Introduction to GNU/Linux

- Brief history of GNU/Linux
- Brief introduction about computer architecture
- Brief introduction to GNU/Linux
- Introdution to bash

contacts: d.mapelli7@campus.unimib.it (mailto:d.mapelli7@campus.unimib.it)

UNIX

AT&T, MIT, General Electrics: Multics (mid 1960s)

AT&T (Bell Labs): UNIX (Ken Thompson, Denis Ritchie. 1970), C

UNIX:

- https://www.youtube.com/watch?v=tc4ROCJYbm0 (https://www.youtube.com/watch?v=tc4ROCJYbm0) : AT&T Archives: The UNIX Operating System

  Kernel - Shell - Utilities

  Pipelines of utilities

GNU / Linux

GNU Manifesto (R. Stallman, 1985): comlete free unix-like system, called GNU FSF (fsf.org) (R. Stallman, 1985)

- license: GNU General Public License
- boring: GNU assembler, GNU C Compiler, GNU Linker, GNU tar
- "not programming tools": bash, GNU C Library
- applications: GNU Cash, GNOME, GNU Chess
- kernel: GNU Hurd

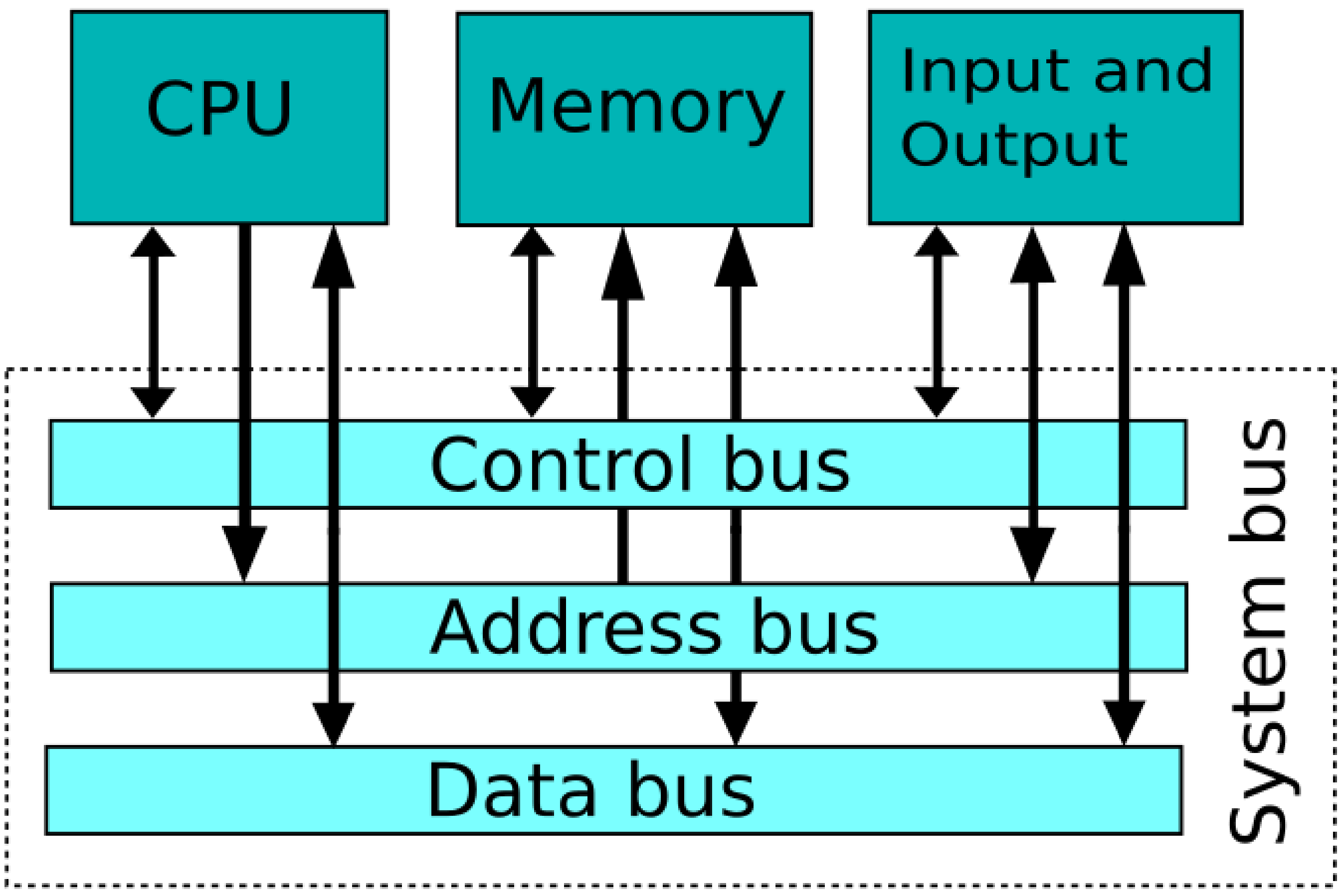
Linux kernel (Linus Torvalds, 1991)

