МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
Учреждение образования
«Витебский государственный технологический университет»

КОМПЬЮТЕРНАЯ ПРАКТИКА

Методические указания по прохождению практики
для студентов специальности 1-55 01 03 «Компьютерная мехатроника»

Витебск
2018

УДК 67/68:682.5

Составители:

А. Г. Кириллов, Т. В. Буевич

Рекомендовано к изданию редакционно-издательским советом УО «ВГТУ», протокол № 7 от 28.09.2018.

Компьютерная практика : методические указания по прохождению практики / сост. А. Г. Кириллов, Т. В. Буевич. – Витебск : УО «ВГТУ», 2018. – 25 с.

В методических указаниях рассмотрены индивидуальные задания по компьютерной практике студентов, обучающихся по специальности «Компьютерная мехатроника».

УДК 67/68:682.5

СОДЕРЖАНИЕ

Введение

Компьютерная (учебная) практика является частью общей образовательной программы высшего профессионального образования по специальности 1-55 01 03 «Компьютерная мехатроника».

Практика организуется в соответствии с образовательным стандартом высшего образования Республики Беларусь (ОСВО 1-55 01 03-2013), учебным планом по специальности и согласно графику учебного процесса.

Компьютерная практика направлена на:

– закрепление, расширение и углубление теоретических знаний, полученных при изучении ряда общепрофессиональных дисциплин учебных дисциплин в семестрах, предшествующих практике, на основе изучения возможностей сред и языков программирования;

– приобретение практических навыков самостоятельной работы, развитие профессионального мышления;

– формирование у студентов профессиональных компетенций в области программирования мехатронных систем.

В ходе прохождения технологической практики студент изучает среду программирования, технологии разработки программного обеспечения, возможности применения данных технологий при моделировании технологических процессов и разработке программного обеспечения для мехатронных систем.

По компьютерной практике предусмотрено выполнение индивидуальных заданий, связанных с написанием программ на объектно-ориентированном языке высокого уровня: Java, C++, C#, Delphi и др. Компьютерная практика проводится в компьютерной лаборатории, оснащенной соответствующим программным обеспечением. Персональные компьютеры в лаборатории объединены в локальную сеть. В лаборатории имеется доступ к виртуальной образовательной среде университета. Предполагается знание студентами одного из языков программирования на базовом уровне, а также владение офисными приложениями, графическими редакторами, приемами работы с операционной системой, системным и прикладным программным обеспечением.

В настоящих методических указаниях приведены рассматриваемые вопросы, задания, порядок их выполнения.

Варианты заданий студентам выдает преподаватель.

1 Цель и задачи компьютерной практики

Цель компьютерной практики – подготовка студентов к осознанному и углубленному практическому изучению учебных дисциплин специальности, закрепление теоретических знаний студентов, полученных на первом курсе, привитие им первичных навыков по выбранной специальности, а также приобретение необходимых практических навыков разработки программного обеспечения на языке высокого уровня.

Задачами практики являются: получение представления о современных средствах разработки программного обеспечения с использованием парадигмы объектно-ориентированного программирования; расширение понимания сущности и социальной значимости будущей специальности; закрепление и расширение знаний, полученных при изучении таких дисциплин, как «Информатика», «Web-технологии»; формирование профессиональных компетенций в области разработки программного обеспечения мехатронных систем; формирование навыков составления отчетов по результатам выполнения самостоятельной работы; расширение и закрепление навыков работы с методической, научной, учебной литературой, системой дистанционного обучения; выработка навыков квалифицированного использования средств и возможностей вычислительной техники при решении конкретных технических, научно-исследовательских и прикладных проблем.

Содержание практики регламентируется учебной программой. Компьютерная практика организуется в учебных лабораториях вычислительного центра УО «ВГТУ».

Продолжительность работы студентов должна составлять шесть часов в день. При проведении практики составляется расписание.

По результатам практики студенты оформляют отчет, который защищают на устном собеседовании. В процессе защиты студенты должны обосновать выбор метода решения поставленной задачи, раскрыть детали реализации задачи, возможности дальнейшего расширения и уточнения программного продукта.

В результате прохождения данной практики обучающийся должен приобрести следующие практические навыки, умения, универсальные и профессиональные компетенции.

1. Академические:

АК-1. Уметь применять базовые научно-теоретические знания для решения теоретических и практических задач.

АК-3. Владеть исследовательскими навыками.

АК-4. Уметь работать самостоятельно.

АК-5. Быть способным порождать новые идеи (обладать креативностью).

АК-6. Владеть междисциплинарным подходом.

АК-7. Иметь навыки, связанные с использованием технических устройств, управлением информацией и работой с компьютером.

2. Социально-личностные:

СЛК-3. Обладать способностью к межличностным коммуникациям.

СЛК-6. Уметь работать в команде.

3. Профессиональные:

ПК-2. Разрабатывать технические задания на проектируемые мехатронные системы с учетом новых достижений науки и техники.

ПК-3. Моделировать динамические и статические процессы мехатронных модулей в машинах.

ПК-4. Владеть компьютерным дизайном мехатронных модулей машин, деталей, оборудования.

ПК-5. Использовать пакеты прикладных программ при решении задач проектирования и создания мехатронных систем.

ПК-6. Применять технологии баз данных для решения конструкторских и проектных работ.

ПК-7. Применять компьютерные технологии, основанные на использовании компьютерных сетей, Internet.

2 Содержание практики

После окончания практики студент предоставляет отчет по установленной форме. Итоги практики обсуждаются на заседании кафедры.

С содержательной точки зрения практика разбивается на пять основных разделов. Студент распределяет время работы на практике в соответствии с таблицей 1.

Таблица 1 – Распределение времени по разделам практики

Наименование раздела или вида работ	Распределение времени, %
1 Анализ темы исследования, прогнозирование его объема и целесообразности его проведения	5
2 Аналитический обзор, сбор исходных данных, выбор оптимального метода решения и программных средств разработки	20
3 Математическое обеспечение исследования	15
4 Содержание и результаты выполненной работы	45
5 Заключение	5
6 Оформление отчета	10

Основные стадии проектирования программного продукта:

1) анализ требований к проекту и разработка технического задания;
2) проектирование структуры системы;
3) реализация программного продукта;
4) тестирование продукта;
5) внедрение и поддержка.

Анализ требований к проекту. На этом этапе формулируются цели и задачи проекта, выделяются базовые сущности и взаимосвязи между ними. То есть создается основа для дальнейшего проектирования системы.

В рамках данного этапа не только фиксируются требования руководителя практики, но и проводится формирование дополнительных требований – подбирается оптимальное решение проблем, определяется необходимая степень автоматизации, выявляются наиболее актуальные для автоматизации вычислительные процессы.

При анализе требований определяются сроки и стоимость разработки ПО, формируется и подписывается техническое задание (ТЗ) на разработку программного обеспечения.

Проектирование структуры системы. На основе предыдущего этапа проводится проектирование системы. Эта методология проектирования соединяет в себе объектную декомпозицию, приемы представления физической, логической, а также динамической и статической моделей системы.

Во время проектирования разрабатываются проектные решения по выбору платформы, где будет функционировать система языка или языков реализации, назначаются требования к пользовательскому интерфейсу, определяется наиболее подходящая система управления базами данных (СУБД). Разрабатывается функциональная спецификация ПО: выбирается архитектура системы, оговариваются требования к аппаратному обеспечению, определяется набор организационных мероприятий, которые необходимы для внедрения ПО, а также перечень документов, регламентирующих его использование.

Реализация программного продукта. Данный этап разработки программного обеспечения организован в соответствии с моделями эволюционного типа жизненного цикла ПО. При разработке применяются экспериментирование и анализ, строятся прототипы как целой системы, так и ее частей. Прототипы дают возможность глубже вникнуть в проблему и принять все необходимые проектные решения еще на ранних этапах проектирования. Такие решения могут затрагивать разные части системы: внутреннюю организацию, пользовательский интерфейс, разграничение доступа и т. д. В результате этапа реализации появляется рабочая версия продукта.

Тестирование продукта. Тестирование тесно связано с такими этапами разработки программного обеспечения, как проектирование и реализация. В систему встраиваются специальные механизмы, которые дают возможность производить тестирование системы на соответствие требований к ней, проверку оформления и наличие необходимого пакета документации.

Результатом тестирования является устранение всех недостатков системы и заключение о ее качестве.

Внедрение и поддержка. Внедрение системы обычно предусматривает следующие шаги: установка системы, обучение пользователей, эксплуатация.

К любой разработке прилагается полный пакет документации, который включает в себя описание системы, руководства пользователей и алгоритмы работы.

3 Теоретическая часть

Среда выполнения Java. Java SE/EE/ME. Java Runtime Environment (JRE). Компилятор Java. Байт-код. Интерпретатор байт-кода. JIT-компиляция байт-кода в машинный код. Виртуальная машина Java (JVM). Сборщик мусора (garbage collector). Структура проекта java. Файлы .java и .class. Компиляция с использованием командной строки. Java API.

Пакет разработчика приложений Java Development Kit (JDK). Стандартные библиотеки классов Java. интегрированные среды разработки приложений на *Java:* JDeveloper, NetBeans IDE, IntelliJ IDEA, Eclipse. Средства отладки приложений в IDE. Unicode. Кодировка UTF-8 (Unicode Transformation Format – формат преобразования Юникода). UTF-8 в проектах Java.

Парадигмы программирования. Директивное, декларативное, объектно-ориентированное программирование – сравнительный анализ.

