МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Донецкий национальный технический университет»

Факультет ИСП

Кафедра ПИ им Л.П.Фельдмана

Лабораторная работа №2

по курсу: «Профессиональная практика программной инженерии»

по теме: «Работа с ветками»

Выполнил:

ст. гр. ПИ-20

Белоконь Л.О.

Проверили:

Филипишин Д.А.

ДОНЕЦК – 2024

**Цель работы:** познакомиться с основами использования веток в системе Git.

**Ход работы**

Сперва из ветки main были созданы 6 веток: code\_editor, program\_setting, search\_module, semantics\_analyzer, terminal, theme\_management (см. рис. 1). Затем, последовательно был осуществлён переход на каждую ветку с помощью команды «checkout <имя ветки>». В каждой ветке было совершено по 5 фиксаций изменений. Просмотреть историю вносимых изменений в ветку можно с помощью команды «log --pretty=format:\"(%h)%ad|%s%d[%an]\" --graph -date=short» (см. рис. 2-7).

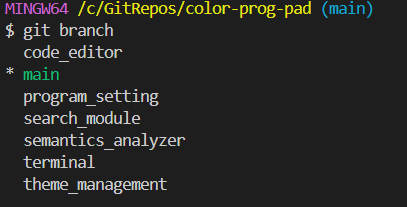


Рисунок 1 — Вывод списка веток

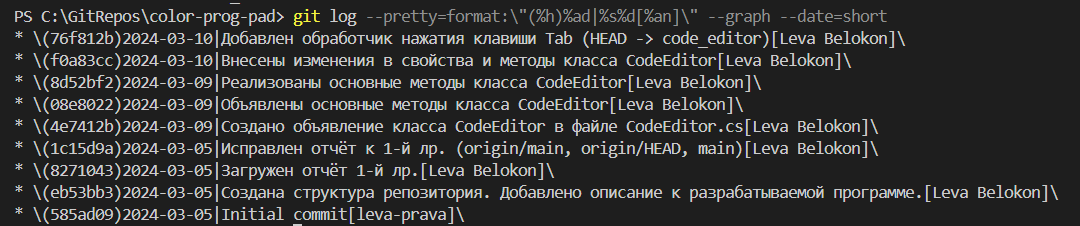


Рисунок 2 — Вывод лога для ветки code\_editor

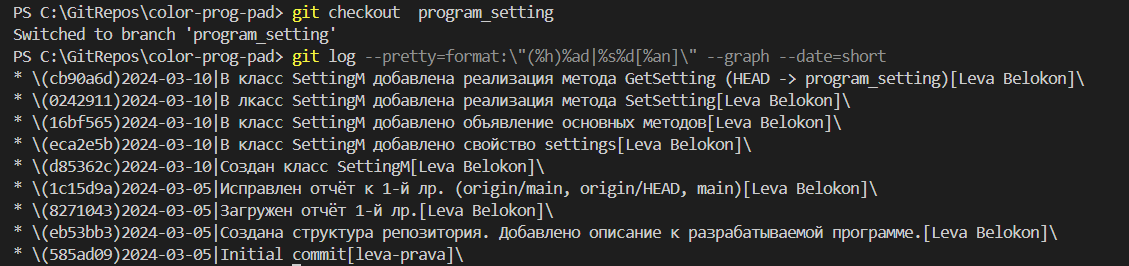


Рисунок 3 — Вывод лога для ветки program\_setting

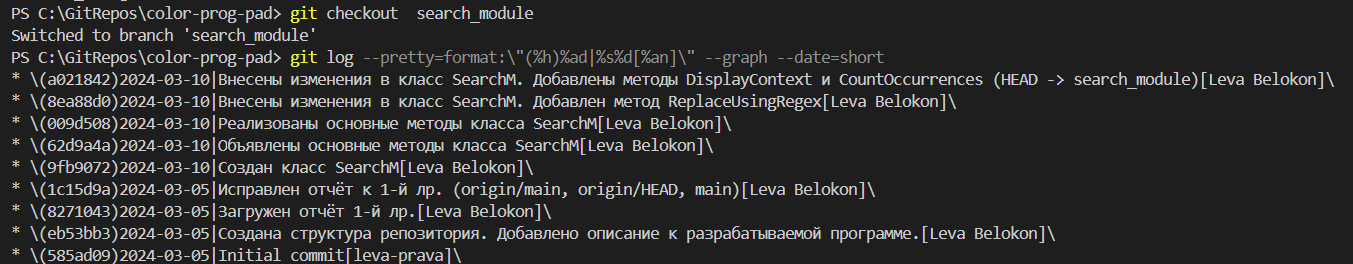


Рисунок 4 — Вывод лога для ветки search\_module

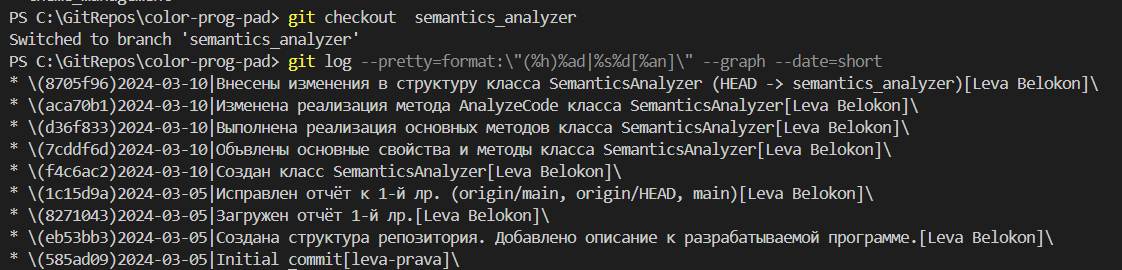


Рисунок 5 — Вывод лога для ветки semantics\_analyzer

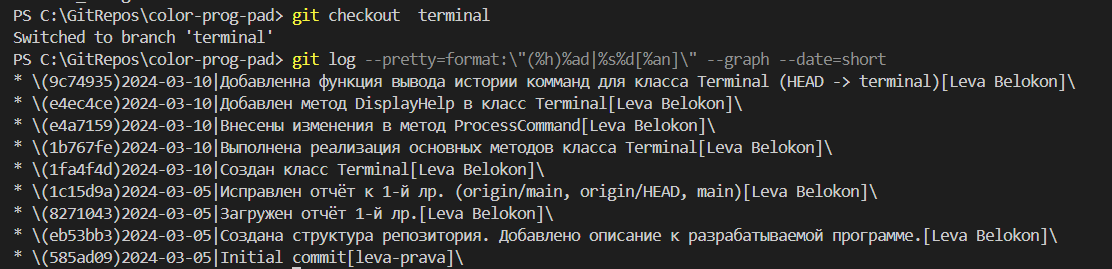


Рисунок 6 — Вывод лога для ветки terminal

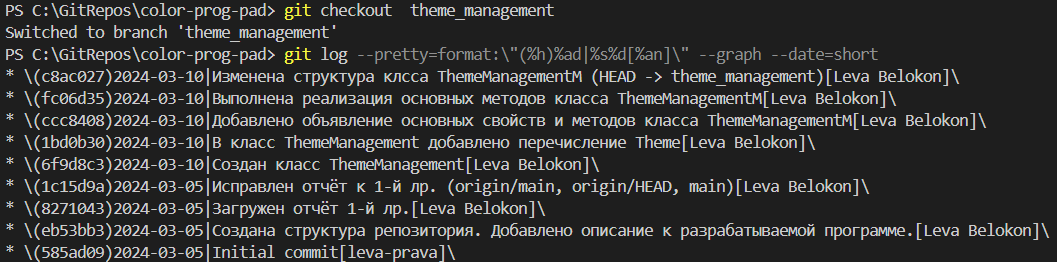


Рисунок 7 — Вывод лога для ветки theme\_management

Затем после того, как была завершена работа во всех ветках, зафиксированные изменения были отправлены на удалённый репозиторий с помощью команды «push origin --all» (см. рис. 8-9).