- GNU system
- kernel: Linux

https://www.gnu.org/gnu/linux-and-gnu.en.html (https://www.gnu.org/gnu/linux-and-gnu.en.html)

https://stallman.org (https://stallman.org)

Computer Architecture: Von Neumann



Di W Nowicki - CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=15258936 (https://commons.wikimedia.org/w/index.php?curid=15258936)

Computer Architecture: adding a bit of realism

Every device has a precise purpose

- smartphone
- laptop
- desktop
- server (cloud, HPC, storage)
- embedded

mobile devices, cloud and IoT are transforming the idea of *computer*, no more only PC!

Multiple souls in GNU/Linux world:

- enterprise - cloud
- private users - privacy/tinkering
- ...

CPU

- instruction pipelining
- speculative execution (ehm ehm)

- multi-layer caching
- Complex instruction set
- multicore
- ...

What about current CPU architecture?

https://moodle.rrze.uni-erlangen.de/pluginfile.php/11671/mod_resource/content/5/01_IntroArchitecture.pdf (https://moodle.rrze.uni-erlangen.de/pluginfile.php/11671/mod_resource/content/5/01_IntroArchitecture.pdf)

Quick comparison between a desktop and laptop intel core CPU: spot the differences

https://ark.intel.com/content/www/us/en/ark/compare.html?productIds=97472,97129 (https://ark.intel.com/content/www/us/en/ark/compare.html?productIds=97472,97129)

CPU: CISC vs RISC

Also known as *why it is 10 years that we are waiting for truly convergent devices*, or also *why no one is doing data analysis on their smartphone*

- type of instruction set, which corresponds to chip complexity

  x86 vs ARM: difference in CPU-intense applications, for example compiling and data analysis
- power consumption

  differences on clock speed, cache dimension, ...

