

ECE 264 Spring 2023

Advanced C Programming

Aravind Machiry
Purdue University

Most of the material is derived from Prof. Lu's slides.

Welcome

Instructor: Aravind Machiry

- 2020 PhD. Computer Science, UCSB
- Join Purdue in 2021 as an assistant professor
- Research areas: Computer Security, Embedded Systems, Program Analysis.
- Webpage: <https://machiry.github.io/>
- Lab page: <https://purs3lab.github.io/>





Binary Analysis

BootStomp [SEC 17]
KARONTONE [SP 20]
BinTrimmer [DIMVA 19]
Boomerang [NDSS 17]
DIANE [SP 21]
GlitchResistor [DSN 21]



Source Code Analysis

DR.CHECKER [SEC 17]
DIFUZE [CCS 17]
Fingerprint [NDSS 18]



Machine Learning

SLAP [ACSAC 18]
ARBITRAR [SP 21]
BRAN [Asia CCS 21]



Systems

Dynodroid [SEC 17]
CLAPP [SP 20]
Boomerang [NDSS 17]
Trust.IO [CNS 20]
PRETENDER [RAID 19]
CONWARE [Asia CCS 21]



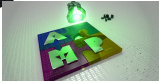
Human interaction

Dynodroid [FSE 13]
Checked C [OOPSLA 22]



Verification

Spider [SP 20]



Our Work

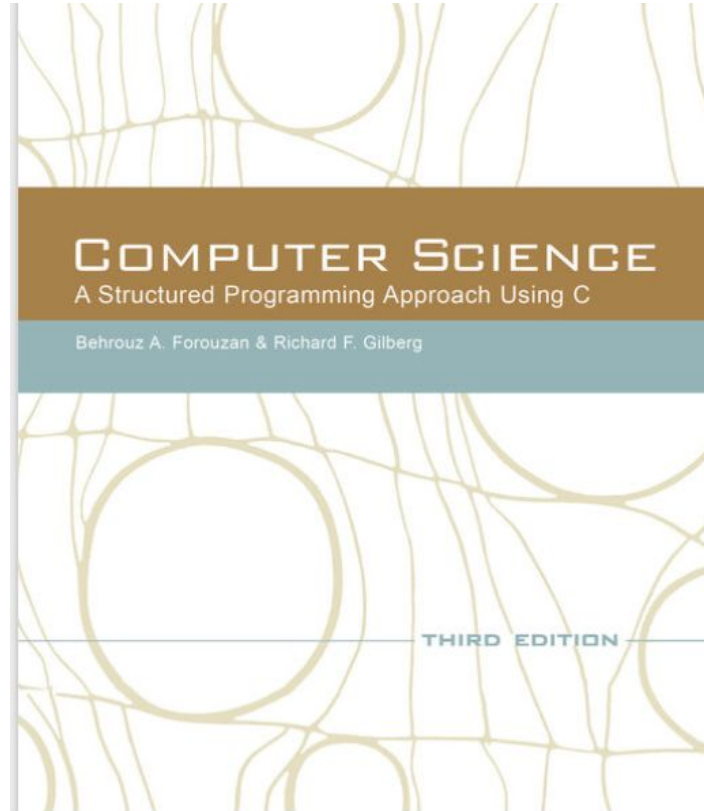
Logistics

Course webpage:

<https://purs3lab.github.io/ece264/>

We will be using GitHub classroom for all our programming assignments.

Textbook



ECE 264 Uses Linux

If you use Windows, MacOS, Android, or anything other than Linux, you will definitely receive F in this class.

Why Linux?

Linux is "UNIX-like"

iOS, Android are also "UNIX-like"

Top 500 (fastest 500 computers in the world)

Linux is widely used in embedded systems

(real-time improvements) Used in autonomous vehicles

"At least" 67% web servers run Linux (wired.com, 2016/08/25)

Many computer courses use Linux (because grading is easier)

Support for Linux and open source technology in Azure

Applies to: Cloud Services (Web roles/Worker roles), MSFC Azure-Azure Apps (IaaS)

Design

[OVERVIEW](#)[ARCHITECTURE](#)[COMPATIBILITY](#)[DISPLAY](#)[SETTINGS](#)[TESTS](#)[Overview](#)

- ▶ [Modular System Components](#)
- ▶ [Hardware Abstraction Layer \(HAL\)](#)
- ▾ [Kernel](#)
 - [Overview](#)
 - ▶ [Stable Releases & Updates](#)
 - [Android Common Kernels](#)

AOSP > Design > Architecture

Kernel

The Linux kernel is an extremely important part of the software on nearly every Android device.

Get Linux

Download Linux and install on your computer

- Brand new computer
- Dual boot if it already has an operating system

Virtual machine (such as Virtualbox), Linux coexists with another operating system (such as Windows)

Cloud (Amazon EC2, Microsoft Azure, Google Cloud ...)

Purdue Engineering students: thin linc



Thin Linc



Connect to the servers managed by ECN (Engineering Computer Networks)

ECN is the IT department of Purdue Engineering

Advantages:

- ECN updates software
- Security is managed by experts
- These are very fast computers
- Many courses use these computers for grading

Linux at Purdue ECN

Many ways to connect:

- Install thinlinc client (recommended)
 - Detailed instructions:
<https://engineering.purdue.edu/ECN/Support/KB/Docs/ECThinlinc>
- Install secureCRT (not recommended)
- Install putty (not recommended)
- <https://desktop.eceprog.ecn.purdue.edu/> (Not recommended)

Strongly discouraged: Cygwin and MinGW (not real Linux)

Do not use telnet (not secure)

[What is ThinLinc](#)[How It Works](#)[Features](#)[Customers](#)[Download](#)[Documentation](#)[Contact](#)

A Linux remote desktop server built on open source technology.

[Try It Now!](#)[Buy / Pricing](#)[What Is ThinLinc](#)[How It Works](#)[Technical Overview](#)[Features](#)[Benefits](#)[Customers](#)[Download](#)[ThinLinc Client Archive](#)[Documentation](#)[White Papers](#)[Support](#)[Training](#)[Partners](#)[How to Buy ThinLinc](#)

ThinLinc Download

Download ThinLinc Server Bundle

Cendio offers these downloads primarily for evaluation and testing purposes. For large organizations, multiple testing licenses and technical support are provided during the evaluation upon request. This download is also available for small user groups and home users for free, with a maximum of 5 concurrent users, for more details see the FAQ.

[FAQ](#)

Large Organisations

Cendio AB works diligently on assisting our potential customers during the evaluation process of our product ThinLinc through offering:

- Free technical support (by phone and email) during the evaluation phase of ThinLinc.

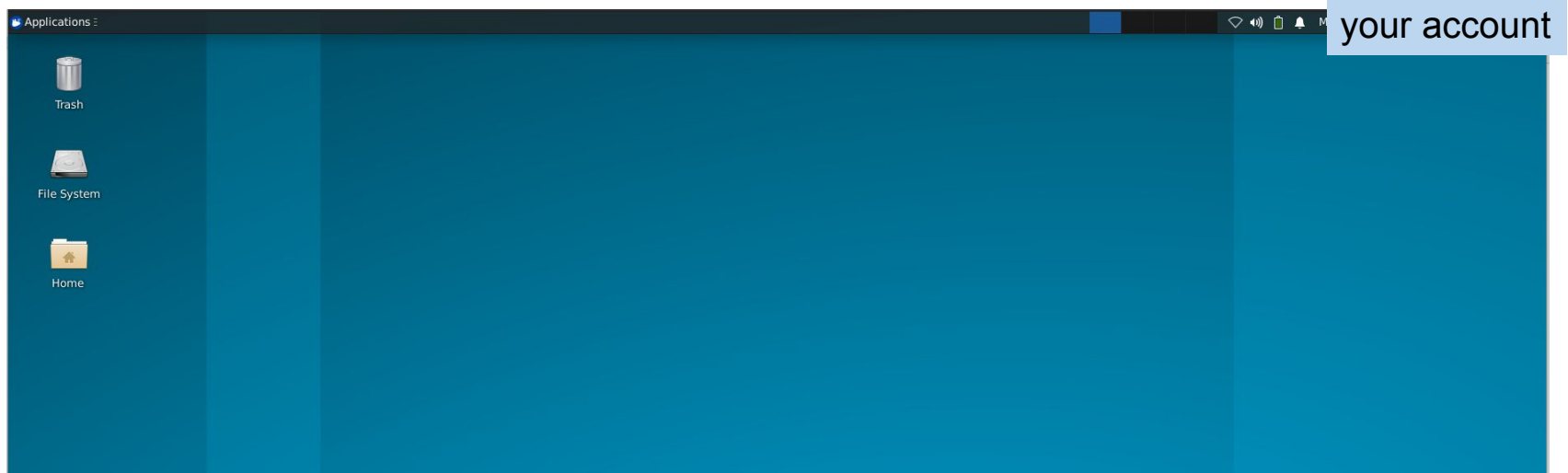
Individual use and small groups

Cendio AB offers this option for individuals and smaller user groups. Further information and general help can be obtained through:

- Joining our mailing list and at Cendio forum.
- Contacting us at demosupport@cendio.com

connect to

desktop.eceprog.ecn.purdue.edu





Trash



File System



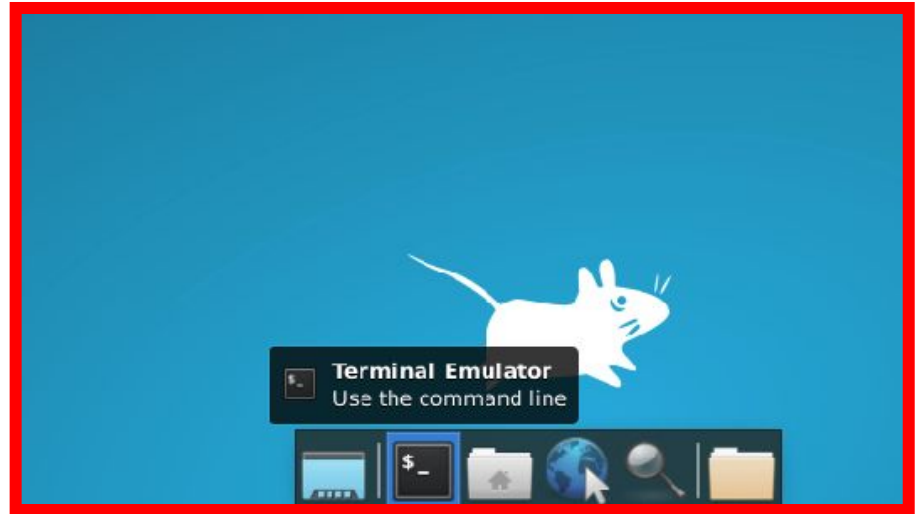
Home



Terminal and Command Line

Combine multiple commands

Automate tools, without human
clicking or dragging



Why terminal? Isn't GUI better?

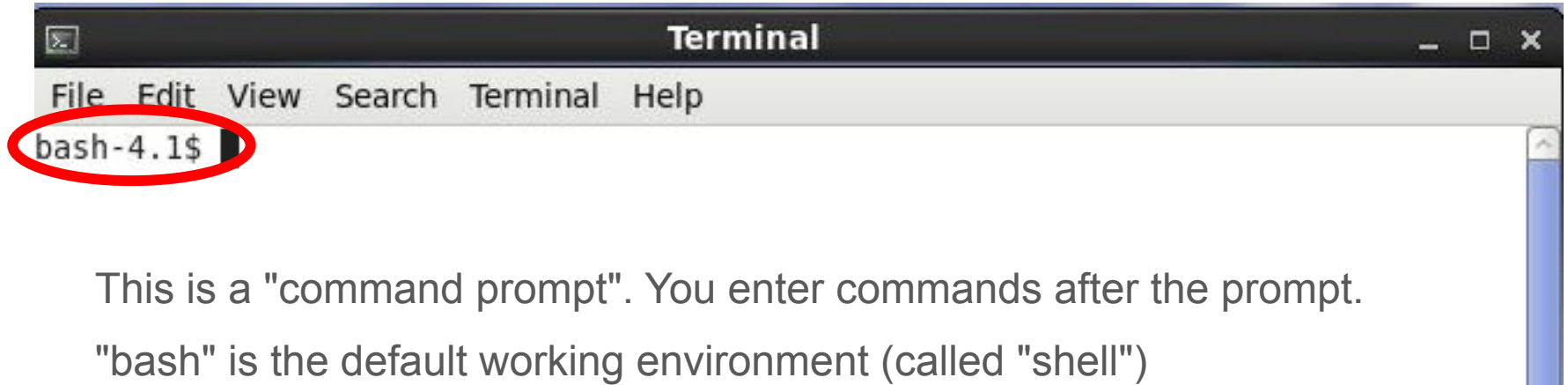
GUI (Graphical User Interfaces) are good for humans, not computers

Terminals allow you to automate many things that are difficult if you use GUI

Terminals allow you to scale up to manage hundreds or thousands of machines

In data centers, everything is based on terminals, no GUI

Knowing how to use terminals give you additional skills (needed for many positions)



This is a "command prompt". You enter commands after the prompt.

