git 보충

About..

컴퓨터소프트웨어공학과
김 원 일

저장소의 최신 정보 유지

- **저장소 정보**
  - "git clone"**으로 복제한 저장소는 로컬 저장소에 별도 보관**
    - **원격 저장소에 내용이 추가되거나 업데이트가 발생한 경우**
    - **최신 내용을** "git clone"**으로 다시 복제하는 것은 효율성이 떨어짐**
  - "git pull" **명령어를 통해 원격 저장소의 내용을 로컬과 동기화 수행**
    - **협업 시에 항상 작업 전에는** "git pull"**로 변화 정보가 있는지 확인부터 수행하는 것이 좋음**

```
MINGW64:/f/202507001/testnew
unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$ git pull
remote: Enumerating objects: 30, done.
remote: Counting objects: 100% (30/30), done.
remote: Compressing objects: 100% (14/14), done.
remote: Total 26 (delta 13), reused 21 (delta 8), pack-reused 0 (from 0)
Unpacking objects: 100% (26/26), 2.52 KiB | 1024 bytes/s, done.
From https://github.com/ycs-wikim/testnew
   df77368..3d81af8  main       -> origin/main
 * [new branch]      new        -> origin/new
Updating df77368..3d81af8
Fast-forward
 3way        | 5 +++++
 list.txt    | 5 -----
 rebase.txt  | 2 ++
 sq.txt      | 3 +++
 4 files changed, 10 insertions(+), 5 deletions(-)
 create mode 100644 3way
 create mode 100644 sq.txt
```

파일 추적 범위 - 1

- **파일 수정 - 1**
  - 파일에 대한 수정은 내용 비교를 통해 항상 수행되고 있음
  - 필요에 의해 파일 이름 변경을 수행한 경우 아래와 같이 나타남
  - 이름 변경 전 파일은 삭제로, 이름 변경 후 파일을 새로운 파일로 나타남

```
MINGW64:/f/202507001/testnew
unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$ ls
3way      LICENSE    list.txt  new.txt      rebase.txt  squash.txt
3way.txt  README.md  main.txt  newfile.txt  sq.txt      work.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$ mv main.txt vi.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

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

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

no changes added to commit (use "git add" and/or "git commit -a")

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$
```

파일 추적 범위 - 2

- **파일 수정** - 2
  - **삭제된 것으로 처리된** main.txt**를** Staged **상태로 변경**
  - **새롭게 추가된** vi.txt**는 여전히** Untracked **상태로 유지**

```
MINGW64:/f/202507001/testnew

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$ git add main.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        deleted:    main.txt

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


unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$
```

파일 추적 범위 - 3

- **파일 수정** - 3
  - **새로 추가된** vi.txt **파일도** Staged **상태로 변경**
  - **파일 삭제와 신규 파일** 2**개로 분리 처리된 파일이 하나로 변경**
  - **최종적으로는** main.txt **파일이** vi.txt **파일로 이름 변경된 것으로 표기**
  - **따라서 파일의 이름 변경 또한 파일 이력으로 취급되어 관리됨**
  - vi.txt **파일 이력을 추적하는 경우, 이름 변경된 이력까지 연결됨**

```
MINGW64:/f/202507001/testnew
unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$ git add vi.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        renamed:    main.txt -> vi.txt


unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$
```

원격 저장소 정보

- **원격 저장소**
  - "git remote -v"**로 원격 저장소에 대한 기본 설정 정보 확인 가능**
    - **원격지이름 접근가능한실제**URL (**접근유형**) **형식으로 출력**
  - "git remote show **원격지이름**"**으로 상제 정보를 확인할 수 있음**

```
MINGW64:/f/202507001/testnew
unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$ git remote -v
origin  https://github.com/ycs-wikim/testnew (fetch)
origin  https://github.com/ycs-wikim/testnew (push)

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$ git remote show origin
* remote origin
  Fetch URL: https://github.com/ycs-wikim/testnew
  Push  URL: https://github.com/ycs-wikim/testnew
  HEAD branch: main
  Remote branches:
    main   tracked
    new    tracked
    rebase tracked
    work   tracked
  Local branches configured for 'git pull':
    main   merges with remote main
    rebase merges with remote rebase
    work   merges with remote work
  Local refs configured for 'git push':
    main   pushes to main   (up to date)
    rebase pushes to rebase (up to date)
    work   pushes to work   (local out of date)

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$
```