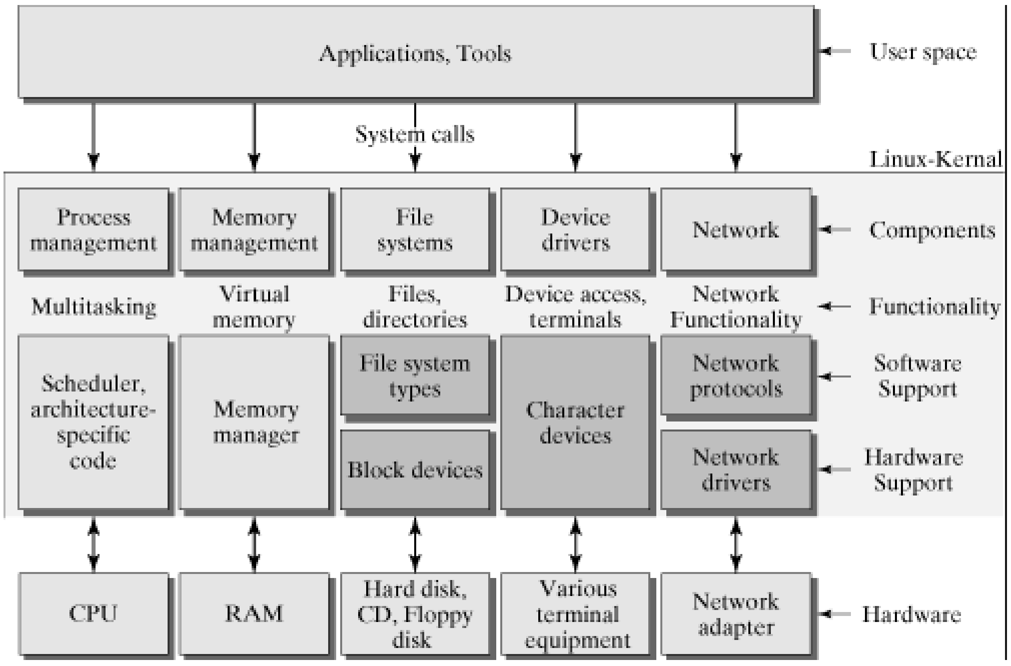
https://en.wikipedia.org/wiki/Comparison_of_instruction_set_architectures (https://en.wikipedia.org/wiki/Comparison_of_instruction_set_architectures)

Linux

Definition (Extract from https://en.wikipedia.org/wiki/Linux (https://en.wikipedia.org/wiki/Linux))

> Linux is a family of free and open-source software operating systems based on the Linux **kernel**, an operating system kernel first released on September 17, 1991 by Linus Torvalds. Linux is typically packaged in a Linux **distribution** (or distro for short).

Kernel:



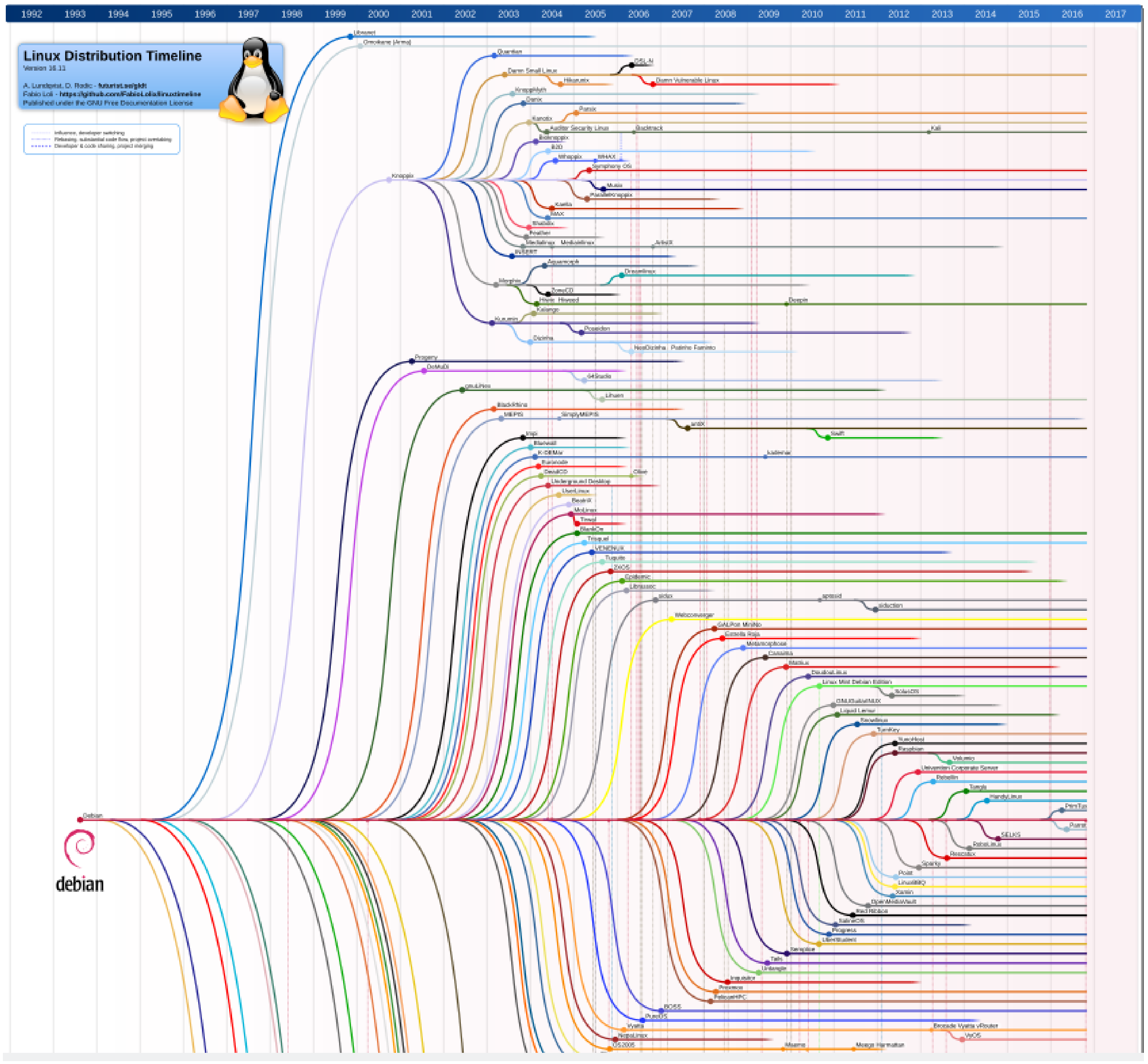
Linux distributions: aka how to obtain Linux

