

**Linux for OSINT. 21-day course for  
beginners**

**@cyb\_detective 28 December 2023**

**<https://t.me/learningnets>**

While working on this course, I called a friend and asked:

*- Remember six months ago when you said you started reading a book about using Kali Linux for forensics?*

*- Yes.*

*- How much have you read?*

*- I stopped at chapter 5.*

*- What's it called?*

*- Installing Kali Linux*

After our conversation ended, I opened the file of this book and cut out all the chapters that require the use of Linux Virtual Machines. To give the reader no reason to stop, for the first 20 days, we will work in a browser only. It's not until day 21 that we think about whether it's time to install Linux.

## Table of contents

**Who Is This Course For?**

**Who should avoid this course?**

**Day 0. Getting ready to work**

**Day 1. Basic skills of working with files and directories in command line**

**Day 2. Basic bash script syntax**

**Day 3. Install and run utilities**

**Day 4. Batch file processing**

**Day 5. Downloading files and working with APIs. Curl.**

**Day 6. Search in files. Grep**

**Day 7. Sed and Awk**

**Day 8. Vim text editor**

**Day 9. Screen and Crone**

**Day 10. Text analyzing and editing utilities**

**Day 11. Video, audio and images**

**Day 12. Analyze PDF**

**Day 13. MS Office files**

**Day 14. JSON, XML, CSV**

**Day 15. Scraping**

**Day 16. Web search automation tools**

**Day 17. File sharing sites, torrents, FTP**

**Day 18. Domain investigation**

**Day 19. Git and Github**

**Day 20. Tools to make Linux easier to use**

**Day 21. Which Linux distribution is better to use?**

**What to do next?**

**Application. Is it possible to do the same thing on Windows?**

# Who Is This Course For?

The course is primarily intended for those **who are professionally involved in or simply interested in OSINT**. And you will find in it a lot about automation of collection and analysis of various data.

On my Twitter ([https://twitter.com/cyb\\_detective](https://twitter.com/cyb_detective)) account, I regularly write about useful tools for OSINT, including various Linux utilities. And many times readers have asked, "Cool! Is it possible to run this on Windows?".

Yes, there are different methods of running Linux utilities on Windows. But they are more complicated than running Linux on a VM (virtual machine), VPS (virtual private server) or in an online development environment.

This short tutorial is created to show you clearly that:

- Linux and the command line are very easy to use. Really easy. Extremely easy.
- hundreds of tasks (OSINT related and not only) can be simplified and automated with Linux.
- using Linux command line is a real, unparalleled pleasure.

This course is designed **for the total beginners**. I won't even ask you to install a VM with Linux until the last day, all examples can be tried in a browser. You also won't need any special knowledge. If you know how to use e-mail, you will be able to handle this course. Just relax and follow the instructions carefully.

When reading, keep in mind that this is a short course and so many things are left out. And really important things have been left out. It is designed primarily to show you the capabilities of Linux and to show you "which way to look" when faced with different tasks.

# Who should avoid this course?

Linux is good because you can solve the same problem in many different ways. I am not sure that the ways described in this course are the best and fastest ways to solve problems. They are written to visually show you the magic of the command line and stimulate your inner inventor (but all examples work and solve the stated problems).

I would advise the following categories of readers to treat this book as critically as possible and certainly read other Linux books:

- students who are preparing for exams
- IT specialists who are preparing for Linux-related job interviews
- people who want to become real Linux experts
- pentesters and bug hunters

This book is primarily about OSINT.

# Day 0. Getting ready to work

Eric Raymond, in his article “How to become a hacker” (1 December 1997, <https://cs.fit.edu/~wds/classes/cdc/Readings/BecomeAHacker.pdf>), recommended going to a local Linuxoid meeting and asking to burn discs of some distro in exchange for beer (I found this article, but link "where to get Linux" no longer working now).

