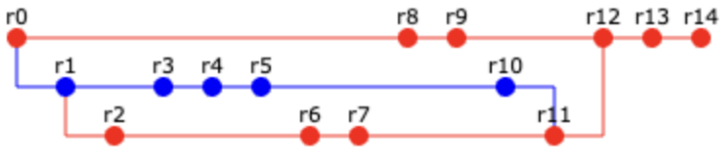

Дополнительное задание 1: переименовать коммиты слияния

Команды для выполнения задания:

git rebase -ip 159eb69 #rebase от revision10
#появляется окно интерактивного перебазирования, в котором у нужных нам коммитов изменяем действие pick на reword (reword edits the commit message)
#сохраняем и выходим из этого окна, и дальше будет предложение изменение сообщений этих двух коммитов
#появляется новое окно, в котором изменяем commit message и сохраняем
#так как это коммиты слияния, при которых уже возникали конфликты, нужно заново решить эти конфликты в обоих коммитах последовательно
#решаем конфликты при revision11
git add .
git commit
#решаем конфликты при revision12
git add .
git commit
git reflog --graph --oneline

```
* bdedcba (HEAD -> main) Revision14 (r14)
* 550c44a Revision13 (r13)
*   f3476b3 Revision12 (r12)
|\
| *   2a0a1c4 Revision11 (r11)
| |\
| | * 159eb69 (branch2) Revision10 (r10)
| | * 55b3281 Revision5 (r5)
| | * 3da622d Revision4 (r4)
| | * 015c02e Revision3 (r3)
| * | 5ce31a3 Revision7 (r7)
| * | 13ce348 Revision6 (r6)
| * | 9bf24a0 Revision2 (r2)
| |/
| * 76c29bd Revision1 (r1)
* | 2bd5a8d Revision9 (r9)
* | fcbb400 Revision8 (r8)
|/
* 4ffd87a Revision0 (r0)
```

Дополнительное задание 2: необходимо скопировать коммиты R2,R6,R7,R3 в ветку main с помощью cherry-pick

Команды для выполнения задания:

git cherry-pick 9bf24a0 #revision2
#solving conflicts
git add .

```
git cherry-pick --continue
git cherry-pick 13ce348 #revision6
#solving conflicts
git add .
git cherry-pick --continue
git cherry-pick 5ce31a3 #revision7
#solving conflicts
git add .
git cherry-pick --continue
git cherry-pick 015c02e #revision3
#solving conflicts
git add .
git cherry-pick --continue
git log --graph --oneline
```

```
* 8deaf59 (HEAD -> main) Revision3 (r3)
* 488c53a Revision7 (r7)
* 35ac30d Revision6 (r6)
* 459ffee Revision2 (r2)
* bdedcba (origin/main) Revision14 (r14)
* 550c44a Revision13 (r13)
*   f3476b3 Revision12 (r12)
|\
| *   2a0a1c4 Revision11 (r11)
| |\
| | * 159eb69 (branch2) Revision10 (r10)
| | * 55b3281 Revision5 (r5)
| | * 3da622d Revision4 (r4)
| | * 015c02e Revision3 (r3)
| * | 5ce31a3 Revision7 (r7)
| * | 13ce348 Revision6 (r6)
| * | 9bf24a0 Revision2 (r2)
| |/
| * 76c29bd Revision1 (r1)
* | 2bd5a8d Revision9 (r9)
* | fcbb400 Revision8 (r8)
|/
* 4ffd87a Revision0 (r0)
```

Так как коммиты, которые было необходимо скопировать, были сделаны на более раннем этапе разработки проекта, то, по понятным причинам, возникли конфликты, которые были решены при копировании каждого из четырех коммитов.

GIT REFLOG:

```
git reflog --no-decorate -4
8deaf59 HEAD@{0}: commit (cherry-pick): Revision3 (r3)
488c53a HEAD@{1}: commit (cherry-pick): Revision7 (r7)
35ac30d HEAD@{2}: commit (cherry-pick): Revision6 (r6)
459ffee HEAD@{3}: commit (cherry-pick): Revision2 (r2)
```

Вывод: два дополнительных задания (переименование коммитов через rebase и копирование коммитов с помощью cherry-pick) помогли изучить git на более глубоком уровне, поскольку два этих процесса основываются на применении обширного числа базовых команд гит.