Systems@ETH zürich

Systems Programming and Computer Architecture (252-0061-00)

Exercise Session 01
Data Lab

Exercise Session

- My exercise session notes **!=** Official exercise session notes
- => exam relevant are only the slides of the official exercise session notes
- Mail: falkbe@ethz.ch
- Website: n.ethz.ch/~falkbe/

My slides vs normal slides

Systems@ETH zürich

Moving in the FHS

- pwd print working directory (where am I)
- ls list all elements within our dir
  - ls -al shows permissions and as list
  - ll often alias for “ls -l”
- cd change directory (to move)
  - cd /home/bfalk direct path
  - cd, cd ~ home directory (/home/bfalk)
  - cd /home/ directory change (down)
  - cd .. dir change (up)

Getting started

You will need a Linux compatible environment to solv

- Use the lab machines, they are running Linux (dua
- **Remote access the lab machines via ssh**
- **Use the Windows Subsystem for Linux (only for W**
- Use a Docker container
- Install Linux in a virtual machine

You can also setup your laptop for dual boot if you li

Dynamic Memory Allocation and data structures

2. Assembly x86-64, Compiling, Linking Loading

Basic x86 Architecture, Compiling C Source Code, Linking (Libraries), Floating Point (IEEE), Compilers

3. Computer Architecture

Processor Design, Exceptions, Virtual Memory, Devices Modern processor design (superscalar), Caches, Exceptions, Virtual Memory, Multiprocessing (Parallel Programming), Devices

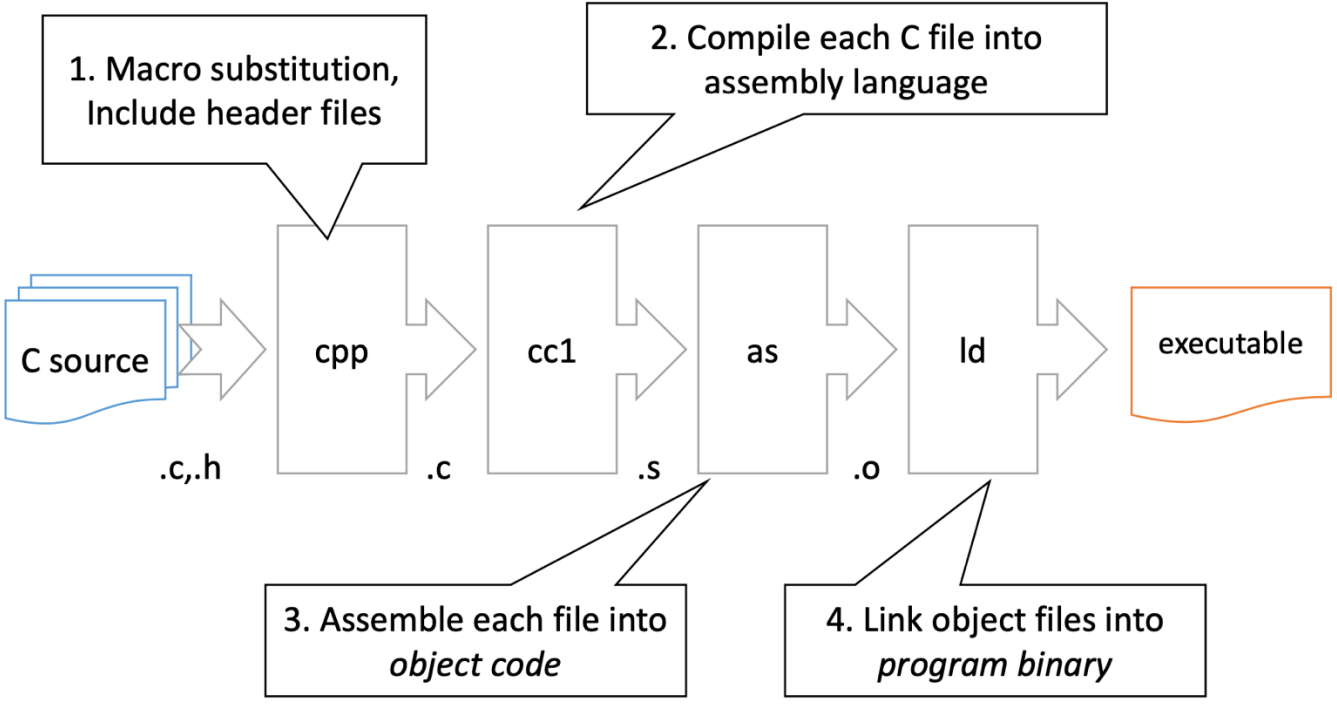
Programming Language C

Systems@ETH zürich

```c
#include <stdio.h>

int fact(int n){
  if(n == 0){
    return 1;
  }else{
    return (n * fact(n-1));
  }
}

int main(int argc, char* argv[]){
  int n,m;
  printf("Enter a number: ");
  scanf("%d", &n);
  m = fact(n);
  printf("Factorial of %d is %d.\n", n, m);
  return 0;
}
```

Assembly x86-64

GNU **gcc** Toolchain



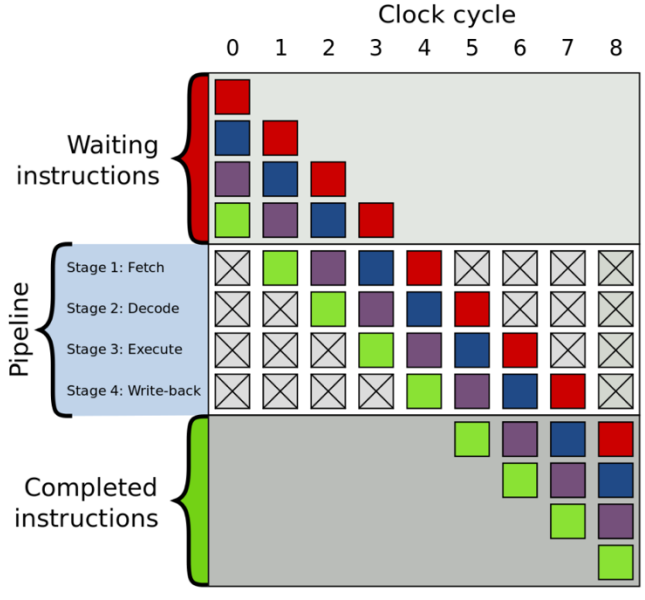
Assembly x86-64

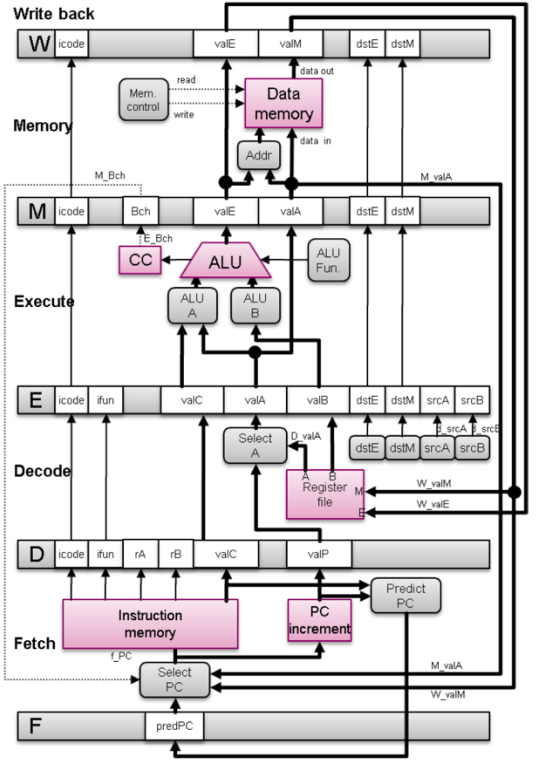
```c
#include <stdio.h>

int fact(int n){
  if(n == 0){
    return 1;
  }else{
    return (n * fact(n-1));
  }
}

int main(int argc, char* argv[]){
  int n,m;
  printf("Enter a number: ");
  scanf("%d", &n);
  m = fact(n);
  printf("Factorial of %d is %d.\n", n, m);
  return 0;
}
```

```
        .section        __TEXT,__text,regular,pure_instructions
        .build_version macos, 14, 0     sdk_version 14, 5
        .globl  _main                           ## -- Begin function main
        .p2align        4, 0x90
_main:                                  ## @main
        .cfi_startproc
## %bb.0:
        pushq   %rbp
        .cfi_def_cfa_offset 16
        .cfi_offset %rbp, -16
        movq    %rsp, %rbp
        .cfi_def_cfa_register %rbp
        subq    $32, %rsp
        movl    $0, -4(%rbp)
        movl    %edi, -8(%rbp)
        movq    %rsi, -16(%rbp)
        leaq    L_.str(%rip), %rdi
        movb    $0, %al
        callq   _printf
        movl    $1, -20(%rbp)
        movl    -20(%rbp), %esi
        leaq    L_.str.1(%rip), %rdi
        movb    $0, %al
        callq   _printf
        leaq    L_.str.2(%rip), %rdi
        leaq    L_.str.3(%rip), %rsi
        movb    $0, %al
        callq   _printf
        xorl    %eax, %eax
        addq    $32, %rsp
        popq    %rbp
        retq
        .cfi_endproc
                                        ## -- End function
        .section        __TEXT,__cstring,cstring_literals
L_.str:                                 ## @.str
        .asciz  "hello, world\n"

L_.str.1:                               ## @.str.1
        .asciz  "%d\n"

L_.str.2:                               ## @.str.2
        .asciz  "%s\n"

L_.str.3:                               ## @.str.3
        .asciz  "true"

.subsections_via_symbols
```

Computer Architecture

Pipelined hardware





Goal

Get familiar with bit level representations,
C and Linux

Outline

- Setting up your work environment
- Introduction to Linux
- Preview of the assignment
- Version Control (git)

Systems@ETH zürich

Setting up your work environment

Setting up Linux

Getting started

Systems@ETH zürich

You will need a Linux compatible environment to solve the exercises, either:

- Use the lab machines, they are running Linux (dual boot)
- **Remote access the lab machines via ssh**
- **Use the Windows Subsystem for Linux (only for Windows devices)**
- Use a Docker container
- Install Linux in a virtual machine

You can also setup your laptop for dual boot if you like or use Live Disks

Using ssh: maximus.inf.ethz.ch

Systems@ETH zürich

Every student of D-INFK can log in to *maximus.inf.ethz.ch*, which has the same Linux setup as the student labs.
https://www.isg.inf.ethz.ch/Main/HelpRemoteAccessSSH

We can access it using the **secure shell protocol**

- SSH creates a secure connection from one device to another (often over the terminal) which allows one to execute commands on the other device

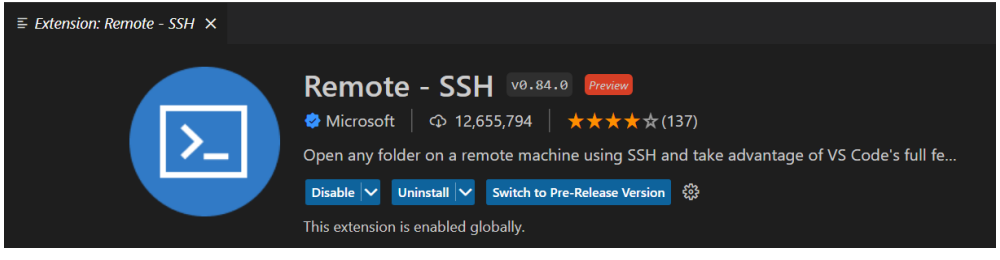
Using ssh: maximus.inf.ethz.ch

Systems@ETH zürich

VS Code Remote Setup Part 1

Install VS Code

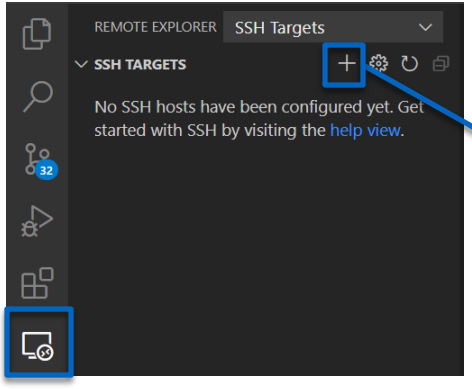
- Install Remote SSH Extension (in VS Code)



