系统开发工具基础实验报告

题目： 版本控制 (git)

学生姓名	周洋迅　学号 24020007175
学部、学院 (中心)	信息科学与工程学部
专业	计算机科学与技术
日期	2025 年 8 月 29 日
github 链接	https://github.com/zysgusg/SysDevelopmentTools

中国海洋大学

目录

1 练习内容

1.1 基础

1. git help <command>: 获取 git 命令的帮助信息
2. git init: 创建一个新的 git 仓库，其数据会存放在一个名为.git 的目录下
3. git status：显示当前的仓库状态
4. git add <filename>: 添加文件到暂存区
5. git commit: 创建一个新的提交
6. git log: 显示历史日志
7. git diff <filename>: 显示与暂存区文件的差异
8. git checkout <revision>: 更新 HEAD（如果是检出分支则同时更新当前分支）

1.2 合并与分支

9. git branch: 显示分支
10. git branch <name>: 创建分支
11. git checkout -b <name>: 创建分支并切换到该分支; 相当于 git branch <name>; git checkout <name>
12. git merge <revision>: 合并到当前分支

1.3 远端操作

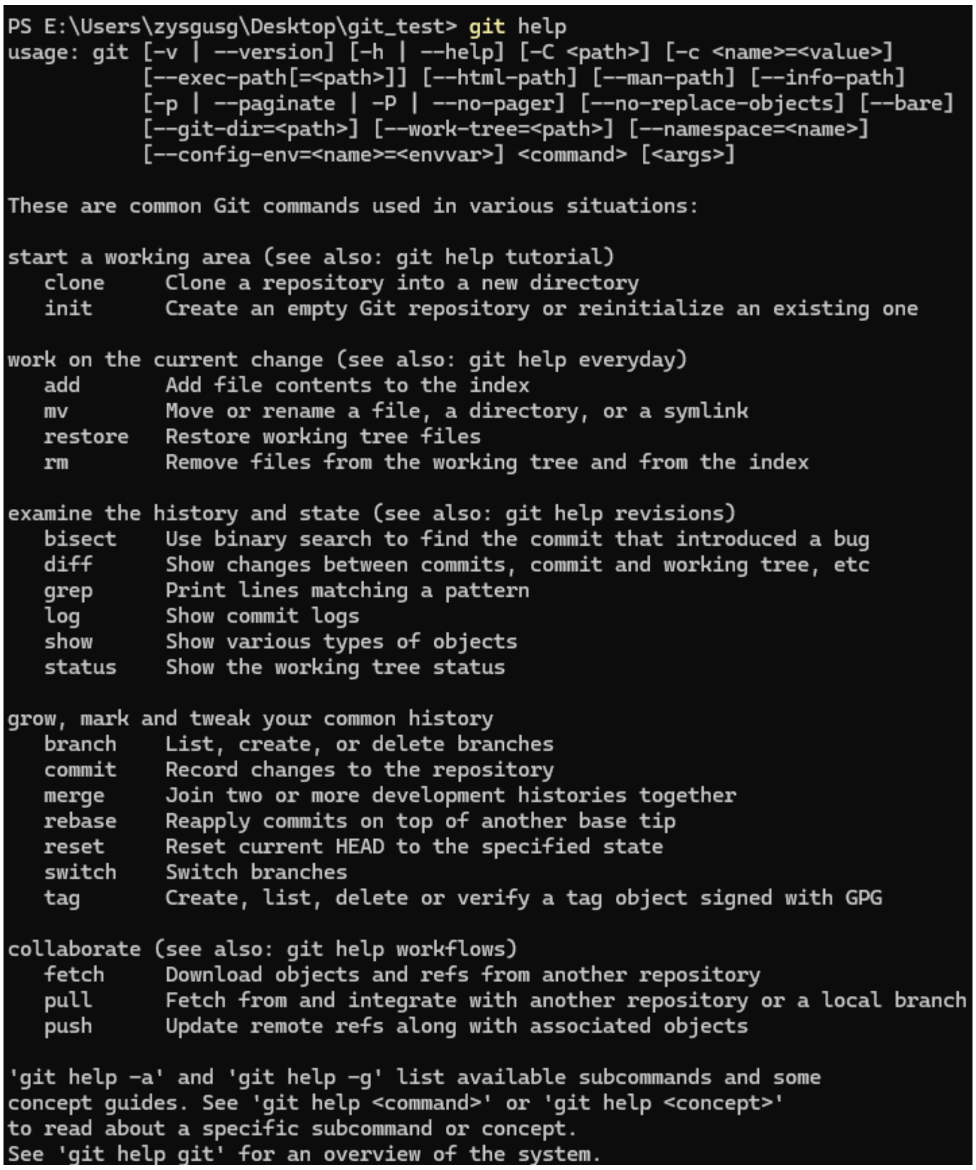
13. git remote: 列出远端
14. git remote add <name> <url>: 添加一个远端
15. git push <remote> <local branch>:<remote branch>: 将对象传送至远端并更新远端引用
16. git branch –set-upstream-to=<remote>/<remote branch>: 创建本地和远端分支的关联关系
17. git fetch: 从远端获取对象/索引
18. git pull: 相当于 git fetch; git merge
19. git clone: 从远端下载仓库

