1. *Spiegare ruolo di “origin*”: origin è il nome di default che viene dato al server, ossia del repository remota, quando viene clonato sulla macchina;
2. *Comandi per creazione repository da riga di comando*: Per creare una nuova repository bisogna prima dirigersi su una working directory in un progetto locale. Da riga di comando si usa **$ git init** per inizializzare una nuova repository, successivamente è possibile aggiungere dei file nei repositori con **$ git add.** e fare il commit con **$commit -m “…”,** successivamente possiamo prendere l’URL del repository remota su GitHub e con il comando **$git remote add origin URL** aggiungiamo l’URL della repository remota nel progetto locale. Dopo aver creato una repository è possibile clonarla su un'altra macchina con **$git clone URL**.
3. *Eseguire una git push force e spiegare le conseguenze di tale azione:* una git push force cancella la commit precedente e viene fatto il push della commit attuale, quindi anche se si ottiene un merge il push force sovrascrive il commit precedente, una git push può essere fatta su repository o branch con il comando **$git push <name> --force.**
4. *Ogni commit ha un id univoco. Documentatevi sul come tale id viene assegnato:* l’id del commit è una codifica SHA-1 contenente le informazioni del commit, tra cui: il contenuto del commit, la data del commit, il nome e l’email di chi ha eseguito il commit, il messaggio di log preso da **$git commit -m ”messaggio”** e l’ID del commit precedente.
5. *Spiegare differenza fra una revert e una reset, con relativi esempi*: **$git reset <file>** rimuove il file dalla staging area lasciando invariato la working directory, può essere applicato anche ai commit annullando la commit precedente ma lasciando la working directory invariata, con **$git reset <commit> , $git revert <commit>** crea un nuovo commit che annulla i cambiamenti fatti dal commit, dopo di che applica i cambiamenti al branch
6. *Spiegare i due metodi di integrazione branch visti finora, con relativi esempi:* ci sono due modi per integrare i branch; merge e rebase. La merge combina la cronologia delle due branch nella branch corrente, viene quindi persa la cronologia delle due branch precedenti e viene creata una nuova cronologia di branch, il comando è **$ git merge --no-ff <branch name>,** se ci sono modifiche su linee differenti, il merge viene applicato normalmente, altrimenti se le linee si sovrappongono è compito di chi effettua il merge gestire i cambiamenti. Il rebase a differenza del merge, prende tutti i cambiamenti che sono stati fatti su un branch e li applica su un altro. Con il rebase è possibile scorrere la cronologia dei cambiamenti uno per volta ed inglobarli manualmente, i comandi per eseguire un rebase sono: **$git rebase** dopo aver fatto delle modifiche, successivamente **$git rebase -i origin/develop** per vedere la lista di tutti i commit, e sostituendo la lettera f alla dicitura pick inglobiamo tutti i commit in uno, infine si può fare il push del nuovo commit.