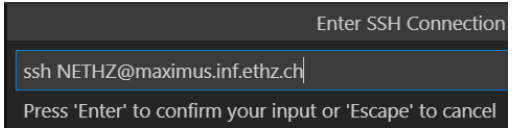
Using ssh: maximus.inf.ethz.ch

VS Code Remote Setup Part 2

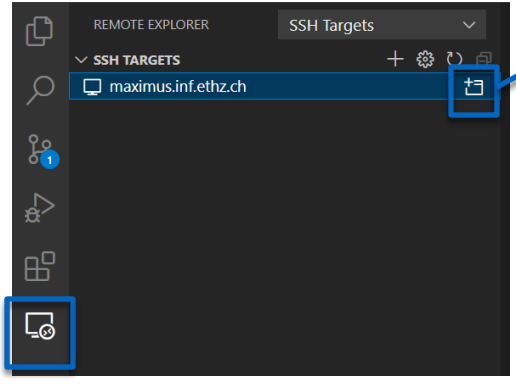
Add a new SSH Target



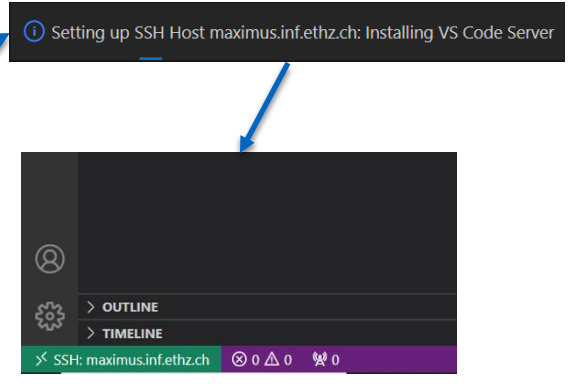
Replace NETHZ



- Connect to SSH Target



Login and Wait



Using ssh: config files and ssh keys

Systems@ETH zürich

- To reduce password prompting you can setup your ssh config file with a ssh key [optional but highly recommended]

Step 1: Key generation

- Open terminal/powershell
- Enter $ ssh-keygen into the terminal (without the dollar sign) and follow the prompts to generate your key
  - check if key pair is in ~/.ssh directory, else move it there
  - On Windows: ~/ corresponds to C:\Users\YOUR_USERNAME

Using a config file and ssh keys

Systems@ETH zürich

Step 2: move key to maximus

- Unix/MacOS:

  $ ssh-copy-id -i ~/.ssh/nameofkey.pub NETHZ@maximus.inf.ethz.ch
- Windows:
  - cat nameofkey.pub -> copy output
  - Connect to maximus and paste into ~/.ssh/authorized_keys
- Test if key was added successfully by running

$ ssh –i ~/.ssh/nameofkey NETHZ@maximus.inf.ethz.ch

Step 3: config file

- Open your config file under ~/.ssh/config
- Add the following lines:

```
1  Host maximus
2          HostName maximus.inf.ethz.ch
3          User NETHZ
4          IdentityFile ~/.ssh/id_rsa_nameofkey
```

Using a config file and ssh keys

Systems@ETH zürich

Step 4: SSH-ing into maximus

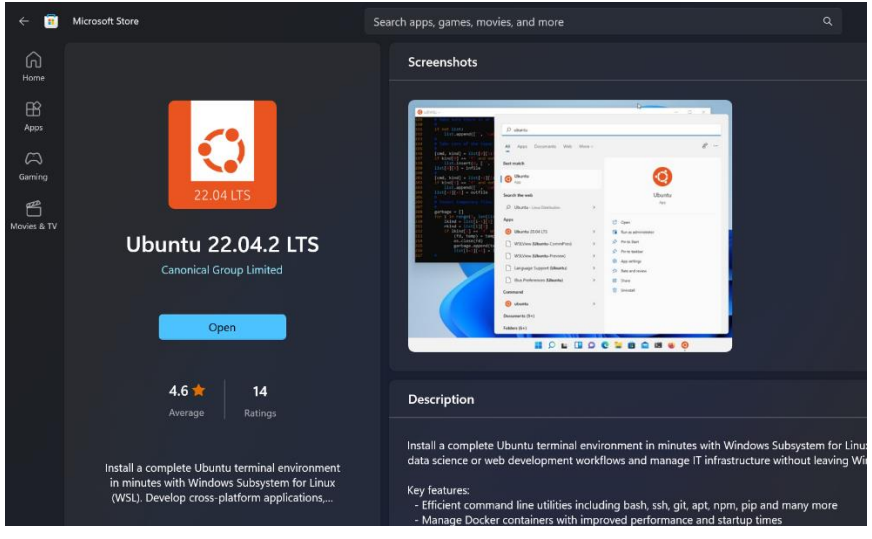
- If everything worked, you should now be able to ssh into maximus without having to enter your username and password every time
- Less time spent doing repetitive tasks -> more time for fun things (like SPCA) :D

Alternative Solution: WSL

The Windows Subsystem for Linux lets you run a GNU/Linux environment directly on Windows without the overhead of a traditional VM or dual boot setup

WSL Setup Part 1

- Install Ubuntu 22.04 LTS (Microsoft Store)

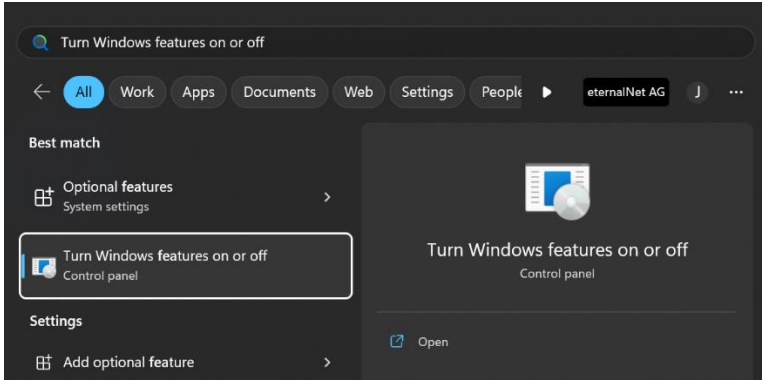


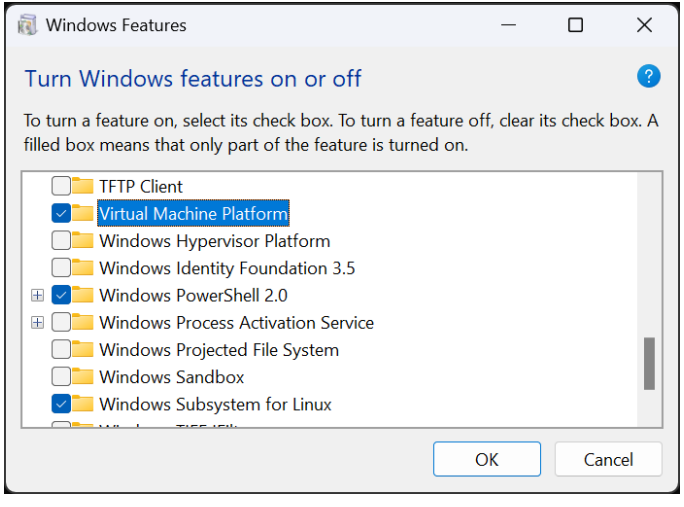
Alternative Solution: WSL

WSL Setup Part 2

- Make sure the Windows feature “Virtual Machine Platform” is enabled

- If this is not the case enable it by marking the checkbox and restart your device when asked



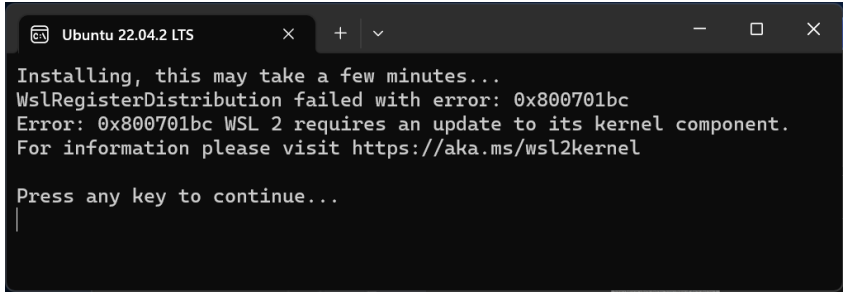


Alternative Solution: WSL

Systems@ETH zürich

WSL Setup Part 3

- Open Ubuntu 22.04
  (the one you installed in Part 1)



- You might encounter an the error message above. In this case install the kernel update. You can download it using this link:
  https://wslstorestorage.blob.core.windows.net/wslblob/wsl_update_x64.msi

- After the installation enter the following command:
  wsl --set-default-version 2

Alternative Solution: WSL

Systems@ETH zürich

WSL Setup Part 4

- You should now be able to successfully start Ubuntu 22.04 and enter Linux commands (described in the following slides)

- To open the current folder with VS Code enter the command:
  code .

- If you want to access your Windows files you can enter
  cd /mnt/c (c is the Windows drive letter)
  Only use this if you really have to, since it reduces performance!

Alternative Solution: Docker container

Systems@ETH zürich

- Install Docker
- Get the Docker file from https://moodle-app2.let.ethz.ch/mod/resource/view.php?id=1096662
- Follow instructions from https://polybox.ethz.ch/index.php/s/LojjBM9YtJLgLxV
- If you have an M1/M2 mac, don't forget

docker build --platform linux/amd64 -t sysprog

Alternative Solution: Virtual machine

Systems@ETH zürich

1. Download VirtualBox https://www.virtualbox.org/
2. Install VirtualBox on your machine
3. Obtain a copy of Ubuntu **22.04 LTS** http://www.ubuntu.com/
4. Create a new machine and install Ubuntu on it. https://docs.oracle.com/cd/E26217_01/E26796/html/qs-create-vm.html


www.virtualbox.org
IT Shop (ITSM)
Oracle VM VirtualBox
Enterprise Open Source and Linux | Ubuntu
6.3. Creating a New Virtual Machine in VirtualBox
VirtualBox
search...
Login
Preferences
Welcome to VirtualBox.org!
About
Screenshots
Downloads
Documentation
End-user docs
Technical docs
Contribute
Community
VirtualBox is a powerful x86 and AMD64/Intel64 virtualization product for enterprise as well as home use. Not only is VirtualBox an extremely feature rich, high performance product for enterprise customers, it is also the only professional solution that is freely available as Open Source Software under the terms of the GNU General Public License (GPL) version 2. See "About VirtualBox" for an introduction.
Presently, VirtualBox runs on Windows, Linux, Macintosh, and Solaris hosts and supports a large number of guest operating systems including but not limited to Windows (NT 4.0, 2000, XP, Server 2003, Vista, Windows 7, Windows 8, Windows 10), DOS/Windows 3.x, Linux (2.4, 2.6, 3.x and 4.x), Solaris and OpenSolaris, OS/2, and OpenBSD.
VirtualBox is being actively developed with frequent releases and has an ever growing list of features, supported guest operating systems and platforms it runs on. VirtualBox is a community effort backed by a dedicated company: everyone is encouraged to contribute while Oracle ensures the product always meets professional quality criteria.
Download VirtualBox 6.1
Hot picks:
Pre-built virtual machines for developers at Oracle Tech Network
Hyperbox Open-source Virtual Infrastructure Manager project site
phpVirtualBox AJAX web interface project site
ORACLE
Contact – Privacy policy – Terms of Use
News Flash
Important May 17th, 2021
We're hiring!
Looking for a new challenge? We're hiring a VirtualBox senior developer in 3D area (Europe/Russia/India).
New July 28th, 2021
VirtualBox 6.1.26 released!
Oracle today released a 6.1 maintenance release which improves stability and fixes regressions. See the Changelog for details.
New July 20th, 2021
VirtualBox 6.1.24 released!
Oracle today released a 6.1 maintenance release which improves stability and fixes regressions. See the Changelog for details.
New April 29th, 2021
VirtualBox 6.1.22 released!
Oracle today released a 6.1 maintenance release which improves stability and fixes regressions. See the Changelog for details.
New April 20th, 2021
VirtualBox 6.1.20 released!
Oracle today released a 6.1 maintenance release which improves stability and fixes regressions. See the Changelog for details.
New January 19th, 2021
VirtualBox 6.1.18 released!
Oracle today released a 6.1 maintenance release which improves stability and fixes regressions. See the Changelog for details.
New October 20th, 2020
VirtualBox 6.1.16 released!
Oracle today released a 6.1 maintenance release which improves stability and fixes regressions. See the Changelog for details.
New September 4th, 2020
VirtualBox 6.1.14 released!
Oracle today released a 6.1 maintenance release which improves stability and fixes regressions. See

