02_基本操作

Git操作（再掲）

- プロジェクトフォルダを開く
- 初期化する
- 「.gitignore」ファイルを作成する。
- ファイルを作成・編集・保存
- Commitしてgitで管理する

練習2（再掲・一部修正）

1. 初期設定

1. 02フォルダで作業する。
2. gitを初期化する。
3. 「.gitignore」を作成する。

2. ファイル作成とコミット

1. 「rensyuu_1.txt」を作成し、以下の内容を入力して保存する。

   ```
   サンプルの文書

   新型コロナウイルスの感染拡大を受けた政府の緊急経済対策の柱で、国民一律に給付される１０万円について、
   もっとも早い自治体では、来月７日にも給付できるよう準備を進めていることがわかりました。
   ```

2. コミットする。
3. 「rensyuu_2.txt」を作成し、以下の内容を入力して保存する。

   ```
   サンプル文書2
   吾輩は猫である。名前はまだ無い。
   　どこで生れたかとんと見当つかぬ。
   ```

4. コミットする。

3. ファイル修正とコミット

1. 「rensyuu_1.txt」を以下の内容に修正して保存する。

```
Yahooニュース

新型コロナウイルスの感染拡大を受けた政府の緊急経済対策の柱で、国民一律に給付される１０万円について、
もっとも早い自治体では、来月７日にも給付できるよう準備を進めていることがわかりました。
```

2. 「rensyuu_2.txt」を以下の内容に修正して保存する。

```
吾輩は猫である
夏目漱石
吾輩は猫である。名前はまだ無い。
　どこで生れたかとんと見当がつかぬ。
何でも薄暗いじめじめした所でニャーニャー泣いていた事だけは記憶している。
```

3. 両方をコミットする。

4. 再修正とファイル削除

1. 「rensyuu_2.txt」を以下の内容に再修正して保存する。

```
吾輩は猫である

夏目漱石

吾輩は猫である。名前はまだ無い。
　どこで生れたかとんと見当がつかぬ。
何でも薄暗いじめじめした所でニャーニャー泣いていた事だけは記憶している。
吾輩はここで始めて人間というものを見た。
```

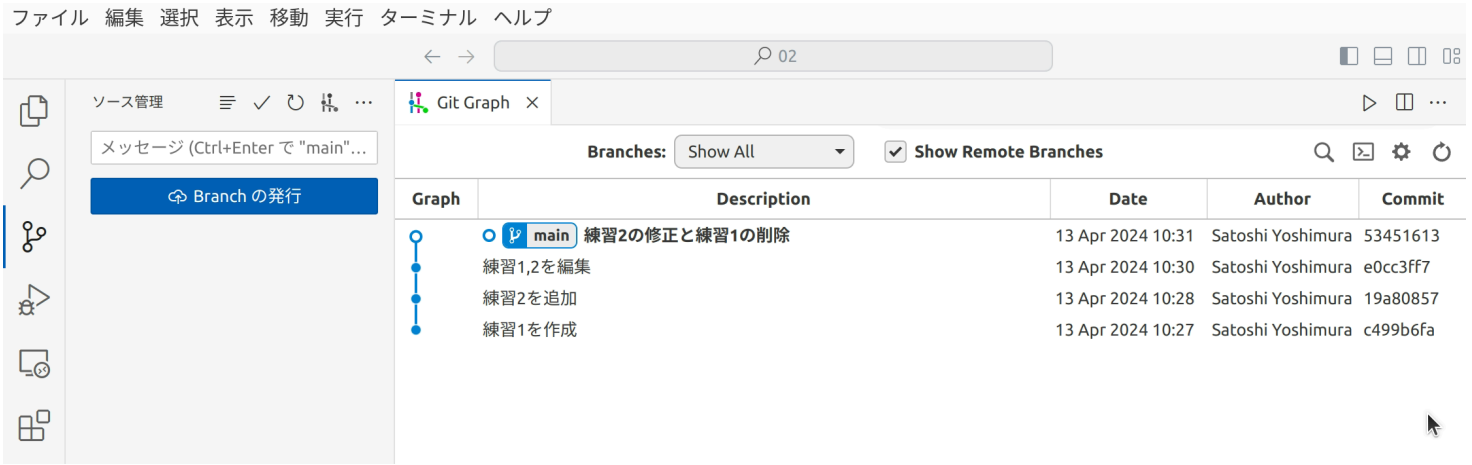
2. 「rensyuu_1.txt」を削除する。
3. コミットする。

5. 状態確認

1. gitの状況を確認する。
2. 各コミットをcheckoutして状態を確認する。

6. 実行結果

- 以下のようになっているだろうか？



- 一番最初の状態（図では「練習1を作成」）をcheckoutする。
    - 「rensyuu_1.txt」が存在しているのを確認する。
- 最後の状態をcheckoutする。
    - 「rensyuu_1.txt」が削除されているのを確認する。
- 途中のコミットもチェックアウトして確認する。

インタープリタの使い方

- コマンドプロンプト（Windows Terminal）で実行

```
> python ↵
Python 3.9.15 (main, Nov 24 2022, 14:31:59)
[GCC 11.2.0] :: Anaconda, Inc. on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

    - 先頭に `>>>` や `...` が表示されているのは、対話モードを意味する。
    - 終了するには、以下の何れかを入力
        - `exit()↵`
        - `quit()↵`
        - `ctrl` + `z`

- ファイルに記述したプログラムを実行

