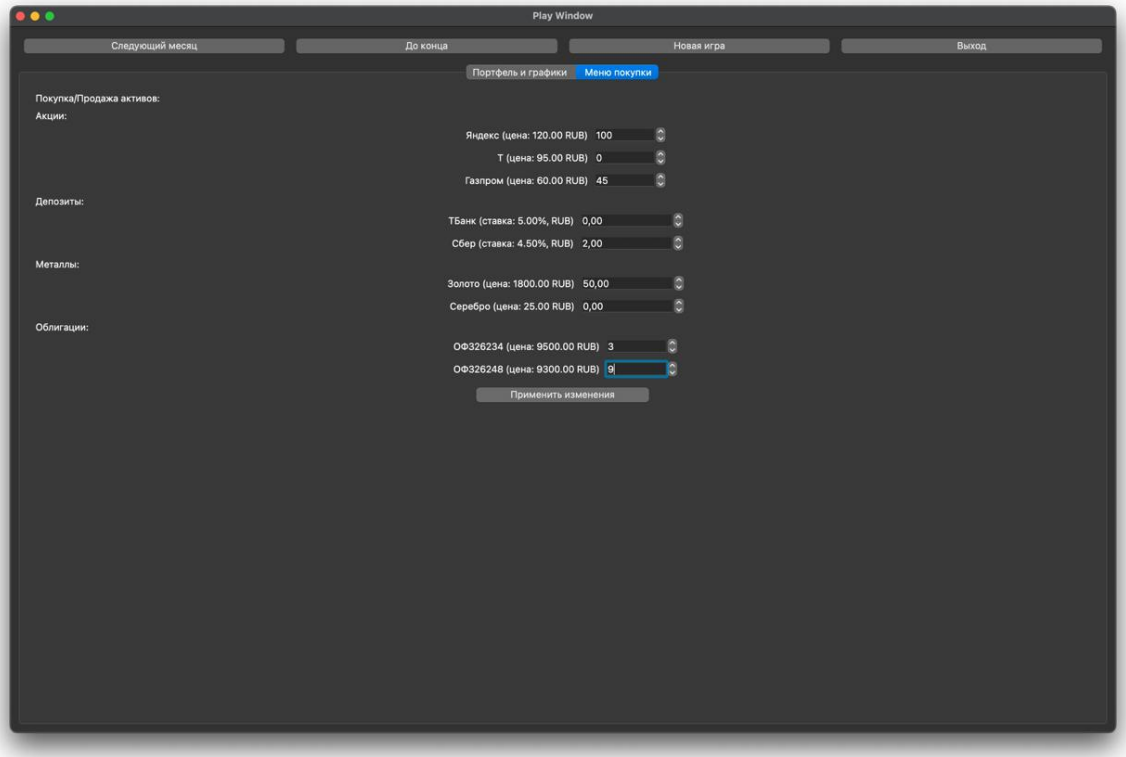


Рис. 5: Интерфейс: меню для реализации инвестиций

7.3 Вкладка «Портфель и графики»

На отдельной вкладке отображается текущее состояние инвестиционного портфеля и итоги за прошедший месяц, а также графики изменения стоимости портфеля и рыночных условий, которые можно переключать. Это позволяет пользователю отслеживать динамику доходности и корректировать стратегию.