1.4 撤销

20. git commit –amend: 编辑提交的内容或信息
21. git reset HEAD <file>: 恢复暂存的文件
22. git checkout – <file>: 丢弃修改

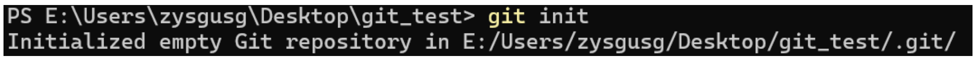
2 练习结果

2.1 结果截图



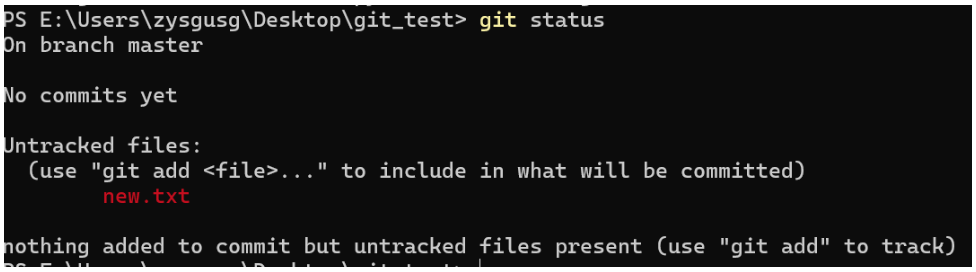
```
PS E:\Users\zysgusg\Desktop\git_test> git help
usage: git [-v | --version] [-h | --help] [-C <path>] [-c <name>=<value>]
           [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
           [-p | --paginate | -P | --no-pager] [--no-replace-objects] [--bare]
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
           [--config-env=<name>=<envvar>] <command> [<args>]

These are common Git commands used in various situations:

start a working area (see also: git help tutorial)
   clone     Clone a repository into a new directory
   init      Create an empty Git repository or reinitialize an existing one

work on the current change (see also: git help everyday)
   add       Add file contents to the index
   mv        Move or rename a file, a directory, or a symlink
   restore   Restore working tree files
   rm        Remove files from the working tree and from the index

examine the history and state (see also: git help revisions)
   bisect    Use binary search to find the commit that introduced a bug
   diff      Show changes between commits, commit and working tree, etc
   grep      Print lines matching a pattern
   log       Show commit logs
   show      Show various types of objects
   status    Show the working tree status

grow, mark and tweak your common history
   branch    List, create, or delete branches
   commit    Record changes to the repository
   merge     Join two or more development histories together
   rebase    Reapply commits on top of another base tip
   reset     Reset current HEAD to the specified state
   switch    Switch branches
   tag       Create, list, delete or verify a tag object signed with GPG

collaborate (see also: git help workflows)
   fetch     Download objects and refs from another repository
   pull      Fetch from and integrate with another repository or a local branch
   push      Update remote refs along with associated objects

'git help -a' and 'git help -g' list available subcommands and some
concept guides. See 'git help <command>' or 'git help <concept>'
to read about a specific subcommand or concept.
See 'git help git' for an overview of the system.
```