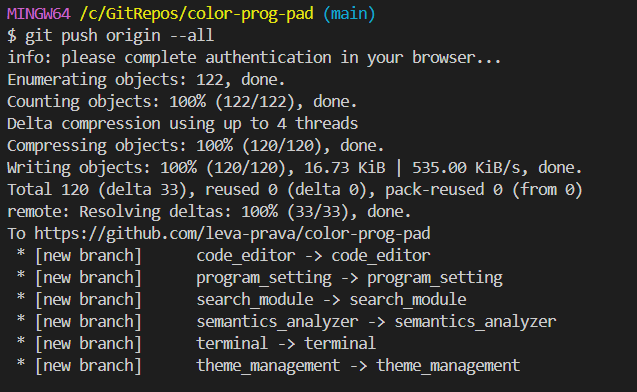


Рисунок 8 — Отправка изменений на удалённый репозиторий

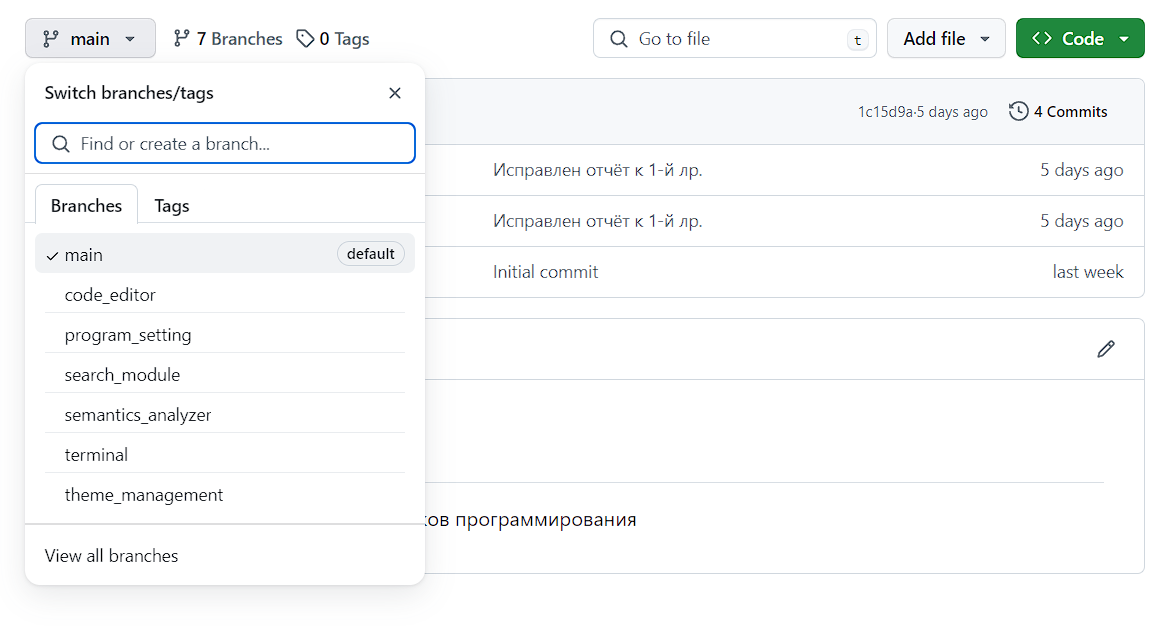


Рисунок 9 — Просмотр списка веток через сервис GitHub

Для того, чтобы продемонстрировать разрешение конфликта слияния веток, создадим конфликтную ситуацию: в ветке main создадим копию файла Terminal.cs из ветки Terminal, изменив некоторые строки. После выполнения команды «git merge terminal» на консоли будет выведено сообщение о том, что git не смог решить конфликт слияния автоматически (см. рис. 10), после чего редактор VS Code предложит в удобном формате выбрать для каждой конфликтной ситуации нужный вариант строк (см. рис. 10-13).

