Лабораторная работа №3

Основы работы с системой управления версиями Git.

1. Создание локального репозитория Git с использованием командной строки.

1.1. Открыть проект графического приложения и сделать его активным.

1.2. Запустить интерпретатор командной строки, выбрав пункт "Открыть консоль в этом каталоге - Системная среда" в локальном меню файла конфигурации проекта. Убедиться, что текущий рабочий каталог, отображаемый в приглашении командной строки, соответствует каталогу проекта.

1.3. Создать в каталоге проекта пустой репозиторий, выполнив команду:

```
git init
```

1.4. Проверить наличие каталога с данными репозитория. Для этого выполнить команду

```
dir /a
```

и убедиться, что в текущем каталоге существует скрытый каталог `.git`.

1.5. Проверить текущее состояние репозитория, выполнив команду

```
git status
```

Все файлы в каталоге проекта должны отображаться как несопровождаемые. Обратите внимание, что файлы модуля работы с простыми числами отсутствуют в списке (так как находятся в другом каталоге).

1.6. Установить имя и адрес электронной почты для пользователя репозитория. Для этого выполнить команды

```
git config user.name "имя_пользователя"
git config user.email "адрес_электронной_почты"
```

1.7. Проверить настройки репозитория, выполнив команду

```
git config --list
```

Значения параметров `user.name` и `user.email` должны совпадать с введенными.

1.8. Исключить файлы конфигурации среды QtCreator (с суффиксом в имени `.pro.user`) из процесса управления версиями. Для этого открыть в текстовом редакторе файл `.git/info/exclude` и добавить в него строку

```
*.pro.user
```

1.9. Проверить текущее состояние репозитория и убедиться, что файл конфигурации среды больше не отображаются в списке доступных.

2. Добавление файлов в систему управления версиями.

2.1. Подготовить все файлы проекта к фиксации, выполнив команду

```
git add .
```

2.2. Проверить состояние репозитория. Все файлы проекта должны иметь состояние "new file".

2.2. Поместить выбранные файлы в репозиторий, выполнив команду

```
git commit -m "initial commit"
```

2.3. Проверить состояние репозитория. В каталоге больше не должно быть подлежащих фиксации файлов.

2.4. Вывести журнал репозитория, выполнив команду

```
git log
```

В журнале должен быть единственный коммит с комментарием "initial commit". Обратите внимание, что для него установлены ранее заданные атрибуты автора.

2.5. Удаление локального репозитория. Для удаления репозитория и всей информации в нем достаточно удалить каталог .git. Выполните команду

```
rmdir /s .git
```

2.6. Убедитесь, что репозиторий больше не сушествует, выполнив команду

```
git status
```

3. Создание репозитория git с использованием IDE QtCreator.

3.1. Открыть проект в QtCreator.

3.2. В главном меню выполнить команду "Инструменты/Git/Создать хранилище" Убедиться, что в открывшемся диалоговом окне выбран каталог активного проекта, и подтвердить выбор.

3.3. Убедиться, что в окне сообщений системы контроля версий появилось сообщение с указанием выполненной команды и ее результата.

3.4. Повторить пункты 1.6 — 1.7.

4. Добавление файлов в репозиторий в IDE QtCreator.

4.1. В главном меню выполнить команду Инструменты/Git/Локальное хранилище/Фиксировать.

4.2. В открывшемся диалоговом окне выбрать все файлы проекта, в поле "Comment" вписать комментарий для коммита, и нажать кнопку "Фиксировать".

4.3. Убедиться, что в окне сообщений системы контроля версий появилось сообщение об успешной фиксации.

4.4. Выполнить команду Инструменты/Git/Локальное хранилище/История. В открывшемся окне убедиться, что в хранилище присутствует один коммит.

4.5. Просмотреть подробности сделанных изменений. Для этого щелкнуть левой кнопкой мыши по идентификатору коммита.

5. Работа с индивидуальными файлами в репозитории.

5.1. Добавить в один из файлов исходного кода проекта произвольный комментарий и сохранить изменения.

5.2. Подготовить текущий файл к помещению в репозиторий. В главном меню выполнить команду Инструменты/Git/Текущий файл/Подготовить.

5.3. Выполнить команду Инструменты/Git/Локальное хранилище/Состояние. Результат выполнения команды отображается в окне сообщений системы контроля версий. Убедиться, что измененный файл отображается как подготовленный к помещению в репозиторий.

5.4. Отменить подготовку файла к коммиту. Для этого выполнить команду "Инструменты/Git/Текущий файл/Не фиксировать". Проконтролировать состояние репозитория, как указано в 5.3 и убедиться, что подготовка к фиксации отменена.

5.5. Отменить сделанные изменения, восстановив версию файла, сохраненную в репозитории. Для этого выполнить команду "Инструменты/Git/Текущий файл/Отменить незафиксированные изменения". В открывшемся диалоговом окне подтвердить изменение файла.

6. Добавить в метод calculate() класса главного окна проверку значений введенных параметров.

6.1. Добавить объявления двух логических переменных, например:

```
bool isAOK, isBOK;
```

6.2. Модифицировать вызов функций toInt(), добавив указатели на эти переменные в качестве параметра:

```
int a = ui->lineEditA->text()->toInt(&isAOK);
int b = ui->lineEditB->text()->toInt(&isBOK);
```

По завершении работы функций в переменных будет содержаться значение true, если исходная строка является представлением целого числа, и false – если это не так.

6.3. Подключить заголовок модуля QMessageBox и добавить условные операторы проверки значений логических переменных таким образом, чтобы при ложном значении каждой из переменных выводилось модальное сообщение об ошибке, например:

```
if(!isAOK) {
     QMessageBox::critical(this, "Ошибка", "A должно быть числом");
     return;
}
if(!isBOK) {
     QMessageBox::critical(this, "Ошибка", "B должно быть числом");
     return;
}
// вызов функции получения списка простых чисел
```

6.4. Выполнить сборку и запуск и проверить работу программы во всех возможных сценариях.

6.5. Зафиксировать сделанные изменения в репозитории.