Universitatea Tehnică a Moldovei

Facultatea Calculatoare Informatică şi Microelectronică

# Raport

la disciplina:

## MIDPS

*Lucrarea de laborator Nr.2*

**Tema:** Version Control Systems si modul de setare a unui server

A efectuat: st.gr.Ti-141 Braga Eugen

### A verificat lect. univ. Cojanu Irina

Chişinău 2016

**Scopul lucrării:** Crearea aplicatiilor interactive in C++Builder

**Obiectivele lucrării:**

* Intelegerea si folosirea CLI (basic level)
* Administrarea remote a masinilor linux machine folosind SSH (remote code editing)
* Version Control Systems (git || mercurial || svn)
* Compileaza codul C/C++/Java/Python prin intermediul CLI, folosind compilatoarele gcc/g++/javac/python

**Controlul versiunilor** (din [engleză:](https://ro.wikipedia.org/wiki/Limba_englez%C4%83) *version control* sau *revision control*) este un domeniu [software](https://ro.wikipedia.org/wiki/Software) care se ocupă cu gestionarea mai multor versiuni (numite și revizii) ale unor[fișiere.](https://ro.wikipedia.org/wiki/Fi%C8%99ier) Este aplicată cu predilecție în [programare,](https://ro.wikipedia.org/wiki/Programare) cu scopul de a păstra versiuni succesive ale [codului sursă](https://ro.wikipedia.org/wiki/Cod_surs%C4%83) al unui [program](https://ro.wikipedia.org/wiki/Software) de [calculator.](https://ro.wikipedia.org/wiki/Calculator) O soluție ar fi arhivarea separată și completă a fiecărei versiuni a programului într-o [bază de date](https://ro.wikipedia.org/wiki/Baz%C4%83_de_date) (pe un purtător de date extern), dar această metodă ar necesita în general prea mult spațiu de memorie. În locul ei se utilizează tehnici speciale, care reduc memoria totală necesară și care facilitează reconstrucția „în zbor”, la cerere, a oricărei versiuni din istoria programului.

#### Prima generatie

Prima generatie de unelete pentru controlul versiunilor foloseau/versionau cate un singur fisier si nu aveau o corespundere intre diferite fisiere din *repository*. Acestea nu aveau suport pentru retea.

Exemple de astfel de unelte: Source Code Control System (SCCS), Revision Control System (RCS).

#### A doua generatie

A doua generatie de unelete pentru controlul versiunilor folosesc/versioneaza mai multe fisiere si aveau o corespundere directa intreb ele. Acestea erau centralizate.

Exemple de astfel de unelte: Concurrent Versions System (CVS), Subversion (SVN), TFS, Perforce, SVK, VSS.

#### A treia generatie

A treia generatie de unelete pentru controlul versiunilor folosesc/versioneaza mai multe fisiere si aveau o corespundere directa intre ele dar sunt descentralizate. Exemple de astfel de unelte: git, BitKeeper (BK), Bazaar.

### Terminologie

#### repository

„depozitul“ în care sunt păstrate fișierele curente și versiunile anterioare. Deseori acest depozit este o bază de date găzduită pe un server.

**working copy (copie de lucru)** copie a fișierelor din repository pe calculatorul de lucru al unui dezvoltator (de unde și numele). Acestea sînt fișierele pe care lucrează un dezvoltator în mod obișnuit. **check-out** operația de creare a unei copii de lucru luate din repository

**commit sau check-in** operația de introducere în repository a schimbărilor din copia de lucru

#### update (actualizare)

introducerea în copia de lucru a schimbărilor făcute de alte persoane (colegi la același proiect) la repository