"bash" is the default working environment (called "shell")

4.1 means the version

Frequently Used Commands

ls (list)

ls: list the files and directories (also called "folders")

ls -l: list with long (more information); beginning 'd': directory

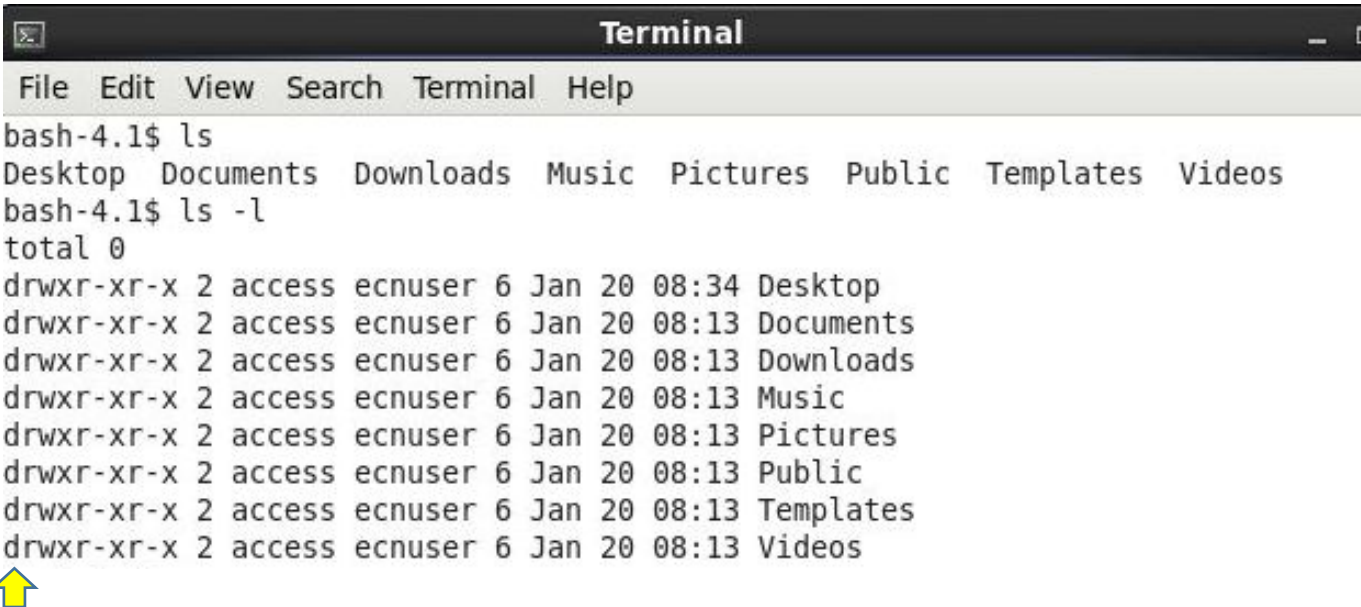


```
→ bash-4.1$ ls
Desktop Documents Downloads Music Pictures Public Templates Videos
→ bash-4.1$ ls -l
total 0
drwxr-xr-x 2 access ecnuser 6 Jan 20 08:34 Desktop
drwxr-xr-x 2 access ecnuser 6 Jan 20 08:13 Documents
drwxr-xr-x 2 access ecnuser 6 Jan 20 08:13 Downloads
drwxr-xr-x 2 access ecnuser 6 Jan 20 08:13 Music
drwxr-xr-x 2 access ecnuser 6 Jan 20 08:13 Pictures
drwxr-xr-x 2 access ecnuser 6 Jan 20 08:13 Public
drwxr-xr-x 2 access ecnuser 6 Jan 20 08:13 Templates
drwxr-xr-x 2 access ecnuser 6 Jan 20 08:13 Videos
```

ls (list)

ls: list the files and directories (also called "folders")

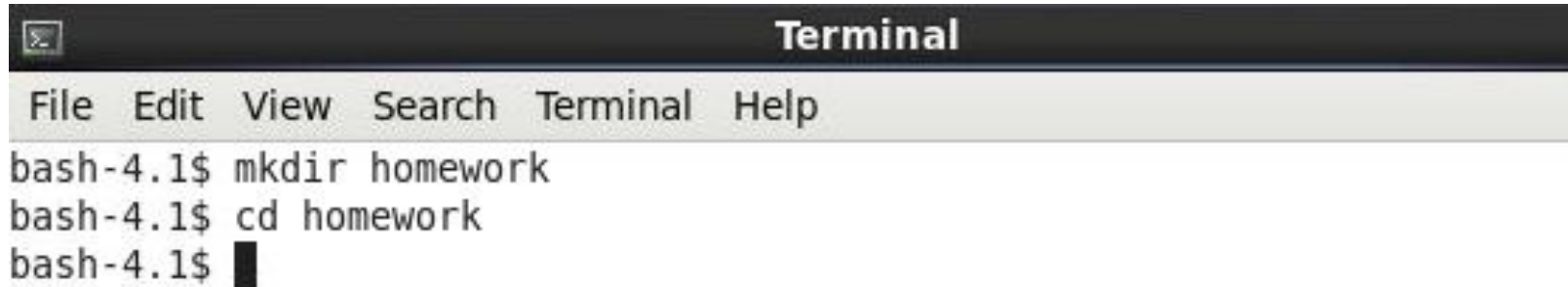
ls -l: list with long (more information); beginning 'd': directory



```
Terminal
File Edit View Search Terminal Help
bash-4.1$ ls
Desktop Documents Downloads Music Pictures Public Templates Videos
bash-4.1$ ls -l
total 0
drwxr-xr-x 2 access ecnuser 6 Jan 20 08:34 Desktop
drwxr-xr-x 2 access ecnuser 6 Jan 20 08:13 Documents
drwxr-xr-x 2 access ecnuser 6 Jan 20 08:13 Downloads
drwxr-xr-x 2 access ecnuser 6 Jan 20 08:13 Music
drwxr-xr-x 2 access ecnuser 6 Jan 20 08:13 Pictures
drwxr-xr-x 2 access ecnuser 6 Jan 20 08:13 Public
drwxr-xr-x 2 access ecnuser 6 Jan 20 08:13 Templates
drwxr-xr-x 2 access ecnuser 6 Jan 20 08:13 Videos
```

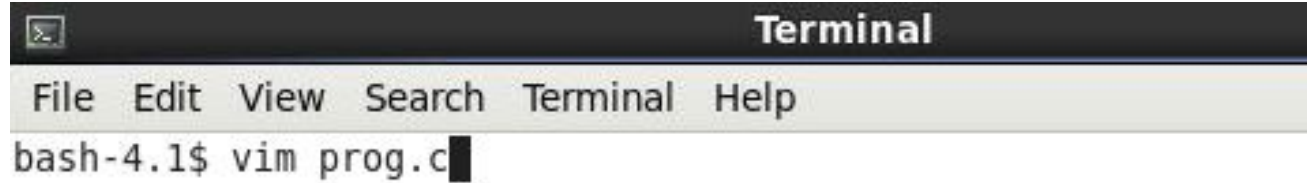
The image shows a terminal window with a menu bar (File, Edit, View, Search, Terminal, Help) and a title bar (Terminal). The terminal output shows the execution of two commands: 'ls' and 'ls -l'. The 'ls' command lists the contents of the current directory: Desktop, Documents, Downloads, Music, Pictures, Public, Templates, and Videos. The 'ls -l' command lists the same contents with long format information, including permissions, number of links, access type, owner, group, size, date, time, and filename. A red arrow points to the first command, and a yellow arrow points to the last line of the output.

mkdir: make new directory (i.e., folder)
cd: change (i.e. enter) directory



```
Terminal
File Edit View Search Terminal Help
bash-4.1$ mkdir homework
bash-4.1$ cd homework
bash-4.1$ █
```

vim: text editor

A terminal window titled "Terminal" with a menu bar containing "File", "Edit", "View", "Search", "Terminal", and "Help". The command prompt shows "bash-4.1\$ vim prog.c" with a cursor at the end of the line.

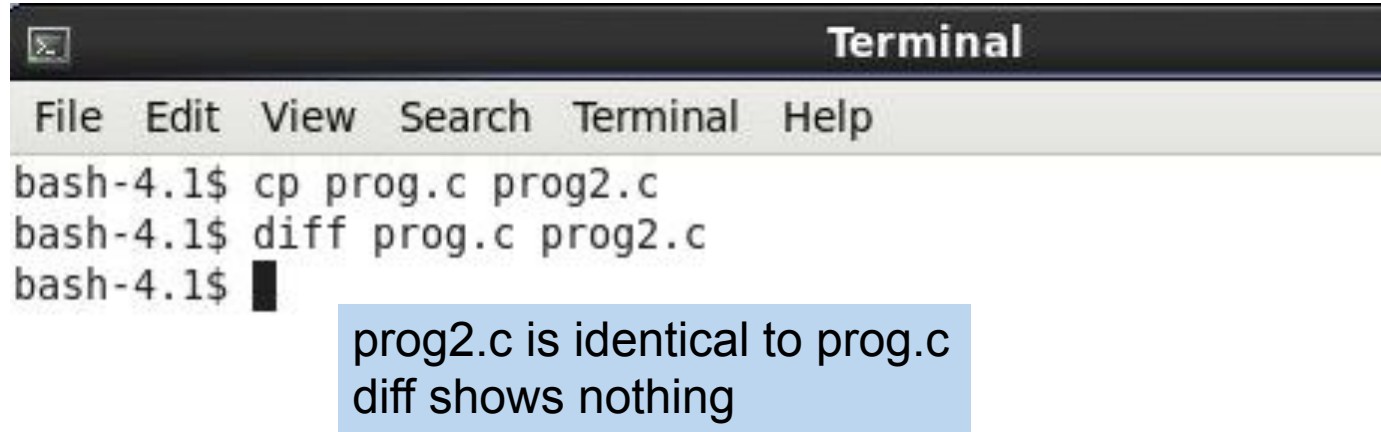
```
Terminal  
File Edit View Search Terminal Help  
bash-4.1$ vim prog.c
```

File Edit View Search Terminal Help

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char * * argv)
{
    int cnt;
    for (cnt = 0; cnt < 10; cnt++)
    {
        printf("%d\n", cnt);
    }
    return EXIT_SUCCESS;
}
```

~
~
~
~
~
~
~
~
~
~
~
~

cp src dest: copy file src to dest
diff file1 file2: compare



```
Terminal
File Edit View Search Terminal Help
bash-4.1$ cp prog.c prog2.c
bash-4.1$ diff prog.c prog2.c
bash-4.1$
```

prog2.c is identical to prog.c
diff shows nothing

echo "message" >> file: add a line
cat -n file: show file with line number

```
Terminal
File Edit View Search Terminal Help
bash-4.1$ echo "// This is a comment" >> prog2.c
bash-4.1$ cat -n prog2.c
 1 #include <stdio.h>
 2 #include <stdlib.h>
 3 int main(int argc, char * * argv)
 4 {
 5     int cnt;
 6     for (cnt = 0; cnt < 10; cnt ++){
 7         printf("%d\n", cnt);
 8     }
 9     return EXIT_SUCCESS;
10 }
11 // This is a comment
bash-4.1$
```

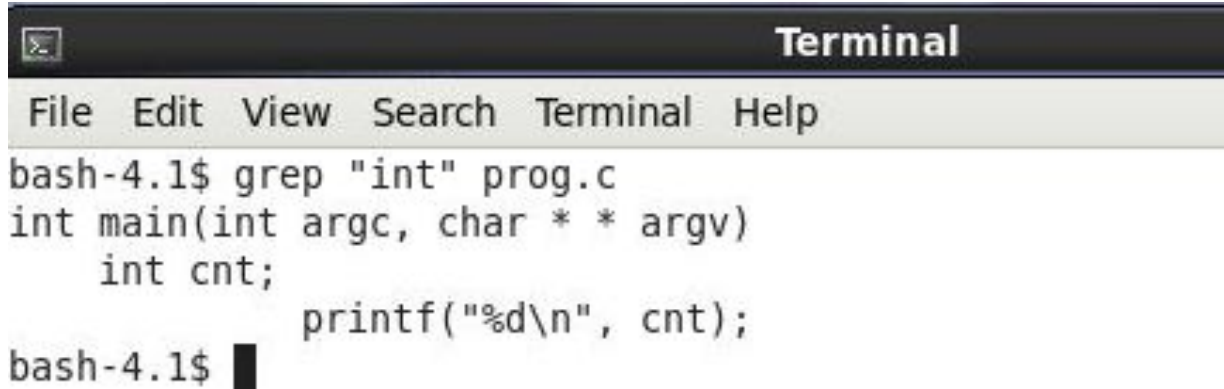
Append at the end of the file

diff shows the difference, with line numbers



```
Terminal
File Edit View Search Terminal Help
bash-4.1$ diff prog.c prog2.c
12a13
> // This is a comment
bash-4.1$
```

grep word file: print the lines with this word in the file



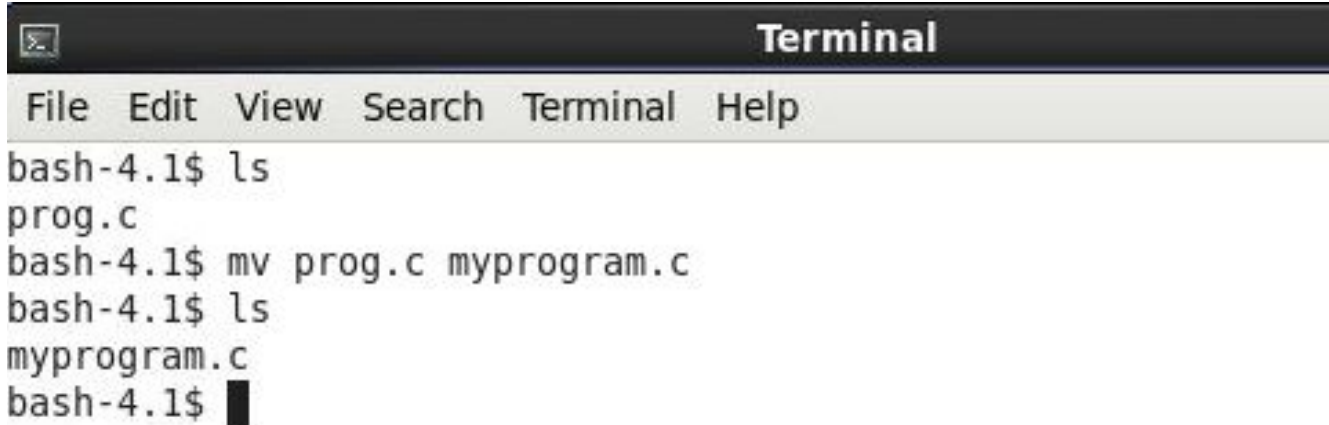
```
Terminal
File Edit View Search Terminal Help
bash-4.1$ grep "int" prog.c
int main(int argc, char * * argv)
    int cnt;
    printf("%d\n", cnt);
bash-4.1$
```

rm: remove a file (irreversible)