图 1: git help



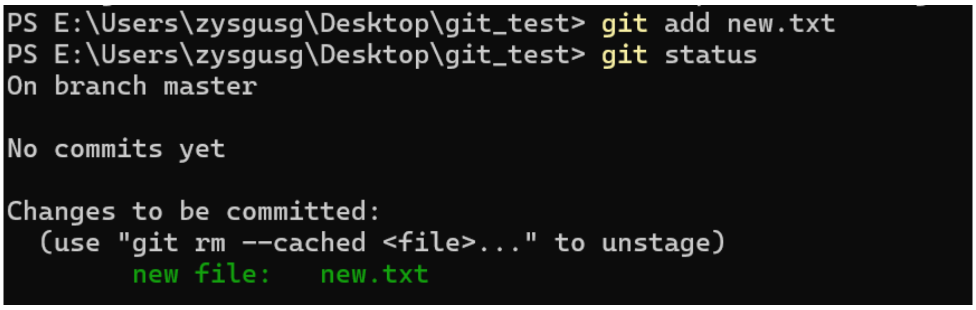
```
PS E:\Users\zysgusg\Desktop\git_test> git init
Initialized empty Git repository in E:/Users/zysgusg/Desktop/git_test/.git/
```

图 2: git init



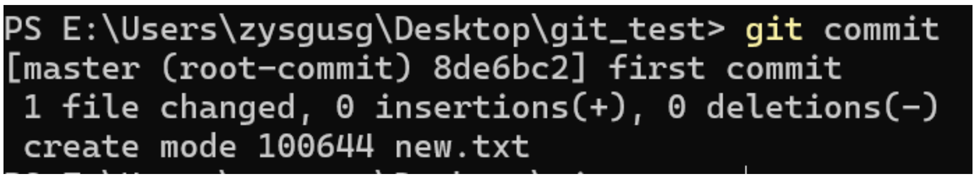
```
PS E:\Users\zysgusg\Desktop\git_test> git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        new.txt

nothing added to commit but untracked files present (use "git add" to track)
```

图 3: git status



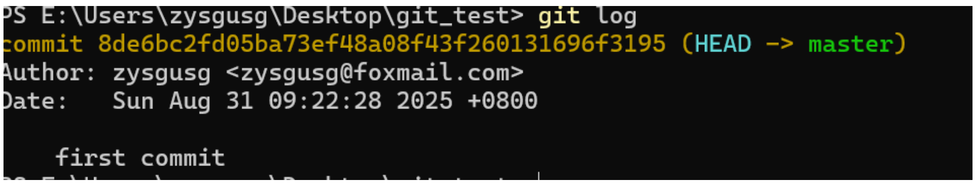
```
PS E:\Users\zysgusg\Desktop\git_test> git add new.txt
PS E:\Users\zysgusg\Desktop\git_test> git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   new.txt
```

图 4: git add



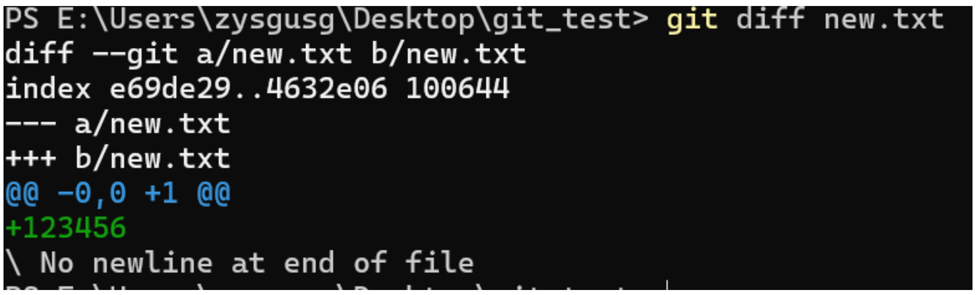
```
PS E:\Users\zysgusg\Desktop\git_test> git commit
[master (root-commit) 8de6bc2] first commit
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 new.txt
```