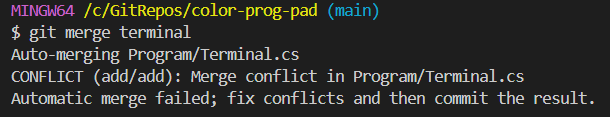


Рисунок 10 — Сообщение о конфликте слияния

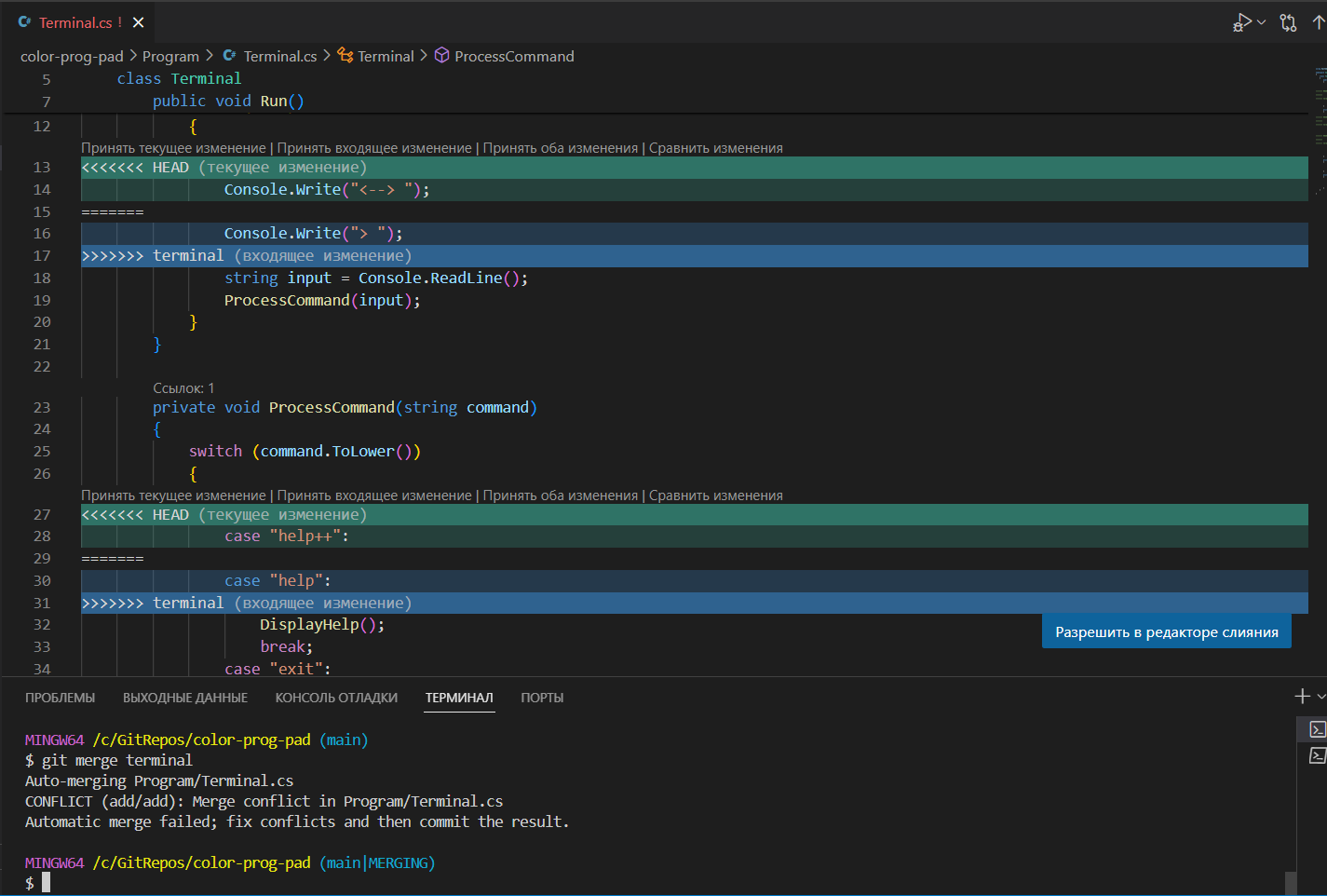


Рисунок 11 — Окно устранения