Санкт-Петербургский политехнический университет

Институт компьютерных наук и технологий

Кафедра «Компьютерные системы и программные технологии»

КУРСОВОЙ ПРОЕКТ

Разработка приложения "Tree Visualizer"

по дисциплине «Технологии программирования»

Выполнил студент

гр. 3530901/20002

Нестеренко С. А.

Преподаватель

Степанов Д. С.

Санкт-Петербург

2023

Санкт-Петербургский политехнический университет

ЗАДАНИЕ

НА ВЫПОЛНЕНИЕ КУРСОВОГО ПРОЕКТА

студенту группы 3530901/20002 Нестеренко Сергею Андреевичу

1. Тема проекта: создание приложения «Tree Visualizer» с графическим интерфейсом. Приложение должно иллюстрировать работу бинарного и префиксного деревьев.
2. Срок сдачи законченного проекта: 31 мая.
3. Содержание пояснительной записки: введение с описанием функционала приложения, описание предложенного решения, тестирование программы, заключение, список использованных источников.

Дата получения задания: «17» апреля 2023 г.

Руководитель: Степанов Д. А.

Задание принял к исполнению: Нестеренко С. А.

«17» апреля 2023

ВВЕДЕНИЕ

Цель работы: создать и протестировать приложение, иллюстрирующее работу с префиксным и бинарным деревьями.

Функционал приложения

Приложение «Tree Visualizer» позволяет работать с двумя структурами данных: бинарным деревом и префиксным деревом. Для каждого из типов деревьев реализуются следующие возможности: добавление новых элементов, поиск по дереву, удаление элементов и очистка. Дерево с его элементами выводится на холсте ниже панели управления.

ОПИСАНИЕ ПРЕДЛОЖЕННОГО РЕШЕНИЯ

Для создания графического пользовательского интерфейса использовалась библиотека JavaFX. Были использованы следующие готовые элементы интерфейса: кнопка (Button), текстовое поле (TextField), холст (Canvas), панель с вкладками (TabPane), а также вертикальный и горизонтальный ряды (VBox и HBox). Взаимодействие пользователя с графическим элементом описывается в событиях этого элемента.

При создании использовался шаблон проектирования MVC (model-view-controller). Поэтому проект содержит классы, описывающие модель каждого дерева (BinaryTree.java и Trie.java), контроллер (Controller.java) и FXML-файл, описывающий представление (view.fxml).

В файле view.fxml описывается внешний вид окна приложения в виде древовидной структуры, содержащей элементы интерфейса и их параметры. К

большинству элементов привязаны поля и методы, содержащиеся в классе Controller.

Класс Controller содержит поля и методы, относящиеся к элементам интерфейса, описанным в представлении. Методы этого класса переносят взаимодействие пользователя с представлением на работу с объектами, описываемыми моделью.

Классы BinaryTree и Trie содержат бизнес-логику работы бинарного и префиксного деревьев соответственно. Помимо этого, первый класс содержит вложенный статический класс BinaryNode, а второй – вложенный статический класс TrieNode, которые описывают узлы бинарного и префиксного деревьев соответственно. Экземпляры классов создаются в контроллере, а взаимодействие с ними описано публичными методами соответствующих классов. Эти классы не работают с графической библиотекой напрямую.

Также имеется класс Application.java, содержащий функцию start, где задаётся сцена и запускается главное окно.

За счёт такого подхода разработка отдельных частей приложения может вестись независимо: разработчик вправе заменить модель на другую, оставив необходимые для работы публичные методы, а представление и контроллер оставить прежними. Можно поступить и наоборот: оставить модель прежней, но заменить контроллер или представление. Таким образом, упрощается дальнейшая поддержка приложения.

ТЕСТИРОВАНИЕ ПРОГРАММЫ

Для тестирования бизнес-логики написано 30 автоматических тестов: 6 для префиксного дерева, 8 для узла префиксного дерева, 6 для бинарного дерева и 10 для узла бинарного дерева. Тесты написаны с использованием библиотеки JUnit.

Данные тесты проверяют работу каждого публичного метода класса. Для примера рассмотрим автоматические тесты, написанные для бинарного дерева.