**branch (ramificare)** bifurcarea unui set de fișiere în două căi de dezvoltare distincte

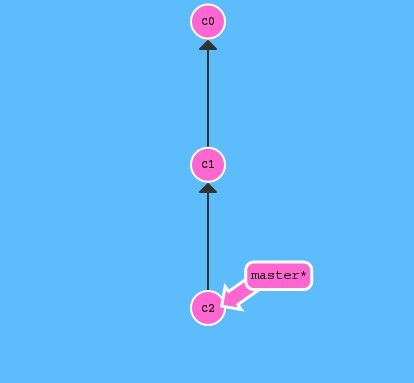
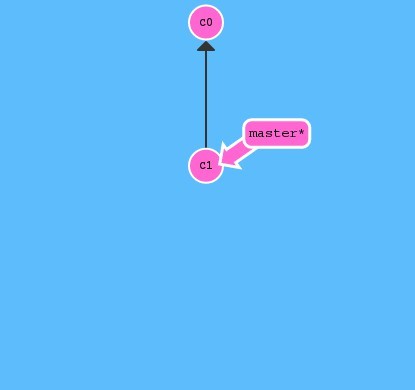
**merge (integrare)** unirea a două versiuni diferite ale unui aceluiași fișier într-o singură versiune

**tag** o „etichetă“ aplicată fișierelor din repository la un anumit moment important din "viața" programului, de exemplu la lansarea unui produs

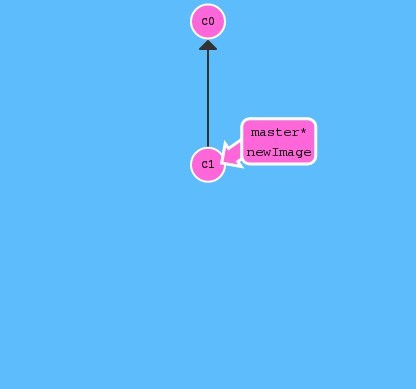
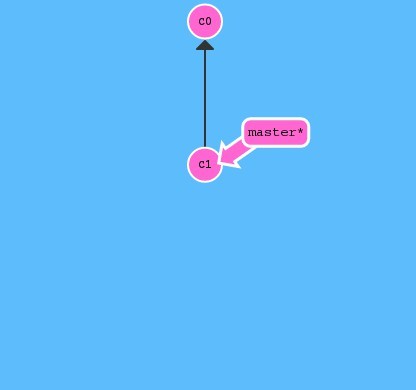
**Git** este un sistem [revision control](https://ro.wikipedia.org/w/index.php?title=Revision_control&action=edit&redlink=1) care rulează pe majoritatea platformelor, inclusiv Linux, POSIX, Windows și OS X. Ca și[Mercurial,](https://ro.wikipedia.org/wiki/Mercurial) Git este un sistem distribuit și nu întreține o bază de date comună. Este folosit în echipe de dezvoltare mari, în care membrii echipei acționează oarecum independent și sunt răspândiți pe o arie geografică mare.