Setting up the Virtual machine

Systems@ETH zürich

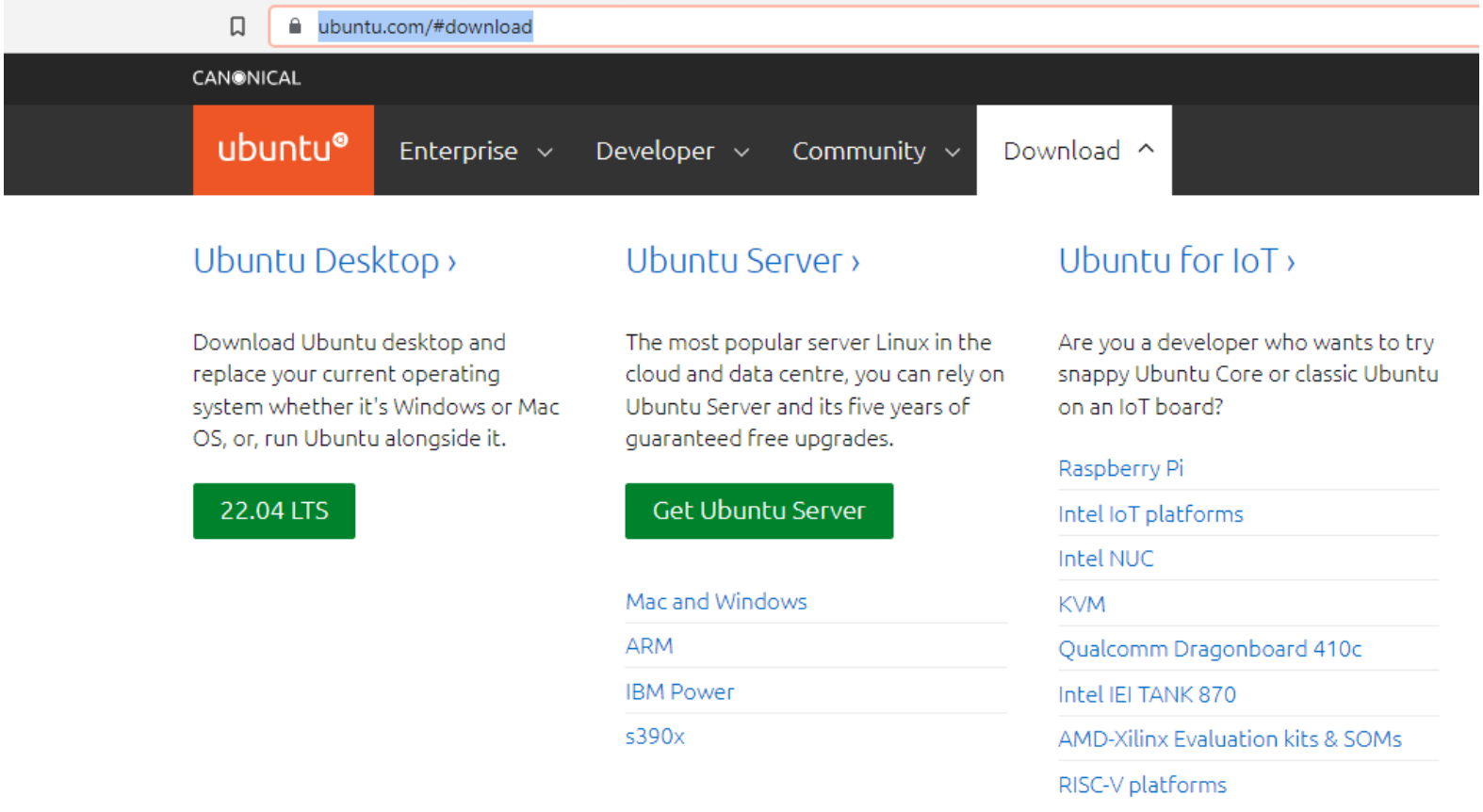
1. Download VirtualBox https://www.virtualbox.org/

2. Install VirtualBox on your machine

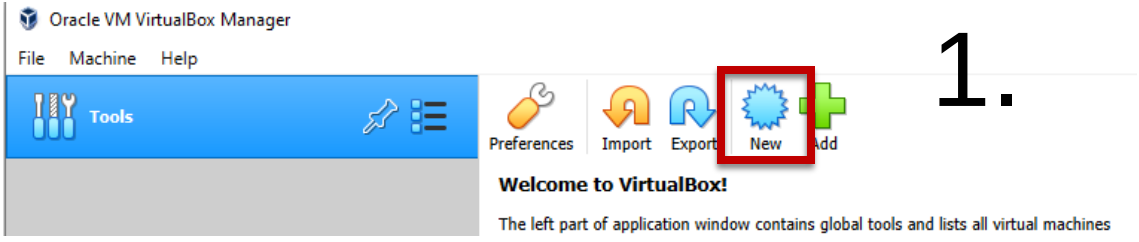
3. Obtain a copy of Ubuntu **22.04 LTS** http://www.ubuntu.com/

4. Create a new machine and install Ubuntu on it. https://docs.oracle.com/cd/E26217_01/E26796/html/qs-create-vm.html

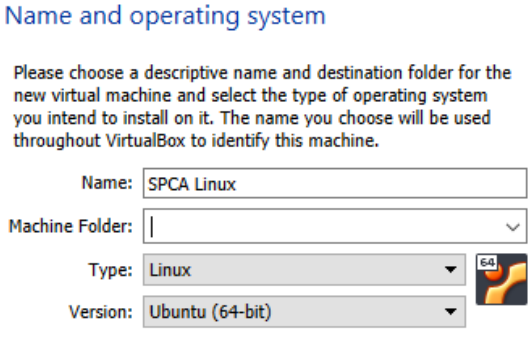
Systems@ETH zürich



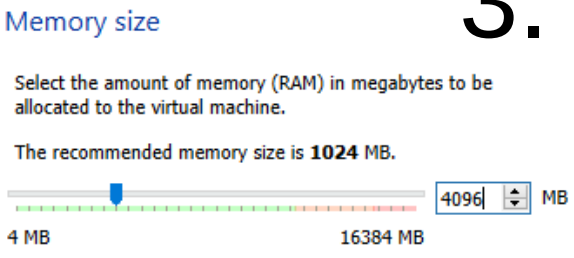
Create a new VM



1.



2.



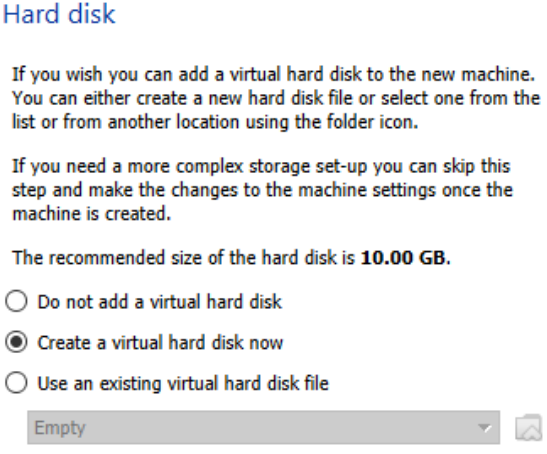
3.

amount depends on how much you have available

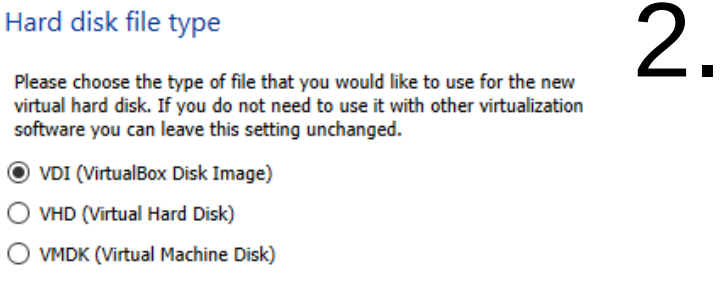
Create a virtual hard disk

Systems@ETH zürich

1.

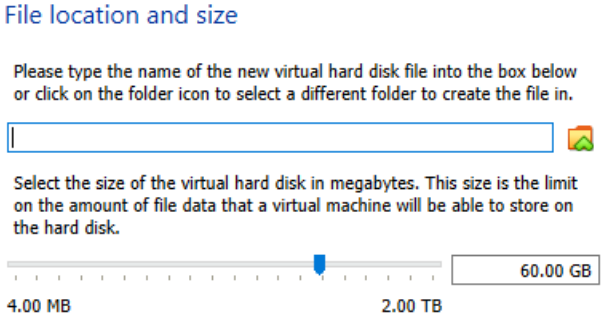


2.



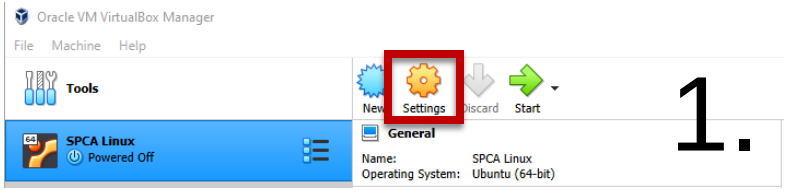
3.

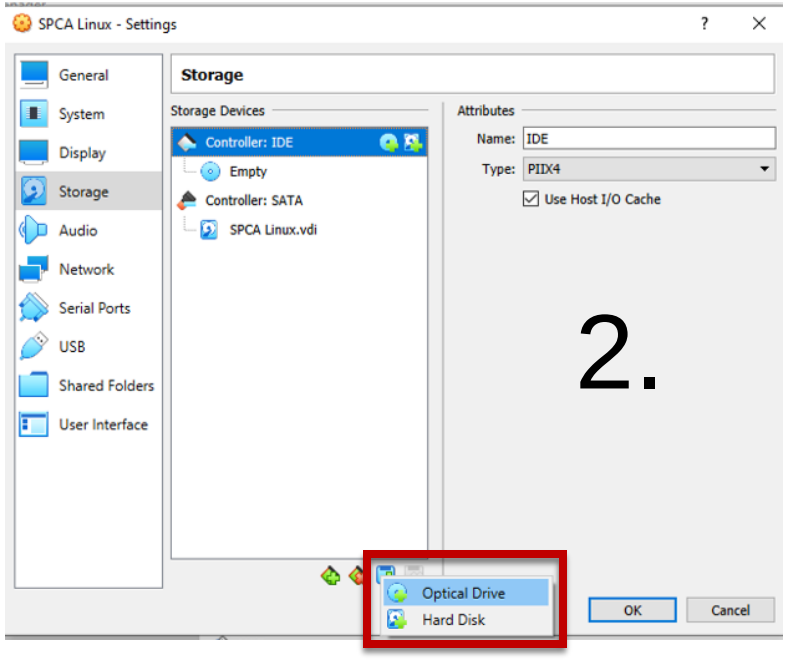
4.