Первый тест проверяет работу метода getRoot(), который должен возвращать экземпляр класса BinaryNode, являющийся корнем дерева. Для проверки правильной работы метода мы создаём новое бинарное дерева и проверяем, является ли его корень null. Далее мы кладём в дерево некоторое число и проверяем, создался ли на месте корня новый узел с этим значением.

Второй тест проверяет работу метода put(). Здесь мы создаём новое бинарное дерево, кладём в него элементы (было взять 3 различных значения) и последовательно проверяем, появляются ли они на тех местах, где должны быть расположены по задумке.

Третий тест проверяет работу метода getPathTo(). Для этого создаётся новое дерево и в него кладутся некоторые значения. Далее мы сперва создаём путь до нужного узла вручную, а затем с помощью метода getPathTo(). Тест проверяет, совпадают ли эти значения.

Четвёртый тест проверяет работу метода remove(). Мы кладём в дерево различные элементы, затем прописываем команду для удаления элемента с заданным числом и проверяем, исчез ли соответствующий узел.

Пятый тест проверяет работу метода clear(). В новое бинарное дерево мы кладём различные элементы, затем очищаем дерево при помощь данного метода и проверяем, является ли корень дерева null.

Шестой тест проверяет работу метода isEmpty(). Сперва мы проверяем, выдаёт ли метод true в случае с новым деревом, а затем – то же самое в случае с только что очищенным.

ЗАКЛЮЧЕНИЕ

Итак, было разработано приложение Tree Visualizer, которое наглядно показывает работу бинарного и префиксного деревьев. Для подтверждения правильной работы бизнес-логики приложения были написаны автоматические тесты. В ходе разработки я самостоятельно ознакомился с библиотекой JavaFX и шаблон проектирования MVC.

В дальнейшем к этому приложению можно добавить другие виды деревьев в новых вкладках. Также при желании есть возможность изменить дизайн приложения, сменив цветовую гамму и расположение элементов. Таким образом, приложение имеет потенциал для дальнейшего развития.

Исходный код приложения расположен по следующему адресу: https://github.com/borov-kaban-nikitovich/TreeVisualizer

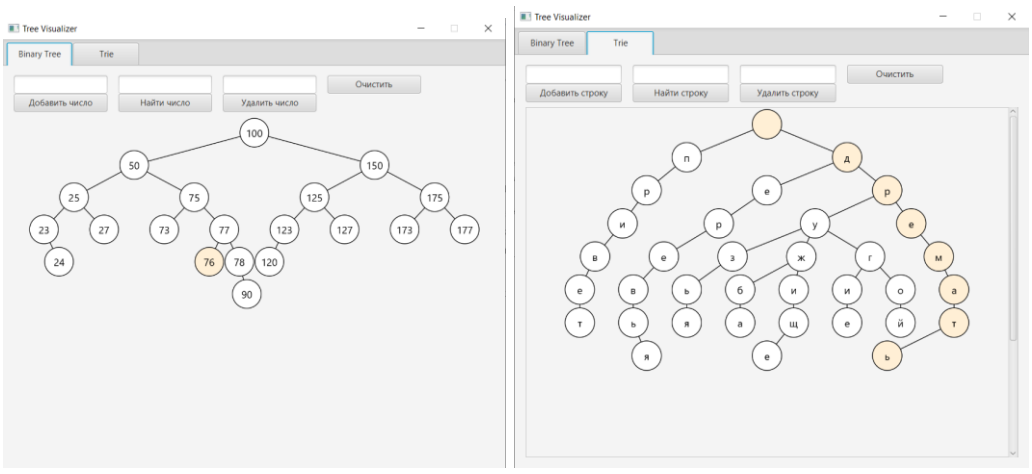

Рис. 1. Скриншоты приложения

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. https://javarush.com – образовательный портал о языке Java.
2. Прохоренок Н.А. – JavaFX. Наиболее полное руководство.
3. Гербер Шилдт – Java. Полное руководство. 12-е издание. Всестороннее раскрытие языка Java.
4. https://javarush.com/groups/posts/2536-chastjh-7-znakomstvo-s-patternom-mvc-model-view-controller – информация о паттерне MVC.
5. https://www.cs.usfca.edu/~galles/visualization/BST.html – визуализатор бинарного дерева.
6. https://www.cs.usfca.edu/~galles/visualization/Trie.html – а префиксного дерева.