Лексическая структура языка. Используемые в языке символы. Комментарии. Однострочные и многострочные комментарии. Javadoc комментарии. Идентификаторы. Правила именования переменных, интерфейсов, классов и методов. Ключевые слова. Литералы. Литерал null. Примитивные типы данных: byte, short, int, long, boolean, char, float, double. Разделители. Операции.

Типы, значения и переменные. Ссылочные типы данных. Объекты. Класс Object. Класс String. Переменные. Виды переменных: переменная класса, переменная экземпляра, компоненты массива, параметры метода, параметры конструктора, параметр обработчика исключений, локальные переменные. Объявление переменной (instantiation). Инициализация переменных (initialization). Преобразование типов. Тождественное, расширяющее и сужающее преобразование. Приведение типов.

Именование. Соглашения именования. Имена пакетов, классов, методов, полей, констант, локальных переменных.

Пакеты (package). Члены пакета: подпакеты и классы. Пакеты в файловой системе. Объявление импорта (import). Объявление одиночного импорта типа. Объявление импорта типа по шаблону.

Блоки и операторы. Управляющие операторы. Условные операторы: if, if-then, if-then-else, switch, тернарная операция «?». Операторы цикла: for, while, do… while. Вложенные циклы. Операторы break, continue.Операторы сравнения. Логические условия. Недостижимые операторы. Побитовые операторы. Оператор return. Оператор throw.

Операции. Арифметические операции. Операции сложения, умножения, деления, остатка. Операции инкрементирования и декрементирования. Операции отношения. Логические операции. Поразрядные операции.

Математические функции (класс Math). Приоритет операций. Постфиксная и префиксная операции инкремента и декремента. Логические операции РАВНО, ИЛИ, И, НЕ.

Массивы. Типы массивов. Одномерные и многомерные массивы. Jagged массивы. Объявление, создание и инициализация массивов. Доступ к элементам массива. Вставка и удаление элементов. Копирование массивов. Сортировка массивов. Методы сортировки. Пузырьковая сортировка. Сортировка вставками, слиянием, с помощью двоичного дерева, сортировка выбором, быстрая сортировка.

Строки. Класс String. Сравнение, конкатенация строк. Выделение подстроки. Поиск в строке. Чтение данных из командной строки. Создание строк с использованием классов StringBuilder и StringBuffer. Поиск в строке, парсинг строки и удаление строк с использованием регулярных выражений.

Ошибки и исключения. Ошибки времени компиляции и времени выполнения. Исключительные ситуации. Обработка исключений. Операторы try-catch и try-catch-finally. Классы Exception, Runtime Exception, Error. Типы исключений. Использование конструкций throw. Использование catch, единожды и многократно. Классы исключений. Создание выборочных исключений и автозакрываемых ресурсов. Использование assertions. Создание пользовательских исключений.

Объектно-ориентированное программирование (ООП). Абстрагирование, инкапсуляция, наследование, полиморфизм. Преимущества и недостатки ООП. Класс и экземпляр класса. Поля и методы. Поля, методы и конструкторы со спецификаторами доступа private, protected, default и public. Статические поля и методы. Ключевое слово static. Классы, поля и методы с модификатором final. Переходные поля (transient). Поля volatile. Абстрактные классы и методы (abstract).

Разработка классов. Использование оператора instanceof для определения типа объекта. Виртуальный вызов методов класса. Перегрузка методов класса Object. Использование абстрактных классов. Вложенные классы. Анонимные классы.

Наследование и классы. Инкапсуляция при разработке классов Java. Моделирование задачи с использованием классов Java. Неизменяемые классы. Подклассы: создание и использование. Перегрузка методов класса. Методы с переменным числом аргументов. Тело класса, члены класса. Наследование (inheritance). Суперкласс и суперинтерфейс. Ключевые слова extends и implements. Ключевые слова this и super. Наследование, замещение и сокрытие методов. Перегрузка методов. Конструктор. Геттеры (get-методы) и сеттеры (set-методы). Перегрузка конструктора. Конструктор по умолчанию.

Наследование и интерфейсы. Интерфейсы в Java, определение интерфейсов. Особенности использования интерфейсов и классов в программах. Расширение интерфейсов. Объявление интерфейса. Абстрактные интерфейсы. Тело интерфейса. Наследование, замещение и перегрузка методов. Сравнение интерфейса и абстрактного класса. Рефакторинг кода.

Обобщённые типы и коллекции значений. Обобщённые типы как способ создания классов в Java. Создание объектов в рамках обобщённого типа. Создание коллекций без использования обобщённых типов и с их использованием. Итераторы. Реализация стека, очереди и дерева. Перечислимые типы. Классы коллекций. Интерфейсы коллекций. Интерфейс Collection. Интерфейсы Iterator и Iterable. Интерфейсы List, Set, SortedSet, Queue. Классы Vector, Stack, ArrayList, LinkedList, HashSet, TreeSet. Интерфейсы Comparable и Comparator. Интерфейс Map. Реализация алгоритмов в коллекциях.

Ввод и вывод. Файловый ввод и вывод. Основы ввода и вывода в Java программах. Чтение данных с консоли и вывод данных на консоль. Использование потоков для чтения и записи файлов. Чтение и запись объектов с использованием сериализации. Использование интерфейса Path для работы с файлами. Работа с классом Files для операций над файлами. Канальный и потоковый ввод-вывод в файлах. Работа с атрибутами файлов. Доступ к дереву каталогов. Поиск файлов с использованием класса PathMatcher. Класс File. Классы FileReader, FileWriter, BufferedReader и BufferedWriter. Исключительные ситуации, возникающие при обращении к файлам. Интерфейсы FileFilter, FilenameFilter.

Локализация. Особенности и задачи локализации (l10n) и интернационализации (i18n) программ. Определение и представление локализуемых данных. Создание файлов .properties. Чтение и установка локализуемых данных с помощью классов Locale и ResourceBoundle. Построение ресурсов. Вызов ресурсов из приложений. Форматирование текста и его локализация с использованием NumberFormat, DateFormat.

Шаблоны проектирования. Преимущества и недостатки шаблонов. Порождающие, структурные и поведенческие шаблоны. Диаграммы классов. Язык Unified Modeling Language (UML). Singleton. Factory method. Façade. Bridge. Adapter. Abstract Factory. Observer. Builder. Mediator. Decorator. Prototype. Proxy. Composite.

Аннотации Java. Чтение и запись конфигурационных файлов. Чтение и запись XML-файлов. Сериализация и десереализация. Маршалинг и демаршалинг.

Разработка оконных приложений. GUI-фреймворки. Контейнеры и компоненты. Библиотеки AWT и SWING. Использование визуального редактора GUI в NetBeans. Пакеты для работы с GUI. Классы пакета Swing. Создание фрейма. Расположение и размеры фрейма. Свойства фрейма. Компоновка и элементы управления пользовательского интерфейса. Менеджеры компоновки (BorderLayout, FlowLayout, BoxLayout, CardLayout, GridLayout). Класс JFrame. Класс JLabel. Класс JButton. Класс JCheckbox. Класс JCheckboxGroup. Класс JList. Класс JScrollbar. Класс JTextField. Класс JTextArea. Стратегии размещения компонентов. Программирование окон. Программирование меню. Модель обработки событий от компонентов (различные способы создания слушателей). Надстройка WindowBuilder.

4 Практическая часть

Индивидуальные задания студентам разрабатывает руководитель практики. Содержание индивидуальных заданий может быть построено на основных темах следующих дисциплин: математика, теоретическая механика, начертательная геометрия, инженерная и машинная графика, информатика, Web-технологии.

Примеры индивидуальных заданий приведены ниже.

1. Написать приложение, в котором пользователь вводит дату рождения. Программа выводит знак зодиака на английском и русском языках. Использовать интернационализацию приложения.

2. Имеется строка: JaggedEdge. Нужно создать на ее основе двумерный массив, состоящий из массивов разной длины («лестница»).

3. Имеется колода игральных карт (массив). Реализовать действия:

– вывод содержимого массива;
– перестановка элементов массива;
– случайное перемешивание массива;
– сортировка;
– выбор некоторого количества элементов из массива;
– формирование всех перестановок элементов массива.

4. Реализовать интерфейсы (например, Running, Swimming, Climbimg) в классах животных (например, Duck, Rabbit, Squirrel).

5. Описать небольшой рассказ в виде объектов Java.

6. Реализовать конвертер из арабских чисел в римские.

7. Реализовать конвертер из арабских чисел в китайские (японские).

8. По введенному номеру мобильного телефона вывести название мобильного оператора.

9. Реализовать приложение, в котором используются коллекции ArrayList и LinkedList. Сравнить производительность операций поиска, добавления и удаления элементов (рис. 1).

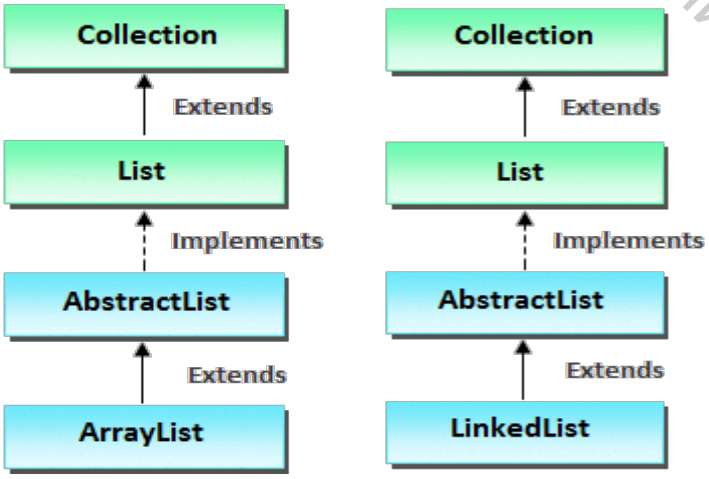

Рисунок 1 – ArrayList vs LinkedList

10. Реализовать приложение с использованием одной из коллекций: HashSet, LinkedHashSet, TreeSet (рис. 2).