**Vizualizarea comenzilor git.**

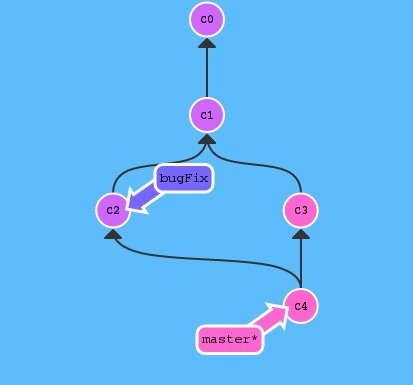
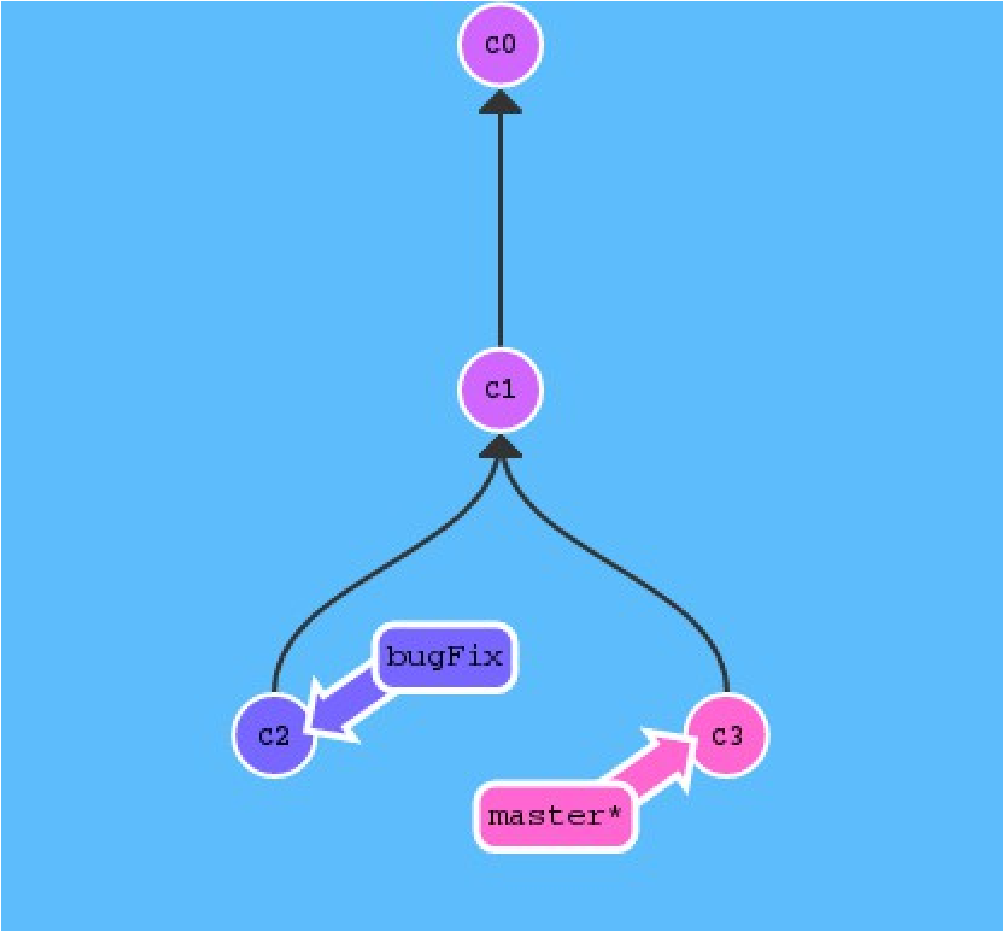
#### Git commit



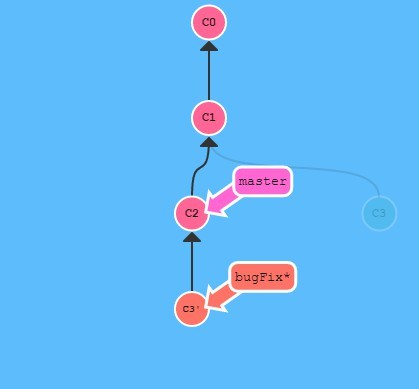
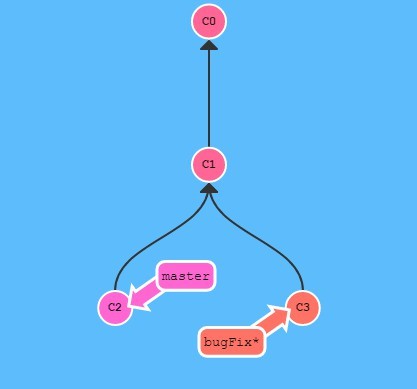
#### git branch newImage



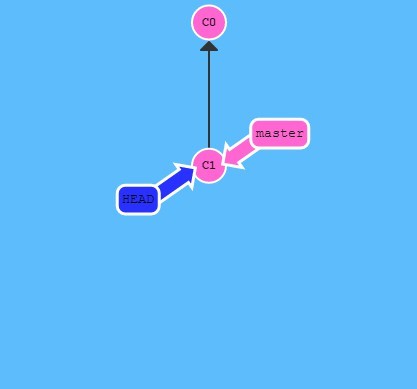
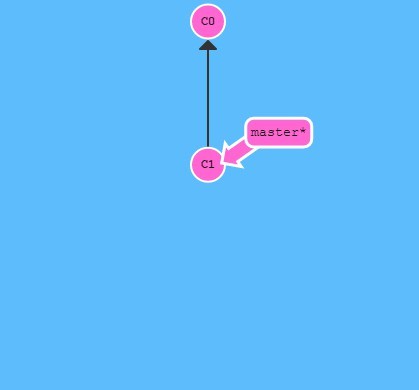
#### git merge bugFix