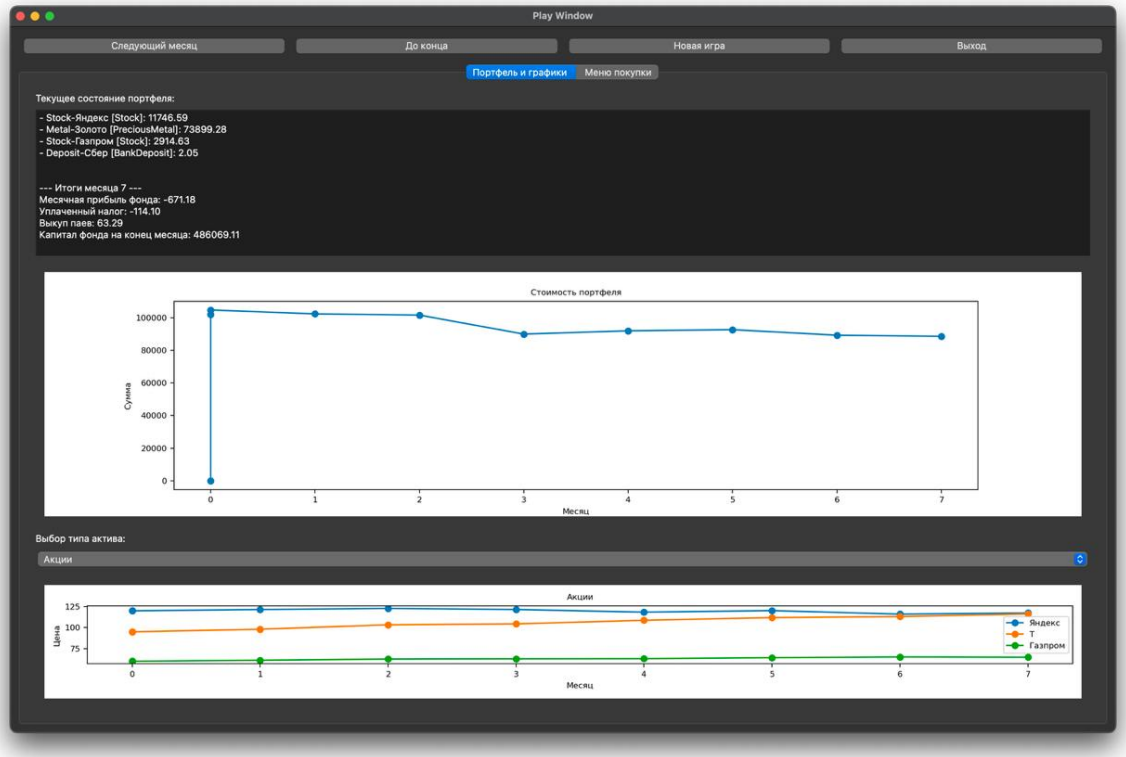


Рис. 6: Интерфейс: графическая визуализация портфеля и рыночных данных

7.4 Панель управления симуляцией

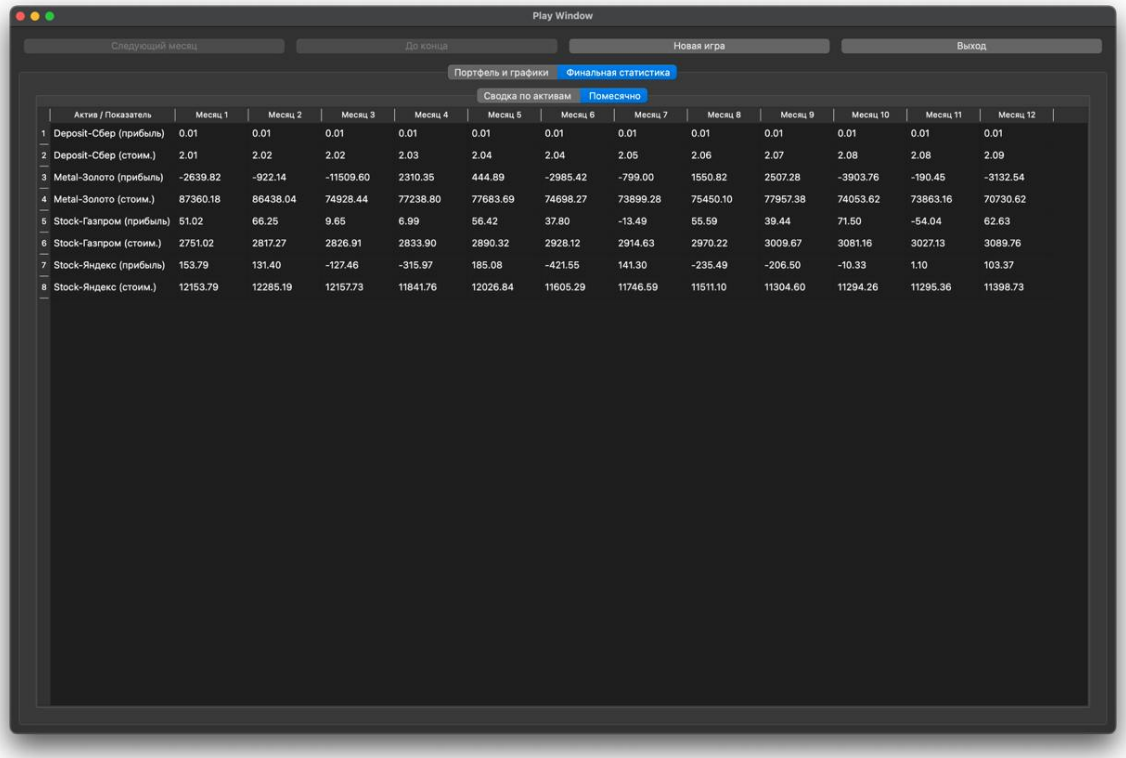
В верхней части главного окна располагается панель с кнопками управления ходом моделирования:

- **Следующий месяц** — выполняет симуляцию следующего временного шага и обновляет состояние портфеля и графиков.
- **До конца** — запускает симуляцию всех оставшихся месяцев.
- **Новая игра** — возвращает пользователя в окно начальной настройки и позволяет запустить симуляцию с новыми параметрами.
- **Выход** — завершает работу приложения.

Эти кнопки доступны независимо от выбранной вкладки и обеспечивают удобное управление процессом моделирования.

7.5 Финальная статистика

По завершении симуляции пользователь может перейти на вкладку *Финальная статистика*, где доступна подробная таблица с результатами моделирования. Статистика может быть представлена в двух режимах: *сводка по активам* и *помесячно*. Таблица содержит данные по каждому активу: его доходность, стоимость и изменения за каждый месяц.



	Актив / Показатель	Месяц 1	Месяц 2	Месяц 3	Месяц 4	Месяц 5	Месяц 6	Месяц 7	Месяц 8	Месяц 9	Месяц 10	Месяц 11	Месяц 12
1	Deposit-Сбер (прибыль)	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
2	Deposit-Сбер (стоим.)	2.01	2.02	2.02	2.03	2.04	2.04	2.05	2.06	2.07	2.08	2.08	2.09
3	Metal-Золото (прибыль)	-2639.82	-922.14	-11509.60	2310.35	444.89	-2985.42	-799.00	1550.82	2507.28	-3903.76	-190.45	-3132.54
4	Metal-Золото (стоим.)	87360.18	86438.04	74928.44	77238.80	77683.69	74698.27	73899.28	75450.10	77957.38	74053.62	73863.16	70730.62
5	Stock-Газпром (прибыль)	51.02	66.25	9.65	6.99	56.42	37.80	-13.49	55.59	39.44	71.50	-54.04	62.63
6	Stock-Газпром (стоим.)	2751.02	2817.27	2826.91	2833.90	2890.32	2928.12	2914.63	2970.22	3009.67	3081.16	3027.13	3089.76
7	Stock-Яндекс (прибыль)	153.79	131.40	-127.46	-315.97	185.08	-421.55	141.30	-235.49	-206.50	-10.33	1.10	103.37
8	Stock-Яндекс (стоим.)	12153.79	12285.19	12157.73	11841.76	12026.84	11605.29	11746.59	11511.10	11304.60	11294.26	11295.36	11398.73

Рис. 7: Интерфейс: финальная статистика по результатам моделирования