```
Terminal
File Edit View Search Terminal Help
bash-4.1$ ls
prog2.c prog.c
bash-4.1$ rm prog2.c
bash-4.1$ ls
prog.c
bash-4.1$
```

← prog2.c has been deleted

mv (move): rename a file



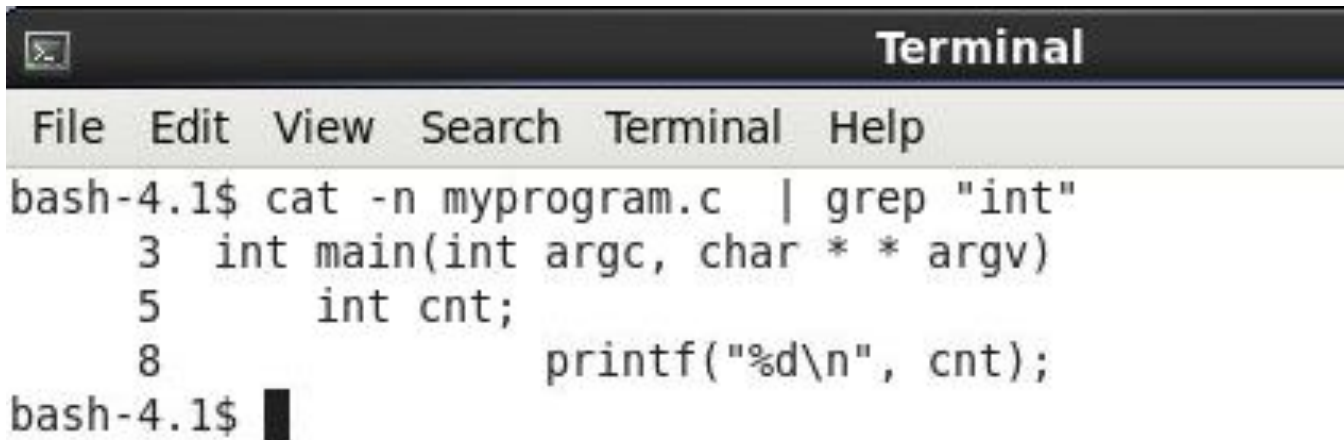
```
Terminal
File Edit View Search Terminal Help
bash-4.1$ ls
prog.c
bash-4.1$ mv prog.c myprogram.c
bash-4.1$ ls
myprogram.c
bash-4.1$ █
```

redirect output using >>

```
Terminal
File Edit View Search Terminal Help
bash-4.1$ cat -n myprogram.c
 1 #include <stdio.h>
 2 #include <stdlib.h>
 3 int main(int argc, char * * argv)
 4 {
 5     int cnt;
 6     for (cnt = 0; cnt < 10; cnt ++)
 7     {
 8         printf("%d\n", cnt);
 9     }
10     return EXIT_SUCCESS;
11 }
12
bash-4.1$ cat -n myprogram.c >> outputfile
bash-4.1$ ls
myprogram.c  outputfile
bash-4.1$
```

```
Terminal
File Edit View Search Terminal Help
bash-4.1$ cat -n outputfile
 1 #include <stdio.h>
 2 #include <stdlib.h>
 3 int main(int argc, char * * a
 4 {
 5     int cnt;
 6     for (cnt = 0; cnt < 10; c
 7     {
 8         printf("%d\n", c
 9     }
10     return EXIT_SUCCESS;
11 }
12
bash-4.1$
```

pipe (|): take output from one program as input of another program



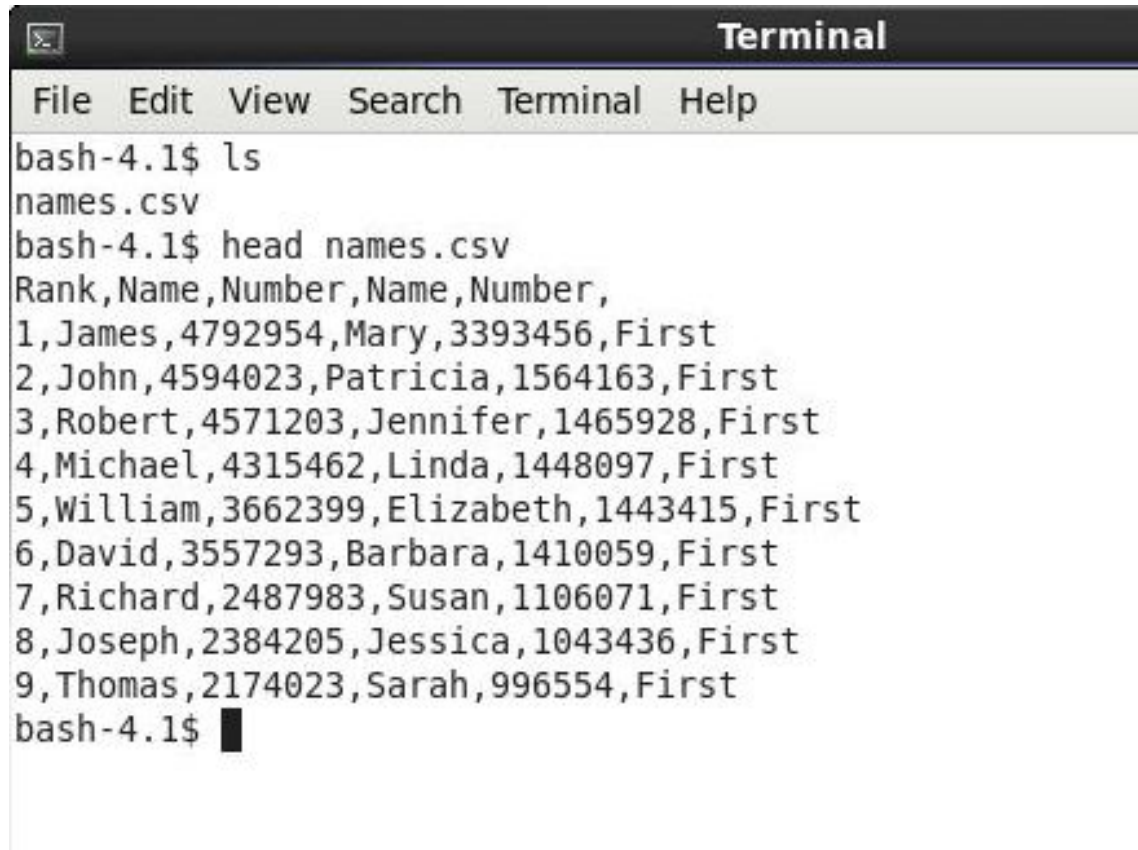
```
Terminal
File Edit View Search Terminal Help
bash-4.1$ cat -n myprogram.c | grep "int"
  3  int main(int argc, char * * argv)
  5      int cnt;
  8          printf("%d\n", cnt);
bash-4.1$ █
```


Popular First and Last Names

	A	B	C	D	E	F
1	Rank	Name	Number	Name	Number	
2	1	James	4792954	Mary	3393456	First
3	2	John	4594023	Patricia	1564163	First
4	3	Robert	4571203	Jennifer	1465928	First
5	4	Michael	4315462	Linda	1448097	First
6	5	William	3662399	Elizabeth	1443415	First
7	6	David	3557293	Barbara	1410059	First
8	7	Richard	2487983	Susan	1106071	First
9	8	Joseph	2384205	Jessica	1043436	First
10	9	Thomas	2174023	Sarah	996554	First
11	10	Charles	2144937	Margaret	993136	First
12	11	Christopher	2018834	Karen	984893	First
13	12	Daniel	1874730	Nancy	973993	First
14	13	Matthew	1582665	Lisa	964399	First
15	14	Anthony	1397889	Betty	957509	First
16	15	Donald	1366903	Dorothy	910134	First
17	16	Mark	1344092	Sandra	873189	First
18	17	Paul	1306645	Ashley	843158	First
19	18	Steven	1277952	Kimberly	834968	First

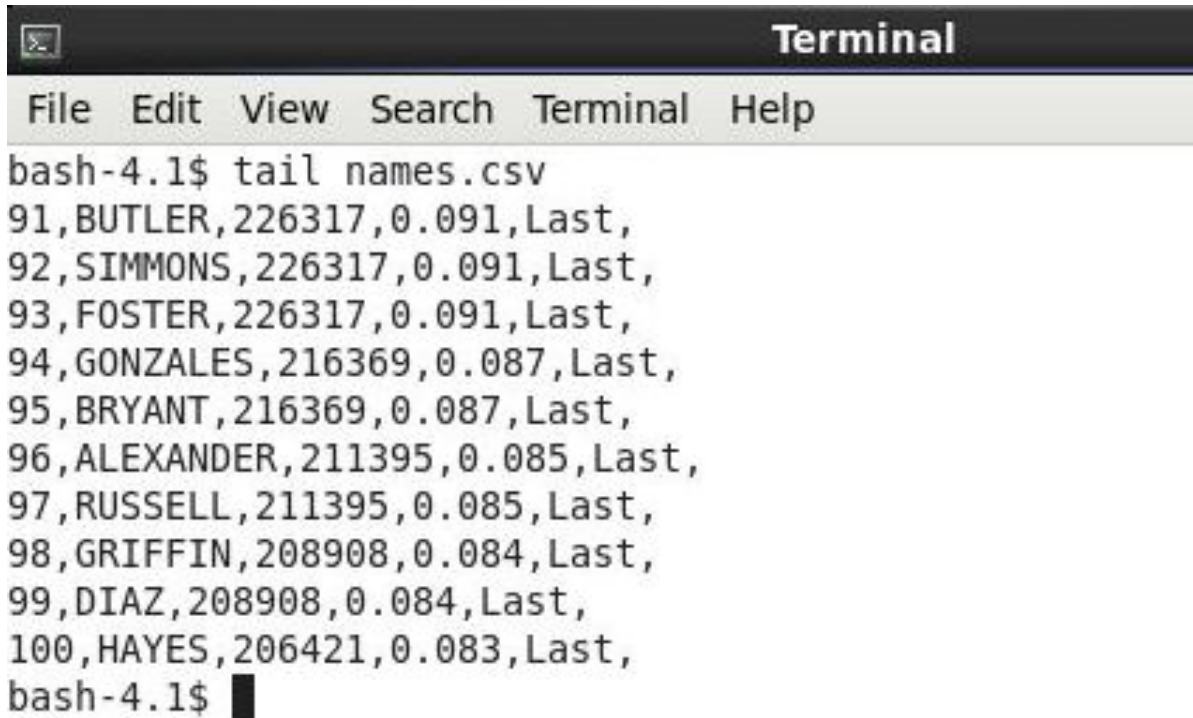
103	1	SMITH	2501922	1.006	Last
104	2	JOHNSON	2014470	0.81	Last
105	3	WILLIAMS	1738413	0.699	Last
106	4	JONES	1544427	0.621	Last
107	5	BROWN	1544427	0.621	Last
108	6	DAVIS	1193760	0.48	Last
109	7	MILLER	1054488	0.424	Last
110	8	WILSON	843093	0.339	Last
111	9	MOORE	775944	0.312	Last
112	10	TAYLOR	773457	0.311	Last
113	11	ANDERSON	773457	0.311	Last
114	12	THOMAS	773457	0.311	Last
115	13	JACKSON	770970	0.31	Last
116	14	WHITE	693873	0.279	Last
117	15	HARRIS	683925	0.275	Last

head: print the first 10 lines



```
Terminal
File Edit View Search Terminal Help
bash-4.1$ ls
names.csv
bash-4.1$ head names.csv
Rank,Name,Number,Name,Number,
1,James,4792954,Mary,3393456,First
2,John,4594023,Patricia,1564163,First
3,Robert,4571203,Jennifer,1465928,First
4,Michael,4315462,Linda,1448097,First
5,William,3662399,Elizabeth,1443415,First
6,David,3557293,Barbara,1410059,First
7,Richard,2487983,Susan,1106071,First
8,Joseph,2384205,Jessica,1043436,First
9,Thomas,2174023,Sarah,996554,First
bash-4.1$ █
```

tail: print the last 10 lines

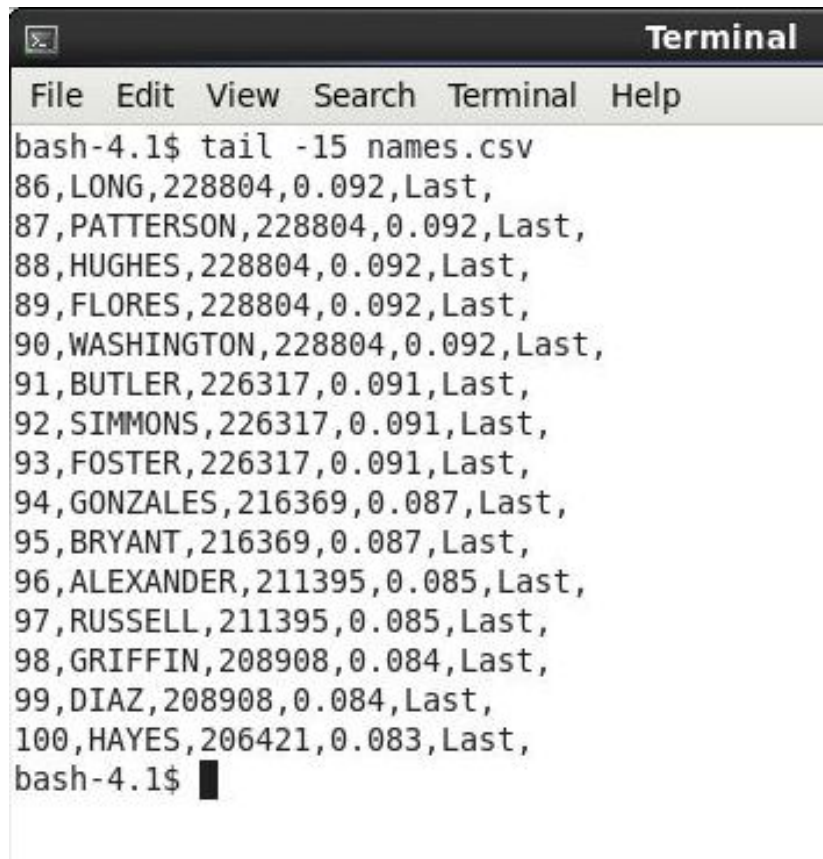