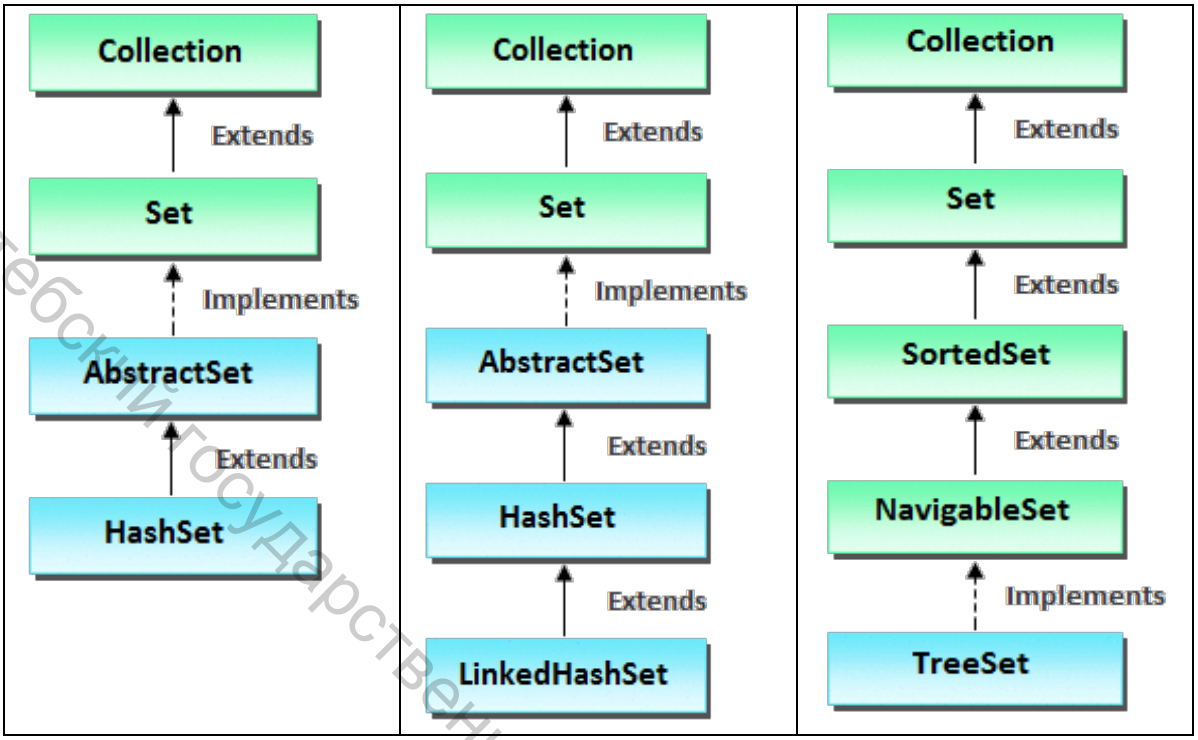

Рисунок 2 – HashSet, LinkedHashSet, TreeSet

11. Реализовать приложение с использованием одной из коллекций: HashMap, LinkedHashMap, TreeMap (рис. 3).

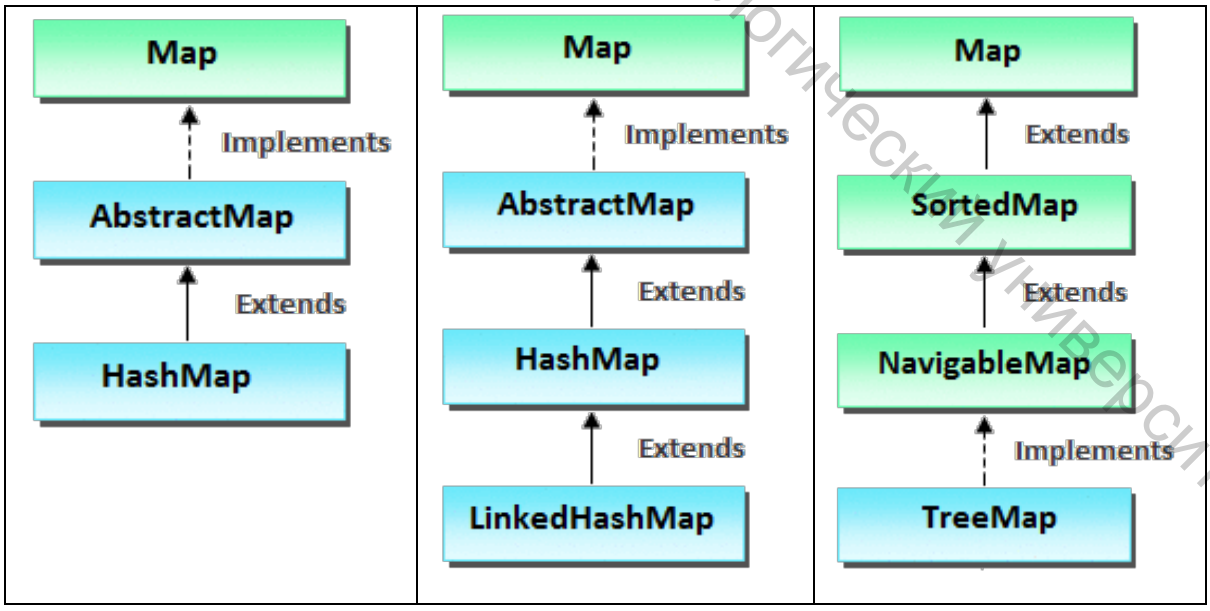

Рисунок 3 – HashMap, LinkedHashMap, TreeMap

12. Реализовать приложение с использованием шаблона Singleton (рис. 4).

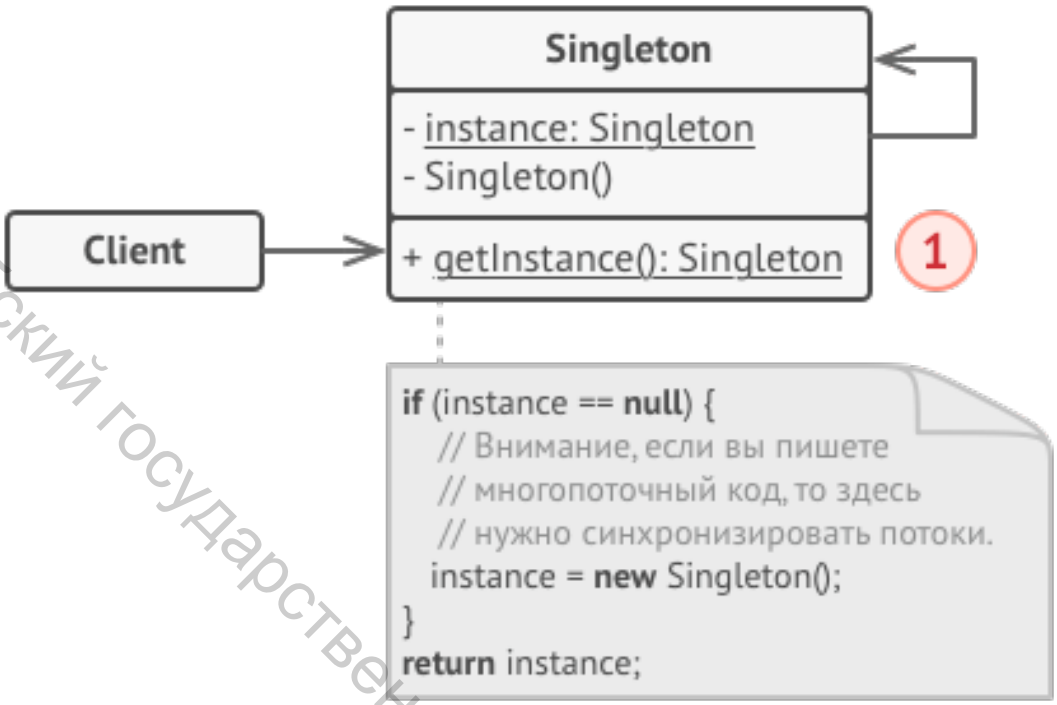

Рисунок 4 – Singleton

13. Реализовать приложение с использованием шаблона Factory method (рис. 5).