конфликтов слияния редактора VS Code

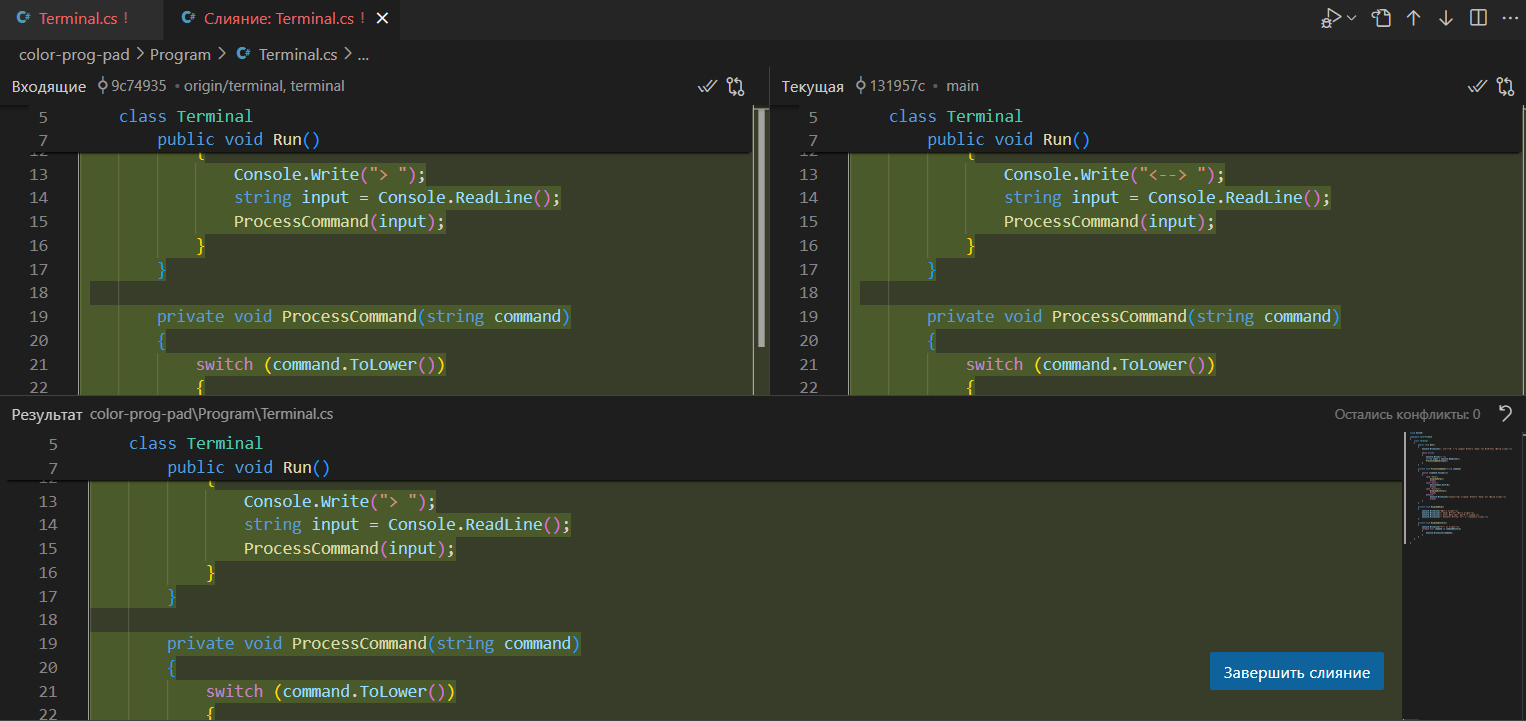


Рисунок 12 — Окно завершения слияния редактора VS Code (1-я конфликтная ситуация)