```
Terminal
File Edit View Search Terminal Help
bash-4.1$ tail names.csv
91,BUTLER,226317,0.091,Last,
92,SIMMONS,226317,0.091,Last,
93,FOSTER,226317,0.091,Last,
94,GONZALES,216369,0.087,Last,
95,BRYANT,216369,0.087,Last,
96,ALEXANDER,211395,0.085,Last,
97,RUSSELL,211395,0.085,Last,
98,GRIFFIN,208908,0.084,Last,
99,DIAZ,208908,0.084,Last,
100,HAYES,206421,0.083,Last,
bash-4.1$ █
```

head -n: first n lines

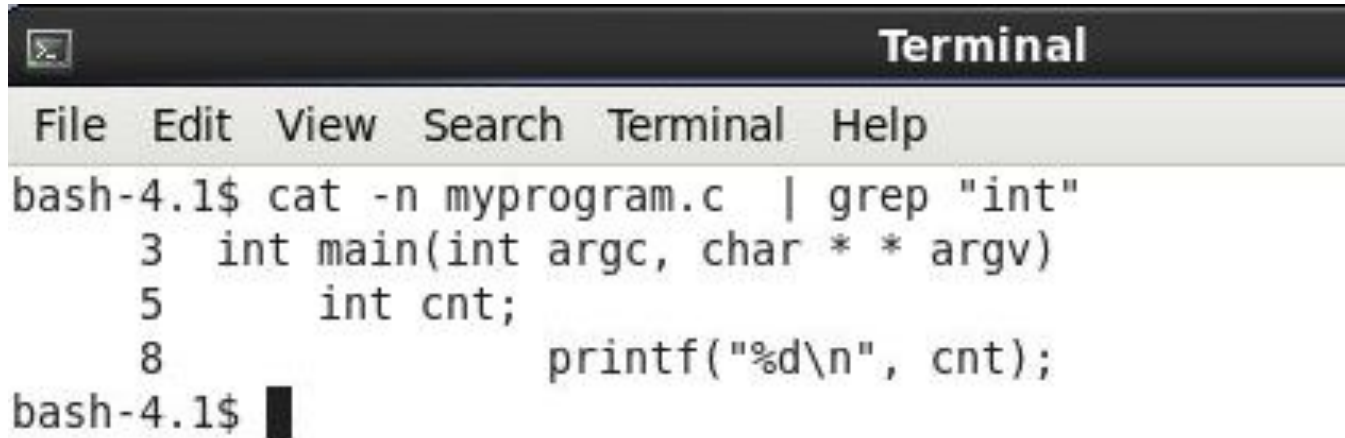
```
Terminal
File Edit View Search Terminal Help
bash-4.1$ head -15 names.csv
Rank,Name,Number,Name,Number,
1,James,4792954,Mary,3393456,First
2,John,4594023,Patricia,1564163,First
3,Robert,4571203,Jennifer,1465928,First
4,Michael,4315462,Linda,1448097,First
5,William,3662399,Elizabeth,1443415,First
6,David,3557293,Barbara,1410059,First
7,Richard,2487983,Susan,1106071,First
8,Joseph,2384205,Jessica,1043436,First
9,Thomas,2174023,Sarah,996554,First
10,Charles,2144937,Margaret,993136,First
11,Christopher,2018834,Karen,984893,First
12,Daniel,1874730,Nancy,973993,First
13,Matthew,1582665,Lisa,964399,First
14,Anthony,1397889,Betty,957509,First
bash-4.1$ █
```

tail -n: last n lines



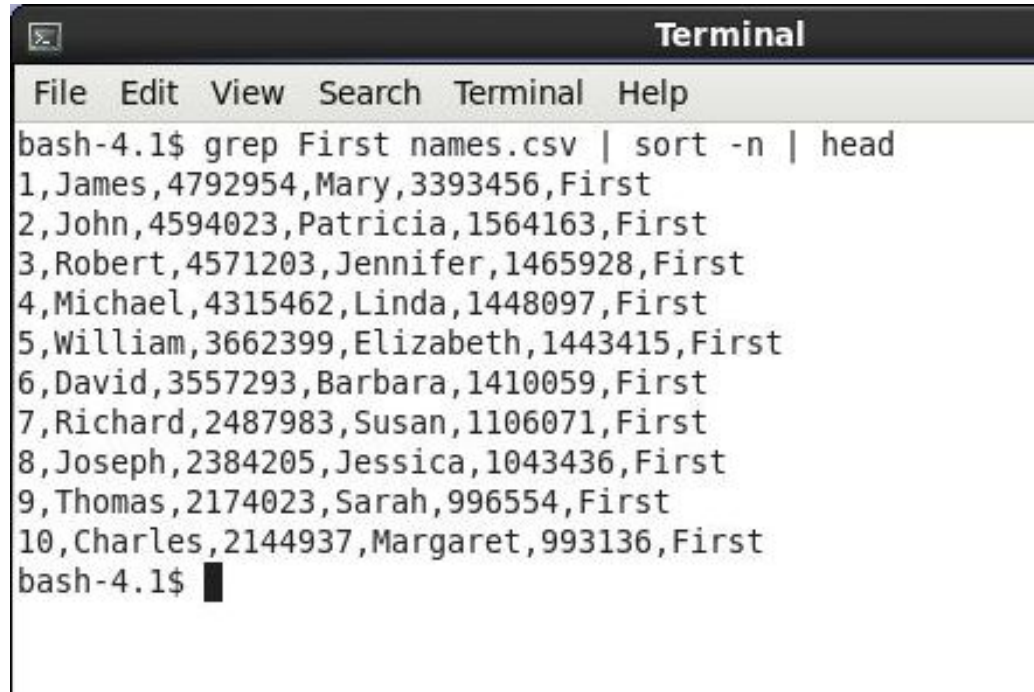
```
Terminal
File Edit View Search Terminal Help
bash-4.1$ tail -15 names.csv
86, LONG, 228804, 0.092, Last,
87, PATTERSON, 228804, 0.092, Last,
88, HUGHES, 228804, 0.092, Last,
89, FLORES, 228804, 0.092, Last,
90, WASHINGTON, 228804, 0.092, Last,
91, BUTLER, 226317, 0.091, Last,
92, SIMMONS, 226317, 0.091, Last,
93, FOSTER, 226317, 0.091, Last,
94, GONZALES, 216369, 0.087, Last,
95, BRYANT, 216369, 0.087, Last,
96, ALEXANDER, 211395, 0.085, Last,
97, RUSSELL, 211395, 0.085, Last,
98, GRIFFIN, 208908, 0.084, Last,
99, DIAZ, 208908, 0.084, Last,
100, HAYES, 206421, 0.083, Last,
bash-4.1$
```

pipe (|): output as input of next



```
Terminal
File Edit View Search Terminal Help
bash-4.1$ cat -n myprogram.c | grep "int"
 3 int main(int argc, char * * argv)
 5     int cnt;
 8         printf("%d\n", cnt);
bash-4.1$ █
```

sort -n: sort as numbers



```
Terminal
File Edit View Search Terminal Help
bash-4.1$ grep First names.csv | sort -n | head
1,James,4792954,Mary,3393456,First
2,John,4594023,Patricia,1564163,First
3,Robert,4571203,Jennifer,1465928,First
4,Michael,4315462,Linda,1448097,First
5,William,3662399,Elizabeth,1443415,First
6,David,3557293,Barbara,1410059,First
7,Richard,2487983,Susan,1106071,First
8,Joseph,2384205,Jessica,1043436,First
9,Thomas,2174023,Sarah,996554,First
10,Charles,2144937,Margaret,993136,First
bash-4.1$
```


sort -k c: sort by the c's column

```
Terminal
File Edit View Search Terminal Help
bash-4.1$ grep First names.csv | sort -t "," -k 2 | head
52,Aaron,569239,Heather,524160,First
57,Adam,543586,Joan,471719,First
87,Alan,344833,Denise,371012,First
76,Albert,395454,Gloria,409027,First
47,Alexander,646463,Rachel,546137,First
19,Andrew,1245696,Donna,824789,First
14,Anthony,1397889,Betty,957509,First
68,Arthur,431511,Olivia,428419,First
74,Austin,403440,Doris,413521,First
42,Benjamin,708478,Samantha,568674,First
bash-4.1$ █
```

```
Terminal
File Edit View Search Terminal Help
bash-4.1$ grep First names.csv | sort -t "," -k 4 | head
94,Randy,327075,Abigail,354708,First
98,Bobby,312923,Alexis,336473,First
71,Sean,416246,Alice,420145,First
23,Kevin,1166544,Amanda,771946,First
89,Eugene,341261,Amber,368623,First
33,Nicholas,885052,Amy,678863,First
67,Roger,431647,Andrea,430394,First
35,Stephen,841135,Angela,657250,First
69,Terry,421661,Ann,422401,First
36,Jonathan,832914,Anna,651333,First
bash-4.1$ █
```


sed: substitute a letter

```
Terminal
File Edit View Search Terminal Help
bash-4.1$ head -5 names.csv
Rank,Name,Number,Name,Number,
1,James,4792954,Mary,3393456,First
2,John,4594023,Patricia,1564163,First
3,Robert,4571203,Jennifer,1465928,First
4,Michael,4315462,Linda,1448097,First
bash-4.1$ head -5 names.csv | sed 's/c/C/g'
Rank,Name,Number,Name,Number,
1,James,4792954,Mary,3393456,First
2,John,4594023,Patricia,1564163,First
3,Robert,4571203,Jennifer,1465928,First
4,Michael,4315462,Linda,1448097,First
bash-4.1$
```

sed: substitute a word

```
Terminal
File Edit View Search Terminal Help
bash-4.1$ head -5 names.csv
Rank,Name,Number,Name,Number,
1,James,4792954,Mary,3393456,First
2,John,4594023,Patricia,1564163,First
3,Robert,4571203,Jennifer,1465928,First
4,Michael,4315462,Linda,1448097,First
bash-4.1$ head -5 names.csv | sed 's/First/Number 1/g'
Rank,Name,Number,Name,Number,
1,James,4792954,Mary,3393456,Number 1
2,John,4594023,Patricia,1564163,Number 1
3,Robert,4571203,Jennifer,1465928,Number 1
4,Michael,4315462,Linda,1448097,Number 1
bash-4.1$
```

awk: keep a column

```
Terminal
File Edit View Search Terminal Help
bash-4.1$ sed 's/,/ /g' names.csv | head
Rank Name Number Name Number
1 James 4792954 Mary 3393456 First
2 John 4594023 Patricia 1564163 First
3 Robert 4571203 Jennifer 1465928 First
4 Michael 4315462 Linda 1448097 First
5 William 3662399 Elizabeth 1443415 First
6 David 3557293 Barbara 1410059 First
7 Richard 2487983 Susan 1106071 First
8 Joseph 2384205 Jessica 1043436 First
9 Thomas 2174023 Sarah 996554 First
bash-4.1$ █
```

```
Terminal
File Edit View Search Terminal Help
bash-4.1$ sed 's/,/ /g' names.csv | awk '{print $2}' | head
Name
James
John
Robert
Michael
William
David
Richard
Joseph
Thomas
bash-4.1$ █
```

sort(1) - Linux manual page

man7.org/linux/man-pages/man1/sort.1.html

man7.org > Linux > man-pages Linux/UNIX system programming training

[NAME](#) | [SYNOPSIS](#) | [DESCRIPTION](#) | [AUTHOR](#) | [REPORTING BUGS](#) | [COPYRIGHT](#) | [SEE ALSO](#) | [COLOPHON](#)

Search online pages

SORT(1) **User Commands** **SORT(1)**

NAME [top](#)

sort - sort lines of text files

SYNOPSIS [top](#)

```
sort [OPTION]... [FILE]...
sort [OPTION]... --files0-from=F
```

DESCRIPTION [top](#)

Write sorted concatenation of all FILE(s) to standard output.

With no FILE, or when FILE is -, read standard input.

Mandatory arguments to long options are mandatory for short options too. Ordering options:

-b, --ignore-leading-blanks
ignore leading blanks

native byte values.

AUTHOR [top](#)

Written by Mike Haertel and Paul Eggert.

REPORTING BUGS [top](#)

GNU coreutils online help: <<https://www.gnu.org/software/coreutils/>>
Report sort translation bugs to
<<https://translationproject.org/team/>>

COPYRIGHT [top](#)

Copyright © 2018 Free Software Foundation, Inc. License GPLv3+: GNU
GPL version 3 or later <<https://gnu.org/licenses/gpl.html>>.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

SEE ALSO [top](#)

shuf(1), uniq(1)

Full documentation at: <<https://www.gnu.org/software/coreutils/sort>>
or available locally via: info '(coreutils) sort invocation'

COLOPHON [top](#)



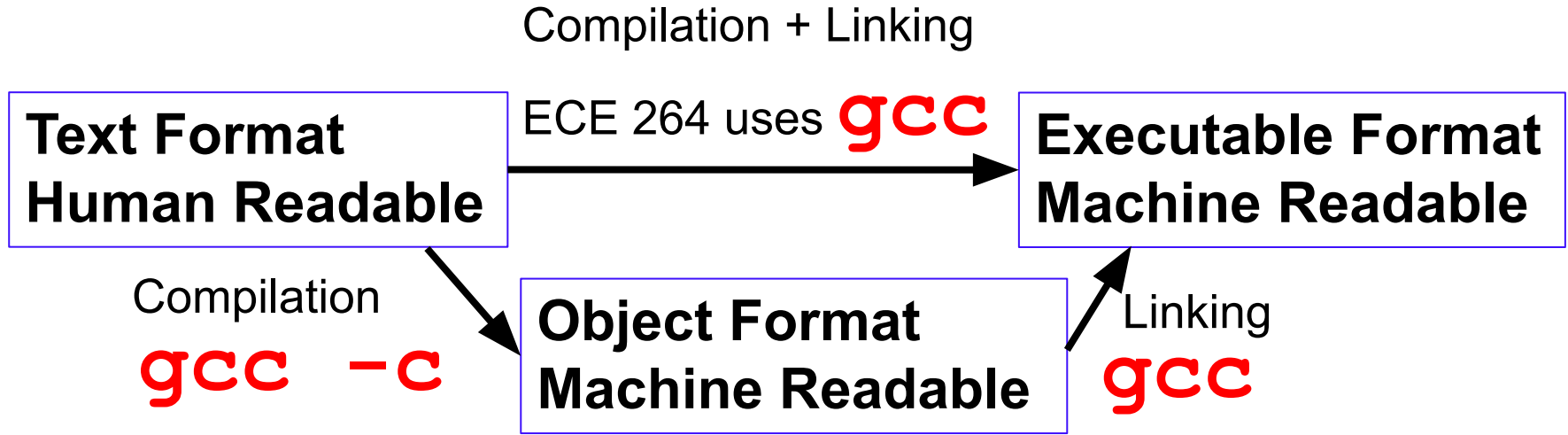
Enjoy Linux



Components of a C Program