로컬 저장소의 원격 설정 - 1

- **원격 저장소 없는 로컬 저장소의 원격 연결**
  - **로컬 저장소를 원격 없이 "git init"으로 생성하여 관리하고 있는 경우**
  - **원격 저장소가 없기 때문에 공유나 다른 PC에서 작업이 불가능**
  - **로컬 저장소에 원격 저장소를 연결하여 사용할 수 있음**
  - **"git remote add 원격지이름 접근가능한실제URL.git"으로 연결 가능**
  - **로컬 저장소에서 저장소가 시작되었으므로 원격 저장소를 생성하고 연결**
  - **경로를 잘못 입력한 경우에는 원격 저장소에 접근이 불가능하므로 주의**

로컬 저장소의 원격 설정 - 2

- **이름 변경과 삭제**
  - "git remote rename **원격지이름 변경할원격지이름**"으로 원격지 이름변경
  - "git remote remove **삭제할원격지이름**"으로 원격지 정보 삭제
  - **원격지가 이미 존재하는 경우는** "git clone"**으로 복제하면 설정 문제 없음**
    - **로컬을 만들고 원격지를 설정해도 되지만 큰 의미는 없음**
  - **원격지에 연결한 다음에는 먼저** "git pull"**로 최신 정보를 받는 것이 좋음**

```
MINGW64:/f/202507001/reset

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/reset (master)
$ git remote rename origin test

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/reset (master)
$ git remote -v
test    https://github.com/ycs-wikim/reset.git (fetch)
test    https://github.com/ycs-wikim/reset.git (push)

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/reset (master)
$ git remote remove test

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/reset (master)
$ git remote -v

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/reset (master)
$
```

branch 이동

About..

컴퓨터소프트웨어공학과
김 원 일

브렌치 사용 주의사항 – 1

- **브렌치 이동 시 주의할 점 – 1**
  - **작업 중인 브렌치에 변경 사항이 발생하면 다른 브렌치 이동이 거부됨**
  - **현재 상태에 대한 변경 적용을 git이 판단할 수 없기 때문에 거부**
    - **변경을 적용해야 하는 것인지, 변경을 취소해야 하는 것인지에 대한 개발자 의도 판단 불가**

```
MINGW64:/d/202507001/testnew
YUHAN@YP12624115 MINGW64 /d/202507001/testnew (work)
$ git status
On branch work
nothing to commit, working tree clean

YUHAN@YP12624115 MINGW64 /d/202507001/testnew (work)
$ vi work.txt

YUHAN@YP12624115 MINGW64 /d/202507001/testnew (work)
$ git status
On branch work
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:   work.txt

no changes added to commit (use "git add" and/or "git commit -a")

YUHAN@YP12624115 MINGW64 /d/202507001/testnew (work)
$ git checkout main
error: Your local changes to the following files would be overwritten by checkout:
        work.txt
Please commit your changes or stash them before you switch branches.
Aborting

YUHAN@YP12624115 MINGW64 /d/202507001/testnew (work)
$
```

브렌치 사용 주의사항 – 2

- **브렌치 이동에서 주의할 점**
  - **저장소에 생성된 새로운 파일(Untracked)은 브렌치 이동에 문제 없음**
    - **추적 대상이 아니기 때문에 해당 파일이 계속 유지**
    - **필요 시에 관리 대상으로 지정 가능하지만 어느 브렌치에 추가한 것인지 알아야 함**

```
MINGW64:/d/202507001/testnew
YUHAN@YP12624115 MINGW64 /d/202507001/testnew (work)
$ git status
On branch work
nothing to commit, working tree clean

YUHAN@YP12624115 MINGW64 /d/202507001/testnew (work)
$ ls -al > new.txt

YUHAN@YP12624115 MINGW64 /d/202507001/testnew (work)
$ git status
On branch work
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)

YUHAN@YP12624115 MINGW64 /d/202507001/testnew (work)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

YUHAN@YP12624115 MINGW64 /d/202507001/testnew (main)
$ ls
LICENSE  README.md  list.txt  new.txt

YUHAN@YP12624115 MINGW64 /d/202507001/testnew (main)
$
```

