Вариант 10

Задание: Использовать 3 ветки, создать 2 конфликта и разрешить их.

https://github.com/Venom342/Lab3

1. Схема хода работы (рисунок 1).

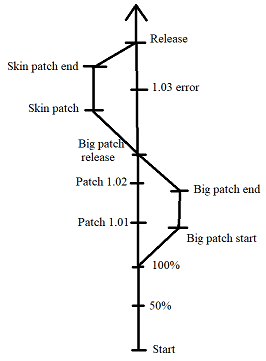


Рисунок 1 – Схема

1. В работе я использовал 2 ветки (рисунок 1.) для реализации работы над большими патчами, в то время как по основные ветки добавляются небольшие патчи.

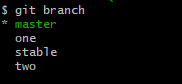


Рисунок 2 – Git

1. Ход создания проекта (рисунок 3.)

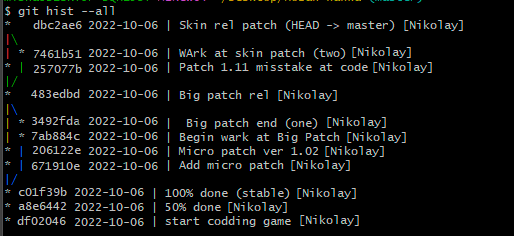


Рисунок 3 – Проект

1. Согласно заданию, необходимо создать 2 конфликта и разрешить их. Приложение Git выдает ошибку при невозможности слияния. (рисунок 4)



Рисунок 4 – Ошибка

1. Приложение автоматически подчеркивает строки, которые необходимо исправить. (Рисунок 5­-6).

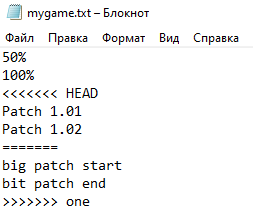


Рисунок 5 – Ошибки

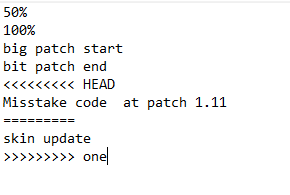


Рисунок 6 – Ошибки

1. Проверка репозитория (рисунок 7).



Рисунок 7 – Проверка

Вывод: Программа Git позволяет легко смотреть за изменениями в проекте. Также можно объединять различные версии кода в один при этом работа в этой ветке не затрагивает основную ветку, что очень удобно, если код не будет работать то основной проект не пострадает. А когда новую ветку можно будет доработать то имеется возможность наложить её на основную.

**Контрольные вопросы**

1. Что представляет собой система GIT?

Это система управления версиями, которая позволяет отслеживать изменения в файловой системе и работы с историей этих изменений. Она позволяет командам разработчиков синхронизировать свою работу и управлять версиями кода.

2. Перечислите возможности систем управления версиями.

* Отслеживание истории изменений файлов и кода;
* Создание новых версий файлов или проектов;
* Переход между разными версиями файлов;
* Слияние изменений с разных веток или ветвей разработки;
* Разрешение конфликтов при объединении изменений;
* Восстановление удаленных файлов или версий.

3. Что такое коммит и когда он выполняется?

Коммит в системе GIT представляет собой сохранение изменений, сделанных в локальном репозитории. Коммит создается, когда разработчик подтверждает или сохраняет свои изменения в исходном коде или файлах проекта. Коммит выполняется, чтобы создать точку возврата и сохранить текущую версию кода, которую можно восстановить в будущем.

4. Как посмотреть историю коммитов?

Для просмотра истории коммитов нужно использовать команду "git log".

5. Как создать пустой GIT репозиторий?

Для создания пустого GIT репозитория нужно перейти в папку, где вы хотите создать репозиторий, с помощью команды "cd". Затем выполните команду "git init".

6. Какую архитектуру имеет система контроля версий GIT?

Система контроля версий GIT имеет распределенную архитектуру.

7. Как создать новую ветку в GIT и соединить ветки?

Чтобы создать новую ветку в GIT, используйте команду "git branch <имя ветки>". Затем, чтобы переключиться на новую ветку, используйте команду "git checkout <имя ветки>". Чтобы соединить ветки, используйте команду "git merge <имя ветки>", которая объединит изменения из выбранной ветки в текущую ветку.

8. Поясните понятие «staging area».

Это промежуточная область, в которой разработчик может подготовить изменения для коммита.

9. Как сравнить версии файла в GIT?

Чтобы сравнить версии файла в GIT, нужно использовать команду "git diff". Команда покажет различия между файлами, такие как добавленные, удаленные или измененные строки кода.

# **Вопросы к лабораторной работе**

1. В чем заключается удобство использования системы контроля версий Git?

Одним из удобств использования системы контроля версий Git является возможность отслеживать историю изменений файлов, что помогает в управлении и откате к предыдущим версиям. Также, Git позволяет работать с удаленными репозиториями, облегчая командную работу и совместный доступ к проекту.

1. Как изменить формат выводимой истории версий?

Формат выводимой истории версий можно изменить с помощью команды git log и различных опций, таких как --oneline, --pretty, --graph и других. Например, команда git log --oneline выводит краткую информацию о коммитах в одной строке.

1. Как сократить время набора команд в системе Git?

Формат выводимой истории версий можно изменить с помощью команды git log и различных опций, таких как --oneline, --pretty, --graph и других. Например, команда git log --oneline выводит краткую информацию о коммитах в одной строке.

1. Если необходимо выделить в истории группы версий, с определенными изменениями, как удобнее всего это сделать?

Для выделения в истории группы версий с определенными изменениями лучше всего использовать ветки. Создание веток позволяет работать над определенным функционалом или исправлением, не затрагивая основную ветку разработки.

1. Что такое тегирование, и для чего его целесообразно использовать?

Это присваивание именованной метки коммитам, что помогает в организации и обозначении важных точек в истории проекта, таких, как релизы или основные версии программного обеспечения.

1. Как можно исправить коммит, если он оказался неверным?

Для исправления коммита, если он оказался неверным, можно использовать команду git commit --amend. Эта команда позволяет внести изменения в последний коммит либо добавить новые изменения к последнему коммиту.

1. Для чего используется ветвление?

Для исправления коммита, если он оказался неверным, можно использовать команду git commit --amend. Эта команда позволяет внести изменения в последний коммит либо добавить новые изменения к последнему коммиту.

1. В каком случае необходимо исправлять файлы, при слиянии веток?

В случае слияния веток, могут возникнуть конфликты, которые нужно разрешить вручную в файле, объединяющем изменения. При разрешении конфликтов необходимо внимательно проанализировать изменения и выбрать правильное сочетание изменений из двух веток.

1. Что такое алиасы? Для чего они применяются?

Алиасы - это сокращенные формы команд Git, которые можно настроить в конфигурационном файле git. Использование алиасов позволяет сократить время набора команд и повысить эффективность работы с Git.

1. Что такое режим «одной строки»?

Режим "одной строки" (single-line mode) в Git отображает информацию о коммитах в одной строке, что сокращает пространство на экране и упрощает навигацию по истории версий. Этот режим может быть полезен при просмотре больших объемов коммитов или при работе с командной строкой.