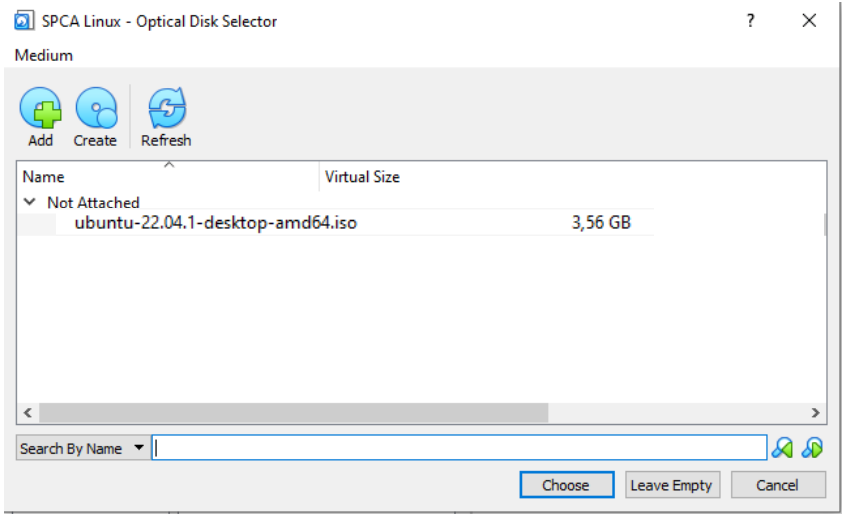


pick a location + size

Create a new VM: Setting the boot media

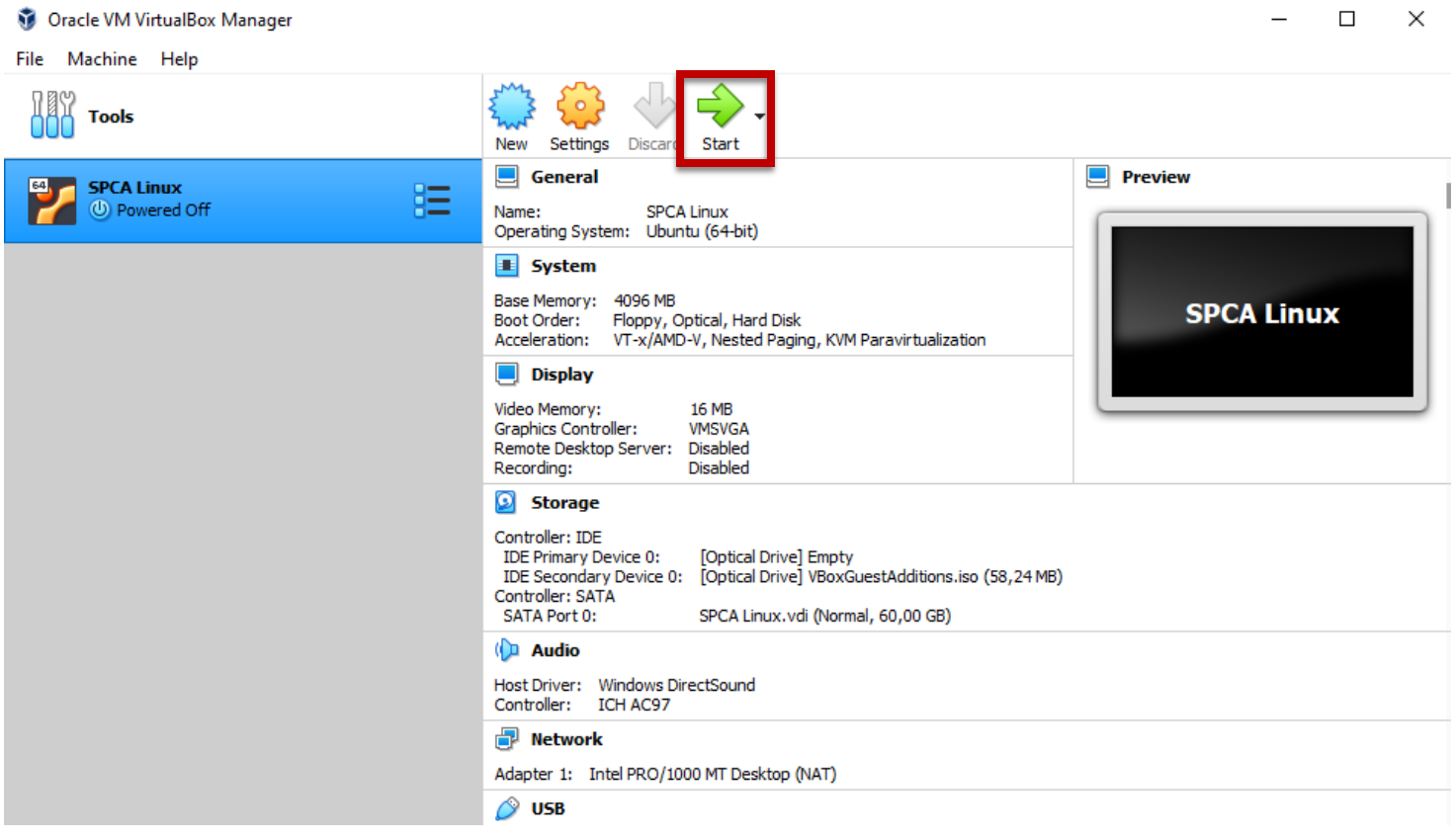




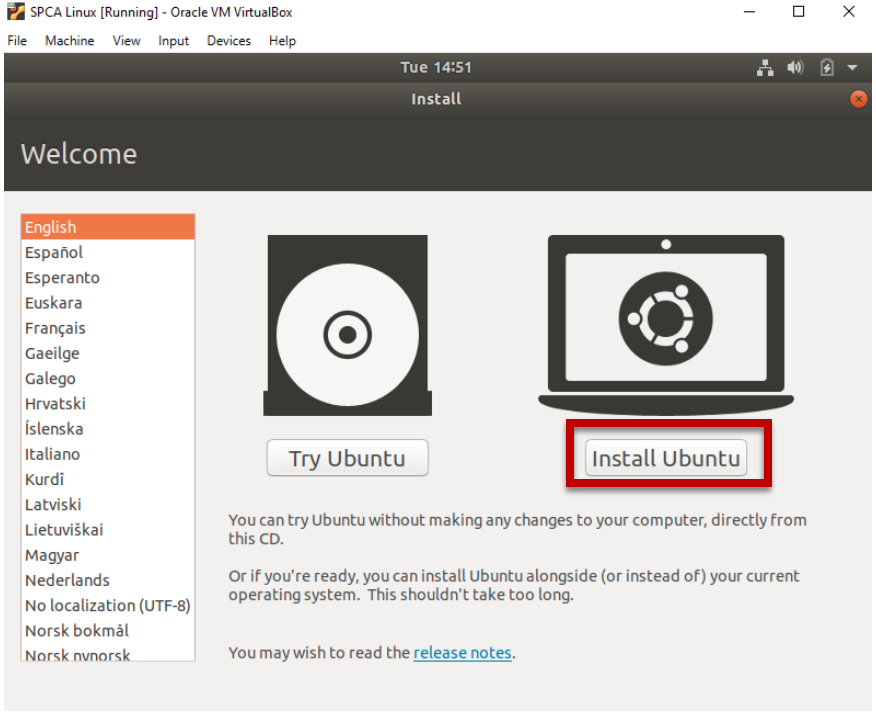


Select the downloaded Ubuntu ISO

Start the VM


Oracle VM VirtualBox Manager
File
Machine
Help
Tools
New
Settings
Discard
Start
SPCA Linux
Powered Off
General
Name: SPCA Linux
Operating System: Ubuntu (64-bit)
System
Base Memory: 4096 MB
Boot Order: Floppy, Optical, Hard Disk
Acceleration: VT-x/AMD-V, Nested Paging, KVM Paravirtualization
Display
Video Memory: 16 MB
Graphics Controller: VMSVGA
Remote Desktop Server: Disabled
Recording: Disabled
Preview
SPCA Linux
Storage
Controller: IDE
IDE Primary Device 0: [Optical Drive] Empty
IDE Secondary Device 0: [Optical Drive] VBoxGuestAdditions.iso (58,24 MB)
Controller: SATA
SATA Port 0: SPCA Linux.vdi (Normal, 60,00 GB)
Audio
Host Driver: Windows DirectSound
Controller: ICH AC97
Network
Adapter 1: Intel PRO/1000 MT Desktop (NAT)
USB

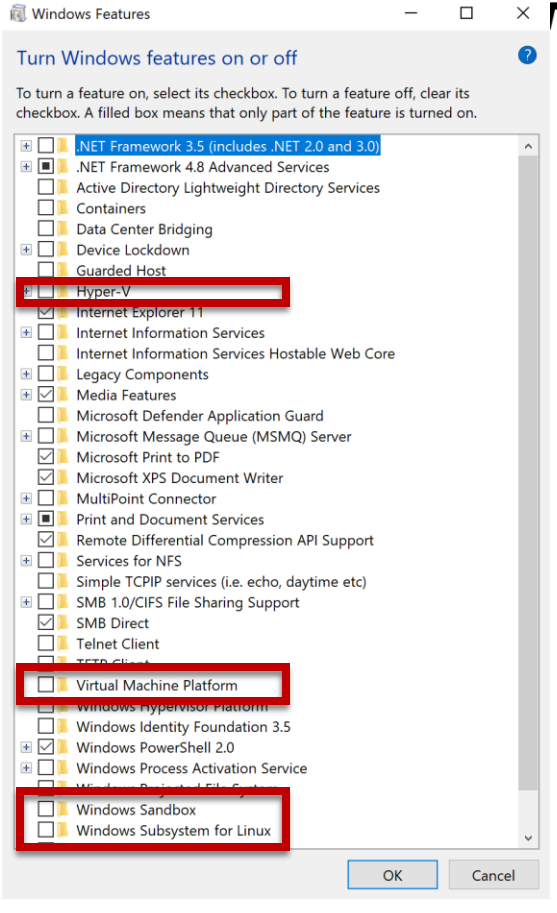
Follow the Ubuntu installation wizard



Troubleshooting: VM Setup

- If you have trouble installing Ubuntu 22.04 in Virtual Box, turn off the Windows Features:
  - Hyper-V
  - Virtual Machine Platform
  - Windows Sandbox
  - Windows Subsystem for Linux

https://stackoverflow.com/a/63229718



Optimal VirtualBox Settings

Systems@ETH zürich

1. Set your graphics controller to VBoxSVGA and 3D acceleration off for automatic resolution scaling with decent performance

2. Give your at least VM 4GB of RAM and 32MB of graphics memory, if possible

3. If your computer allows for it, give the VM two CPU cores.
4. For people using laptops: VMs use a lot of performance. Try to either be plugged in or, on Windows, set your energy options to the performance setting for a better experience.

Systems@ETH zürich

Introduction to Linux

Ubuntu 22.04 LTS

Terminal Introduction

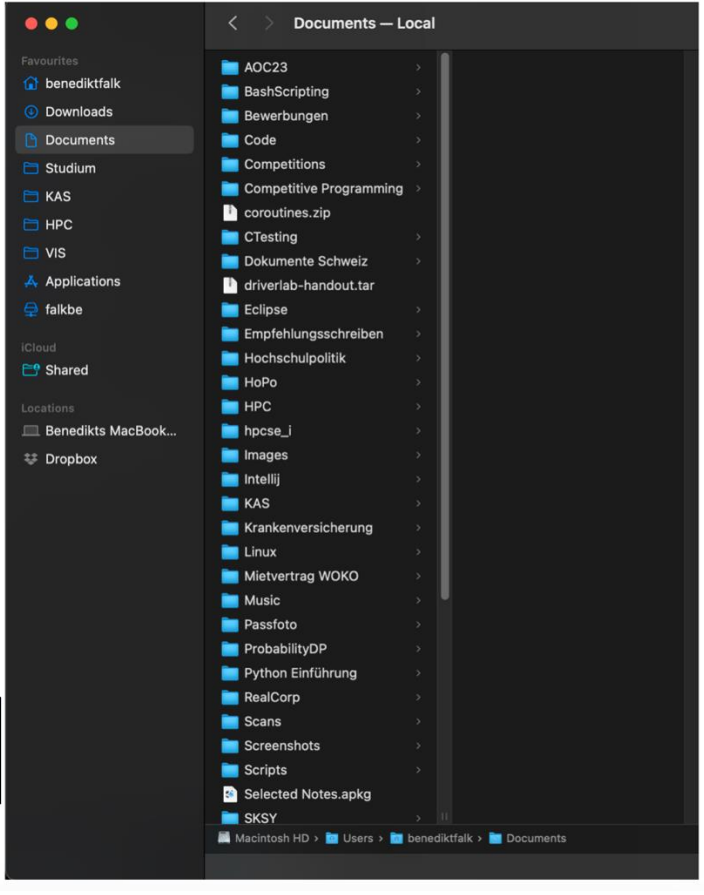
Interact with computer (without GUI)

- We can type in commands, write script

Terminal Introduction

- Everyone used to this overview
- You just see a subset of the actual pc, where are we currently?
- Same folders as we see on RHS?

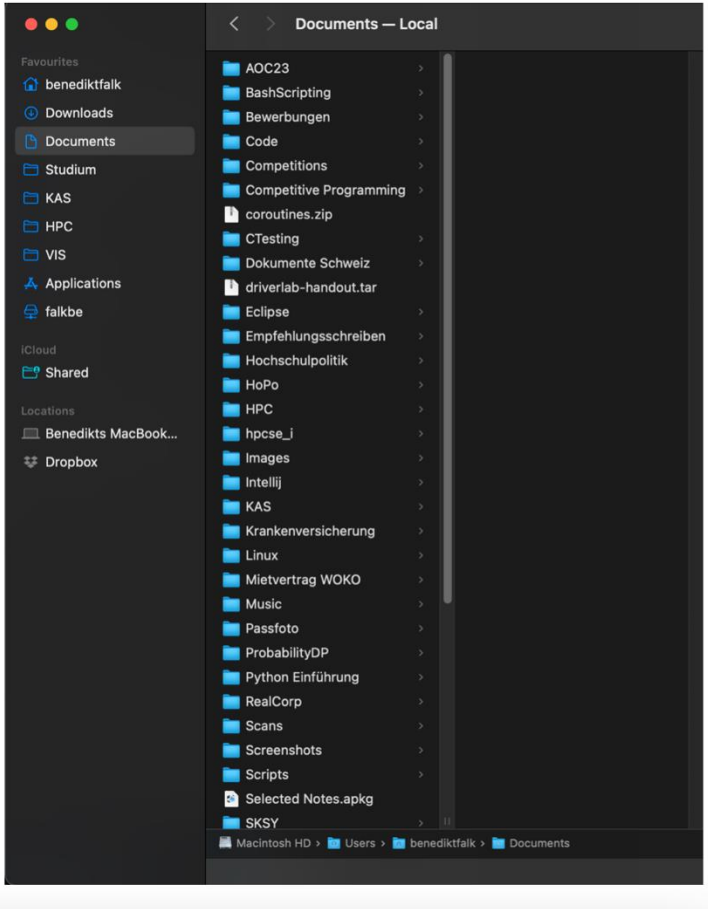
```
student-net-cx-3753:Documents benediktfalk$ pwd
/Users/benediktfalk/Documents
student-net-cx-3753:Documents benediktfalk$
```