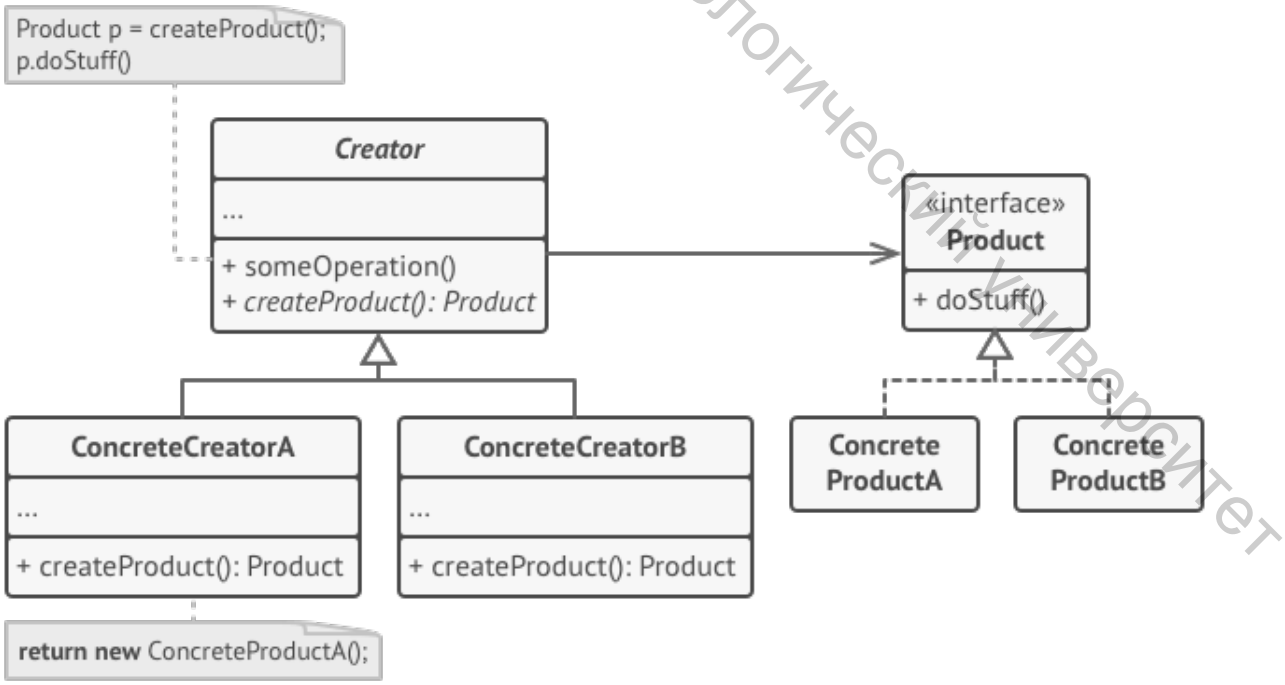

Рисунок 5 – Factory method

14. Реализовать приложение с использованием шаблона Facade (рис. 6).

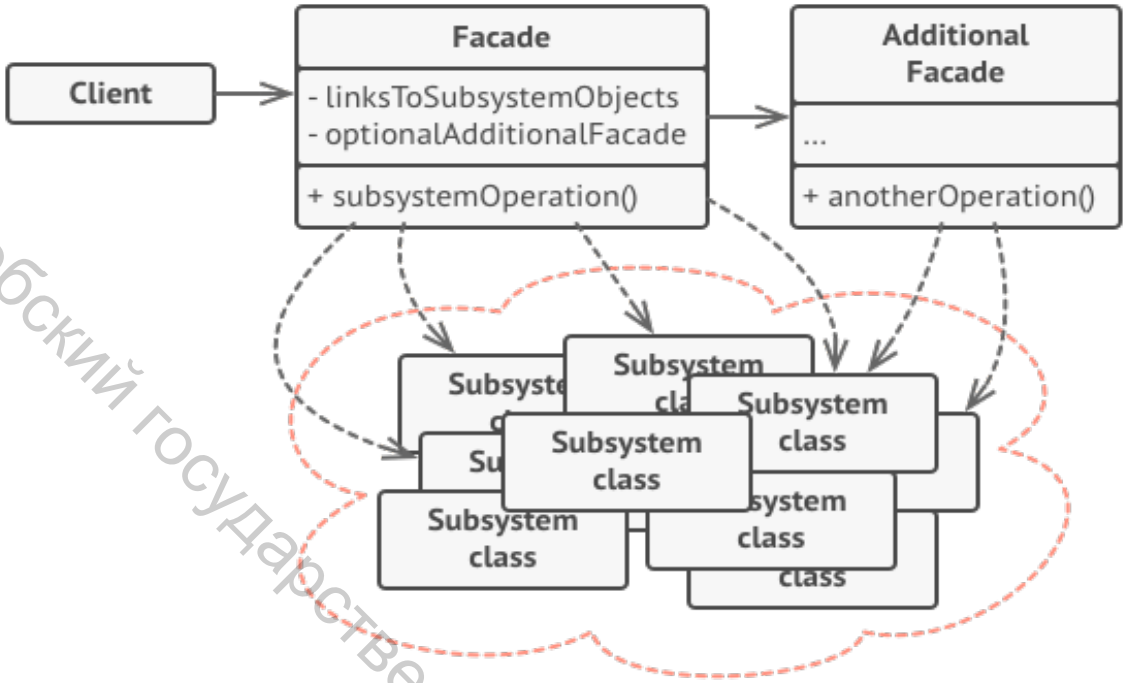

Рисунок 6 – Facade

15. Реализовать приложение с использованием шаблона Bridge (рис. 7).

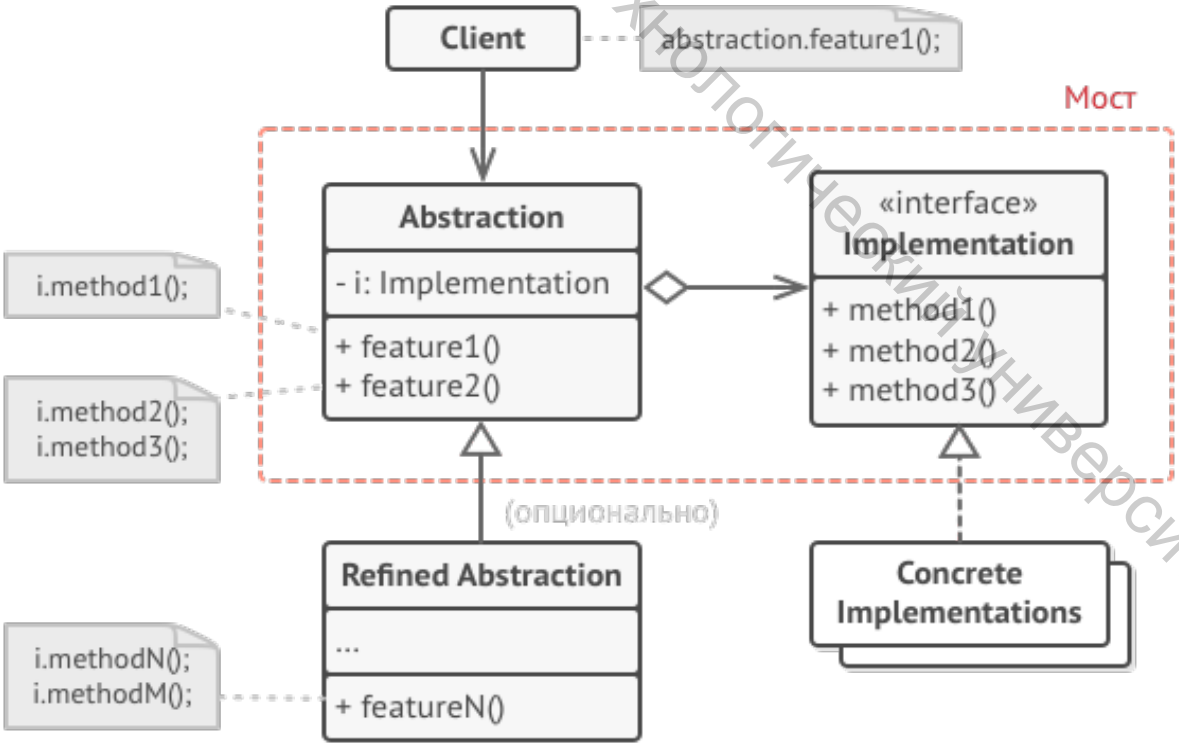

Рисунок 7 – Bridge

16. Реализовать приложение с использованием шаблона Adapter (рис. 8).

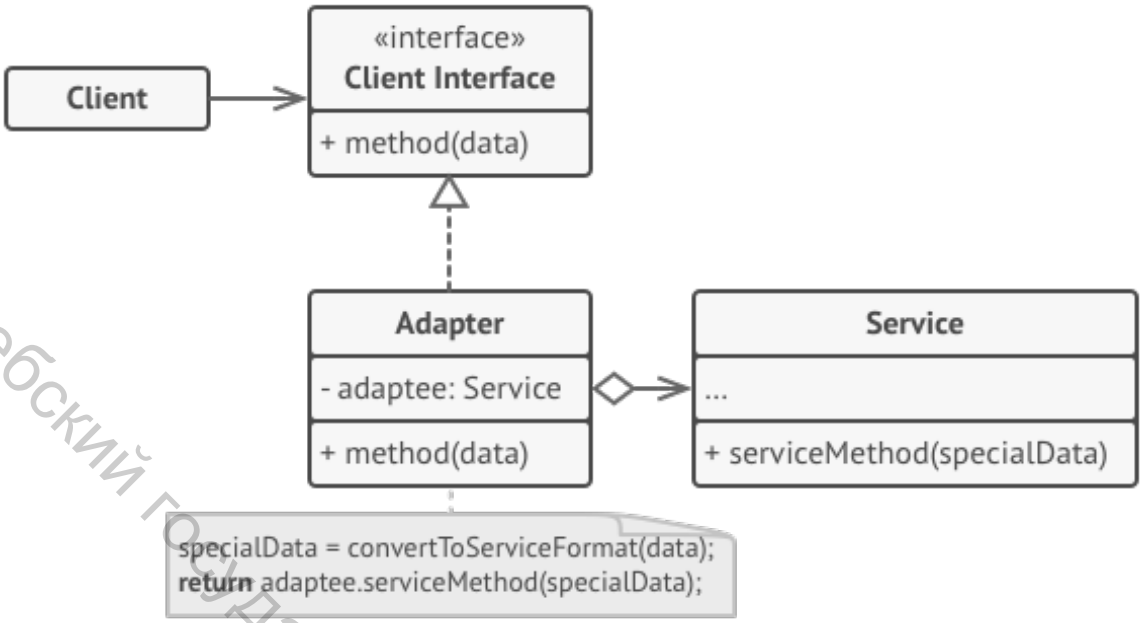

Рисунок 8 – Adapter

17. Реализовать приложение с использованием шаблона Abstract Factory (рис. 9).