브렌치 이동 - 1

- **이동 제약 조건**
  - **브렌치의 이동은 현재 브렌치에 내용이 모두 관리되고 있는 경우에 가능**
  - **관리되고 있는 파일이 수정된 경우에는 브렌치 이동이 불가능**
  - Commit**으로 모두 관리 상태로 만들거나** stash(**잠시 숨김**) **후 이동 가능**

```
MINGW64:/f/202507001/testnew
unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (rebase)
$ vi rebase.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (rebase)
$ git status
On branch rebase
Your branch is up to date with 'origin/rebase'.

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:   rebase.txt

no changes added to commit (use "git add" and/or "git commit -a")

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (rebase)
$ git switch main
error: Your local changes to the following files would be overwritten by checkout:
        rebase.txt
Please commit your changes or stash them before you switch branches.
Aborting
```

브렌치 이동 - 2

- **작업 숨김** - 1
  - **개발 도중 갑작스럽게 다른 브렌치 변경이 필요한 경우**
  - **현재 작업 중인 상태를** stash(**숨김**)**하고, 관리 상태로 변경 가능**
  - git status**로 확인하면 작업 디렉터리에 변경 사항이 없음으로 표시**
    - **작업 디렉터리에 변화가 없는 것으로 인식되어 브렌치 변경이 가능해짐**
  - "git stash list"**로 숨겨진 상태 목록을 확인할 수 있음**

```
MINGW64:/f/202507001/testnew
unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (rebase)
$ git stash
Saved working directory and index state WIP on rebase: df77368 rebase : add rebase.tx
t

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (rebase)
$ git stash list
stash@{0}: WIP on rebase: df77368 rebase : add rebase.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (rebase)
$ git status
On branch rebase
Your branch is up to date with 'origin/rebase'.

nothing to commit, working tree clean

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (rebase)
$
```