Terminal Introduction

Systems@ETH zürich

```
student-net-cx-3753:Documents benediktfalk$ ll
total 5104
drwxr-xr-x     3 benediktfalk  staff       96 Dec  1 15:39 AOC23
drwxr-xr-x    10 benediktfalk  staff      320 Sep 29  2023 BashScripting
drwx------@    6 benediktfalk  staff      192 Oct 11 20:50 Bewerbungen
drwxr-xr-x    11 benediktfalk  staff      352 Jan 17 14:26 CTesting
drwxr-xr-x     4 benediktfalk  staff      128 Apr  5  2023 Code
drwxr-xr-x     4 benediktfalk  staff      128 Mar  5  2023 Competitions
drwxr-xr-x     3 benediktfalk  staff       96 May 20  2023 Competitive Programming
drwxr-xr-x    12 benediktfalk  staff      384 Nov 21  2022 Dokumente Schweiz
drwxr-xr-x     7 benediktfalk  staff      224 Nov 16 12:51 Eclipse
drwxr-xr-x     6 benediktfalk  staff      192 Jul 10  2022 Empfehlungsschreiben
drwxr-xr-x    12 benediktfalk  staff      384 Apr  8 19:11 HPC
drwxr-xr-x     2 benediktfalk  staff       64 Feb 24 18:01 HoPo
drwxr-xr-x     6 benediktfalk  staff      192 Mar  8 09:32 Hochschulpolitik
drwxr-xr-x     6 benediktfalk  staff      192 Apr  7 18:32 Images
drwxr-xr-x     5 benediktfalk  staff      160 Dec  1 14:44 Intellij
drwxr-xr-x     6 benediktfalk  staff      192 Feb 12 23:22 KAS
drwxr-xr-x     5 benediktfalk  staff      160 Nov 21  2022 Krankenversicherung
drwxr-xr-x     3 benediktfalk  staff       96 Jan  9  2023 Linux
drwxr-xr-x     8 benediktfalk  staff      256 Mar  5  2023 Mietvertrag WOKO
drwxr-xr-x@ 116 benediktfalk  staff     3712 Jul 12  2023 Music
drwxr-xr-x    13 benediktfalk  staff      416 May  1  2023 Passfoto
drwxr-xr-x@    7 benediktfalk  staff      224 Jun  8  2023 ProbabilityDP
drwxr-xr-x     3 benediktfalk  staff       96 Oct  4  2022 Python Einführung
drwxr-xr-x     4 benediktfalk  staff      128 Nov 21  2022 RealCorp
drwxr-xr-x     4 benediktfalk  staff      128 Apr  8  2023 SKSY
drwxr-xr-x     3 benediktfalk  staff       96 Sep 29  2023 SPCADocker
drwxr-xr-x     9 benediktfalk  staff      288 Sep 22  2022 Scans
drwxr-xr-x@ 188 benediktfalk  staff     6016 Mar 30 11:18 Screenshots
drwxr-xr-x     5 benediktfalk  staff      160 Mar  5  2023 Scripts
-rw-------@    1 benediktfalk  staff  1935347 Dec 14 09:16 Selected Notes.apkg
drwxr-xr-x@    9 benediktfalk  staff      288 May 28  2023 Sonstiges
drwx------@   16 benediktfalk  staff      512 Nov 21  2022 Stipendium am HLRS
-rw-r--r--@    1 benediktfalk  staff   589702 Mar 15 09:48 StudentID.jpg
drwxr-xr-x    10 benediktfalk  staff      320 Dec 10 11:25 Studium
drwxr-xr-x     6 benediktfalk  staff      192 Sep 27  2023 Test
drwxr-xr-x     4 benediktfalk  staff      128 Feb 25 17:59 Tutorials
drwxr-xr-x     4 benediktfalk  staff      128 Mar  7  2023 Ubuntu
drwxr-xr-x     7 benediktfalk  staff      224 Apr  9 10:22 VIS
drwxr-xr-x    23 benediktfalk  staff      736 Oct 11 21:24 Zeugnisse
drwxr-xr-x@    2 benediktfalk  staff       64 Mar 14 13:46 Zoom
-rw-r--r--@    1 benediktfalk  staff    15306 Jan  8 15:47 coroutines.zip
-rw-r--r--@    1 benediktfalk  staff    30720 Feb  6 12:54 driverlab-handout.tar
drwx------@    7 benediktfalk  staff      224 Sep  4  2023 hpcse_i
-rw-r--r--@    1 benediktfalk  staff     6497 Oct 11 21:55 submissionhpcgtrivial.txt
-rw-r--r--@    1 benediktfalk  staff    25907 Sep 30  2023 submissionrun14.txt
student-net-cx-3753:Documents benediktfalk$
```



Terminal Introduction

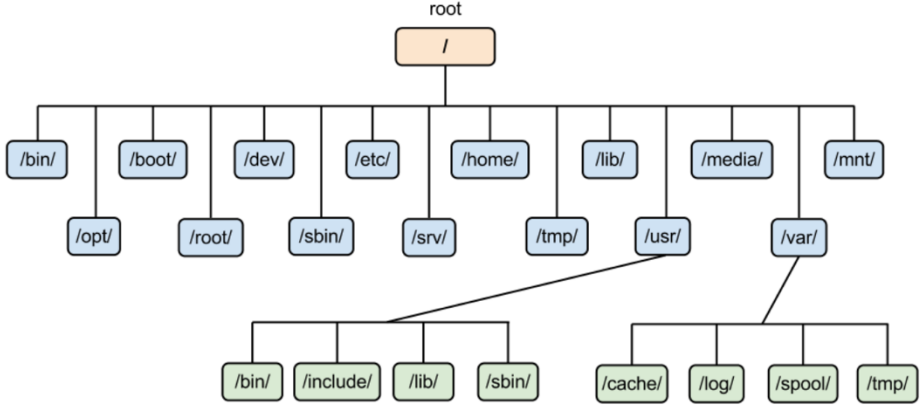
Systems@ETH zürich

- Just as we can create Folders and Texts in GUI, we can do in the terminal
- Now we are going to look from a more general perspective on the terminal as a whole

Linux FHS (File Hierarchy Standard)

Systems@ETH zürich

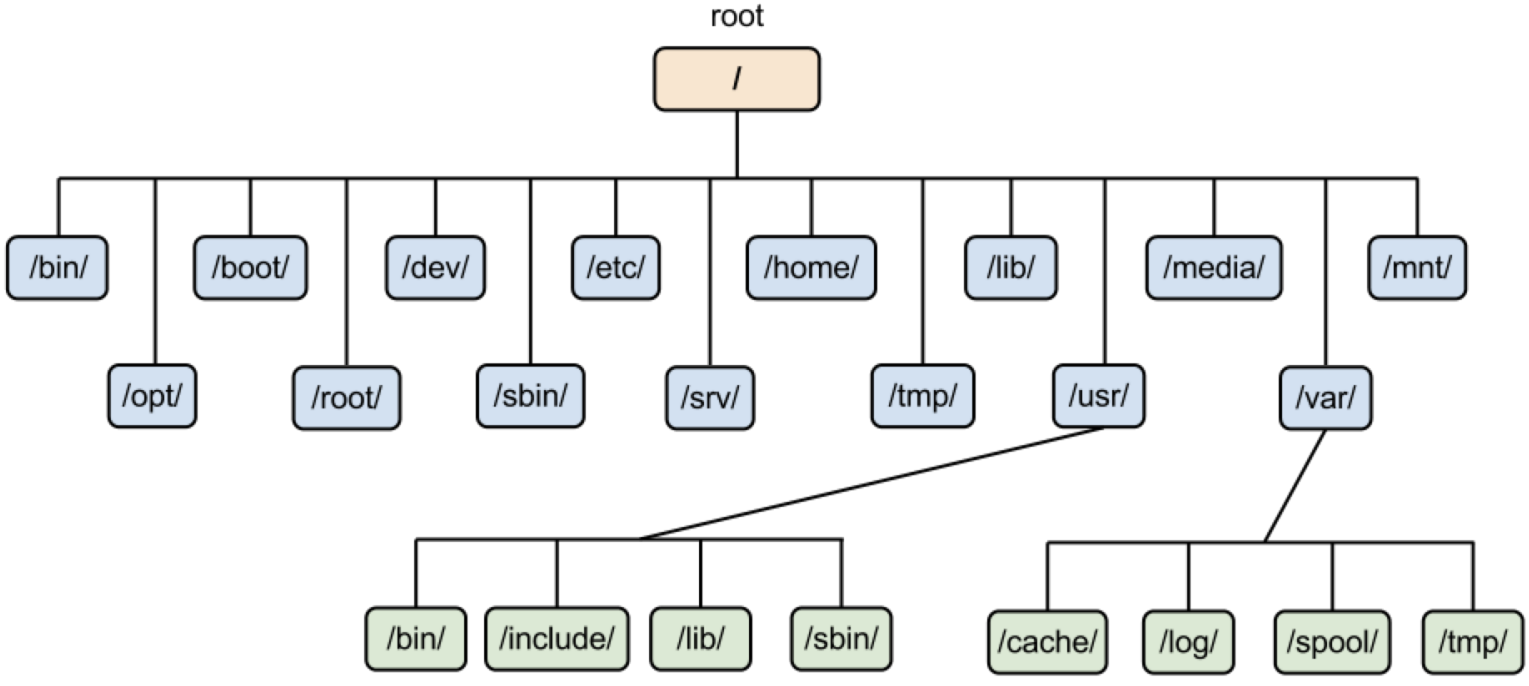
1. / Root directory (top level dir): all other files contained in it
2. /home your personal home directory
3. /dev device files (used to interact with hardware like USB, /dev/sda SATA/SCSI)
4. /etc configuration files for system
5. /bin, /sbin, /usrbin, /usr/bin contain executable binaries (programs(used by user
6. /lib, /lib64 directories contain libraries
7. /var contains variable data files that are expected to change frequently
8. /proc, /sys directories for interface to kernel ata structures



https://nepalisupport.wordpress.com/2016/06/29/linux-file-system-hierarchy/

Linux FHS

Systems@ETH zürich



https://nepalisupport.wordpress.com/2016/06/29/linux-file-system-hierarchy/

Moving in the FHS

Systems@ETH zürich

- pwd print working directory (where am I)
- ls list all elements within our dir
  - ls -al shows permissions and as list
  - ll often alias for “ls -l”
- cd change directory (to move)
  - cd /home/bfalk direct path
  - cd, cd ~ home directory (/home/bfalk)
  - cd /home/ directory change (down)
  - cd .. dir change (up)

```
[bfalk@piora newdirectory]$ pwd
/home/bfalk/newdirectory
[bfalk@piora newdirectory]$ ll
total 0
drwxr-xr-x 2 bfalk g113 6 Apr  8 10:27 dir1
-rw-r--r-- 1 bfalk g113 0 Apr  8 10:27 file1.txt
[bfalk@piora newdirectory]$ cd dir1/
[bfalk@piora dir1]$ pwd
/home/bfalk/newdirectory/dir1
[bfalk@piora dir1]$ cd ..
[bfalk@piora newdirectory]$ pwd
/home/bfalk/newdirectory
[bfalk@piora newdirectory]$
```

Changes in the FHS (Directories, Files)

Directories

- mkdir <name> create a directory
  - mkdir newDirectory creates directory called “newDirectory”
- rmdir <dir name>, rm -rf <dir name> remove a directory