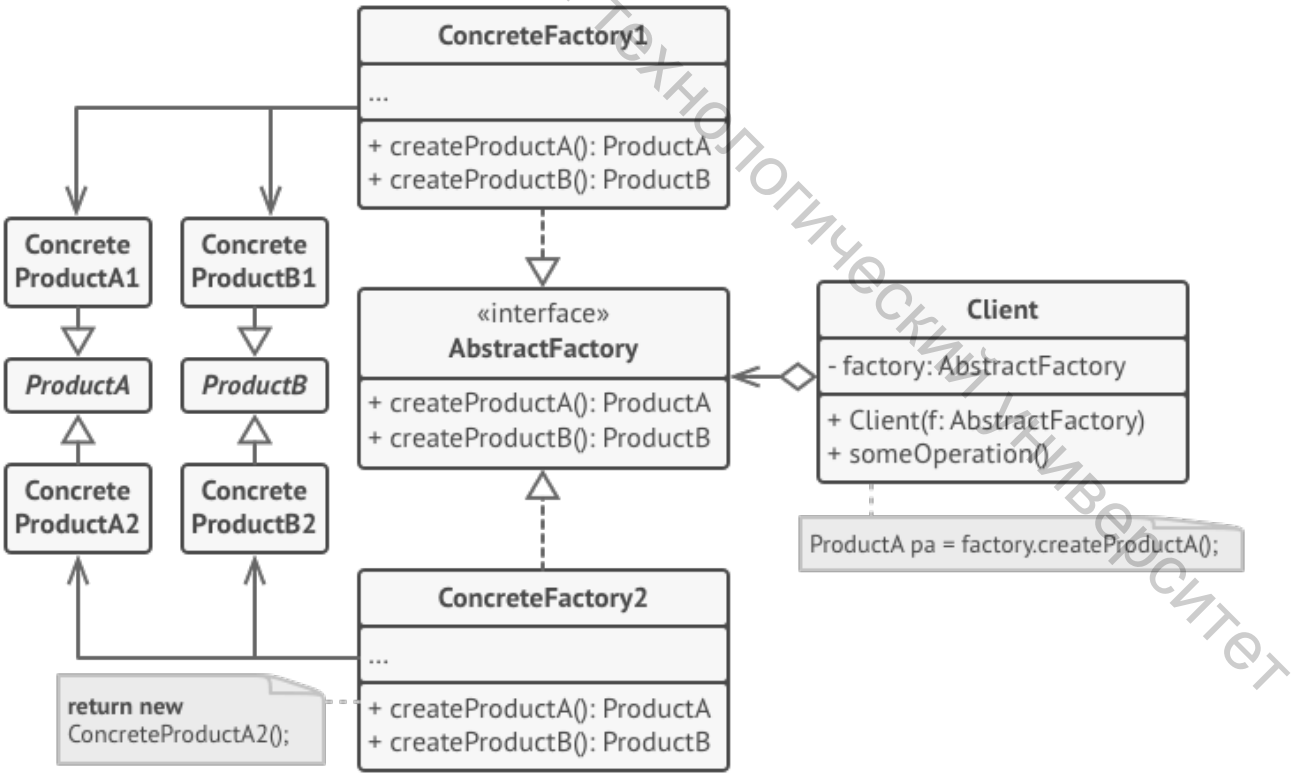

Рисунок 9 – Abstract Factory

18. Реализовать приложение с использованием шаблона Observer (рис. 10).

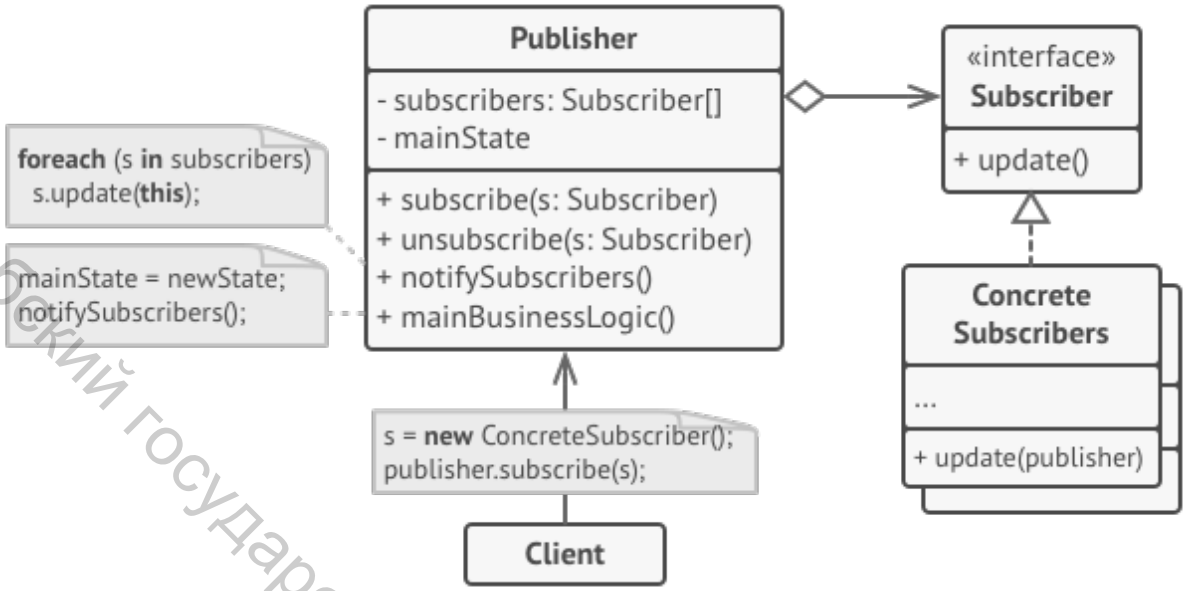

Рисунок 10 – Observer

19. Реализовать приложение с использованием шаблона Builder (рис. 11).

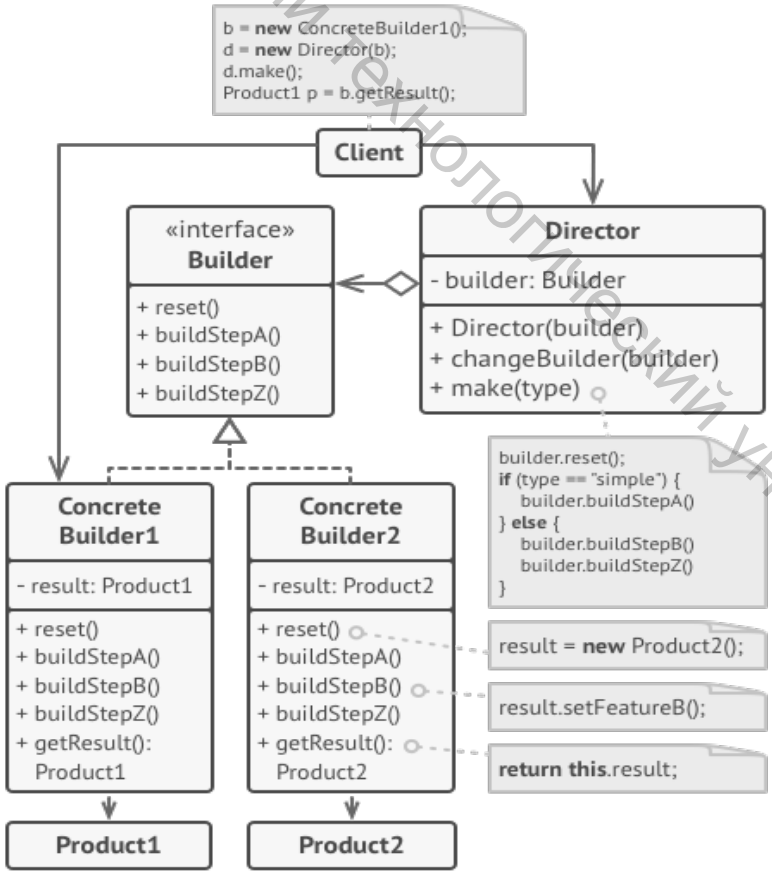

Рисунок 11 – Builder

20. Реализовать приложение с использованием шаблона Mediator (рис. 12).