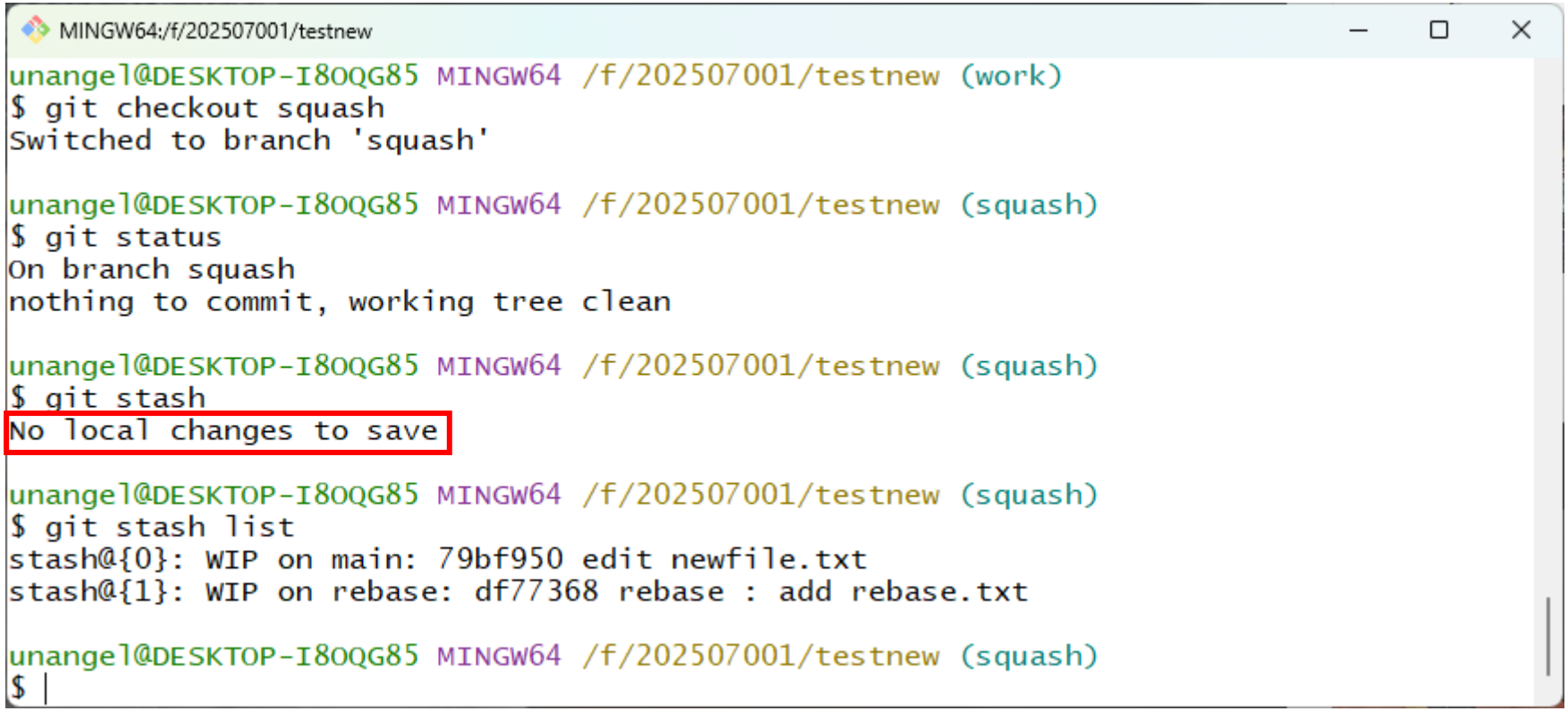
브렌치 이동 - 3

- **작업 숨김 - 2**
  - 작업 중인 브렌치에서는 어디서나 현재 작업을 숨길 수 있음
  - 목록 정보에서 어느 브렌치의 정보가 보관되어 있는지 정보 확인이 가능
  - 너무 많은 정보를 숨길 경우, 어떤 작업을 수행하고 있었는지 혼동할 수 있음
  - 급하게 브렌치를 변경하여 작업을 진행하는 경우에만 사용하는 것이 좋음

```
MINGW64:/f/202507001/testnew
unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (rebase)
$ git switch main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$ vi main.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$ git stash
Saved working directory and index state WIP on main: 79bf950 edit newfile.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$ git stash list
stash@{0}: WIP on main: 79bf950 edit newfile.txt
stash@{1}: WIP on rebase: df77368 rebase : add rebase.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$
```

브렌치 이동 - 4

- **작업 숨김** - 3
  - **작업 디렉터리에 변경 내역이 없는 경우에는** stash**가 동작하지 않음**
  - stash **목록에도 추가되지 않기 때문에 무분별한** stash **생성을 막음**
  - stash**는 저장될 때 마지막에** stash**한 내용이** 0**번으로 생성** (Stack **이용**)



브렌치 이동 - 5

- **메시지 보관**
  - stash **생성 시,** git **생성 메시지가 아니라 별도 메시지 적용이 가능**
  - git stash -m "**메시지 내용**" **형식으로 메시지를 지정할 수 있음**
  - **메시지에 작업 내역 또는 진행 상황 등에 대해 간단히 기록 가능**

```
MINGW64:/f/202507001/testnew
unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (squash)
$ git stash list
stash@{0}: WIP on main: 79bf950 edit newfile.txt
stash@{1}: WIP on rebase: df77368 rebase : add rebase.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (squash)
$ vi squash.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (squash)
$ git stash -m "main 브 렌 치  작 업 을  위 해  stash 생 성 "
Saved working directory and index state On squash: main 브 렌 치  작 업 을  위 해  stash 생 성

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (squash)
$ git stash list
stash@{0}: On squash: main 브 렌 치  작 업 을  위 해  stash 생 성
stash@{1}: WIP on main: 79bf950 edit newfile.txt
stash@{2}: WIP on rebase: df77368 rebase : add rebase.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (squash)
$
```

브렌치 이동 - 6

- stash **복원** - 1
  - "git stash apply"**로 임시 저장된 내용을 복원할 수 있음**
  - **스택 자료구조를 이용하기 때문에** top**의 내용을 현재 브렌치에 복원**
  - "git stash list"**에는 여전히 임시 저장된 정보가 존재하는 것을 확인**

```
MINGW64:/f/202507001/testnew
unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$ git stash list
stash@{0}: On squash: main 브 렌 치  작 업 을  위 해  stash 생 성
stash@{1}: WIP on main: 79bf950 edit newfile.txt
stash@{2}: WIP on rebase: df77368 rebase : add rebase.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$ git stash apply
On branch main
Your branch is up to date with 'origin/main'.

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:   squash.txt

no changes added to commit (use "git add" and/or "git commit -a")

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$ git stash list
stash@{0}: On squash: main 브 렌 치  작 업 을  위 해  stash 생 성
stash@{1}: WIP on main: 79bf950 edit newfile.txt
stash@{2}: WIP on rebase: df77368 rebase : add rebase.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$
```