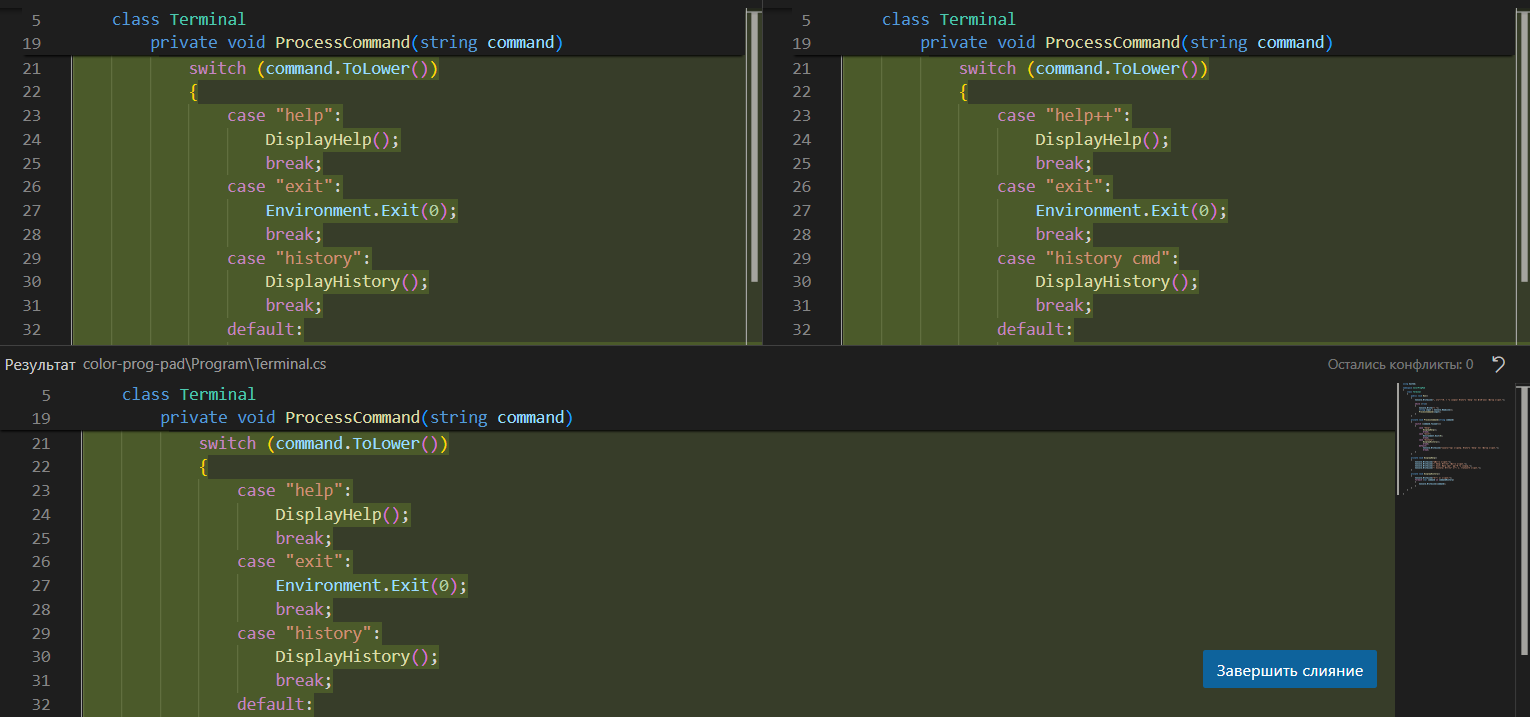


Рисунок 13 — Окно завершения слияния редактора VS Code (2-я конфликтная ситуация)

После того, как будут выбраны нужные варианты конфликтующих строк, необходимо зафиксировать изменения. В редакторе VS Code это предлагается сделать с помощью нажатия кнопки (см. рис. 14-15).

