Để giúp bạn nghiên cứu tốt về Git về mặt lý thuyết và thực hành. Biến máy bạn thành một Repository.

Làm cách nào? → Tạo thư mục rỗng, thực hiện lệnh command line tại thư mục đó: **git init**
Thư mục bạn vừa tạo đã trở thành một Repository và bắt đầu tìm hiểu Git

Online documents:

Tiếng Anh: https://git-scm.com/book/en/v2

Chú ý:

- *Bạn phải cài đặt Git và thiết lập môi trường trước khi tìm hiểu về các khái niệm và tính năng trong Git*
- *Sử dụng command line*
- *Bạn nên ghi chú lại những lệnh git đã tìm hiểu để sử dụng sau này*

1. **Repository** là gì? Phân biệt **Remote** repository và **Local** repository?
2. **Working directory** hay **Working tree** là gì? **Index** là gì?
3. **Clone** là gì? Để thực hiện clone cần yếu tố gì và sử dụng câu lệnh gì?
4. Phân biệt các trạng thái của file sau:
   - Committed
   - Unmodified - Modified
   - Untracked
   - Unstaged – Staged

   Và sử dụng lệnh gì để kiểm tra trạng thái file hiện tại?
5. Cách để thêm (add) một file vào trạng thái Staged là gì? Cách hủy bỏ chức năng add vừa rồi là gì? (undo add)
6. Lệnh ***git status -s*** để xem trạng thái của file dưới dạng ngắn gọn. Giải thích ý nghĩa của các ký hiệu trạng thái trước tên file:

   M a.txt
   M b.txt
   MM c.txt
   A d.txt
   AM e.txt
   ?? f.txt
7. Giải thích ý nghĩa của lệnh ***git diff*** và ***git diff --staged***
8. **Commit** là gì? Câu lệnh đơn giản để thực hiện Commit là gì?
9. Giải thích ý nghĩa của lệnh sau:
   - ***git rm -f file.txt***
   - ***git rm -cached file.txt***

   (biết rằng ***file.txt*** không phải là trạng thái untracked)
10. Nêu các cách thay đổi tên file và di chuyển file?

11. Mục đích **Ignore** là gì? Nêu cách Ignore file hoặc folder?
12. Giải thích ý nghĩa của lệnh sau:
    - ***git log***
    - ***git log --oneline***
    - ***git show***
    - ***gitk***
13. Giải thích ý nghĩa của lệnh sau:
    - ***git remote add origin <url>***
    - ***git remote add <shortname> <url>***
    - ***git remote -v***
    - ***git remote remove <short name>***
14. **Branch** là gì? Mục đích sử dụng Branch là gì?
15. Nêu cách thực hiện:
    - Liệt kê các branch hiện tại
    - Tạo mới một branch
    - Chuyển tới một branch để làm việc
    - Xóa branch
    - Đổi tên branch
16. **Merge** là gì? Nêu cách thực hiện
17. **Conflict** là gì? Conflict xảy ra trong những trường hợp nào và nêu cách giải quyết?
18. Bạn có thể làm việc trên nhánh master không? Vì sao? Nếu bạn được giao làm một tính năng trong game, bạn sẽ sử dụng quy trình làm việc của git như thế nào?
19. So sánh lệnh **Fetch** và **Pull**?
20. **Push** là gì?
21. **Merge request** là gì? Sử dụng khi nào?
22. **Revert** là gì? Nêu cách hủy bỏ (revert) những gì đang thay đổi của một file, một số file, toàn bộ dự án?
23. Giả sử có các commit theo thứ tự thời gian từ mới đến cũ:

    commit_Id_4
    commit_Id_3
    commit_Id_2
    commit_Id_1

    - Làm thế nào để revert tới ***commit_Id_2***?
    - Làm thế nào để revert những gì đã thay đổi trong ***commit_Id_1***?
24. Phân biệt
    - ***git reset --hard***
    - ***git reset --soft***

    - ***git reset –mix***
    - ***git reset***

25. Giả sử Repository của bạn có 3 branch: ***master, issue1, issue2***. Bạn đang ở ***issue1*** để làm việc và đang có những sự thanh đổi dang dở. Trong khi đó ***issue2*** cần ưu tiên để hoàn thiện trước. Làm thế nào để lưu lại những gì đang làm ở ***issue1*** và chuyển qua ***issue2***, sau đó trở lại ***issue1*** để tiếp tục với những gì đang làm?
26. Giải thích ý nghĩa của lệnh sau:
    - ***git clean -f***
    - ***git clean -f -d***
    - ***git clean -n -d***
27. **Rebase** là gì? Nên sử dụng rebase trong trường hợp nào?