图 5: git commit



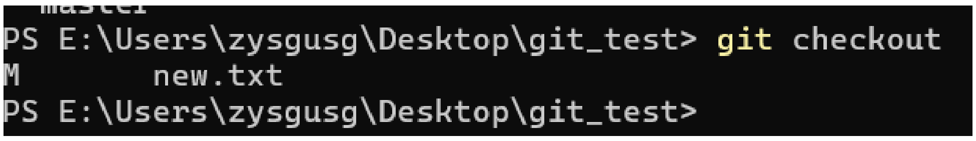
```
PS E:\Users\zysgusg\Desktop\git_test> git log
commit 8de6bc2fd05ba73ef48a08f43f260131696f3195 (HEAD -> master)
Author: zysgusg <zysgusg@foxmail.com>
Date:   Sun Aug 31 09:22:28 2025 +0800

    first commit
```

图 6: git log



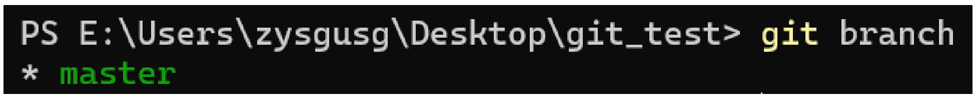
```
PS E:\Users\zysgusg\Desktop\git_test> git diff new.txt
diff --git a/new.txt b/new.txt
index e69de29..4632e06 100644
--- a/new.txt
+++ b/new.txt
@@ -0,0 +1 @@
+123456
\ No newline at end of file
```

图 7: git diff <filename>



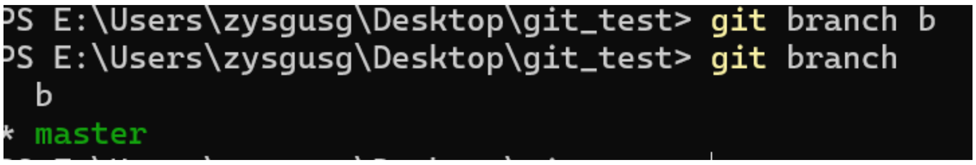
```
PS E:\Users\zysgusg\Desktop\git_test> git checkout
M       new.txt
PS E:\Users\zysgusg\Desktop\git_test>
```

图 8: git checkout <revision>



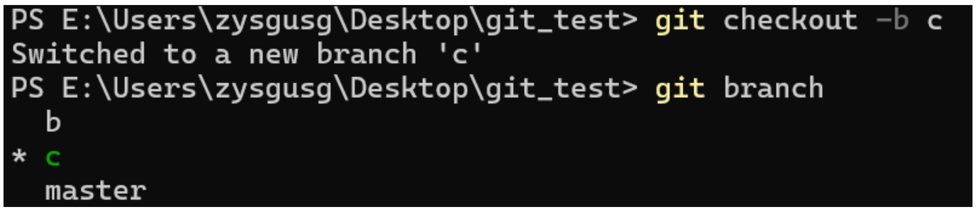
```
PS E:\Users\zysgusg\Desktop\git_test> git branch
* master
```

图 9: git branch



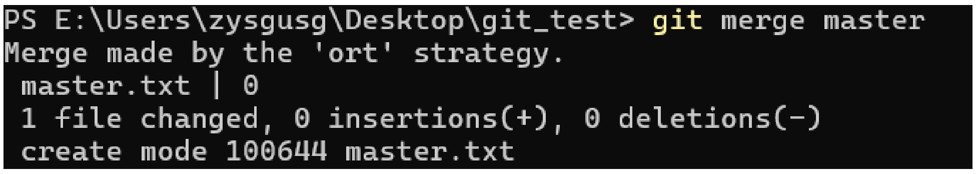
```
PS E:\Users\zysgusg\Desktop\git_test> git branch b
PS E:\Users\zysgusg\Desktop\git_test> git branch
  b
* master
```

图 10: git branch <name>



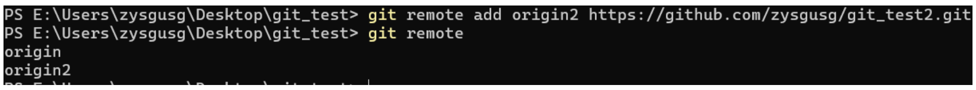
```
PS E:\Users\zysgusg\Desktop\git_test> git checkout -b c
Switched to a new branch 'c'
PS E:\Users\zysgusg\Desktop\git_test> git branch
  b
* c
  master
```