브렌치 이동 - 7

- stash **복원** - 2
  - **복원은 현재 브렌치를 기준으로 수행하기 때문에 브렌치 정보에 주의**
  - squash **브렌치에서 임시 저장한 내용이** main **브렌치에 복원**
  - **의도하지 않은 작업이** main**에서 발생하였기 때문에 주의 필요**

```
MINGW64:/f/202507001/testnew
unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$ git stash list
stash@{0}: On squash: main 브렌치 작업을 위해 stash 생성
stash@{1}: WIP on main: 79bf950 edit newfile.txt
stash@{2}: WIP on rebase: df77368 rebase : add rebase.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$ git status
On branch main
Your branch is up to date with 'origin/main'.

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:   squash.txt

no changes added to commit (use "git add" and/or "git commit -a")

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$
```

브렌치 이동 - 8

- stash **복원** - 3
  - main**에서의 작업이 아니므로** restore **후** squash **브렌치로 이동**
  - **또 다른 복원 방법인** "git stash pop"**으로 복원을 수행**
    - **스택 자료구조에서** pop**은 마지막에 입력된 노드의 삭제의 의미**
  - apply**로 복원하면 임시 저장 정보가 남지만,** pop**으로 복원하면 삭제 발생**

```
MINGW64:/f/202507001/testnew
unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (main)
$ git switch squash
Switched to branch 'squash'

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (squash)
$ git stash pop
On branch squash
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:   squash.txt

no changes added to commit (use "git add" and/or "git commit -a")
Dropped refs/stash@{0} (6bbd1221494d8911a88bbe2dd7dc8069842d5b94)

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (squash)
$ git stash list
stash@{0}: WIP on main: 79bf950 edit newfile.txt
stash@{1}: WIP on rebase: df77368 rebase : add rebase.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (squash)
$
```

브렌치 이동 - 9

- **복원 정보 삭제**
  - "git stash drop"**으로 저장된 복원 정보를 삭제할 수 있음**
  - **스택 자료구조의 특성상 나중에 저장된** main **브렌치 복원 정보가 먼저 삭제됨**
  - **리스트에서 완전히 삭제되기 때문에 복원 정보 또한 완전 삭제**
  - **삭제된 이후에는 남아 있는** rebase **브렌치의 정보가** 0**번으로 설정**
  - "git stash clear"**로 저장된 복원 정보 전체를 삭제할 수 있음**

```
MINGW64:/f/202507001/testnew
unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (squash)
$ git stash list
stash@{0}: WIP on main: 79bf950 edit newfile.txt
stash@{1}: WIP on rebase: df77368 rebase : add rebase.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (squash)
$ git stash drop
Dropped refs/stash@{0} (1106fd381de101e176eb8f10d196676a2064b978)

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (squash)
$ git stash list
stash@{0}: WIP on rebase: df77368 rebase : add rebase.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (squash)
$ git stash clear

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (squash)
$ git stash list

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (squash)
$
```

브렌치 이동 - 10

- **번호 지정 및 주의 사항**
  - stash 수행 시에 지정된 번호를 이용하여 stash 관련 작업 수행 가능
  - 번호를 지정하는 방법은 stash 관련 명령어 마지막에 번호를 입력하는 형태
  - apply, drop, pop 명령 모두 번호를 지정하여 처리를 수행할 수 있음
  - 항상 현재 브렌치에 복원을 시도하기 때문에 의도하지 않는 문제 발생 가능
  - 아래와 같이 main 브렌치 복원 정보를 stash 브렌치에 적용할 수 없는 경우
  - 브렌치의 파일 상태와 존재 유무가 다를 수 있기 때문에 주의해야 함

```
MINGW64:/f/202507001/testnew

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (squash)
$ git stash list
stash@{0}: WIP on main: 79bf950 edit newfile.txt
stash@{1}: WIP on rebase: df77368 rebase : add rebase.txt

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (squash)
$ git stash apply 0
rebase.txt: needs merge
error: could not write index

unangel@DESKTOP-I8OQG85 MINGW64 /f/202507001/testnew (squash)
$
```