Main components of a Linux Distribution

- Linux kernel
- GNU tools
- Package manager (almost always)
- Desktop Environment (often)

How to gather informations on current Linux distributions

https://distrowatch.com/ (https://distrowatch.com/)



[https://github.com/FabioLolix/LinuxTimeline (https://github.com/FabioLolix/LinuxTimeline)](https://github.com/FabioLolix/LinuxTimeline)

Which distribution should you chose?

Main differences

- Release policy: Standard vs Rolling
- Installation: Installer (easy) vs manual (a bit of effort) vs re-compile by hand (sadistic)

```
|           | standard                      | rolling                      |
|-----------+-------------------------------+------------------------------|
| manual    |                               | Arch                         |
|-----------+-------------------------------+------------------------------|
| installer |                               | Manjaro                      |
|           | Debian, Ubuntu, KDE Neon, ... |                              |
|           | OpenSUSE Leap                 | OpenSUSE Tumbleweed          |
|           | RHEL, CentOS, Fedora, ...     |                              |
```

More informations: Package managers

Main package types

- Debian: apt, .deb
- RHEL: yum, .rpm
- OpenSUSE: zypper, rmp
- Arch: pacman, .tar

Distribution package managers are very similar to one another.

Main differences are in the management of *repositories* .

- updates release policy: standard (veloce o lento), rolling (puro o parziale), ...
- How to add external repositories: apt sources, ubuntu PPA, arch community, arch AUR, ...

Disclaimer

A Package Manager - Repository is not a concept exclusive to Linux distributions. Here are some other examples:

- Android Google Play app and repository
- Apple App Store
- CTAN: Comprehensive TeX Archive Network
- CRAN: Comprehensive R Archive Network
- pip: package manager for python packages
- npm: Node package manager
- ...

More informations: Desktop Environment

	Debian	Ubuntu	Arch	Manjaro
GNOME	x	x	x	x
plasma, KDE	x	x	x	x
MATE	x	x	x	x
xfce	x	x	x	x
lxde	x	x	x	
...				