Files

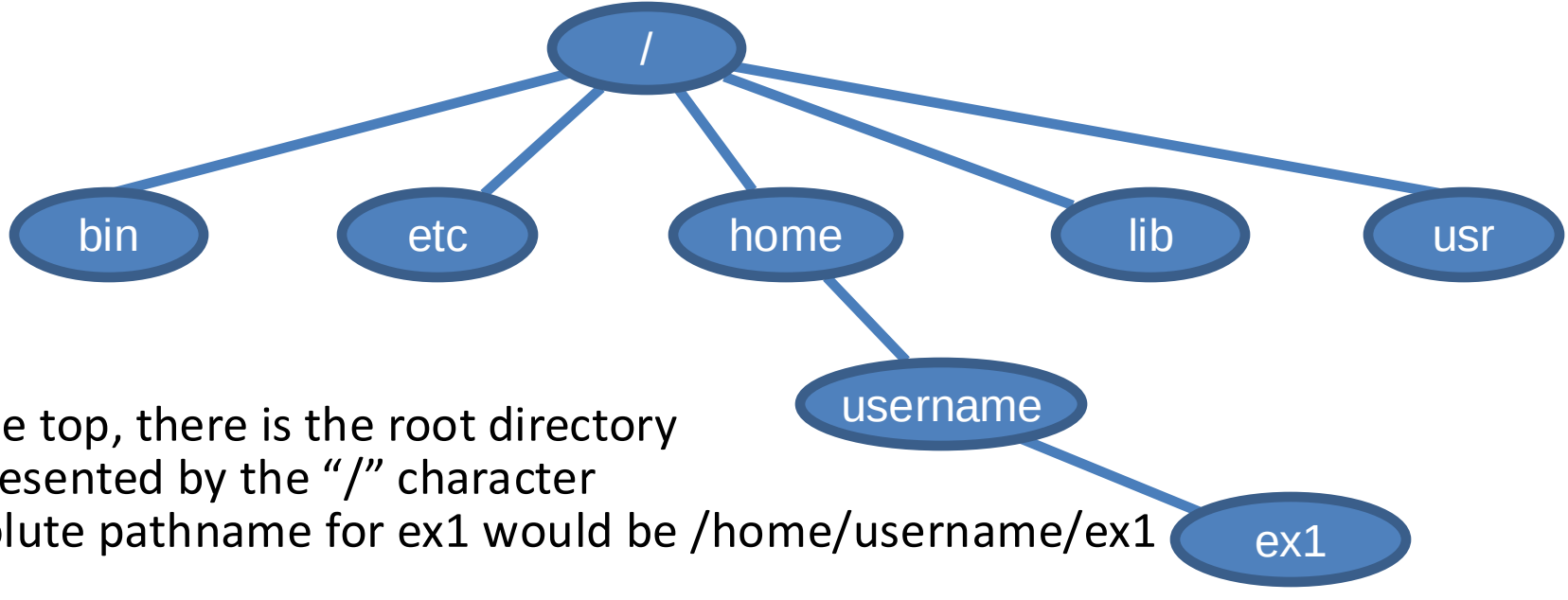
- touch <name> creates file <name>
  - touch test.txt
- cp <path dir1>/<name> <path dir2> copies a file from dir1 to dir 2
- rm <name> removes a file

```
[bfalk@piora newdirectory]$ ll
total 0
drwxr-xr-x 2 bfalk g113 6 Apr  8 10:27 dir1
-rw-r--r-- 1 bfalk g113 0 Apr  8 10:27 file1.txt
[bfalk@piora newdirectory]$ mkdir dir2
[bfalk@piora newdirectory]$ ll
total 0
drwxr-xr-x 2 bfalk g113 6 Apr  8 10:27 dir1
drwxr-xr-x 2 bfalk g113 6 Apr  9 23:10 dir2
-rw-r--r-- 1 bfalk g113 0 Apr  8 10:27 file1.txt
[bfalk@piora newdirectory]$ rmdir dir2
[bfalk@piora newdirectory]$ ll
total 0
drwxr-xr-x 2 bfalk g113 6 Apr  8 10:27 dir1
-rw-r--r-- 1 bfalk g113 0 Apr  8 10:27 file1.txt
[bfalk@piora newdirectory]$ mkdir dir2
[bfalk@piora newdirectory]$ ll
total 0
drwxr-xr-x 2 bfalk g113 6 Apr  8 10:27 dir1
drwxr-xr-x 2 bfalk g113 6 Apr  9 23:10 dir2
-rw-r--r-- 1 bfalk g113 0 Apr  8 10:27 file1.txt
[bfalk@piora newdirectory]$ rm -rf dir2
[bfalk@piora newdirectory]$ ll
total 0
drwxr-xr-x 2 bfalk g113 6 Apr  8 10:27 dir1
-rw-r--r-- 1 bfalk g113 0 Apr  8 10:27 file1.txt
[bfalk@piora newdirectory]$ touch test.txt
[bfalk@piora newdirectory]$ ll
total 0
drwxr-xr-x 2 bfalk g113 6 Apr  8 10:27 dir1
-rw-r--r-- 1 bfalk g113 0 Apr  8 10:27 file1.txt
-rw-r--r-- 1 bfalk g113 0 Apr  9 23:11 test.txt
[bfalk@piora newdirectory]$
```

File System

Systems@ETH zürich

- UNIX organizes user data, programs, etc. into structures called files.
- Files are placed in directories.
- Directories are organized into a hierarchical structure.



- At the top, there is the root directory
- Represented by the “/” character
- Absolute pathname for ex1 would be /home/username/ex1

Browsing the Filesystem

Systems@ETH zürich

- **whoami**: prints the login name of the current user

- **pwd**: prints the working directory

- **ls**: lists files and directories
  - Has more options such as –F, -a, -l, -all.

- **cd**: changes the current working directory to the given pathname
- e.g.: cd /home/username/ex1

- **“.”** is the current directory and **“..”** stands for the parent directory, both can be used with cd
- **“~”** stands for your home directory

Browsing the Filesystem

Systems@ETH zürich

- **mkdir**: creates a directory
  - mkdir /home/username/ex1/newfolder
- **rmdir**: removes a directory
  - will only remove empty directories
- **cp**: copies files/folders from one location to another
  - cp /etc/hosts /home/username
- **mv**: move/rename existing files/folders
  - mv /home/username/hosts /home/username/ex1/newfolder
- **rm**: removes files/folders
  - rm /home/username/ex1/newfolder/hosts

Processes

Systems@ETH zürich

- **ps**: see the processes associated with the current shell
  - ps -ef to get a full listing of all processes in the system

- **top**: display the processes using the most CPU time
  - Quit with q

- **kill**: terminates a process
  - Used as ‘kill <ProcessID>’.
  - -9 option to force kill

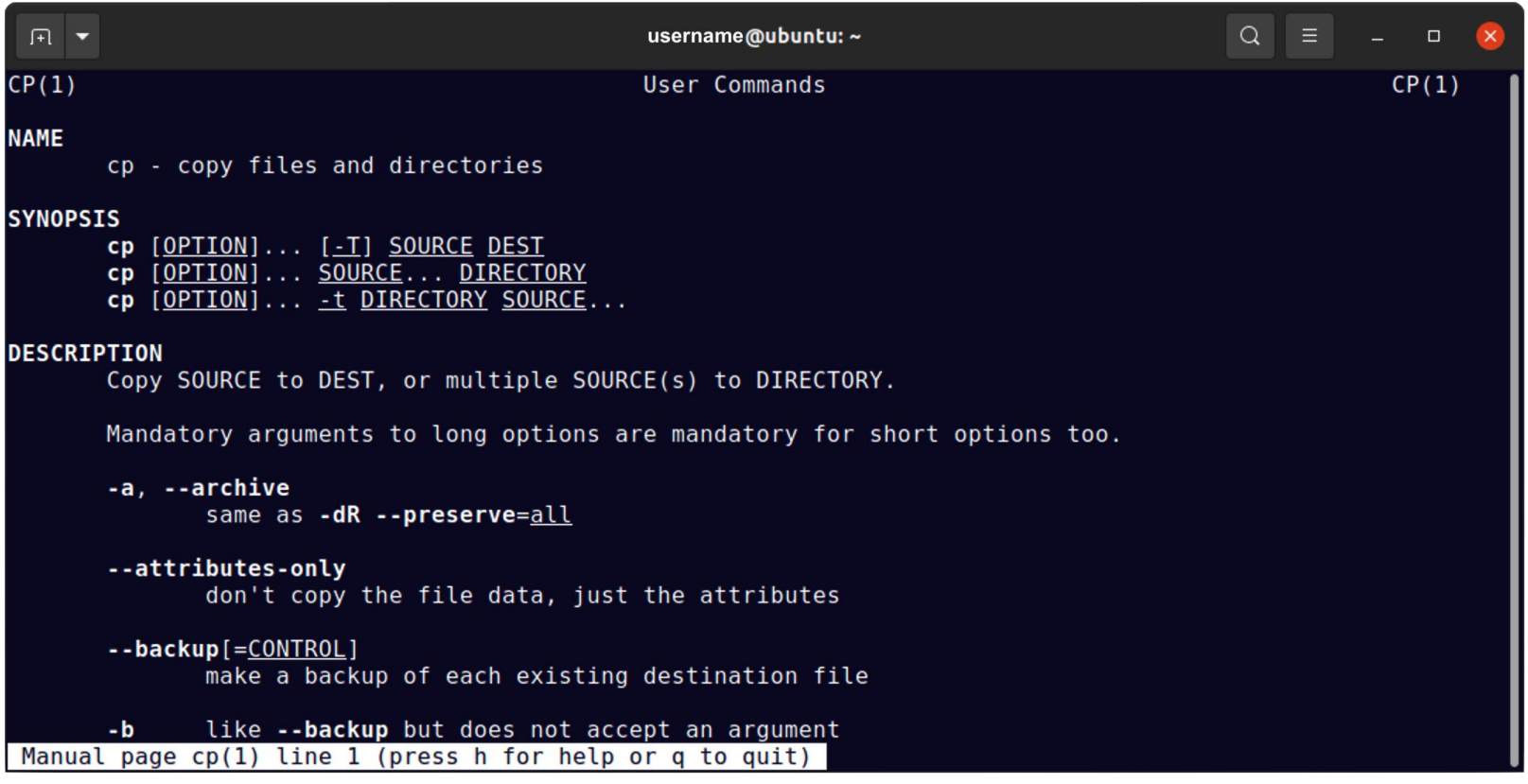
Miscellaneous

Systems@ETH zürich

- **nano, gedit, emacs, vi/vim**: useful text editors for writing your programs and editing files.
- **cat, more, less**: useful to view files
- **grep**: useful for searching text files
- **gcc/gdb:** compilers and debuggers

Lost? Try “man”.

man cp

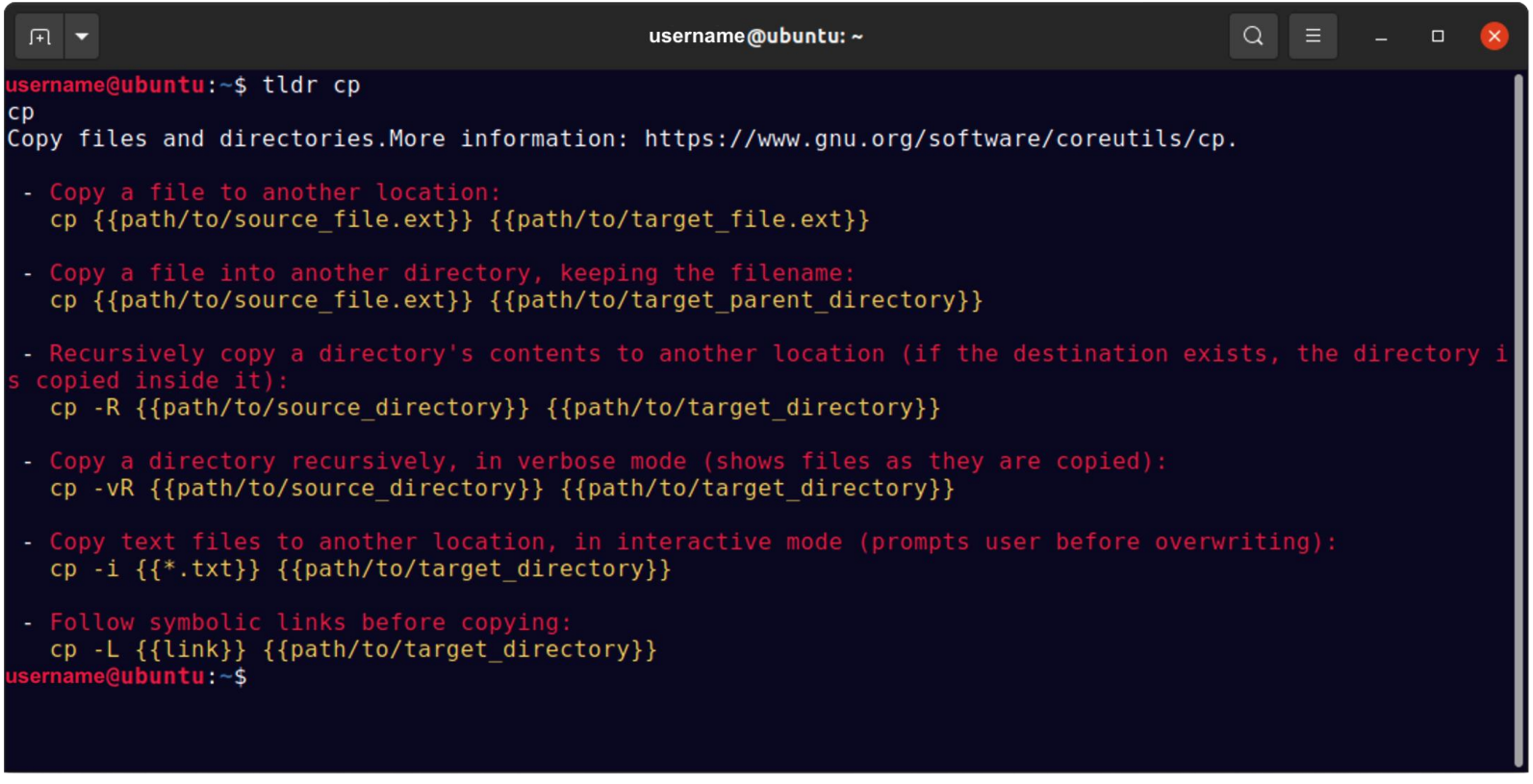


Still lost? Try “tldr”.