```
/*
 * first.c
 *
 * Created by yunglu@purdue.edu
 *
 */
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char * * argv)
{
    printf("Hello C\n");
    return EXIT_SUCCESS;
}
```

C Programs has three formats

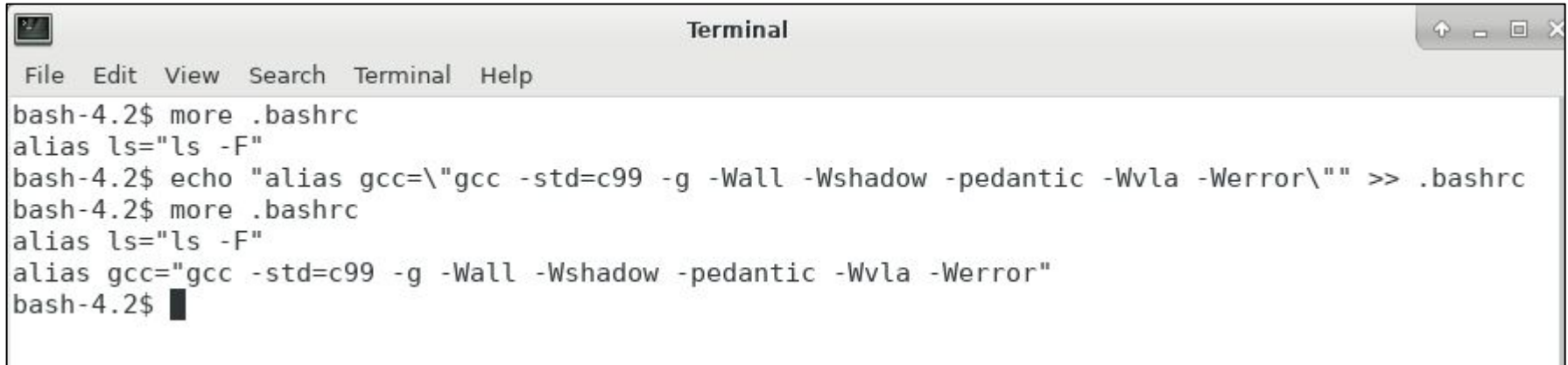


These formats allow the same programs (text format) to run on different types of machines.

gcc compiler

- Convert text file (human readable) to executable file (machine)
- Detect likely mistakes if you ask gcc to do that
- gcc **-std=c99 -g -Wall -Wshadow -pedantic -Wvla -Werror**
- -std=c99: using the C standard announced in 1999
- -g : enabling debugging
- -Wall : enable warning messages
- -Wshadow : detecting shadow variables
- -pedantic : restricting standard C
- -Wvla : detecting variable length array
- -Werror : treating warnings as errors

create an alias for gcc



```
Terminal
File Edit View Search Terminal Help
bash-4.2$ more .bashrc
alias ls="ls -F"
bash-4.2$ echo "alias gcc=\"gcc -std=c99 -g -Wall -Wshadow -pedantic -Wvla -Werror\"" >> .bashrc
bash-4.2$ more .bashrc
alias ls="ls -F"
alias gcc="gcc -std=c99 -g -Wall -Wshadow -pedantic -Wvla -Werror"
bash-4.2$ █
```

Warning: unused variable

Compiler warnings are your “first-line of defense” detecting erroneous code.

Unused variables are likely caused by mistyping.

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char * * argv)
{
    int counter;
    int unused; ←
    for (counter = 0; counter < 10; counter ++ )
    {
        printf("%d\n", counter);
    }
    return EXIT_SUCCESS;
}
```

```
bash-4.2$ gcc prog02.c
prog02.c: In function 'main':
prog02.c:6:7: error: unused variable 'unused' [-Werror=unused-variable]
    int unused;
    ^
cc1: all warnings being treated as errors
bash-4.2$ █
```

Shadow Variables

```
bash-4.2$ gcc prog03.c
bash-4.2$ ./a.out
0
var = 3
1
var = 3
var = 5
```

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char * * argv)
{
    int counter;
    int var = 5;
    for (counter = 0; counter < 2; counter ++)
    {
        int var = 3;
        printf("%d\n", counter);
        printf("var = %d\n", var); // should be 3
    }
    printf("var = %d\n", var); // should be 5
    return EXIT_SUCCESS;
}
```

Shadow Variables

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char * * argv)
{
    int counter;
    int var = 5;
    for (counter = 0; counter < 2; counter++)
    {
        int var = 3;
        printf("%d\n", counter);
        printf("var = %d\n", var);
        var++;
    }
    printf("var = %d\n", var);
    return EXIT_SUCCESS;
}
```

Shadow Variables

```
bash-4.2$ gcc prog03.c
bash-4.2$ ./a.out
0
var = 3
1
var = 3
var = 5
```

**without gcc
warning**

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char * * argv)
{
    int counter;
    int var = 5;
    for (counter = 0; counter < 2; counter++)
    {
        int var = 3;
        printf("%d\n", counter);
        printf("var = %d\n", var);
        var++;
    }
    printf("var = %d\n", var);
    return EXIT_SUCCESS;
}
```

Shadow Variables

```
bash-4.2$ gcc prog03.c
bash-4.2$ ./a.out
0
var = 3
1
var = 3
var = 5
```

without gcc warning

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char * * argv)
{
    int counter;
    int var = 5;
    for (counter = 0; counter < 2; counter++)
    {
        int var = 3;
        printf("%d\n", counter);
        printf("var = %d\n", var);
        var++;
    }
    printf("var = %d\n", var);
    return EXIT_SUCCESS;
}
```

```
bash-4.2$ gcc prog03.c
prog03.c: In function 'main':
prog03.c:9:11: error: declaration of 'var' shadows a previous local [-Werror=shadow]
    int var = 3;
        ^
prog03.c:6:7: error: shadowed declaration is here [-Werror=shadow]
    int var = 5;
        ^
cc1: all warnings being treated as errors
```

with gcc warning



Terminal

File Edit View Search Terminal Help

```
bash-4.2$ ls
```

```
prog01.c
```

```
bash-4.2$ gcc prog01.c
```


```
bash-4.2$ ls
```

```
a.out prog01.c
```

```
bash-4.2$ █
```




```
Terminal
File Edit View Search Terminal Help
bash-4.2$ ls
prog01.c
bash-4.2$ gcc prog01.c
bash-4.2$ ls
a.out prog01.c
bash-4.2$
```



```
Terminal
File Edit View Search Terminal Help
bash-4.2$ ls
prog01.c
bash-4.2$ gcc prog01.c
bash-4.2$ ls
a.out prog01.c
bash-4.2$ file prog01.c
prog01.c: C source, ASCII text
bash-4.2$ file a.out
a.out: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked (
uses shared libs), for GNU/Linux 2.6.32, BuildID[sha1]=9ed96d95061cf2d5258f13e0b
bcba8fc827a595c, not stripped
bash-4.2$
```



```
Terminal
File Edit View Search Terminal Help
bash-4.2$ ls
prog01.c
bash-4.2$ gcc prog01.c
bash-4.2$ ls
a.out prog01.c
bash-4.2$ file prog01.c
prog01.c: C source, ASCII text
bash-4.2$ file a.out
a.out: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked (
uses shared libs), for GNU/Linux 2.6.32, BuildID[sha1]=9ed96d95061cf2d5258f13e0b
bcba8fc827a595c, not stripped
bash-4.2$ ./a.out
0
1
2
3
4
5
6
7
8
9
bash-4.2$ █
```



**./ ensures that you are running
the program in this directory**

```
Terminal
File Edit View Search Terminal Help
bash-4.2$ gcc prog01.c -o myprog01
bash-4.2$ ls
a.out myprog01 prog01
bash-4.2$ file myprog01
myprog01: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked (uses shared libs), for GNU/Linux 2.6.32, BuildID[sha1]=9ed96d95061cf2d5258f13e0bbcba8fc827a595c, not stripped
bash-4.2$ ./myprog01
0
1
2
3
4
5
6
7
8
9
bash-4.2$
```

-o (lower case) specifies the name of the executable



```
Terminal
File Edit View Search Terminal Help
bash-4.2$ gcc prog01.c -o myprog01
bash-4.2$ ls
a.out myprog01 prog01
bash-4.2$ file myprog01
myprog01: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked (uses shared libs), for GNU/Linux 2.6.32, BuildID[sha1]=9ed96d95061cf2d5258f13e0bbcb8a8fc827a595c, not stripped
bash-4.2$ ./myprog01
0
1
2
3
4
5
6
7
8
9
bash-4.2$
```

-o (lower case) specifies the name of the executable

Caution:

- Do not call your executable “test”. Test is a Linux command.
- Do not put your text file after -o.

Doing so **erases** your text file.

conditional compilation

Notice the difference
ifdef and
ifndef (with n)

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char * * argv)
{
#ifdef DEBUG
    printf("This is debugging message\n");
#endif
#ifndef DEBUG
    printf("This is not debugging message\n");
#endif
    return EXIT_SUCCESS;
}
```