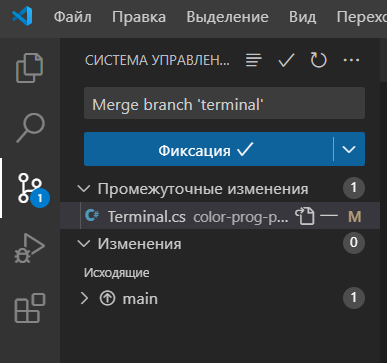


Рисунок 14 — Фиксация разрешенного конфликта слияния

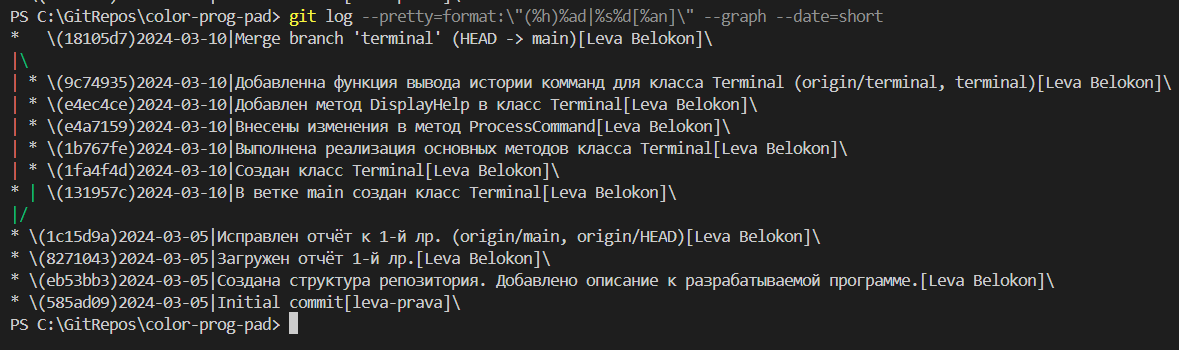


Рисунок 15 — Лог ветки main после слияния с веткой terminal

Затем с помощью команды «git merge -m “сообщение” <имя ветки> » было выполнено последовательное слияние остальных веток с веткой main. На рисунке 16 представлен лог ветки main после слияния со всеми второстепенными ветками.