图 11: git checkout -b <name>



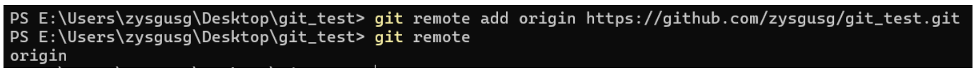
```
PS E:\Users\zysgusg\Desktop\git_test> git merge master
Merge made by the 'ort' strategy.
 master.txt | 0
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 master.txt
```

图 12: git merge <revision>



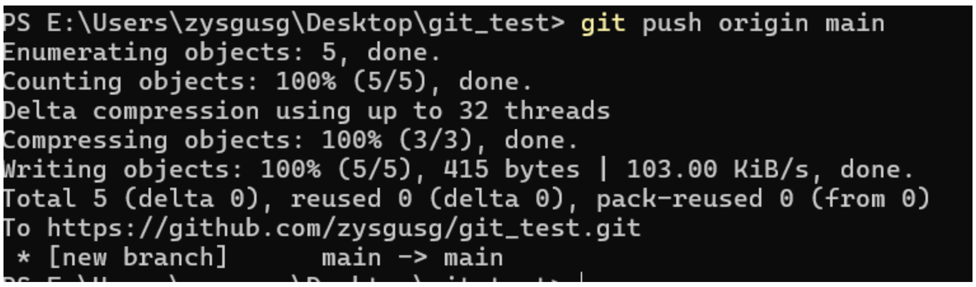
```
PS E:\Users\zysgusg\Desktop\git_test> git remote add origin2 https://github.com/zysgusg/git_test2.git
PS E:\Users\zysgusg\Desktop\git_test> git remote
origin
origin2
```

图 13: git remote



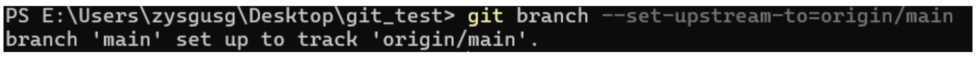
```
PS E:\Users\zysgusg\Desktop\git_test> git remote add origin https://github.com/zysgusg/git_test.git
PS E:\Users\zysgusg\Desktop\git_test> git remote
origin
```

图 14: git remote add <name> <url>



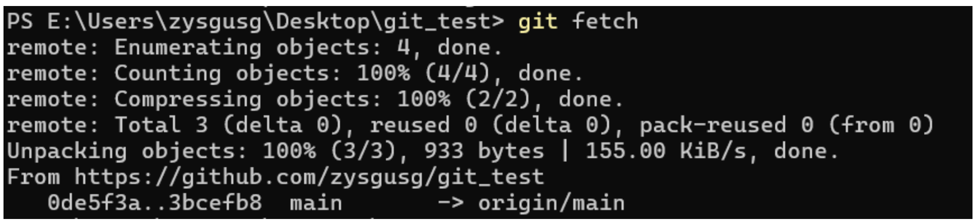
```
PS E:\Users\zysgusg\Desktop\git_test> git push origin main
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 32 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (5/5), 415 bytes | 103.00 KiB/s, done.
Total 5 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/zysgusg/git_test.git
 * [new branch]      main -> main
```

图 15: git push <remote> <local branch>:<remote branch>



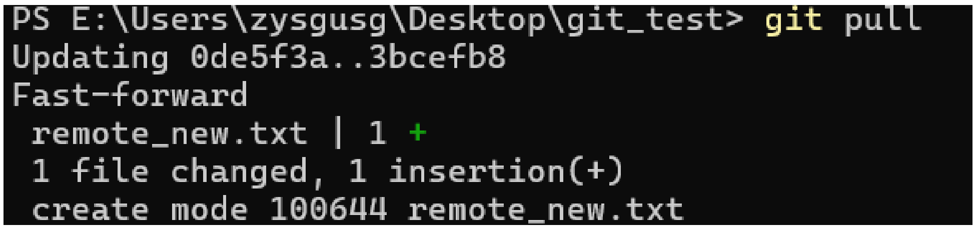
```
PS E:\Users\zysgusg\Desktop\git_test> git branch --set-upstream-to=origin/main
branch 'main' set up to track 'origin/main'.
```