Internet full of examples of what they look like: https://www.youtube.com/user/linuxscoop/ (https://www.youtube.com/user/linuxscoop/)

Other informations

- Debian: Desktop, Server. default "free/libre", but also "non-free" repositories
- Ubuntu: Desktop, Server, Cloud. Ci sono versioni LTS
- KDE Neon: based on Ubuntu LTS, with desktop environment plasma by KDE in semi-rolling
- Fedora: Workstation, Server, Atomic

You installed Linus. Now what?

Two main ways to interact with a compute: TUI (Text-based User Interface), GUI (Graphical User Interface)

Let's open a terminal: `ctrl + alt + t`

Consider installing `yakuake` or `guake` .

Example of commands to manage files

```
pwd
ls

mkdir temp_dir
cd temp_dir
touch temp_file
file temp_file
touch temp_file.txt
file temp_file.txt
cd ..
rm -r temp_dir
```

Hidden files

```
touch .temp_file
ls
ls -a
rm .temp_file
```

- Filesystem: pay attention, this may have different meanings.
- path: relative and absolute

Edit a file

```
mkdir temp
cd temp

nano esempio.txt
vim esempio.txt
emacs -nw esempio.txt
base64 /dev/urandom | head -c 1000000 > esempio.txt

file esempio.txt
cat esempio.txt
less esempio.txt
head esempio.txt
tail esempio.txt
```

Concurrently

```
base64 /dev/urandom >> esempio.txt
```

```
tail -f esempio.txt
```

Retrieve informations: `man`

```
man pwd
man ls
man file
man base64

man man
```

Everything is a file

```
file ~/temp
file esempio.txt

file /dev/null
file /dev/urandom
file /dev/sda
```

https://unix.stackexchange.com/questions/60034/what-are-character-special-and-block-special-files-in-a-unix-system (https://unix.stackexchange.com/questions/60034/what-are-character-special-and-block-special-files-in-a-unix-system)

Users and privileges

```
ls -l
stat -c %a filename
```

https://en.wikipedia.org/wiki/File_system_permissions#Notation_of_traditional_Unix_permissions (https://en.wikipedia.org/wiki/File_system_permissions#Notation_of_traditional_Unix_permissions)

Intermezzo: What have we been using so far?

All Unix-like systems are based on the kernel-shell-utils pattern.

We have just written some basic commands, the computer did some actions and sometimes answered with some message.

What is happening here behind the curtains?

TTY

`tty` comes from TeleTYpewriter (aka teleprinter), from telegraph communication.

Similar machines have been adapted in the early ages of computing to interface users with mainframes.

Technology changed, but the name stuck.

Try hitting `ctrl alt F1` , or any other function number. what do you see?

Login and run this command

```
tty
```

Switch back to a GUI.

You do not need to switch to a TTY every time you need to interact to the computer: there are **Terminal Emulators**!

This programs emulate the behaviour of a tty. Common example are xterm, konsole, ...

What happens when you open a tty or terminal emulator?

It launches a **shell**. This is a program that allows you to interact with the operative system, exactly as descrbed for all good Unix-like OSs.

Many different shells emerged during the years. The de-facto standard is **bash**, which is a Bourne-like shell.

Other alternatives are **zsh**, which is not fully bourne-like, and this allowed it to be more flexible.

Another popular alternative (which ROOT\CERN has setup scripts for since v6.16) is **fish**. It is nothing like a bourne shell, but it offers some very nice features.

How to configure a shell

Every shell has different configuration files.

bash and zsh

https://medium.com/@rajsek/zsh-bash-startup-files-loading-order-bashrc-zshrc-etc-e30045652f2e (https://medium.com/@rajsek/zsh-bash-startup-files-loading-order-bashrc-zshrc-etc-e30045652f2e)

https://shreevatsa.wordpress.com/2008/03/30/zshbash-startup-files-loading-order-bashrc-zshrc-etc/ (https://shreevatsa.wordpress.com/2008/03/30/zshbash-startup-files-loading-order-bashrc-zshrc-etc/)

http://www.solipsys.co.uk/new/BashInitialisationFiles.html (http://www.solipsys.co.uk/new/BashInitialisationFiles.html)

fish

Everything in `~/.config/fish` .

