- **git log**: mostra la cronologia di tutti i commit raggiungibili dal branch selezionato
    - **git log -n <numero>**: Mostra solo gli n commit passati
- **git log <branch1> … <branchn>**: Mostra tutti i commit di tutti i branch specificati
    - **git log --graph <branch1> … <branchn>**: Mostra un grafico affollato dei branch specificati
    - **git log --graph --oneline <branch1> .. <branchn>**: Grafico su singola linea dei commit sui branch specificati
- **git checkout <commit/branch>**: permette di ripristinare la situazione attuale al commit o al branch selezionato
- **git add <file>**: Permette di aggiungere un file alla staging area.
    - **git add -u**: Aggiunge tutti i file in tracking con git che hanno subito modifiche dall'ultimo commit alla staging area
- **git init**: Crea una nuova repository all'interno della cartella
- **git clone <link/.git link/other>**: permette di copiare tutti i files e i commit di una repository in una nuova cartella con lo stesso nome.
  Se si clona da una repository online si crea in automatico un remote che punta alla repository online.
- **git diff <commit1> <commit2>**: Permette di visualizzare tutte le differenze nei files tra i due commit specificati
    - **git diff**: permette di visualizzare le differenze tra i files in locale e la staging area
    - **git diff --staged**: permette di visualizzare le differenze tra la staging area e HEAD, ovvero l'ultimo commit selezionato
- **git reset [--mode= --mixed] [commit=HEAD]**: riporta lo stato della cartella al commit selezionato
    - **git reset --soft [commit=HEAD]**: riporta solamente la storia dei commit al commit selezionato rendendo irragiungibili i commit succesivi a quello specificato.
      Lascia invariata la staging area ed i files in locale.
    - **git reset --mixed [commit=HEAD]**: è l'opzione di default. Riporta la staging area e la storia dei commit al commit specificato.
      Lascia invariati i files in locale.
    - **git reset --hard [commit=HEAD]**: riporta TUTTA la repository, locale e non, al commit selezionato.
- **git commit**: Apre l'editor per inserire un messaggio ed effettua un commit con le modifiche della staging area.
    - **git commit -m "..."**: Effettua un commit "al volo" con la descrizione specificata tramite parametro.
- **git show <commit>**: Mostra le differenze tra il commit specificato ed il suo genitore (nel vecchio branch) nell'albero dei commit. Utile dopo un merge di due branch per vedere diff di commit passati intervallati da commit del branch che è stato unito.
- **git gc**: chiama il Garbage Collector che elimina i commit irraggiungibili
- **git branch**: Mostra tutti i branch esistenti e asterisca quello attuale.
    - **git branch <branch_name>**: Crea un nuovo branch

    - **git branch -d <branch>**: Cancella un branch. Cancella solo l'etichetta del branch non i commit. Se il branch non è stato unito a nessun altro branch allora tutti i commi vanno persi.
- **git merge <branch1> .. <branch2>**: unisce i branch specificati dentro quello selezionato al momento del comando che diventa il nuovo branch finale.
  In caso di conflitti il merging rimane in attesa che i conflitti venga risolti manualmente e viene concluso effettuando un commit.
    - **git merge --abort**: annulla il merge in attesa di risoluzione
- **git remote**: Mostra tutti i remote presenti.
    - **git remote -v**: mostra tutti i remote con più info (link).
- **git remote add <remote_name> <remote_link>**: aggiunge un nuovo remote.
- **git push <remote> <branch>**: Carica sul remote tutti i commit raggiungibili dal branch specificato ed effettua il merge tra il branch locale e quello remoto.
- **git pull <remote> <branch>**: Scarica dal remote tutti i commit raggiungibili dal branch sul remote ed effettua il merge tra il branch remoto e quello locale.
- **git fetch <remote>**: Scarica dal remote tutti i branch in locale senza perdere i propri branch, cioè senza effettuare il merge con i branch locali.
  Le copie in locale dei branch da remote si chiamano *<remote_name>/<branch_name>*