图 16: git branch –set-upstream-to=<remote>/<remote branch>



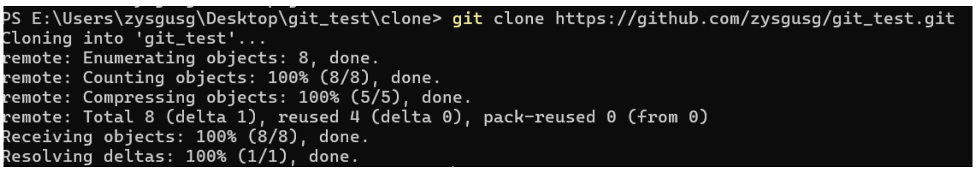
```
PS E:\Users\zysgusg\Desktop\git_test> git fetch
remote: Enumerating objects: 4, done.
remote: Counting objects: 100% (4/4), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (3/3), 933 bytes | 155.00 KiB/s, done.
From https://github.com/zysgusg/git_test
   0de5f3a..3bcefb8  main       -> origin/main
```

图 17: git fetch



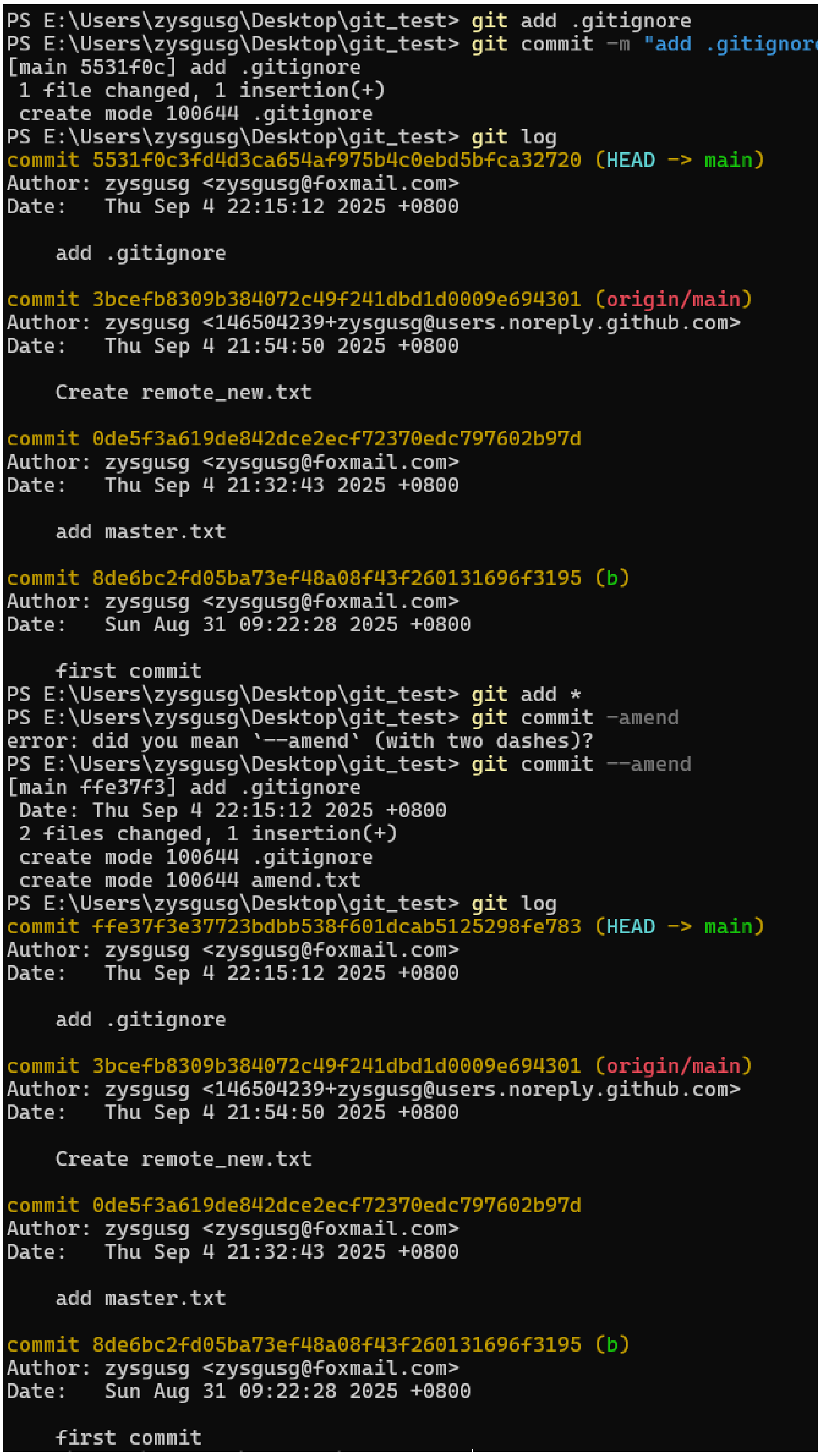
```
PS E:\Users\zysgusg\Desktop\git_test> git pull
Updating 0de5f3a..3bcefb8
Fast-forward
 remote_new.txt | 1 +
 1 file changed, 1 insertion(+)
 create mode 100644 remote_new.txt
```