```
bash-4.2$ gcc -DDEBUG prog04.c
bash-4.2$ ./a.out
This is debugging message
bash-4.2$ gcc prog04.c
bash-4.2$ ./a.out
This is not debugging message
```

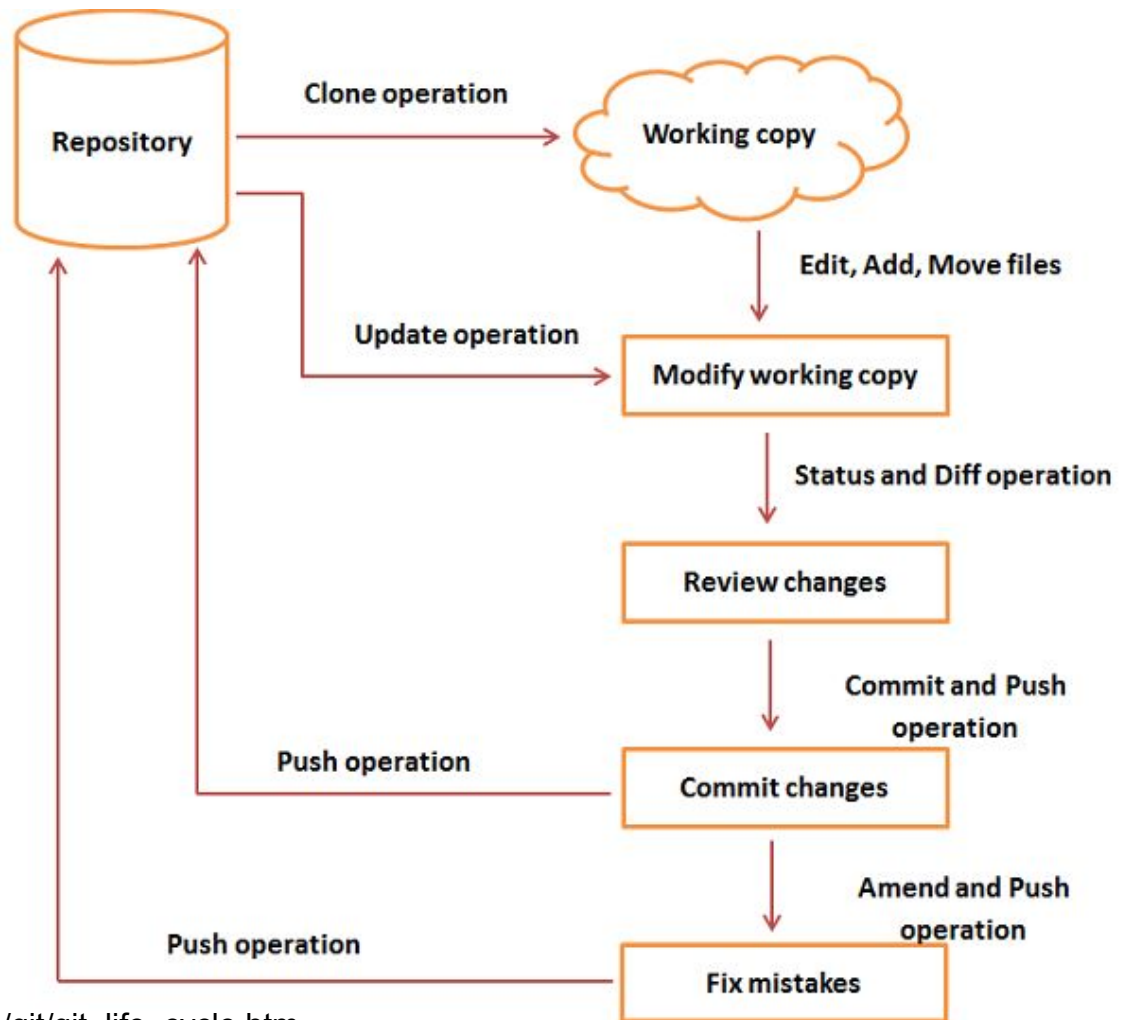
Version Control using `git`

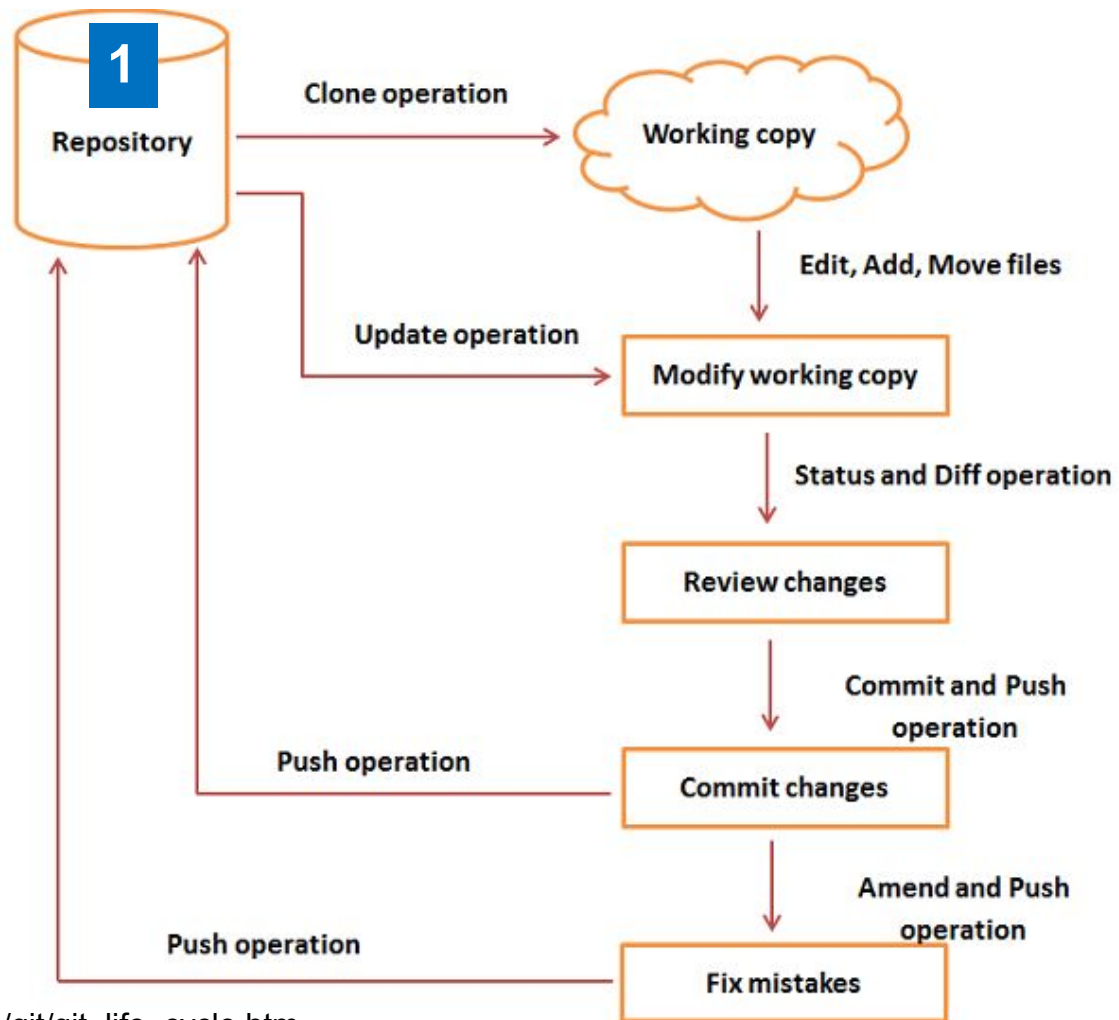
git vs github

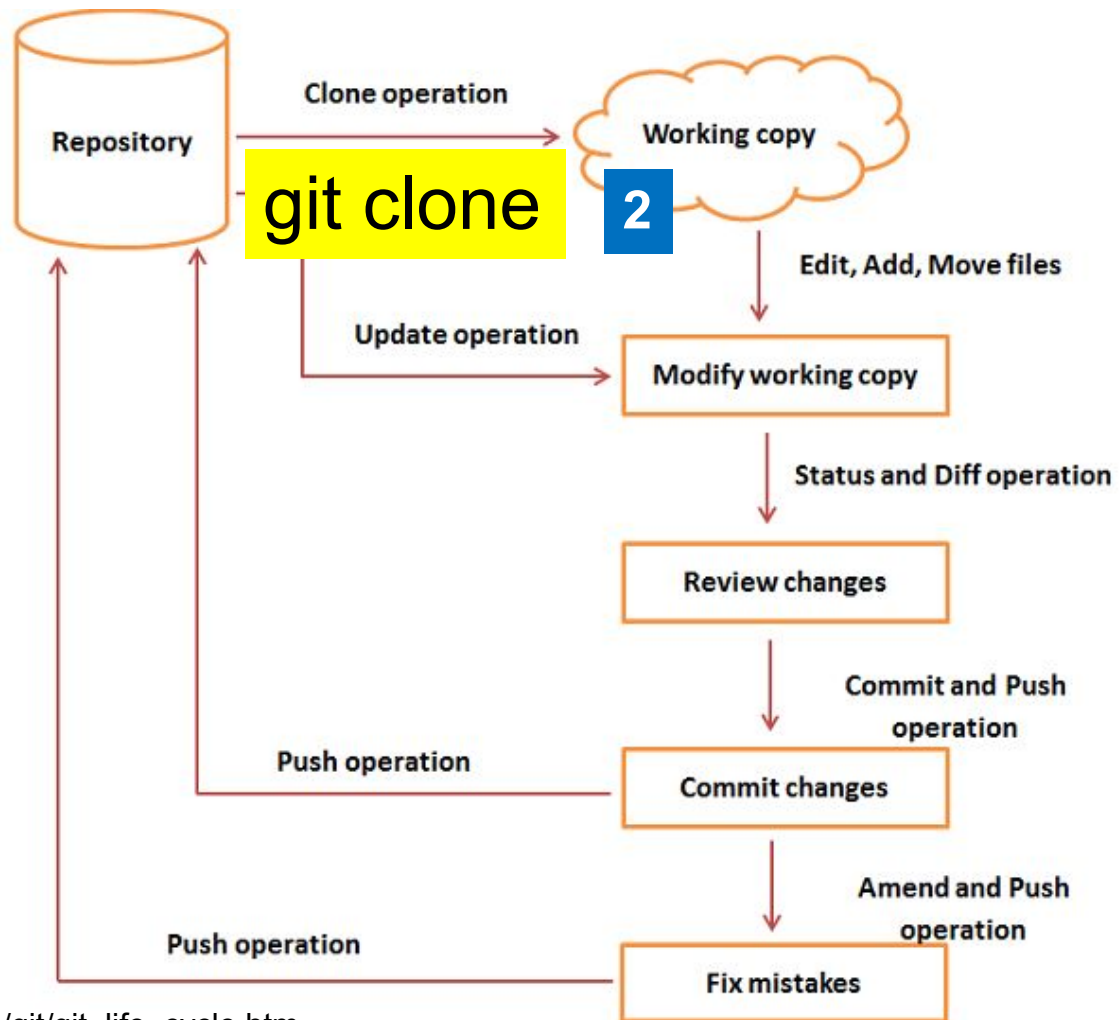
- git: distributed version control tool
- github: cloud service for git
- You may use git without using github
- ECE 264 uses github to distribute homework
- You need to use git to receive assignments
- You are encouraged to use version control for your own work; version control is not a requirement.

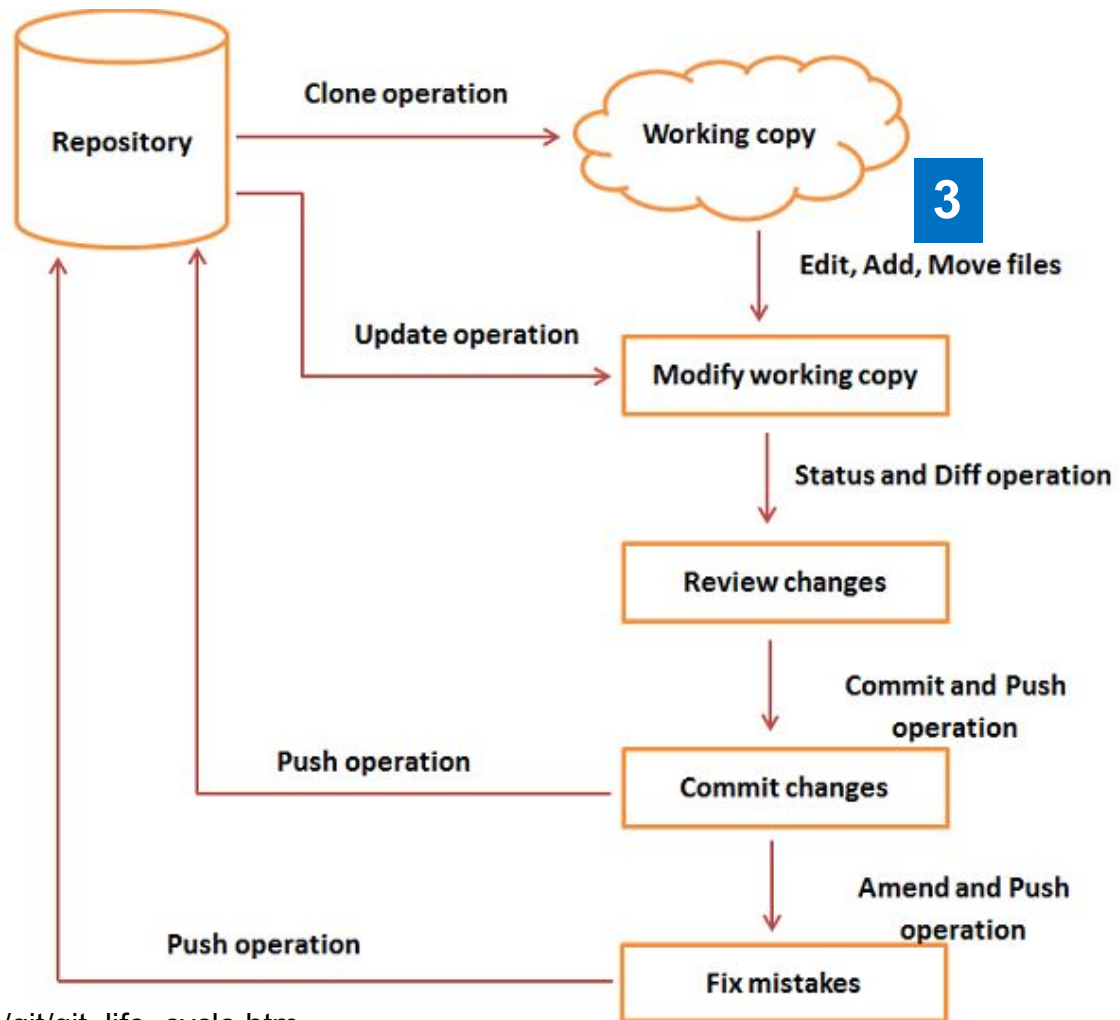
Why is github not required?

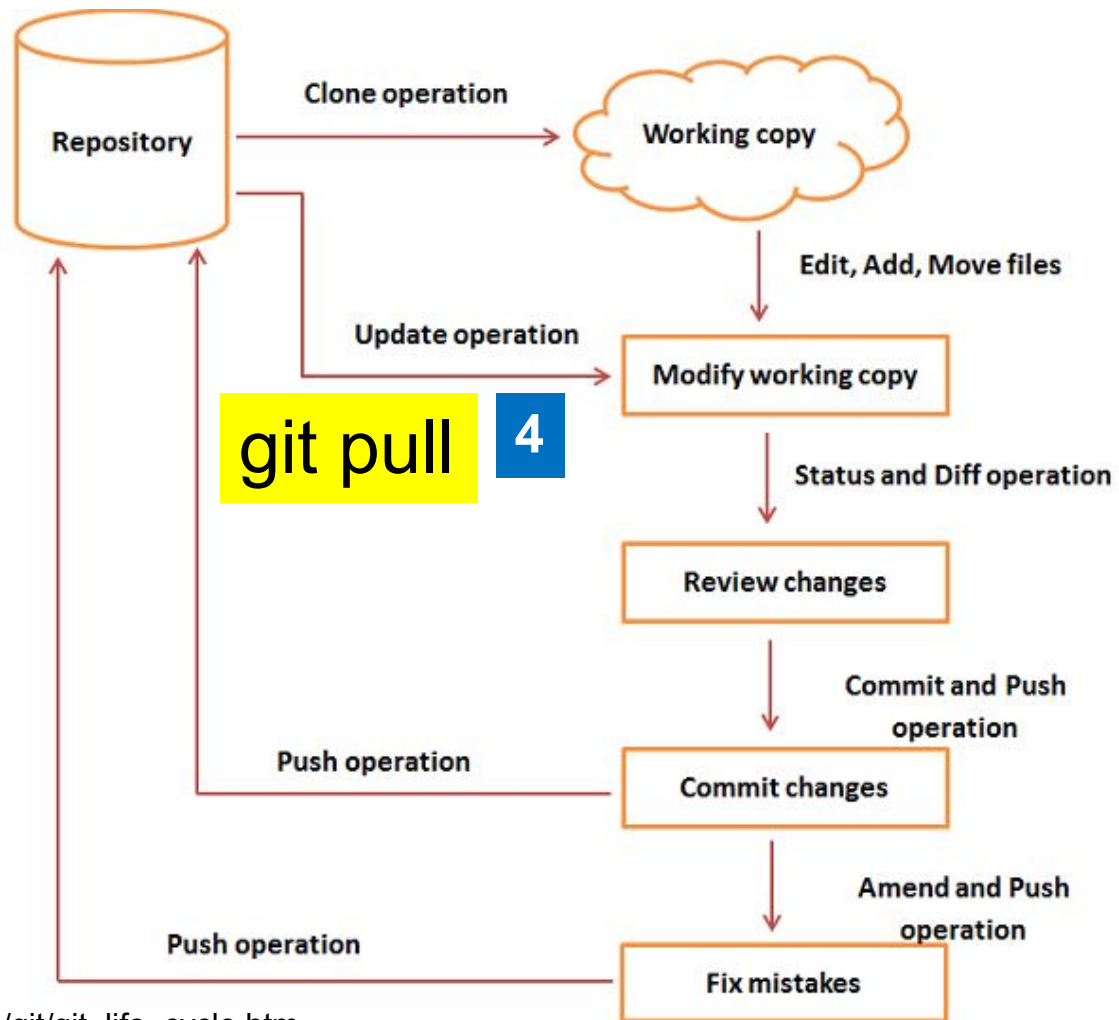
- github is a commercial entity and ECE 264 does not force students to use commercial services.
- If you decide to use github (or any other version control service), please ensure your work is ***private***.
- Public repository for ECE 264 is **academic dishonesty**.

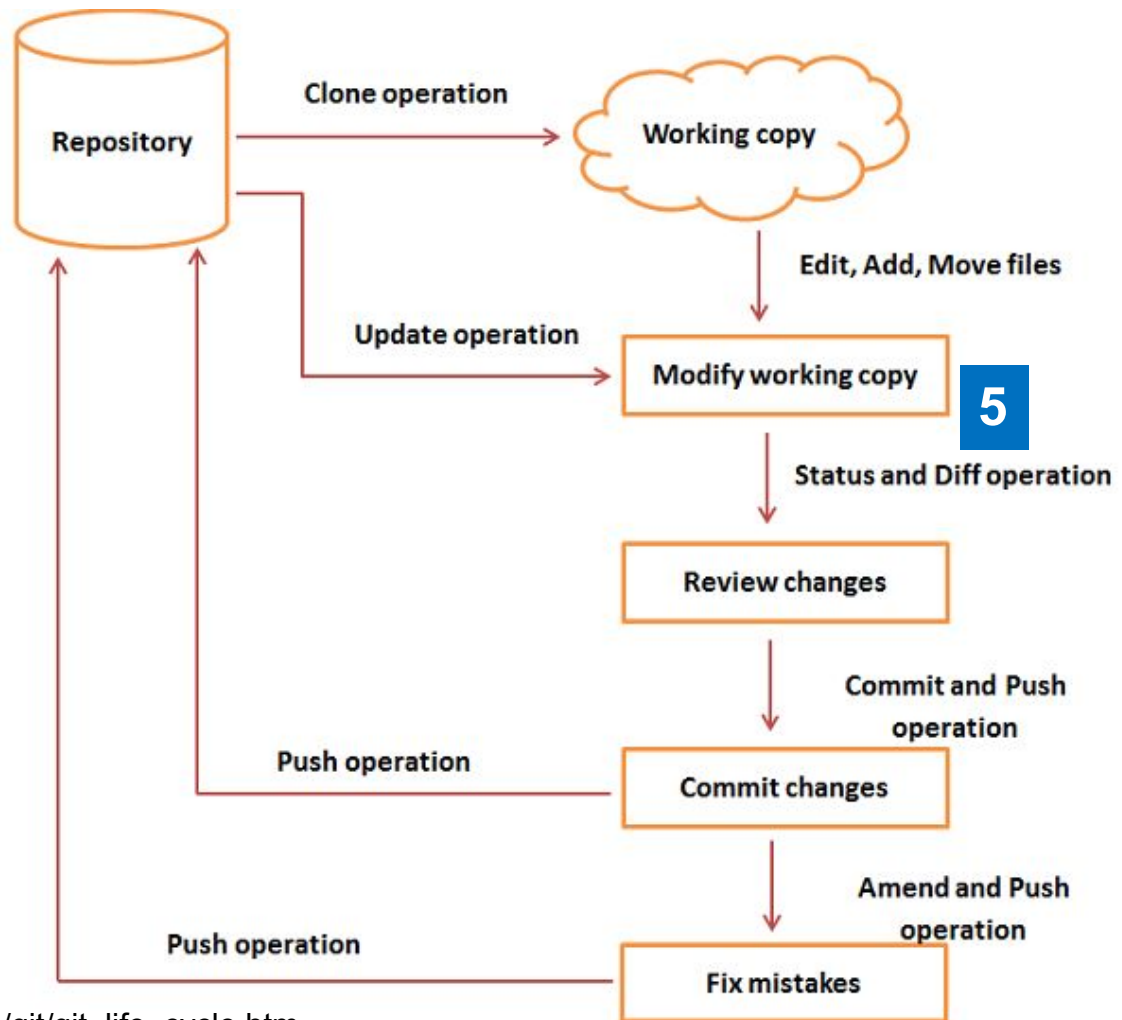


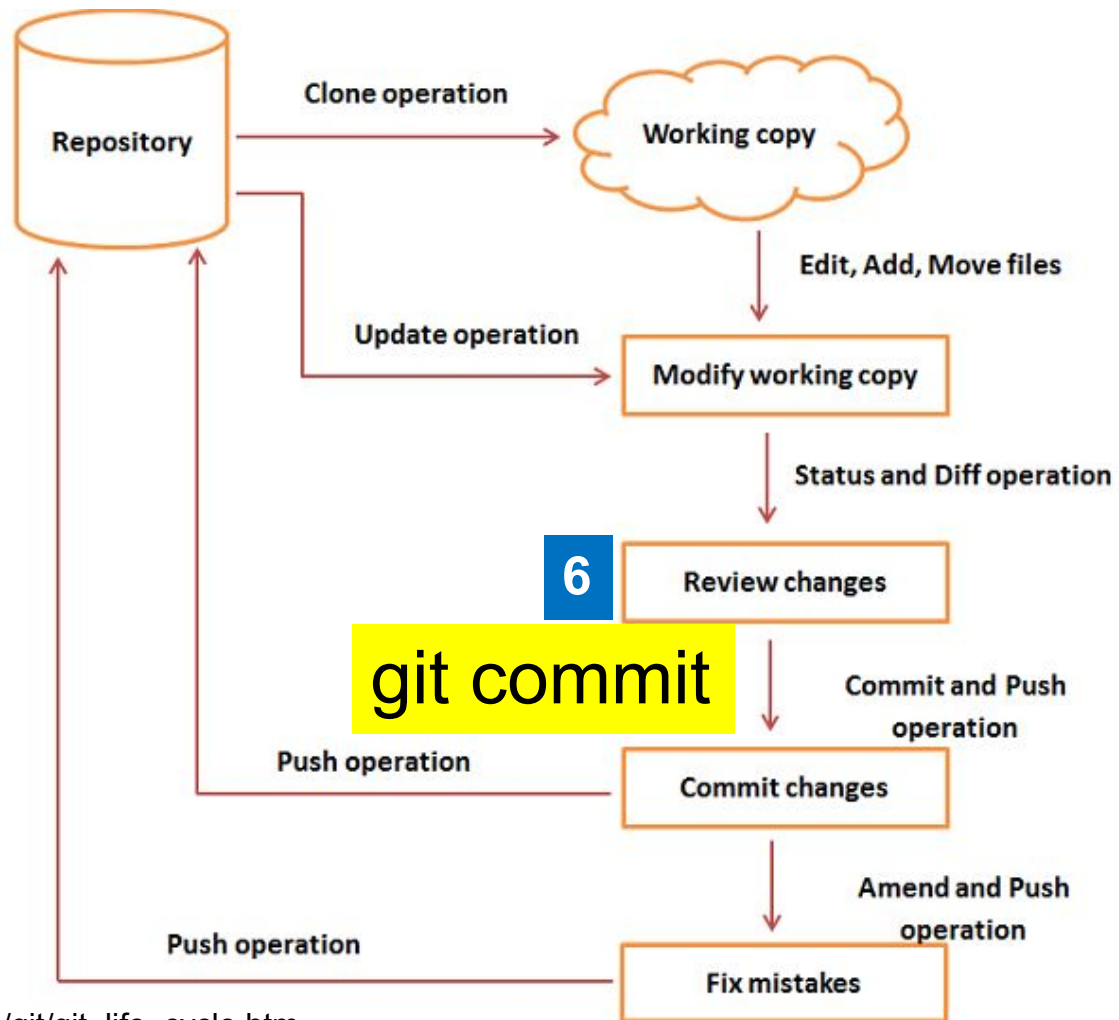


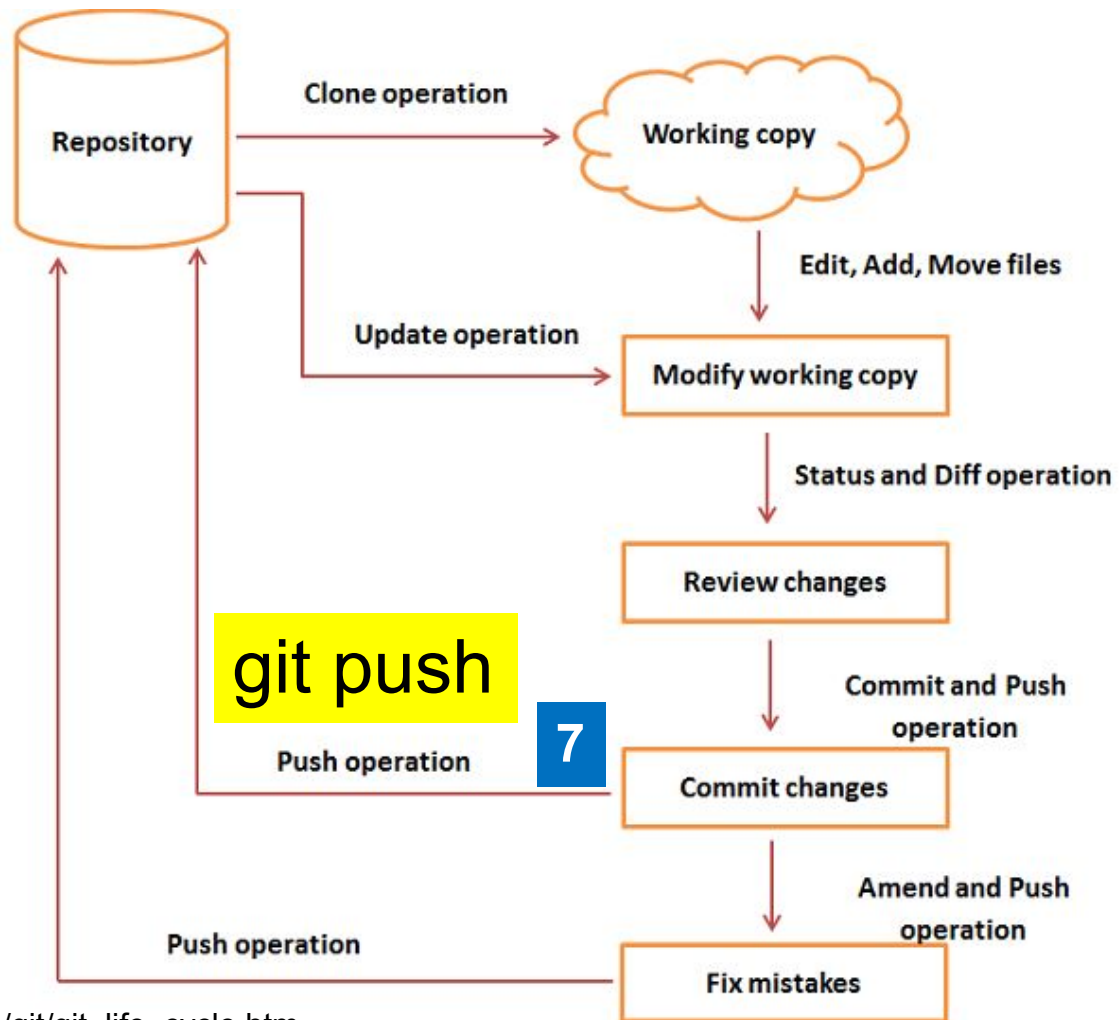




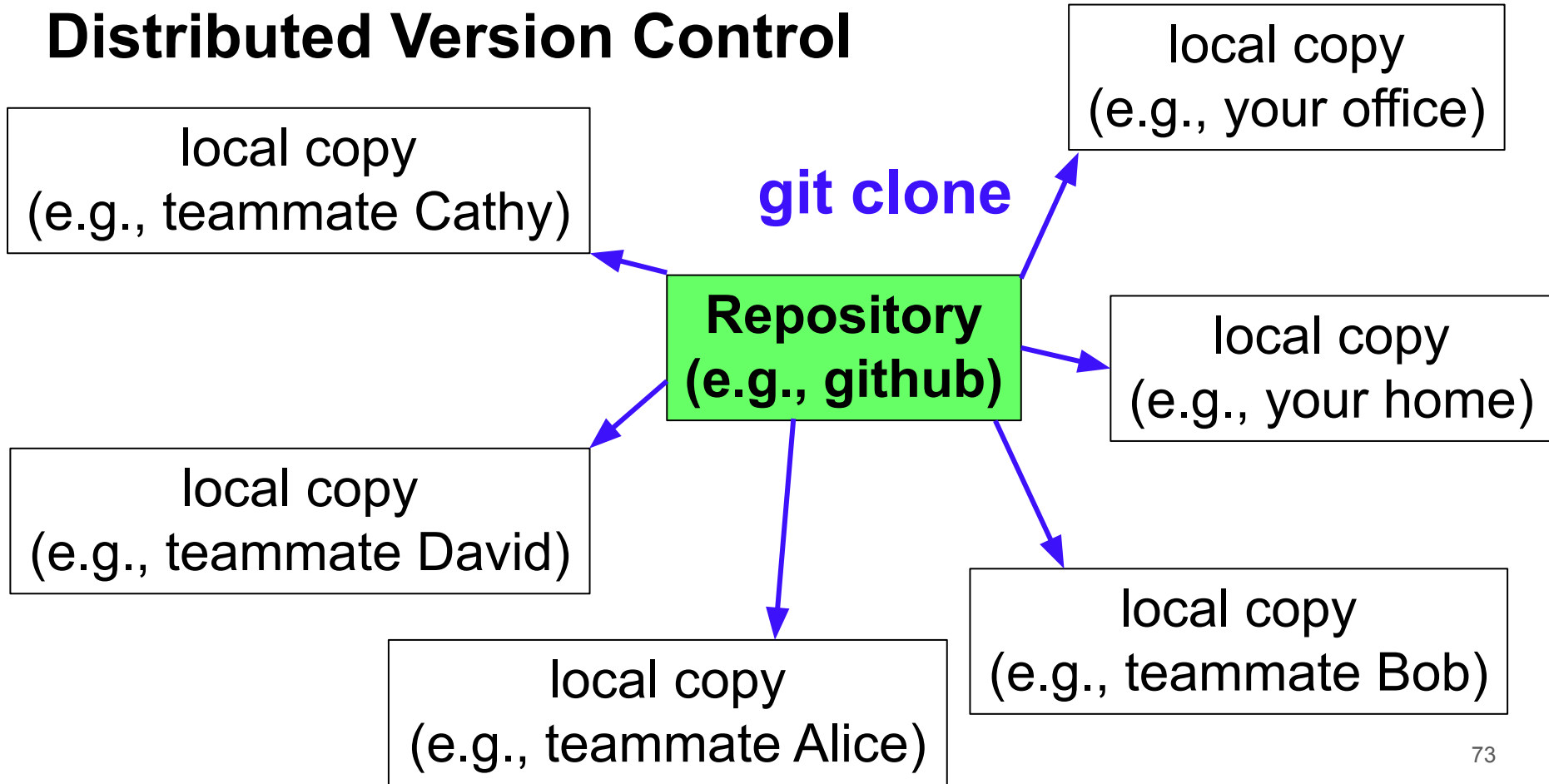








Distributed Version Control



Distributed Version Control

local copy
(e.g., teammate Cathy)

git commit

local copy
(e.g., your office)

git commit

**Repository
(e.g., github)**

local copy
(e.g., your home)

local copy
(e.g., teammate David)

local copy
(e.g., teammate Alice)

local copy
(e.g., teammate Bob)

Distributed Version Control

local copy
(e.g., teammate Cathy)

git push

local copy
(e.g., your office)

**Repository
(e.g., github)**

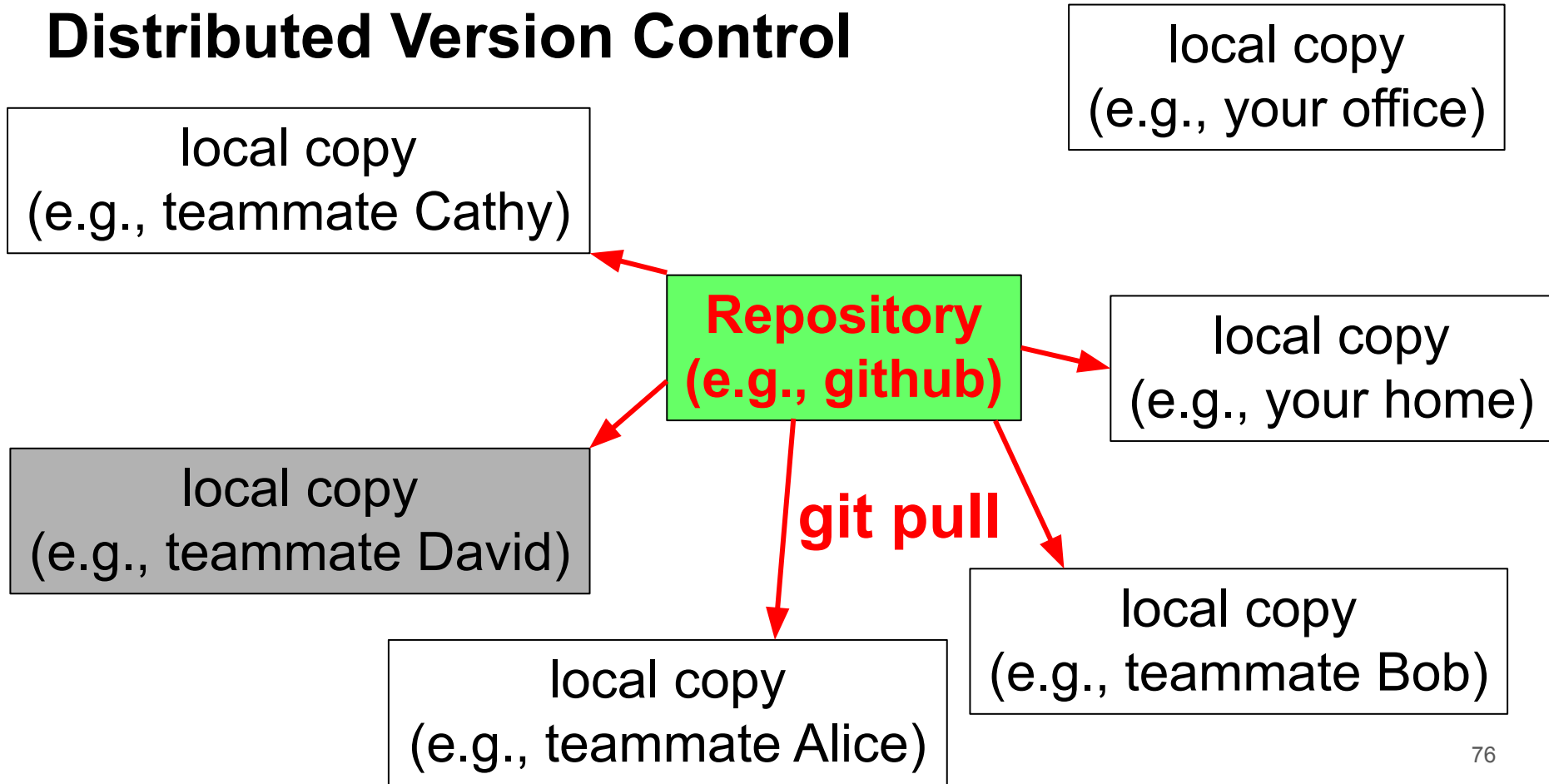
local copy
(e.g., your home)

local copy
(e.g., teammate David)

local copy
(e.g., teammate Alice)

local copy
(e.g., teammate Bob)

Distributed Version Control



git commands

- “git commit” creates on local computer a snapshot of the changes
- “git push” modifies the repository
- “git pull” obtains the latest changes in the repository

Use git effectively

- “git commit” and “git pull” often. If you work alone, “git push” often.
- If you work in a team, “git push” only working code.
- Do one thing at a time. Finish the work, test it, and push it.
- Do not commit or push too many changes at once.

Programming Assignments

We will be using GitHub classroom for all our programming assignments.

Knowledge of git is necessary to be successful in this course.

Programming Assignments

- Will be released in batches -- ~ 5 assignments at a time.