```
> python 2_1_1_argv.py
ファイル名: 2_1_1_argv.py
```

  - プログラム実行後、pythonを終了し、コマンド入力に戻る

- ファイルに記述したプログラムを実行後、対話モードに入る
  `-i` をファイル名の前に記述する

```
> python -i 2_1_1_argv.py
ファイル名: 2_1_1_argv.py
>>>
```

- 引数を渡す場合は、次のように実行する

```
> python -i 2_1_1_argv.py test sample 012
ファイル名: 2_1_1_argv.py
```

  - 対話モードで、実行中のプログラムを確認をしてみよう。

```
>>> sys
<module 'sys' (built-in)>
>>> sys.argv
['2_1_1_argv.py', 'test', 'sample', '012']
>>> sys.argv[0]
'2_1_1_argv.py'
>>> sys.argv[1]
'test'
>>>
```

練習1

- 2_1_2_var.pyを実行後、対話モードに入って操作してみよう

```
i = 10
j = 20
s = 'string'
ary = [2, 4, 6, 8]
```

- 対話モードで、変数の内容を確認する場合は、変数名を指示すればOK
- 計算なども行ってみよう

調査

1. 日本語Windowsが標準で使用している文字エンコーディングは何ですか？
2. vscodeの標準の文字エンコーディングは何ですか？
3. cp932とShift-JISの違いは何ですか？
4. UNICODEとUTF-8の違いは何ですか？

> https://docs.python.org/3/library/codecs.html#standard-encodings

練習2

- 以下のプログラムを実行してください。
    - 2_2_1_utf8.py
- このファイルをshift-jisで保存したものが、以下のファイルです。同様に実行してください。
    - 2_2_1_sjis.py
    - このファイルは実行することができない。（エラーが発生）
- 上記の `2_2_1_sjis.py` を実行できるようにしたものが、以下のファイル。同様に実行してください。
    - 2_2_1_sjis2.py

- vscodeでは、文字コードを指定して開くことができます。
    - 操作方法を確認してください。
- vscodeでは、文字コードを指定して、保存することができます。
    - 操作方法を確認してください。

練習3

- gitのBranch（ブランチ）について、調べてください。
    - どのような働きなのか？
    - どのように使用するのか？
- 以下を実際に行ってください。

1. ファイル：sample.txt を作成

```
sampleファイル
1,あいうえお
2,かきくけこ
3,さしすせそ
```

   - 保存・コミット

2. ブランチ「案１」を作成

   - ファイル「sample.txt」に追記

   ```
   # 案1
   4,たちつてと
   5,なにぬねの
   ```

   - 保存・コミット

3. mainブランチに戻る（checkout）

   - ファイル：sample2.txtを作成

   ```
   # 案2
   6,はひふへほ
   7,まみむめも
   ```

   - 保存・コミット

4. sample2.txtに追記

```
# 案3
8,やゆよ
9,らりるれろ
```

   - 保存・コミット

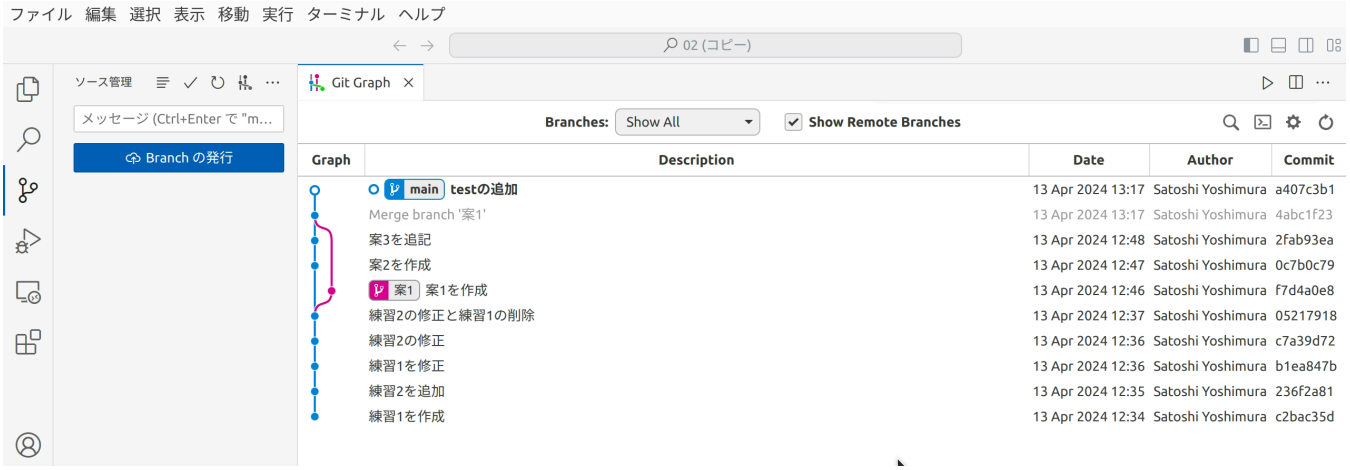
5. 案1のブランチとマージする

   - mainをcheckoutしている
   - 案1を選択し「Merge into current branch...」を選択

6. test.txtを作成する

```
test
```

    - 保存・コミット
7. これまでの結果



このようになっていない場合は、何度も繰り返して練習してみてください。

調査

- 以下の事項について調べてください。
    - `git branch` でブランチ一覧を確認する方法。
    - `git checkout -b <branch_name>` で新しいブランチを作成して切り替える方法。
    - マージ時の競合が発生した場合の対処方法。