图 18: git pull



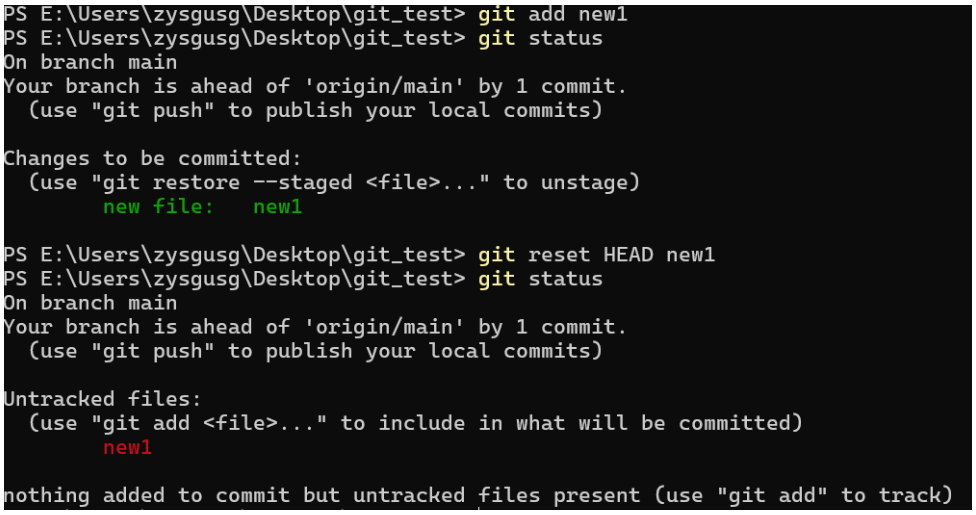
```
PS E:\Users\zysgusg\Desktop\git_test\clone> git clone https://github.com/zysgusg/git_test.git
Cloning into 'git_test'...
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 8 (delta 1), reused 4 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (8/8), done.
Resolving deltas: 100% (1/1), done.
```