Systems@ETH zürich

Can be installed with sudo apt install tldr

tldr cp



More tutorials online

Systems@ETH zürich

- http://people.ischool.berkeley.edu/~kevin/unix-tutorial/toc.html
- http://www.ee.surrey.ac.uk/Teaching/Unix/
- http://www.unixtutorial.org/commands/
- … just Google/ChatGPT for more!
- A lot to take in, but it will become second nature over time :)

Preview of Assignment 1

The Data Lab

Absolute Basics: Bits, Bytes and Hex

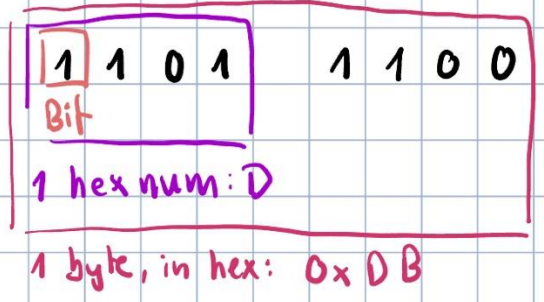
Systems@ETH zürich

Decimal: $(203)_{10} \hat{=} 2 \cdot 10^2 + 0 \cdot 10^1 + 3 \cdot 10^0$

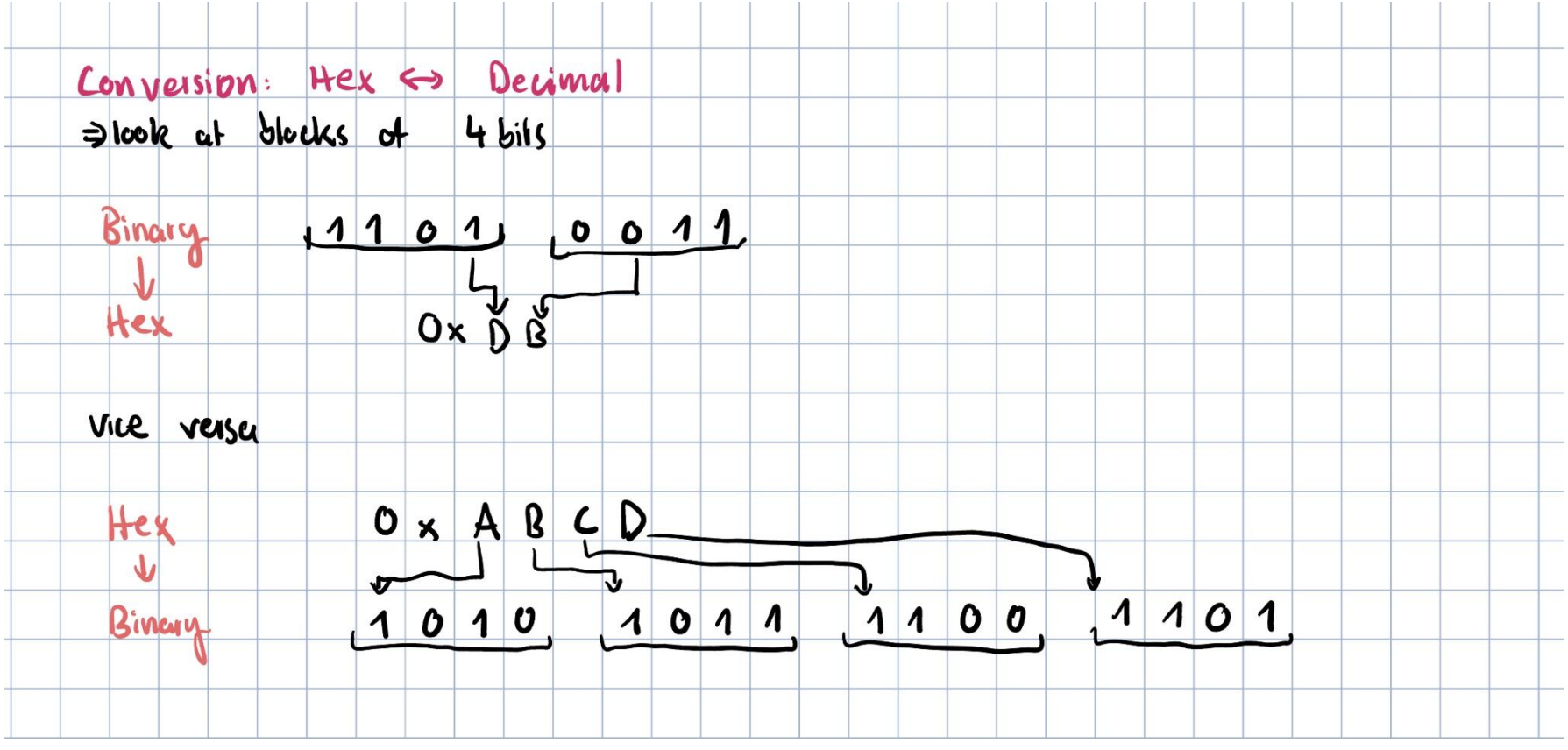
Binary: $(1101)_2 \hat{=} 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 \overset{\text{Dec.}}{\Rightarrow} 8 + 4 + 1 = 13$

Hex: $(0x\,a)_{16} = (0x\,1010)$

1 Byte = 8 bit num = 2 hex num



Absolute Basics: Conversion



Preview lecture

Encoding integers

Unsigned

$$B2U(X) = \sum_{i=0}^{w-1} x_i \cdot 2^i$$

Two's complement

$$B2T(X) = -x_{w-1} \cdot 2^{w-1} + \sum_{i=0}^{w-2} x_i \cdot 2^i$$

Sign Bit

```
short int x =  15213;
short int y = -15213;
```

- A C `short` is 2 bytes long:

	Decimal	Hex	Binary
x	15213	3B 6D	00111011 01101101
y	-15213	C4 93	11000100 10010011

- Sign bit
  - For 2's complement, most significant bit = 1 indicates negative

Preview lecture

Integers and floats

- Types and sizes:

C data type	Typical 32-bit	ia32	Intel x86-64
char	1	1	1
short	2	2	2
int	4	4	4
long	4	4	8
long long	8	8	8
float	4	4	4
double	8	8	8
long double	8	10/12	10/16

Sizes are implementation defined!

- Integers are signed by default
  - use `signed` or `unsigned` to clarify

Preview lecture

C99 extended integer types

```
#include <stdint.h>

int8_t          a;
int16_t         b;
int32_t         c;
int64_t         d;

uint8_t         x;
uint16_t        y;
uint32_t        z;
uint64_t        w;
```

Signed integers, precise size in bits

Unsigned integers,

Preview lecture

Encoding integers

Unsigned

$$B2U(X) = \sum_{i=0}^{w-1} x_i \cdot 2^i$$

Two's complement

$$B2T(X) = -x_{w-1} \cdot 2^{w-1} + \sum_{i=0}^{w-2} x_i \cdot 2^i$$

Sign Bit

```
short int x =  15213;
short int y = -15213;
```

- A C `short` is 2 bytes long:

	Decimal	Hex	Binary
x	15213	3B 6D	00111011 01101101
y	-15213	C4 93	11000100 10010011

- Sign bit
  - For 2's complement, most significant bit = 1 indicates negative

Preview lecture

Encoding integers

Unsigned

$$B2U(X) = \sum_{i=0}^{w-1} x_i \cdot 2^i$$

Two's complement

$$B2T(X) = -x_{w-1} \cdot 2^{w-1} + \sum_{i=0}^{w-2} x_i \cdot 2^i$$

Sign Bit

```
short int x =  15213;
short int y = -15213;
```

Exercise: for a char (1byte = 8bit)

- Umin:
- Umax:
- Tmin:
- Tmax:
- -1:
- 0:

Preview lecture

Encoding integers

Unsigned

$$B2U(X) = \sum_{i=0}^{w-1} x_i \cdot 2^i$$

Two's complement

$$B2T(X) = -x_{w-1} \cdot 2^{w-1} + \sum_{i=0}^{w-2} x_i \cdot 2^i$$

Sign Bit

```
short int x =  15213;
short int y = -15213;
```

Exercise: for a char (1byte = 8bit)

- Umin: 0000 0000
- Umax: 1111 1111
- Tmin: 1000 0000
- Tmax: 0111 1111
- -1: 1111 1111
- 0: 0000 0000

Preview lecture

Numeric ranges

- Unsigned values
  - UMin = 0
    - 000…0
  - UMax = $2^w - 1$
    - 111…1
- Two's complement values
  - TMin = -2^{w-1}
    - 100…0
  - TMax = $2^{w-1} - 1$
    - 011…1

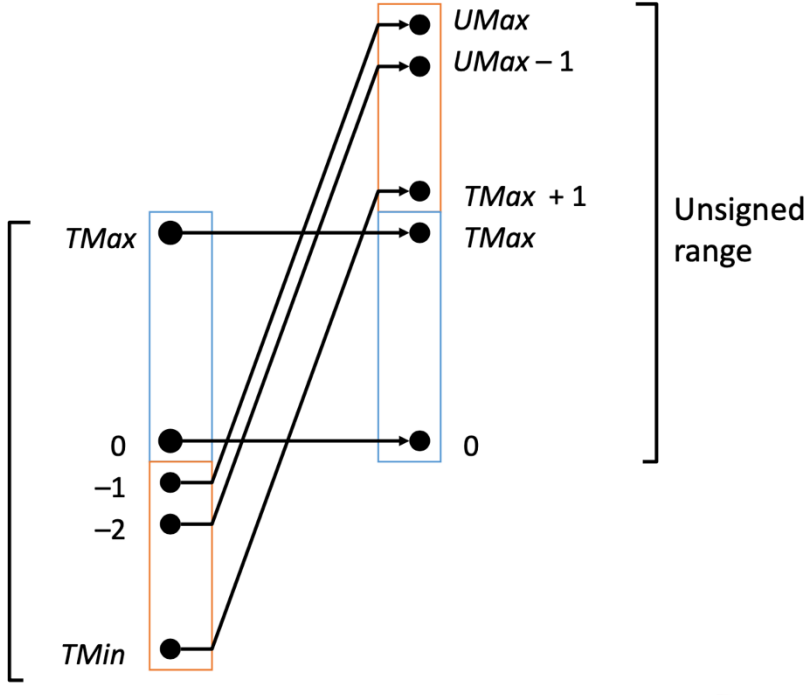
Values for w=16

	Decimal	Hex	Binary
UMax	65535	FF FF	11111111 11111111
TMax	32767	7F FF	01111111 11111111
TMin	-32768	80 00	10000000 00000000
-1	-1	FF FF	11111111 11111111
0	0	00 00	00000000 00000000

Preview lecture

Conversion visualized

- 2's Complement → Unsigned
  - Ordering inversion
  - Negative → big positive



Preview lecture

- Left shift: x << y
  - Shift bit-vector x left y positions
    - Throw away extra bits on left
    - Fill with 0's on right
- Right shift: x >> y
  - Shift bit-vector x right y positions
    - Throw away extra bits on right
  - Logical shift
    - Fill with 0's on left
  - Arithmetic shift
    - Replicate most significant bit on right
- Undefined behavior
  - Shift amount < 0 or ≥ word size

Argument x	01100010
<< 3	00010*000*
Log. >> 2	*00*011000
Arith. >> 2	*00*011000

Argument x	10100010
<< 3	00010*000*
Log. >> 2	*00*101000
Arith. >> 2	*11*101000

Java writes this ">>>".

Preview lecture

Negation: complement & increment

- Recall the following holds for 2's complement:

```
~x + 1 == -x
```

- Complement

  Observation: `~x + x == 1111…111 == -1`

```
      x   1 0 0 1 1 1 0 1
+    ~x   0 1 1 0 0 0 1 0
-------------------------
     -1   1 1 1 1 1 1 1 1
```

- Complete proof?