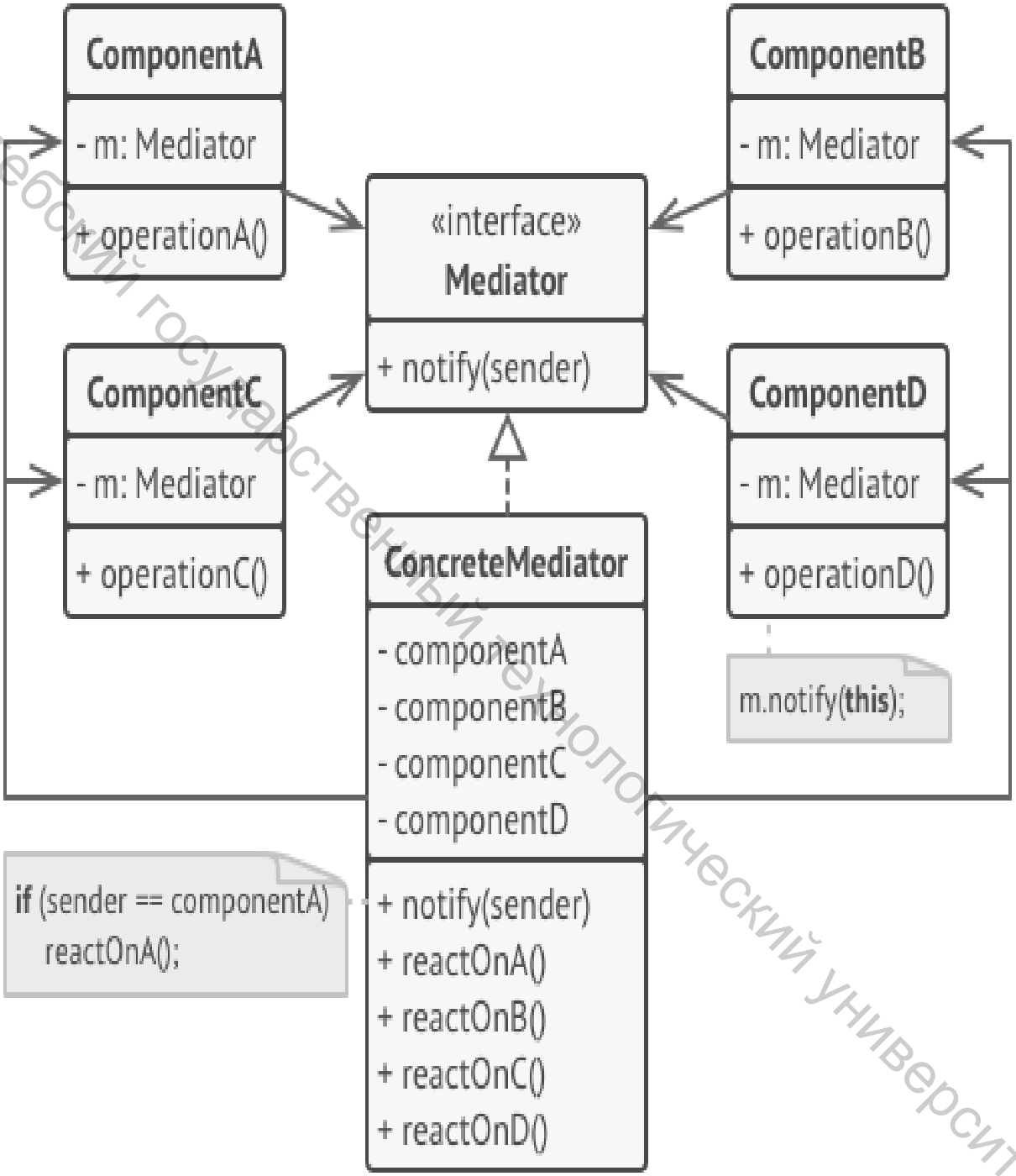

Рисунок 12 – Mediator

21. Реализовать приложение с использованием шаблона Decorator (рис. 13).

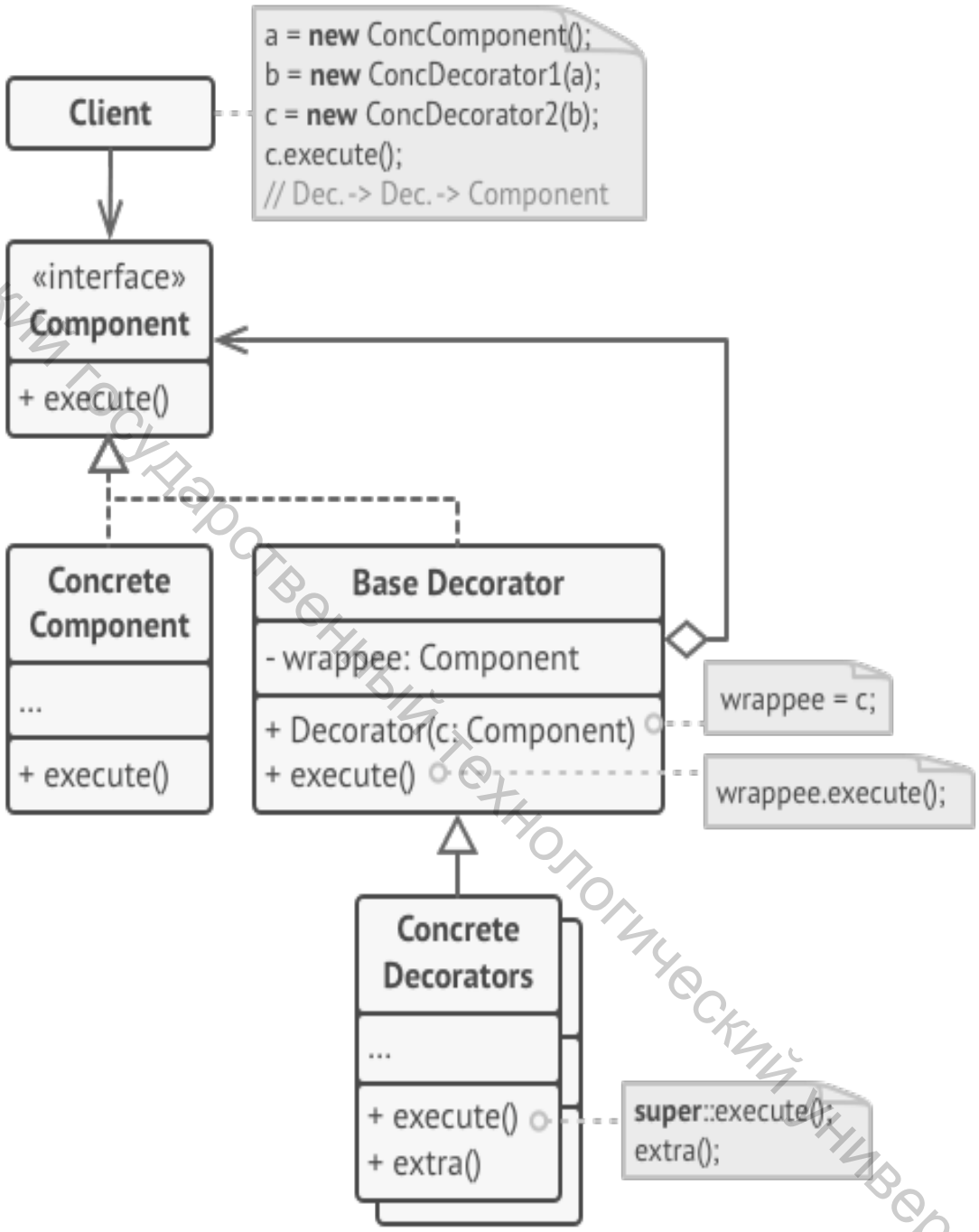

Рисунок 13 – Decorator

22. Реализовать приложение с использованием шаблона Prototype (рис. 14).