图 19: git clone



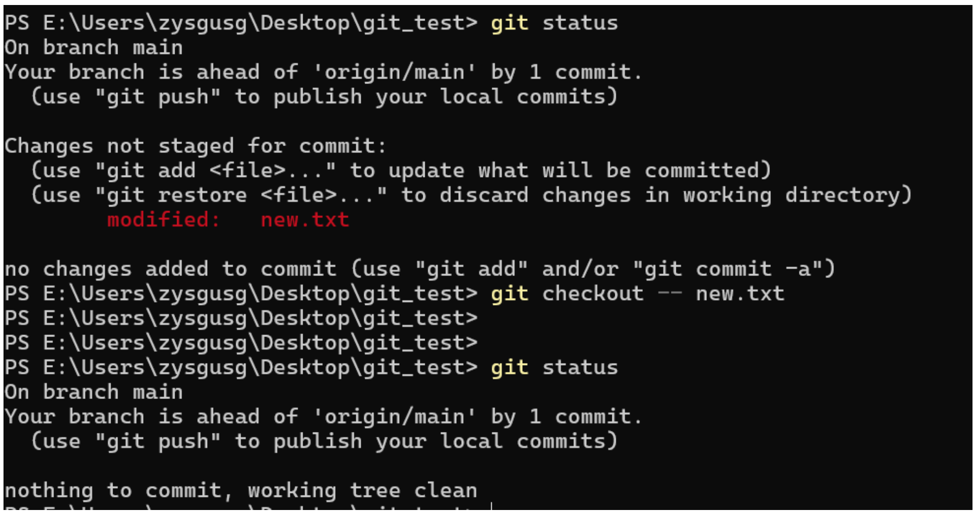
```
PS E:\Users\zysgusg\Desktop\git_test> git add .gitignore
PS E:\Users\zysgusg\Desktop\git_test> git commit -m "add .gitignor
[main 5531f0c] add .gitignore
 1 file changed, 1 insertion(+)
 create mode 100644 .gitignore
PS E:\Users\zysgusg\Desktop\git_test> git log
commit 5531f0c3fd4d3ca654af975b4c0ebd5bfca32720 (HEAD -> main)
Author: zysgusg <zysgusg@foxmail.com>
Date:   Thu Sep 4 22:15:12 2025 +0800

    add .gitignore

commit 3bcefb8309b384072c49f241dbd1d0009e694301 (origin/main)
Author: zysgusg <146504239+zysgusg@users.noreply.github.com>
Date:   Thu Sep 4 21:54:50 2025 +0800

    Create remote_new.txt

commit 0de5f3a619de842dce2ecf72370edc797602b97d
Author: zysgusg <zysgusg@foxmail.com>
Date:   Thu Sep 4 21:32:43 2025 +0800

    add master.txt

commit 8de6bc2fd05ba73ef48a08f43f260131696f3195 (b)
Author: zysgusg <zysgusg@foxmail.com>
Date:   Sun Aug 31 09:22:28 2025 +0800

    first commit
PS E:\Users\zysgusg\Desktop\git_test> git add *
PS E:\Users\zysgusg\Desktop\git_test> git commit -amend
error: did you mean `--amend` (with two dashes)?
PS E:\Users\zysgusg\Desktop\git_test> git commit --amend
[main ffe37f3] add .gitignore
 Date: Thu Sep 4 22:15:12 2025 +0800
 2 files changed, 1 insertion(+)
 create mode 100644 .gitignore
 create mode 100644 amend.txt
PS E:\Users\zysgusg\Desktop\git_test> git log
commit ffe37f3e37723bdbb538f601dcab5125298fe783 (HEAD -> main)
Author: zysgusg <zysgusg@foxmail.com>
Date:   Thu Sep 4 22:15:12 2025 +0800

    add .gitignore

commit 3bcefb8309b384072c49f241dbd1d0009e694301 (origin/main)
Author: zysgusg <146504239+zysgusg@users.noreply.github.com>
Date:   Thu Sep 4 21:54:50 2025 +0800

    Create remote_new.txt

commit 0de5f3a619de842dce2ecf72370edc797602b97d
Author: zysgusg <zysgusg@foxmail.com>
Date:   Thu Sep 4 21:32:43 2025 +0800

    add master.txt

commit 8de6bc2fd05ba73ef48a08f43f260131696f3195 (b)
Author: zysgusg <zysgusg@foxmail.com>
Date:   Sun Aug 31 09:22:28 2025 +0800

    first commit
```

图 20: git commit –amend



```
PS E:\Users\zysgusg\Desktop\git_test> git add new1
PS E:\Users\zysgusg\Desktop\git_test> git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   new1

PS E:\Users\zysgusg\Desktop\git_test> git reset HEAD new1
PS E:\Users\zysgusg\Desktop\git_test> git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        new1

nothing added to commit but untracked files present (use "git add" to track)
```

图 21: git reset HEAD <file>



```
PS E:\Users\zysgusg\Desktop\git_test> git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   new.txt

no changes added to commit (use "git add" and/or "git commit -a")
PS E:\Users\zysgusg\Desktop\git_test> git checkout -- new.txt
PS E:\Users\zysgusg\Desktop\git_test>
PS E:\Users\zysgusg\Desktop\git_test>
PS E:\Users\zysgusg\Desktop\git_test> git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```

图 22: git checkout – <file>

3 心得体会

- 通过本次实验，我学到了版本控制相关的知识，加深了对 git 命令及其实现机制的理解。