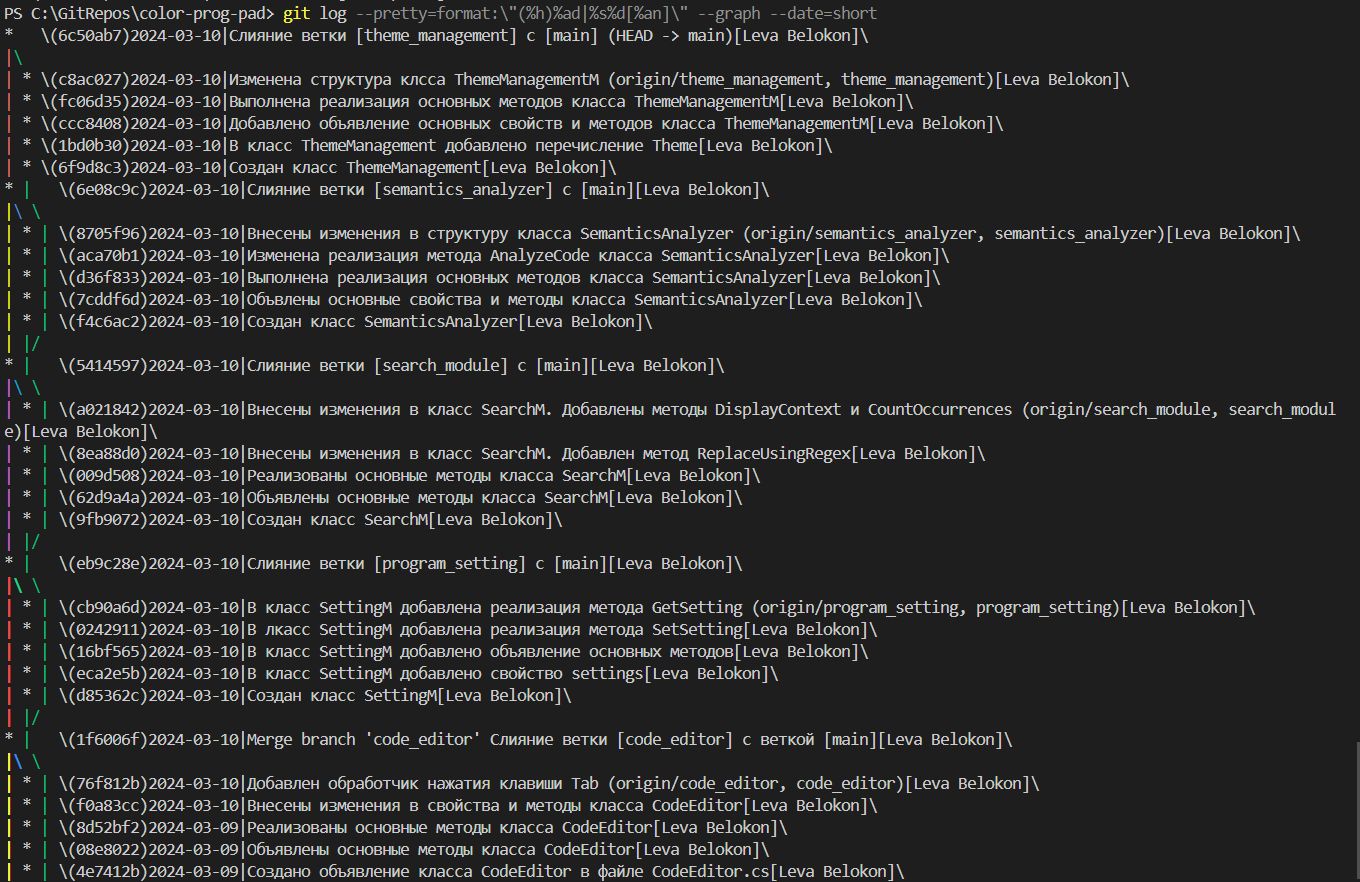


Рисунок 16 — Лог ветки main после слияния с второстепенными ветками

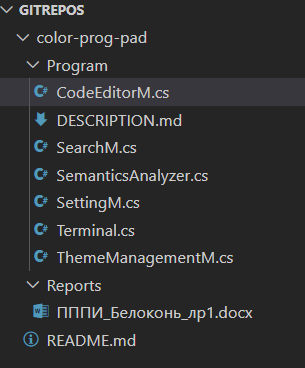


Рисунок 17 — Дерево файлов ветки main после слияния

После слияния, каждая из веток была удалена командой «branch -d» (см. рис. 18-19).

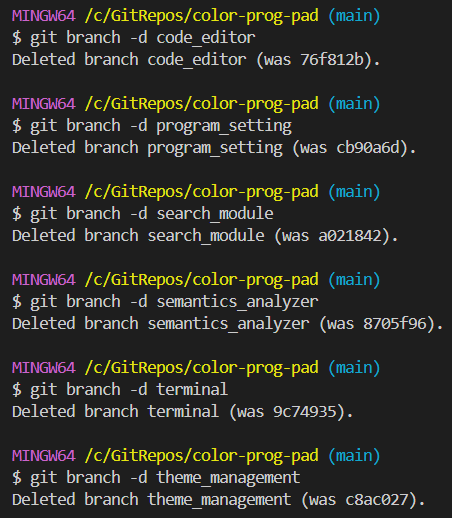


Рисунок 18 — Процесс удаления второстепенных веток

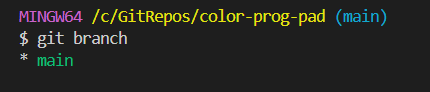


Рисунок 19 — Список веток

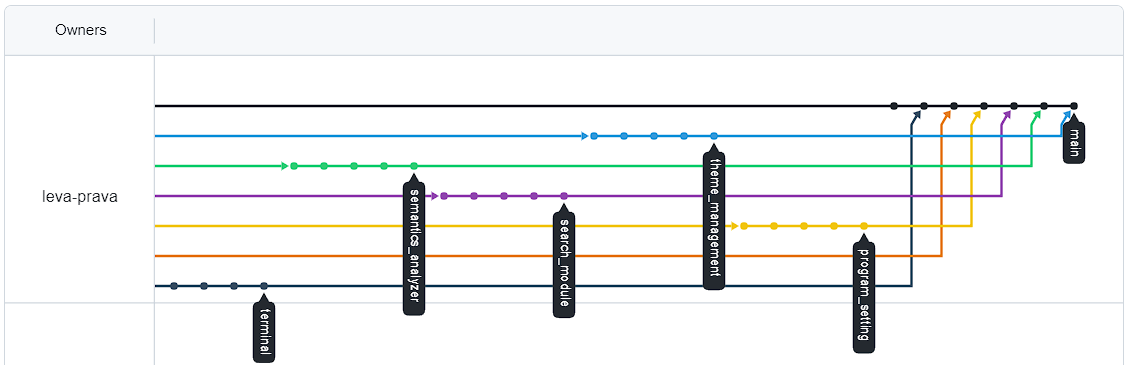


Рисунок 20 — График слияния веток