- Submission deadline: 1 assignment per week -- Refer course webpage for more details.
- No extensions.
- Other rules:

Refer: <https://purs3lab.github.io/ece264/labs/>

Accessing Programming Assignments

- Step 1: Fill up this form: <https://forms.gle/afjQoHubsh67GKA1A> by Tomorrow.
- Step 2 (After 13th Friday): Click on the link provided under each assignment to access the homework.

Ac

ECE264Spre

<> Code ⌵ Is



Join the classroom:

PurdueECE264-Spring2023

To join the GitHub Classroom for this course, please select yourself from the list below to associate your GitHub account with your school's identifier (i.e., your name, ID, or email).

Can't find your name? [Skip to the next step](#) →

Identifiers	
sefabot	>

om



Ac

ECE264Spre

<> Code ⌵ Is



PurdueECE264-Spring2023

Accept the assignment — purdueece264-spring2023-hw01

Once you accept this assignment, you will be granted access to the `purdueece264-spring2023-hw01-sefabot` repository in the [ECE264Spring2023](#) organization on GitHub.

Accept this assignment

om

Ac

ECE264Spri

<> Code ⌵ Is



You accepted the assignment, **purdueece264-spring2023-hw01** . We're configuring your repository now. This may take a few minutes to complete. Refresh this page to see updates.

📅 Your assignment is due by **Jan 23, 2023, 23:00 EST**

om




Ac



You're ready to go!

You accepted the assignment, **purdueece264-spring2023-hw01**.

Your assignment repository has been created:

 <https://github.com/ECE264Spring2023/purdueece264-spring2023-hw01-sefabot>

We've configured the repository associated with this assignment ([update](#)).

 Your assignment is due by **Jan 23, 2023, 23:00 EST**

ECE264Sprir

<> Code ⓘ Iss

oom

Accessing Programming Assignments

ECE264Spring2023 / **purdueece264-spring2023-hw01-sefabot** Private








[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Security](#) [Insights](#)

master 1 branch 0 tags

Go to file

Add file

Code

 github-classroom[bot] Initial commit	f8da84f 1 minute ago	🕒 1 commit
 testcases	Initial commit	1 minute ago
 Makefile	Initial commit	1 minute ago
 README.md	Initial commit	1 minute ago
 main.c	Initial commit	1 minute ago
 sort.c	Initial commit	1 minute ago
 sort.h	Initial commit	1 minute ago

☰ README.md 

Homework 1: git and make and gcc, Oh My!

Due 01/23 at 11:59pm

Goals

This homework serves several purposes:

About

purdueece264-spring2023-hw01-sefabot created by GitHub Classroom

📖 Readme

☆ 0 stars

👁 1 watching

🍴 0 forks

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

Languages

● C 87.9% ● Makefile 12.1%

Submitting Programming Assignments