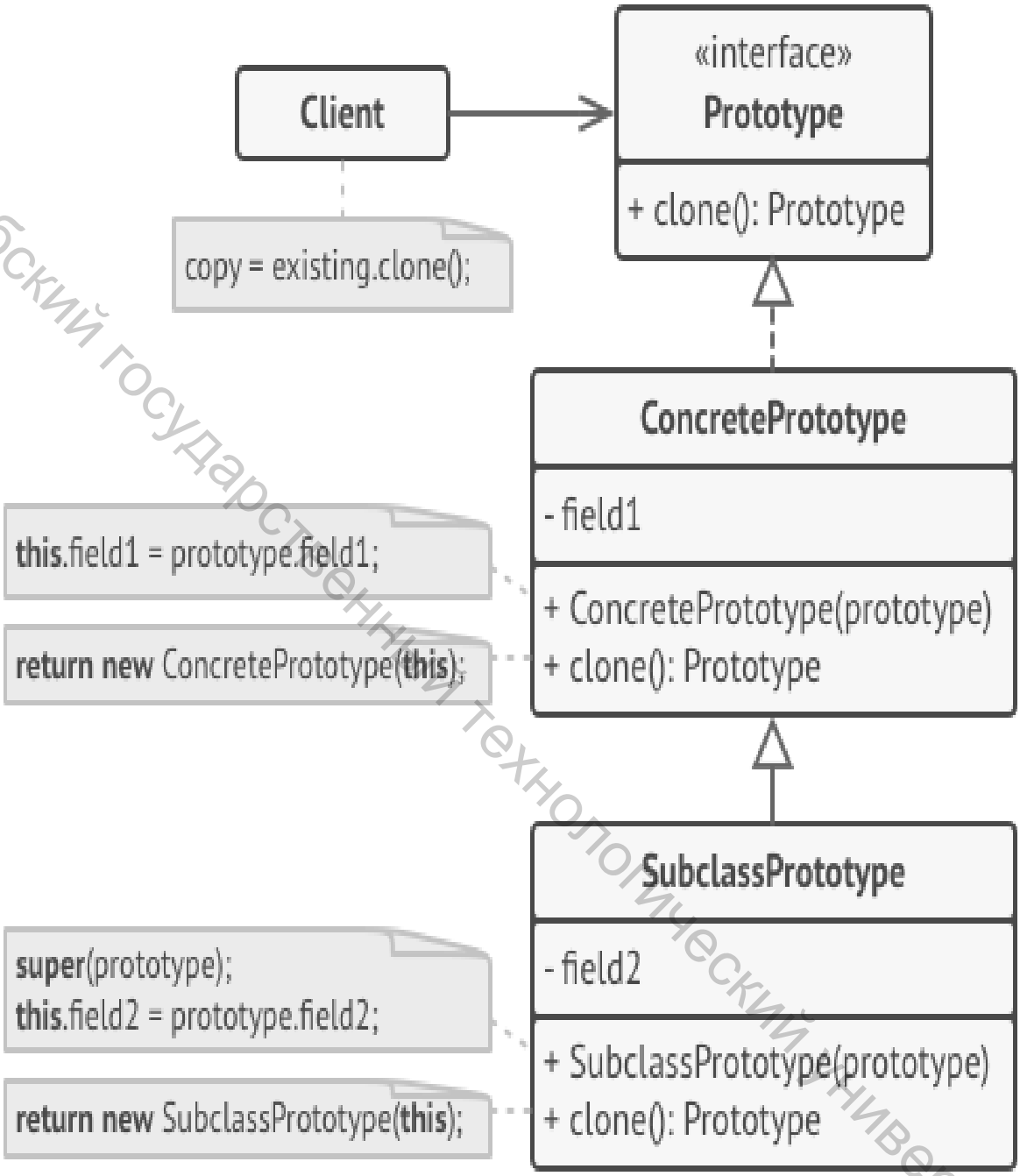

Рисунок 14 – Prototype

23. Реализовать приложение с использованием шаблона Proxy (рис. 15).

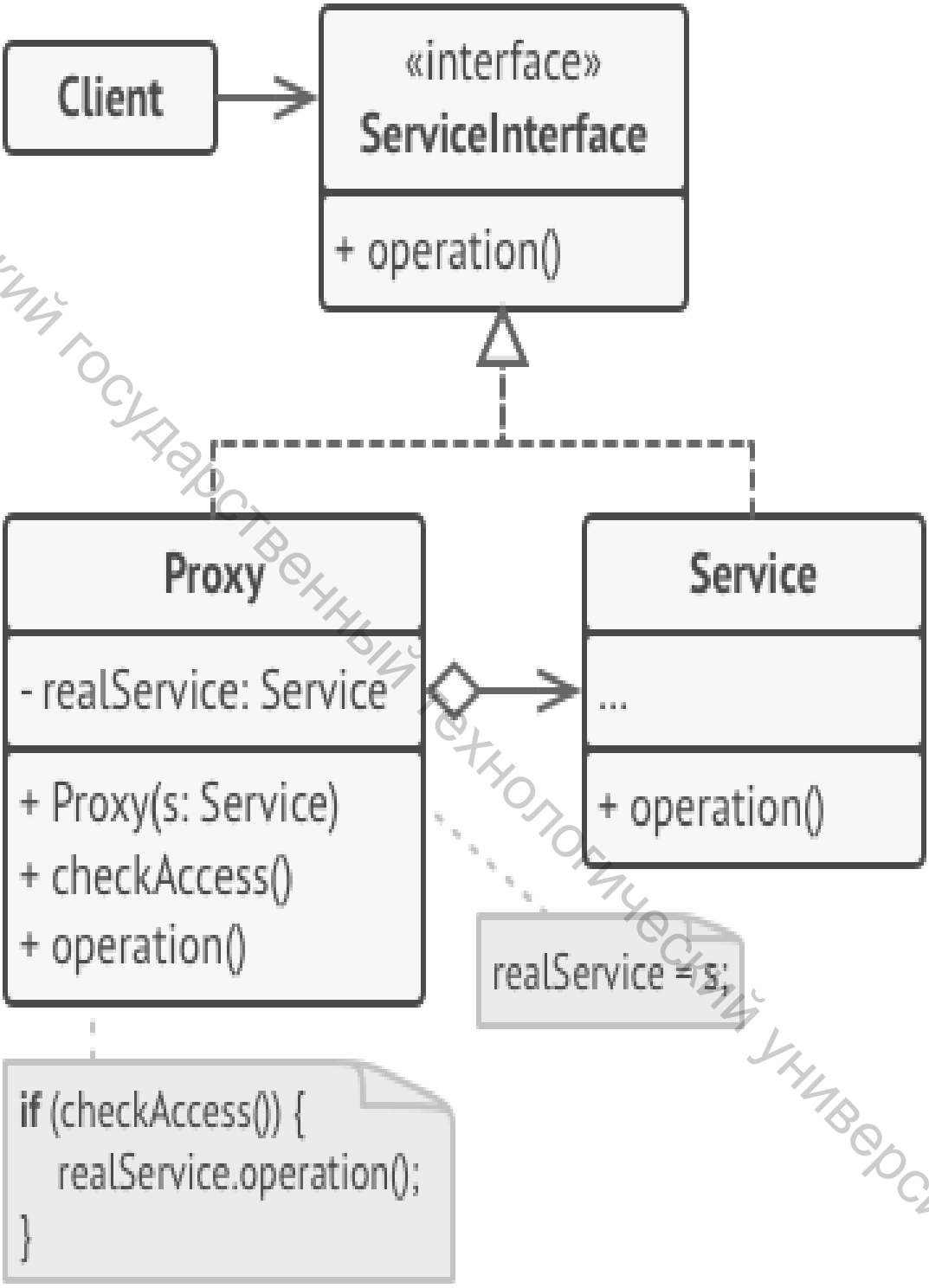

Рисунок 15 – Proxy

24. Реализовать приложение с использованием шаблона Composite (рис. 16).

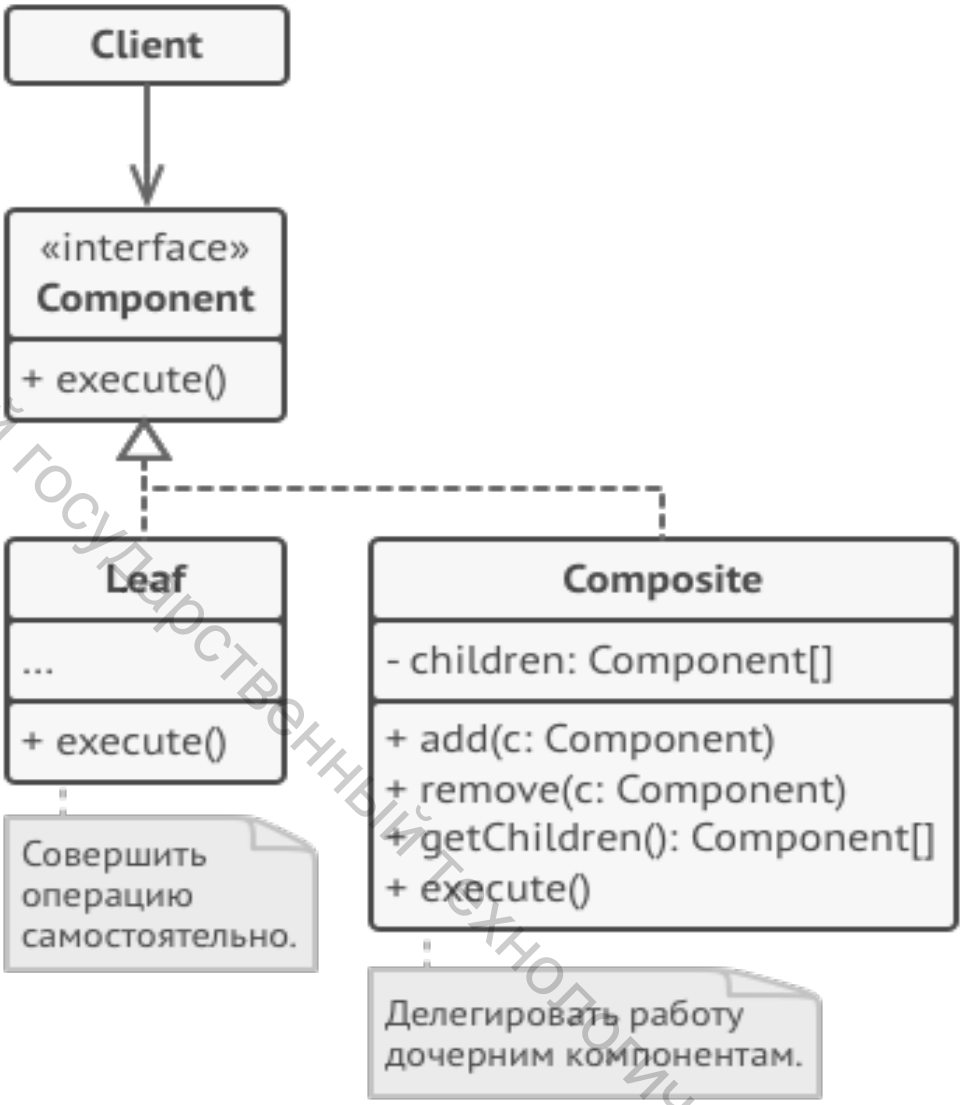

Рисунок 16 – Composite

25. Реализовать один из шаблонов проектирования, не приведенных выше (по усмотрению руководителя).

26. Реализовать приложение для чтения-записи полей объектов в конфигурационный файл.

27. Реализовать приложение для чтения-записи полей объектов в XML-файл.