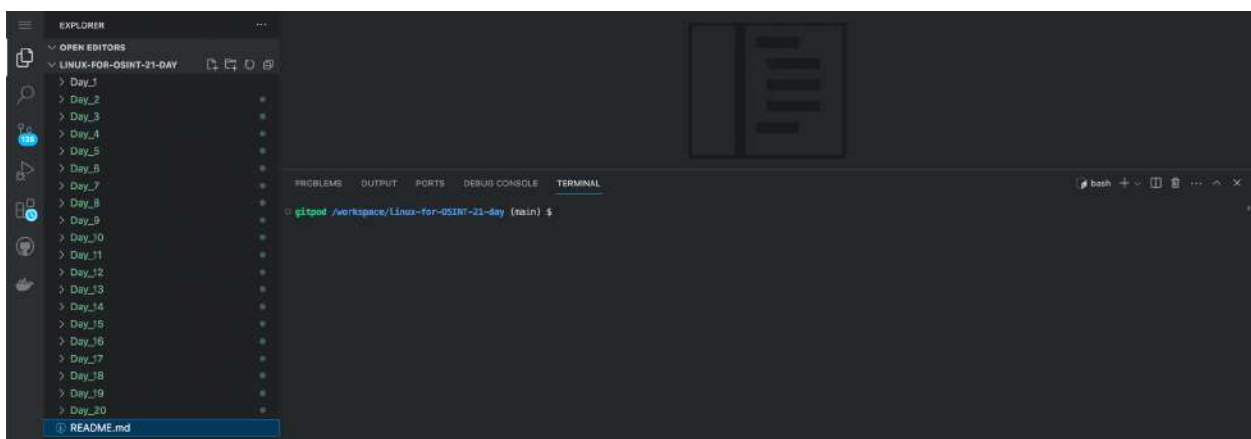
In the last 26 years progress has gone a long way and I'll just ask you to open a new tab in your browser.

Gitpod is a service that provides online development environments based on the Linux distribution Ubuntu (<https://ubuntu.com>). You can use it for 50 hours a month for free. This is definitely enough time for you to take this course **many times**.

You should use a Github (<https://github.com/>) account for authorization (registration is very fast and also free).

Open this link and create a workspace with standard settings:

<https://gitpod.io#https://github.com/cipher387/linux-for-OSINT-21-day>



Welcome to Gitpod! Now you're ready to start the first lesson!

If you are already using some Linux distribution, you can simply clone the <https://github.com/cipher387/linux-for-OSINT-21-day> repository to your computer or server.

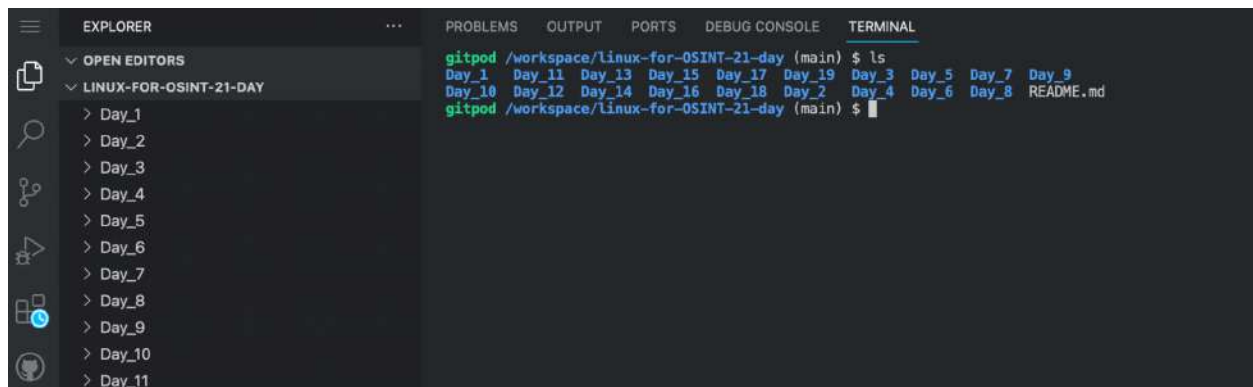
If you don't know how to clone repositories yet, I recommend using Gitpod for now. On **Day 19** you will learn how to clone repositories, and on **Day 21** you will think about which Linux distribution is best for you.

# Day 1. Basic skills of working with files and directories in command line

If you did not miss **Day 0**, you should now have the command line open and the directory **linux-for-OSINT-21-day** active.