Preview lecture

Complement & increment examples

x = 15213

	Decimal	Hex	Binary
x	15213	3B 6D	00111011 01101101
~x	-15214	C4 92	11000100 10010010
~x+1	-15213	C4 93	11000100 10010011
y	-15213	C4 93	11000100 10010011

x = 0

	Decimal	Hex	Binary
0	0	00 00	00000000 00000000
~0	-1	FF FF	11111111 11111111
~0+1	0	00 00	00000000 00000000

Preview lecture

Systems@ETH Zürich

Summary

- Signed/unsigned multiply:

$$x * 2^k = x << k$$

- Unsigned divide:

$$u / 2^k = u >> k$$

Logical shift

- Signed divide:

$$s / 2^k = s >> k \quad \text{for } s > 0$$

$$s / 2^k = s + (2^k - 1) >> k \quad \text{for } s < 0$$

Arithmetic shift

Pre-requisites

Systems@ETH zürich

- You will need a working Linux environment
  - If you just installed Ubuntu on a VM, you still need to install some tools (gcc, etc.)

```
$ sudo apt update
$ sudo apt install build-essential
$ sudo apt install flex bison
```

- Download the assignment sheet and follow the instructions carefully.
- All you need to change is in **bits.c**

Introduction Bit-Operators in C

- Memory is organized as an array of bits
- Smallest addressable memory unit: byte
- The type of a variable determines its value
- e.g.: integers are represented with two's complement

x

1	0	1	1	1	0	0	1

signed char x = -71

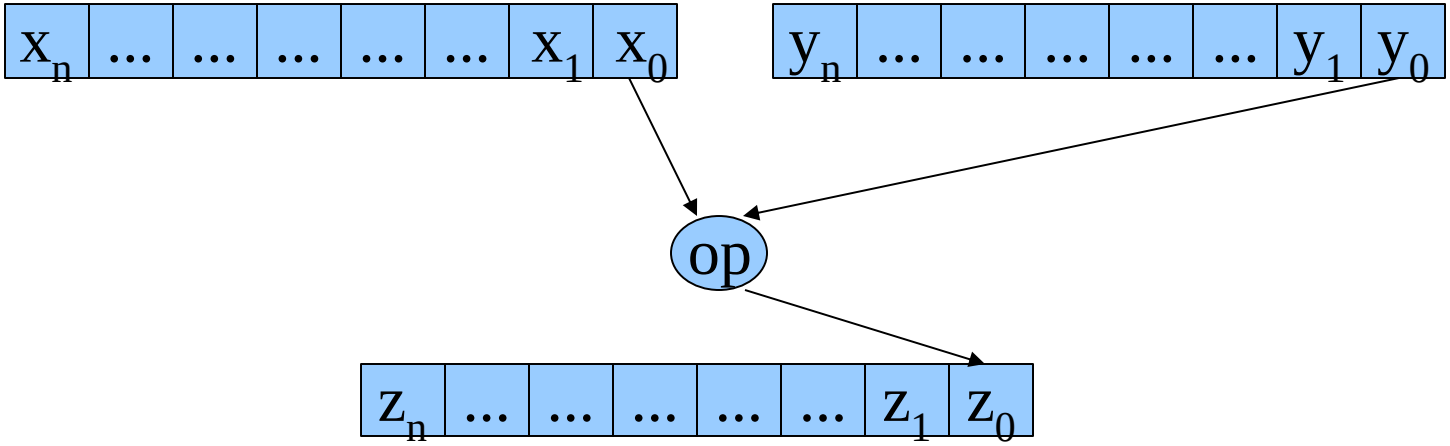
x

1	0	1	1	1	0	0	1

unsigned char x = 185

Introduction Bit-Operators in C

- Bitwise operations are performed on every bit of the two operands individually
- Can be applied to any “integral” datatype
- $Z = X \text{ op } Y \quad \rightarrow \quad Z_i = X_i \text{ op } Y_i$



Logical vs Bitwise Operators

Systems@ETH zürich

- **Logical operators** evaluate the truth or falsity of **an expression**
  - The result is either **true or false**
  - In C: 0 is false, **anything else** is true
  - Logical AND: && Logical OR: || Logical NOT: !

- **Bit operators** perform the operation on **each bit**
- The result can be an **arbitrary** value
  - Bit-wise AND: & Bit-wise OR: | Bit-wise NOT: ~

Preview lecture

- Operations &, |, ~, ^ available in C
  - Apply to any "integral" data type
    - `long`, `int`, `short`, `char`, `unsigned`
  - View arguments as bit vectors
  - Arguments applied bit-wise

&	Bitwise AND
|	Bitwise OR
~	Bitwise NOT
^	Bitwise XOR

- Examples (using `char` data type):
  - `~0x41` → `0xBE`
    `~01000001`$_2$ → `10111110`$_2$
  - `~0x00` → `0xFF`
    `~00000000`$_2$ → `11111111`$_2$
  - `0x69` & `0x55` → `0x41`
    `01101001`$_2$ & `01010101`$_2$ → `01000001`$_2$
  - `0x69` | `0x55` → `0x7D`
    `01101001`$_2$ | `01010101`$_2$ → `01111101`$_2$

Preview lecture

- &&, ||, !
  - View 0 as “False”
  - Anything nonzero as “True”
  - Always return 0 or 1
  - Early termination

- Examples (char data type)
  - `!0x41  →  0x00`
  - `!0x00  →  0x01`
  - `!!0x41 →  0x01`

  - `0x69 && 0x55 →  0x01`
  - `0x69 || 0x55 →  0x01`

&&	Logical AND
||	Logical OR
!	Logical NOT

Bit Masks

- Used to set/delete/test single bits
  - **Delete** and test bits with AND
  - **Set** bits with OR
  - **Flip** bits with XOR
- Example: x is either '0' or '1'

```
  xxxxxxxx      xxxxxxxx      01101001
& 01010101    | 01010101    ^ 01010101
----------    ----------    ----------
  0x0x0x0x      x1x1x1x1      00111100
```

Bit Masks

- Test if i-th bit is 1
  - result = (input & (1<<i))
- Flip i-th bit
  - result = (input ^ (1<<i))
- Set i-th bit
  - result = (input | (1<<i))

Shift Operators

- Right Shift *"Division by a power of two"*
  - Logical (Java: >>>): fill left-end with 0's, used with unsigned types
  - Arithmetic (Java: >>): fill left-end with MSB, used with signed types

    WARNING: not all compilers do arithmetic shift with signed types, thus shift with signed types considered to be **UNDEFINED.**
- Left Shift *"Multiplication by a power of two"*

```
x = 0b0011;      // x = 3
z = x << 2;      // z = 0b001100 = 12 = 3 * 2^2
```

Your Turn! Do the homework

Systems@ETH zürich

- Complete function skeletons in **bits.c**
- **Restrictions**
  - **No loops, conditions or jumps**
  - **Use the following operators only: ! ~ & ^ | + << >>**
  - **Constants must not be longer than 8 bits**

- **Contest**: “Beat the professor”

- **Goal**: Use as few operations as possible

Example

- Return the min. value *Tmin* of a signed integer

Example

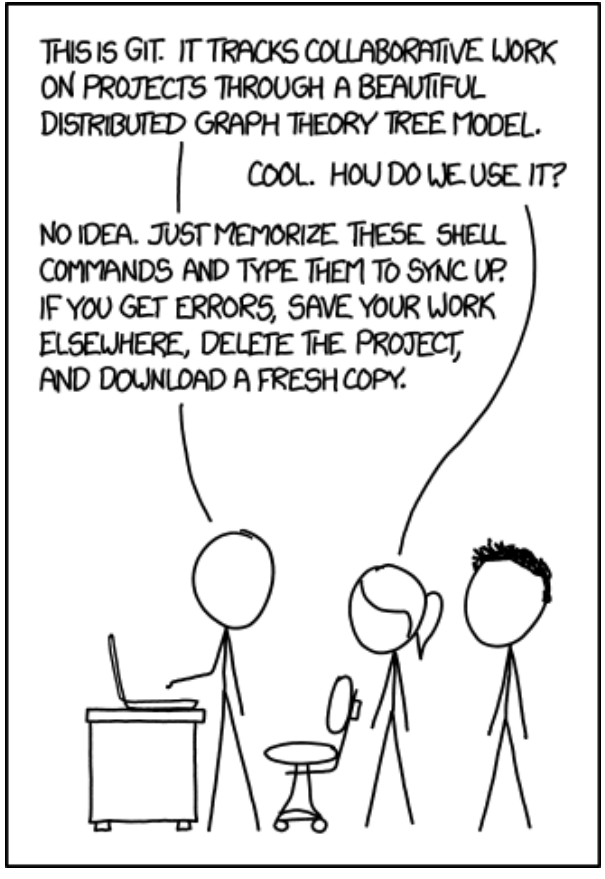
- Return the min. value *Tmin* of a signed integer
- Tmin is 0x80000000
- Idea: shift 1 31 positions to the left

```
int Tmin() {
    return (1 << 31);
}
```

- Note: return (0x80000000); is not legal, since constants must not be longer than 1 byte!

Version Control using git

How to submit your solution



Preparation

- You will need to install git and ssh:

  $ sudo apt install git openssh-client

- You will need to generate and put your SSH key to gitlab and clone your repo.
  (Instructions also in assignment1).

Tell git about you

```
$ git config --global user.name "Jane Doe"
$ git config --global user.email "jdoe@student.ethz.ch"
```

Generate an SSH key pair

Systems@ETH zürich

- If you haven't used ssh before, generate a new key
  $ ssh-keygen

- Confirm defaults with enter three times (or use a passphrase). Then display your public key
  $ cat .ssh/id_rsa.pub
  ssh-rsa AAAAB3NzaC1yc2EAAAADAQ…

- Copy the key (in the terminal, copy/paste with ctrl-c/ctrl-v doesn't work. Select the text and use right-click, copy)

Upload SSH key to gitlab

- Open https://gitlab.inf.ethz.ch/-/profile/keys
- Login with your nethz credentials
- Paste your key and save

Checkout your repository

(replace the placeholder NETHZ below with your NETHZ)

Systems@ETH zürich

- Clone your repository

  $ git clone git@gitlab.inf.ethz.ch:course-spca2024/spca2024-NETHZ-hand-in.git

- This will create a folder “spca2024-NETHZ-hand-in”

  $ cd spca2024-NETHZ-hand-in

Submitting your solution

Systems@ETH zürich

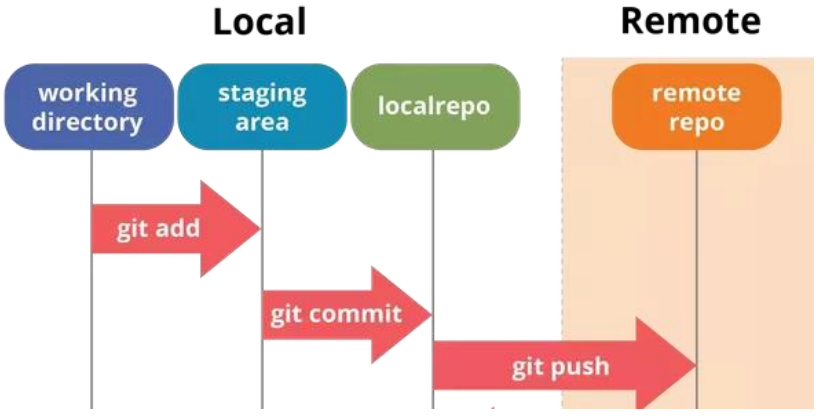
- You need to copy the file bits.c into your git repository

- Make a new directory and copy your solution into it
  $ mkdir assignment1
  $ cp bits.c assignment1

- Add, commit and push
  $ git add bits.c
  $ git commit –m “assignment1”
  $ git push

Add, commit, push?

- add
  Add to staging area
- commit
  turn staging area into a commit
- push
  push commit(s) to the server



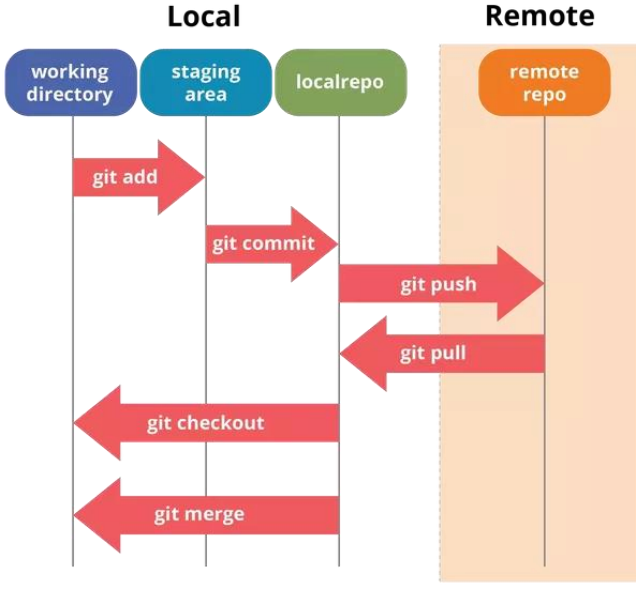
- Commits = savegame
- add and commit do not do any network access

What if push fails?

Should not happen in this assignment

Systems@ETH zürich

- Probably the server has a more recent version than you (somebody else pushed a newer commit)
- To get new commits from the server
  $ git pull
- ***If*** there are no conflicts, you're done!
  $ git push



Submitting your solution

- You can repeat these steps to update your solution

- Check your score (only from ETH network)
  -> http://spca.ethz.ch