```
git add sort.c  
git commit -m "Adding solution"
```

**Commit your changes
to local repo.**

Submitting Programming Assignments

```
git add sort.c  
git commit -m "Adding solution"
```

**Commit you changes
to local repo.**

```
git push
```

Push changes to your git repo.

Submitting Programming Assignments

```
git add sort.c  
git commit -m "Adding solution"
```

**Commit your changes
to local repo.**

```
git push
```

Push changes to your git repo.

```
git tag final_ver  
git push origin final_ver
```

**Tag your repo to say that it is
ready to grade.**

Important: if you don't tag, we don't grade!

Submitting Programming Assignments

```
git add sort.c  
git commit -m "Adding solution"
```

**Commit you changes
to local repo.**

```
git push
```

Push changes to your git repo.

```
git tag final_ver  
git push origin final_ver
```

**Tag your repo to say that it is
ready to grade.**

```
git tag final_ver -f  
git push origin final_ver -f
```

**If you make more changes..do
not forget to re_tag**

Accessing your grades

ECE264Spring2023 / **purdueece264-spring2023-hw01-sefabot** Private
generated from ECE264Spring2023/PurdueECE264-Spring2023-HW01-template

<> Code Issues Pull requests Actions Projects Security Insights Settings

master 1 branch 2 tags Go to file Add file <> Code

Machiry Grades updated for your homework. 25cdc4c 4 minutes ago 3 commits

testcases	Initial commit	2 days ago
Makefile	Initial commit	2 days ago
README.md	Initial commit	2 days ago
gradeReport.txt	Grades updated for your homework.	4 minutes ago
main.c	Initial commit	2 days ago
sort.c	Making submission	1 hour ago
sort.h	Initial commit	2 days ago

README.md

Homework 1: git and make and gcc, Oh My!

About purdueece264-spring2023-hw01-sefabot created by GitHub Classroom

- Readme
- 0 stars
- 1 watching
- 0 forks

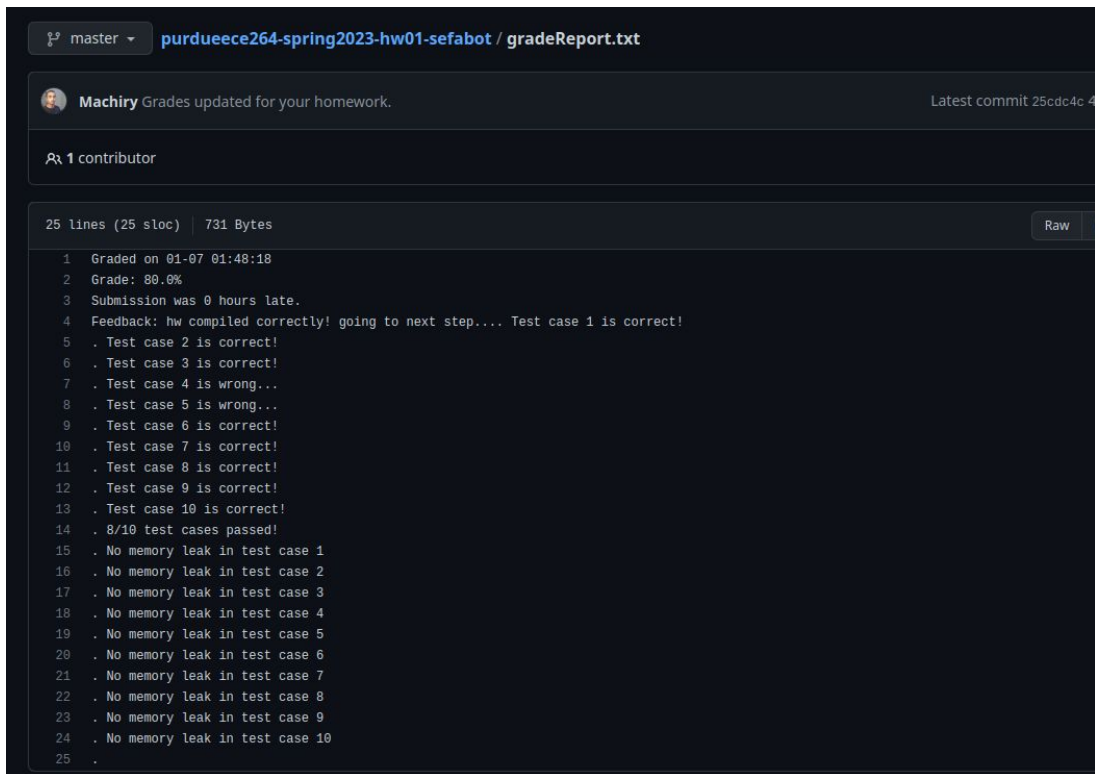
Releases 1

final_ver Latest
1 hour ago

Packages

No packages published
Publish your first package

Accessing your grades



The screenshot shows a GitHub file viewer interface for a file named `gradeReport.txt` in the repository `purdueece264-spring2023-hw01-sefabot`. The file is on the `master` branch. A commit by `Machiry` is shown with the message "Grades updated for your homework." and the commit hash `25cdc4c`. The file is 731 Bytes and 25 lines long. The content of the file is a text-based report with the following text:

```
1 Graded on 01-07 01:48:18
2 Grade: 80.0%
3 Submission was 0 hours late.
4 Feedback: hw compiled correctly! going to next step... Test case 1 is correct!
5 . Test case 2 is correct!
6 . Test case 3 is correct!
7 . Test case 4 is wrong...
8 . Test case 5 is wrong...
9 . Test case 6 is correct!
10 . Test case 7 is correct!
11 . Test case 8 is correct!
12 . Test case 9 is correct!
13 . Test case 10 is correct!
14 . 8/10 test cases passed!
15 . No memory leak in test case 1
16 . No memory leak in test case 2
17 . No memory leak in test case 3
18 . No memory leak in test case 4
19 . No memory leak in test case 5
20 . No memory leak in test case 6
21 . No memory leak in test case 7
22 . No memory leak in test case 8
23 . No memory leak in test case 9
24 . No memory leak in test case 10
25 .
```

Welcome to Computer Programming

It is challenging and you will enjoy it.