- `~/.config/fish/fishd.$hostname`
- `~/.config/fish/config.fish`
- `~/.config/fish/functions/*.fish`

I want more!

https://www.tldp.org/LDP/Bash-Beginners-Guide/html/ (https://www.tldp.org/LDP/Bash-Beginners-Guide/html/)

Other guides on https://www.tldp.org/ (https://www.tldp.org/)

Nice talking about it, but what can I use it for?

(credits to Riccardo Gilardi)

We simulate a 2-body system. We want to plot the kinetic energy, the potential energy, and the total energy.

The simulation returns such a file:

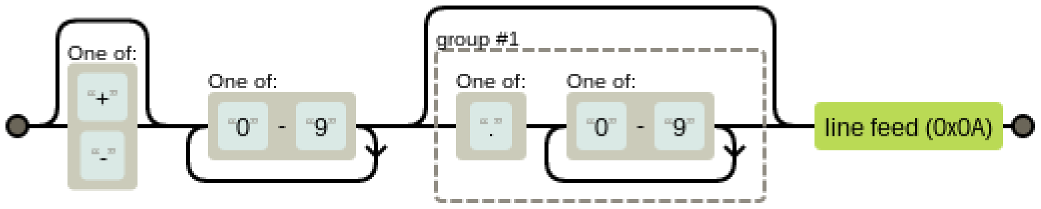
```
Starting a Hermite integration for a 2-body system,
  from time t = 0 with time step control parameter dt_param = 0.03  until time
 18.1 ,
  with diagnostics output interval dt_dia = 0.362,
  and snapshot output interval dt_out = 0.362.
at time t = 0 , after 0 steps :
  E_kin = 1.63228 , E_pot = -6.73401 , E_tot = -5.10172
              absolute energy error: E_tot - E_init = 0
              relative energy error: (E_tot - E_init) / E_init = -0
at time t = 0.369991 , after 24 steps :
  E_kin = 2.55889 , E_pot = -7.66061 , E_tot = -5.10172
              absolute energy error: E_tot - E_init = 1.80336e-07
              relative energy error: (E_tot - E_init) / E_init = -3.53481e-08
at time t = 0.726828 , after 62 steps :
  E_kin = 8.04706 , E_pot = -13.1488 , E_tot = -5.10172
              absolute energy error: E_tot - E_init = -5.91385e-07
              relative energy error: (E_tot - E_init) / E_init = 1.15919e-07
[...]
```

We can approach this task in many ways. We start in "full shell mode".

```
cat error.csv | grep E_kin | tr ',' '\n' | grep E_kin | egrep -Eo '[+-]?[0-9]+
([.][0-9]+)?' > ekin.txt

gnuplot
> plot 'ekin.txt'
```

Regular Expression

[https://unix.stackexchange.com/questions/290974/extracting-positive-negative-floating-point-numbers-from-a-string (https://unix.stackexchange.com/questions/290974/extracting-positive-negative-floating-point-numbers-from-a-string)](https://unix.stackexchange.com/questions/290974/extracting-positive-negative-floating-point-numbers-from-a-string)