28. Реализовать оконное приложение с различными компонентами, например, как показано на рисунке 17. Настройки сохранить в конфигурационный файл.

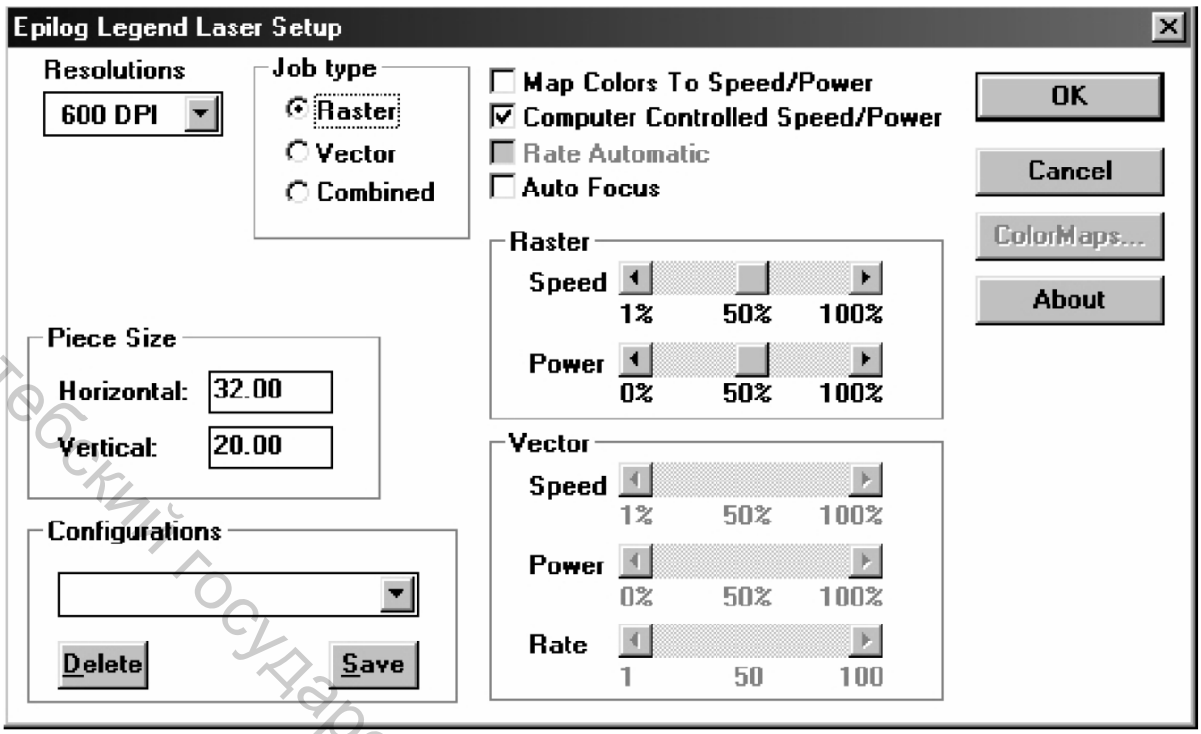

Рисунок 17 – Оконное приложение

5 Требования к содержанию отчета

К моменту окончания практики студент завершает подготовку отчета, на окончательное оформление которого выделяется в конце практики 2–3 дня.

Текстовая часть отчета оформляется в соответствии с общими требованиями к структуре и правилам оформления текстовых документов.

Отчет должен по объему быть не менее 25–30 страниц и содержать информацию о задании, предложенном к выполнению в ходе учебной практики, а также результаты выполнения этого задания. Необходимо отразить все проанализированные методы решения, полученные для исследования данные, подробно осветить выбор конкретного метода или схемы исследования. Обосновать и подтвердить теоретически свой выбор решения.

В отчет включаются количественные и качественные характеристики задания, описание практических методов, примененных в ходе его выполнения, все приложения и необходимые таблицы как с данными задания, так и с его результатами.

Отчет состоит из титульного листа, реферата (рекомендуется), введения, содержания (несколько разделов, которые еще делятся на подразделы, пункты), заключения, списка использованных источников, приложений (при необходимости). Тексты всех разделов необходимо разбить на подразделы, а подразделы на пункты. Например, 3.1.2 – второй пункт первого подраздела третьего раздела. Введение и заключение не нумеруются. Разделы и подразделы должны иметь заголовки (у разделов – все буквы прописные, у подразделов – строчные буквы, кроме первой прописной). Реферат – краткое

связное изложение основного содержания выполненной работы. Начинается с указания объема, количества иллюстраций и таблиц, количества использованных источников. Затем располагается перечень ключевых слов, которые в совокупности дают представление о содержании работы. Ключевыми словами являются слова или словосочетания из текста работы, которые несут существенную смысловую нагрузку. Перечень включает от 5 до 15 ключевых слов, напечатанных в строчку, через запятые в именительном падеже прописными буквами. Дальше идет текст реферата: цель работы; объект исследования; метод исследования и используемые средства; полученные результаты и их новизна; основные конструктивные и технико-эксплуатационные характеристики; эффективность; степень внедрения; область применения и рекомендации.

Содержание отчета практики состоит из разделов (обычно от 2 до 4), которые часто разбиваются на подразделы. Заранее следует продумать возможную логическую связь между разделами (подразделами).

Первый раздел можно посвятить теории рассматриваемой системы, обзору экспериментальной, методической, теоретической литературы с обязательной ссылкой на источники. Во втором разделе обычно дают обоснование, описание методики со всеми подробностями. В третьем разделе, например, могут быть описаны условия, особенности и результаты задания. Математическое объяснение, обсуждение итогов задания обязательно. Особо следует обратить внимание на взаимосвязь исходных данных с полученными результатами.

Требования к оформлению кода программ:

– абзац одинарный;

– шрифт Courier 10pt;

– абзац одинарный, без интервалов «перед» и «после»;

– код отформатирован автоматически в IDE.

Примерная структура отчета:

1. Тема индивидуального задания, научно-исследовательской работы студента. Формируется тема самостоятельного исследования, приводится обоснование целесообразности его проведения. Отмечается новизна, актуальность работы, ее место в исследуемых проблемах.

2. Аналитический обзор. Выяснение состояния вопроса на основе изученных литературных и экспериментальных данных. Следует обратить внимание на полноту и объективность проведенного обзора.

3. Математическое обеспечение исследований. Оценивают сильные и слабые стороны метода, имеющихся данных. Необходимо обратить внимание на целесообразность и экономический эффект проведенной работы.

4. Содержание и результаты выполненной работы.

5. Заключение. Оценить результаты работы (положительные и отрицательные). Отметить возникающие проблемы. Стараться наметить пути и цели дальнейшей работы или мотивируется нецелесообразность ее продолжения.

Литература

1. Герберт Шилдт. Java. Полное руководство / Герберт Шилдт. – Москва : Вильямс, 2012. – 1104 с.
2. Берт Бейтс, Кэтти Сьерра. Изучаем Java / Берт Бейтс, Кэтти Сьерра. – Москва : ЭКСМО, 2012. – 720 с.
3. Кей Хорстманн, Гари Корнелл. Java. Библиотека профессионала. Том 1. Основы / Кей Хорстманн, Гари Корнелл. – Москва : Вильямс, 2014. – 864 с.
4. Айвор Хортон. Java 2 (книга 1 и книга 2) / Айвор Хортон. – Москва : Лори, 2013. – 1020 с.
5. Берт Бейтс, Кэтти Сьерра, Элизабет Фримен, Эрик Фримен. Паттерны проектирования / Берт Бейтс, Кэтти Сьерра, Элизабет Фримен, Эрик Фримен. – Санкт-Петербург : Питер, 2014. – 656 с.
6. Бенджамин Эванс, Мартин Вербург. Java. Новое поколение разработки / Бенджамин Эванс, Мартин Вербург. – Санкт Петербург : Питер, 2014. – 560 с.
7. Патрисия Лигуори, Роберт Лигуори. Java 8. Карманный справочник / Патрисия Лигуори, Роберт Лигуори. – Москва : Вильямс, 2015. – 256 с.
8. Хелм, Р. Приемы объектно-ориентированного проектирования. Паттерны проектирования / Р. Хелм, Эрик Гамма. – Санкт-Петербург : Питер, 2013. – 368 с.
9. Брюс Эккель. Философия Java. Библиотека программиста / Брюс Эккель. – Санкт-Петербург : Питер, 2014. – 640 с.
10. Попов, И. И. Компьютерные сети / И. И. Попов, Н. В. Максимов. – Москва : Инфра-М, 2013. – 464 с.

Учебное издание

КОМПЬЮТЕРНАЯ ПРАКТИКА

Методические указания по прохождению практики

Составители:
Кириллов Алексей Геннадьевич
Буевич Татьяна Владимировна

Редактор *Н. В. Медведева*
Корректор *Т. А. Осипова*
Компьютерная верстка *А. Г. Кириллов*

Подписано к печати 11.10.2018. Формат 60x90 $^{1}/_{16}$. Усл. печ. листов 1,6.
Уч.-изд. листов 1,8. Тираж 25 экз. Заказ № 287.

Учреждение образования «Витебский государственный технологический университет»
210038, г. Витебск, Московский пр., 72.
Отпечатано на ризографе учреждения образования
«Витебский государственный технологический университет».
Свидетельство о государственной регистрации издателя, изготовителя,
распространителя печатных изданий № 1/172 от 12 февраля 2014 г.
Свидетельство о государственной регистрации издателя, изготовителя,
распространителя печатных изданий № 3/1497 от 30 мая 2017 г.