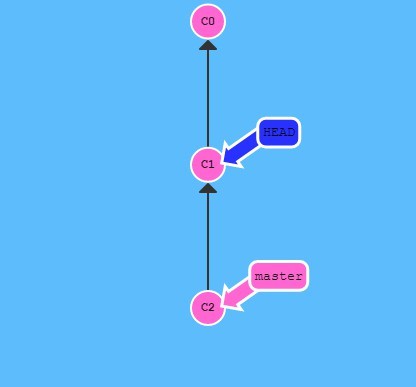
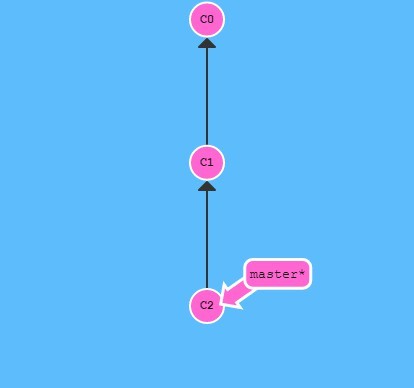
#### git rebase master



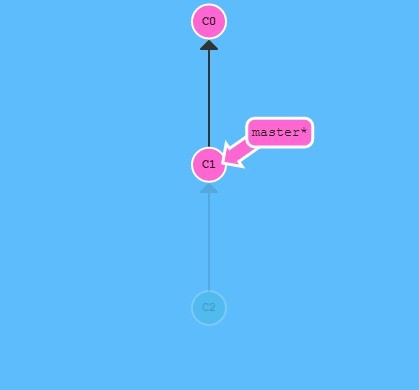
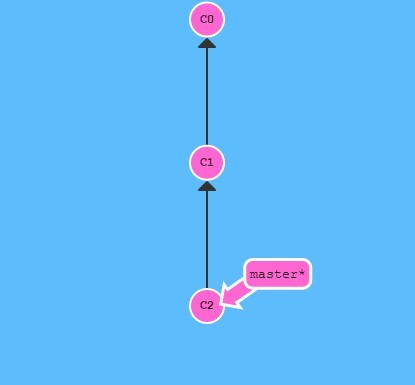
#### git checkout C1



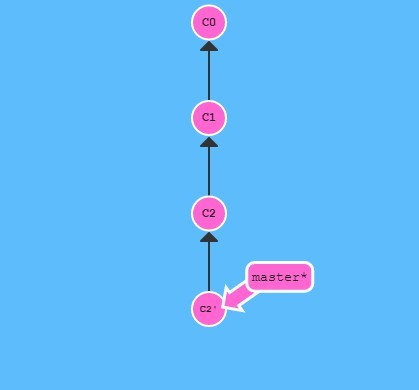
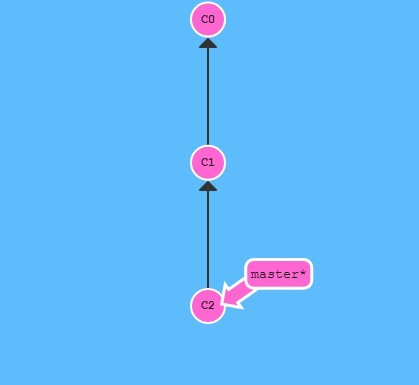
#### git checkout master^