We can also use some python

We can preparse the file:

```
grep 'E_kin' error.csv >> error_trimmed.txt
```

And we only have some lines such as

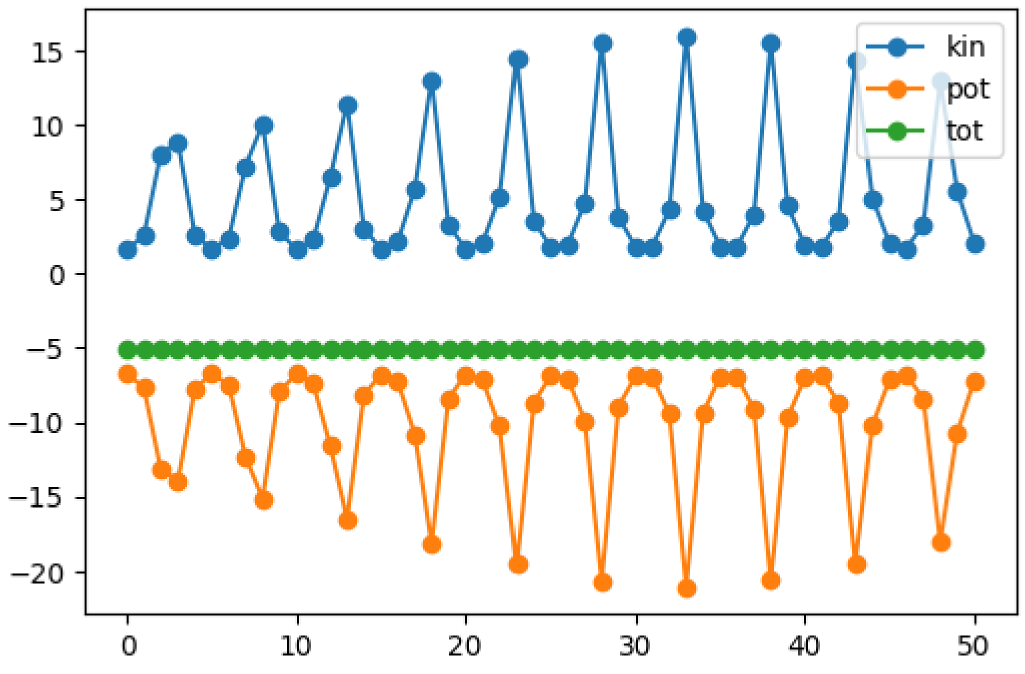
```
'  E_kin = 1.63228 , E_pot = -6.73401 , E_tot = -5.10172\n'
```

In [1]:

```
def get_energy(string, energy='kin'):
    valid = ('kin', 'pot', 'tot')
    if energy not in valid:
        raise ValueError('argument energy must be in ' + str(valid))
    return float(string.split(',')[valid.index(energy)].strip(" ").split("=")[1].strip(

# def get_pot_energy(string):
#    return float(string.split(',')[1].strip(" ").split("=")[1].strip(" "))

f = open('examples/error_trimmed.txt')
lines = f.readlines()
kin_energies = [get_energy(line, energy='kin') for line in lines]
pot_energies = [get_energy(line, energy='pot') for line in lines]
tot_energies = [get_energy(line, energy='tot') for line in lines]
```

In [4]:

```
from matplotlib import pyplot as plt

plt.figure(dpi=100)
plt.plot(kin_energies, marker='o', label='kin')
plt.plot(pot_energies, marker='o', label='pot')
plt.plot(tot_energies, marker='o', label='tot')
plt.legend()
plt.show()
```



In [5]:

```
# this is a simple case and the preprocessing is not really necessary

f1 = open("examples/error.csv")
lines = f1.readlines()
kin_energies = [get_energy(line, energy='kin') for line in lines if line.find("E_kin")
```

ASCII vs UTF8

Hardware monitoring

cpu

```
top
```

ram

```
free -m
```

dischi

```
sudo fdisks -l
lsblk
df -hT
du -sh ~
```

Network

```
ip addr
ping 8.8.8.8
host wikipedia.org
```

hardware

```
lshw -short
```

Altre referenze

- https://www.youtube.com/user/Computerphile (https://www.youtube.com/user/Computerphile)

  Buon canale, una serie di brevi spiegazioni su argomenti molto specifici o su concetti di base dell'Informatica. Ci sono anche una serie di interviste a Kernighan.