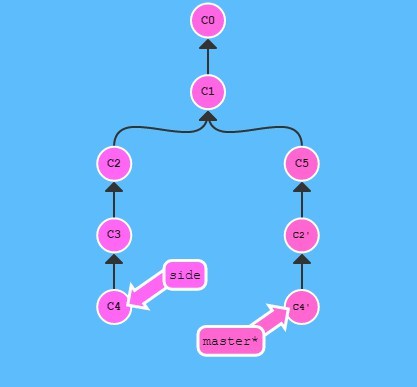
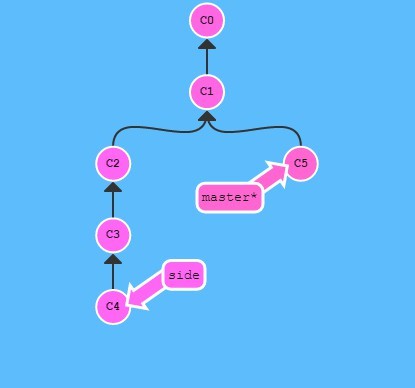
#### git reset HEAD~1



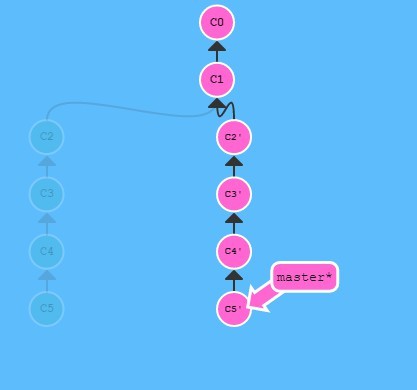
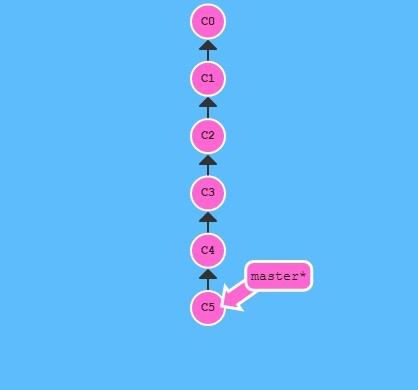
#### git revert HEAD



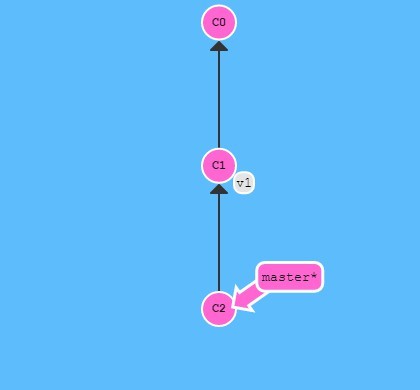
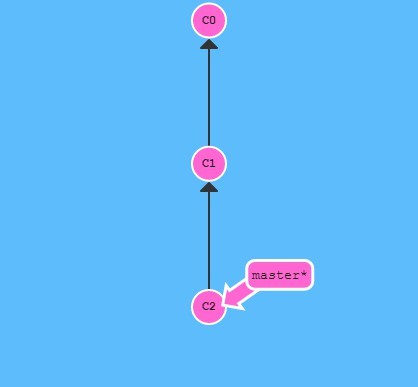
#### git cherry-pick C2 C4



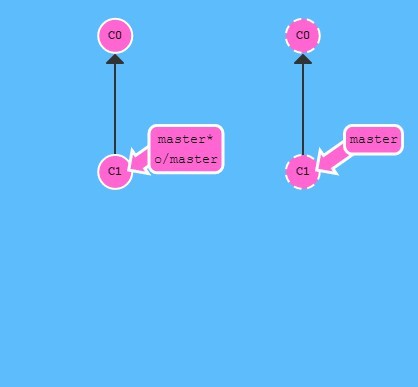
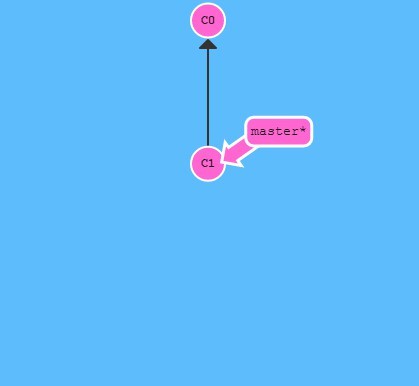
#### git rebase -i HEAD~4 –aboveAll



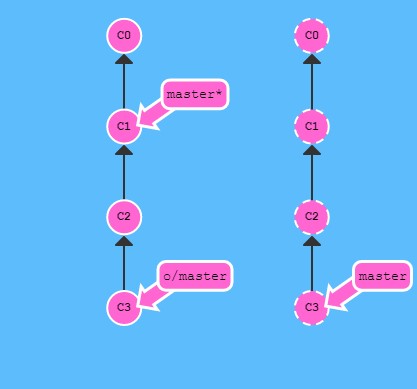
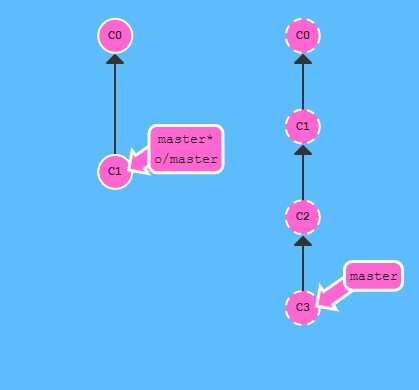
#### git tag v1 C1



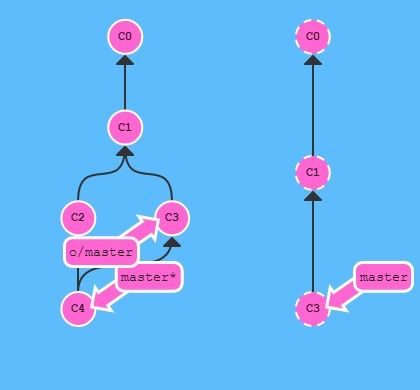
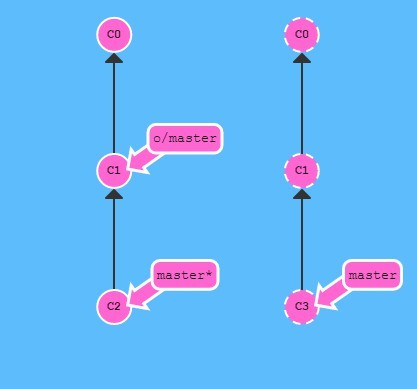
#### git clone



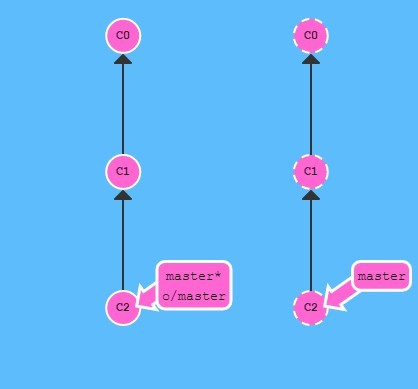
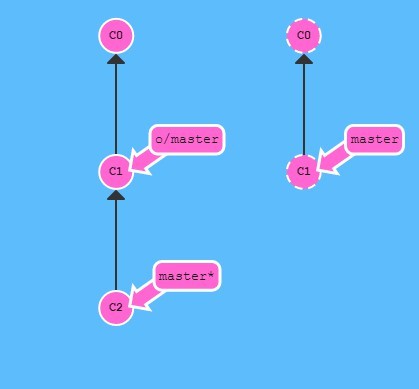
#### git fetch



#### git pull



#### git push



**Shell script ce compileaza HelloWordPrograms projects.**

#!/bin/bash

echo -e "\n|--------C-------------|" cd c gcc hello.c ./a.out cd ..

echo -e "\n|--------C++-----------|"

cd cpp

g++ hello.cpp ./a.out cd ..

echo -e "\n|--------JAVA----------|"

cd java javac hello.java java HelloWorld cd ..

echo -e "\n|--------Python--------|"

cd python python hello.py cd ..

echo -e "\n|--------Ruby----------|"

cd ruby ruby hello.rb

#### Concluzii

In urma efectuarii lucrarii de laborator Nr.2 la MIDPS am obtinut capacitati practice de utilizarea a sistemului git . Acum pot folosi sitemul git pentru a crea commit-uri si branch-uri noi si a le gestiona , creind astfel un workflow de orice complexitate al unui proiect modern.