Let's see what folders and files are inside it. Type **ls** at the command line and press **Enter**:

**ls**

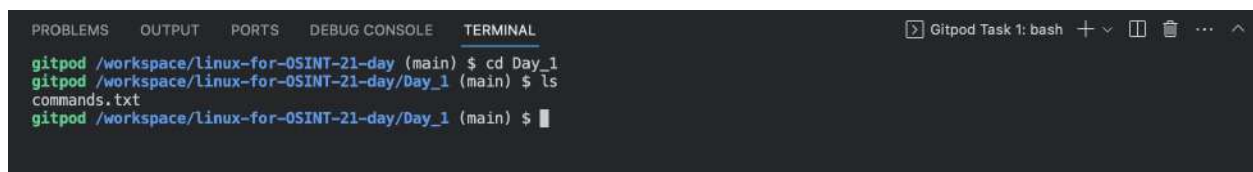


```
gitpod /workspace/linux-for-OSINT-21-day (main) $ ls
Day_1 Day_11 Day_13 Day_15 Day_17 Day_19 Day_3 Day_5 Day_7 Day_9
Day_10 Day_12 Day_14 Day_16 Day_18 Day_2 Day_4 Day_6 Day_8 README.md
gitpod /workspace/linux-for-OSINT-21-day (main) $
```

Now let's go to the **Day\_1** folder:

**cd Day\_1**

**ls**



```
gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_1
gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $ ls
commands.txt
gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $
```

At the moment there is only a text file with all the commands from this book. Let's add something else to the **Day\_1** folder.

**mkdir new\_folder**

## ls

```
PROBLEMS  OUTPUT  PORTS  DEBUG CONSOLE  TERMINAL
● gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_1
● gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $ ls
  commands.txt
● gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $ mkdir new_folder
● gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $ ls
  commands.txt  new_folder
○ gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $ █
```

Let's create a new empty text file in this folder.

## cat>new\_file.txt

Enter some text and press “**Ctrl+D**”.

Let's check if the file was created successfully:

## ls

```
PROBLEMS  OUTPUT  PORTS  DEBUG CONSOLE  TERMINAL
● gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_1
● gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $ cat>new_file.txt
  Hello!
● gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $ ls
  commands.txt  new_file.txt  new_folder
○ gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $ █
```

To go to the parent directory use this command:

## cd ..

```
● gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $ cd ..
● gitpod /workspace/linux-for-OSINT-21-day (main) $ cd ..
○ gitpod /workspace $ █
```

Go to the directory located several levels below by specifying the path to it:

**cd linux-for-OSINT-21-day/Day\_1**

```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
● gitpod /workspace/linux-for-OSINT-21-day (main) $ cd ..
● gitpod /workspace $ cd linux-for-OSINT-21-day/Day_1
```

Now let's change the text of the file `new_file.txt` and display it on the screen (write the results of the `ls` command to a file):

**ls>new\_file.txt**  
**cat new\_file.txt**

```
● gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $ ls>new_file.txt
● gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $ cat new_file.txt
commands.txt
new_file.txt
new_folder
○ gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $ █
```

Note that this is the second time we have used the `>` symbol to write the results of a command to a file. Pay special attention to this.

If you just want to copy text from one file to another, use the `cp` command:

**cp new\_file.txt copy\_new\_file.txt**

```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
● gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_1
● gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $ cp new_file.txt copy_new_file.txt
● gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $ ls
commands.txt copy_new_file.txt new_file.txt new_folder
● gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $ cat copy_new_file.txt
commands.txt
new_file.txt
new_folder
○ gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $ █
```

Directories can be copied in the same way:

```
cp -r new_folder copy_new_folder
```

```
● gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $ cp -r new_folder copy_new_folder
● gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $ ls
  commands.txt  copy_new_file.txt  copy_new_folder  new_file.txt  new_folder
○ gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $
```

Now let's delete the file and directory copies we created:

```
rm copy_new_file.txt
```

```
rm -d copy_new_folder
```

```
● gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $ ls
  commands.txt  copy_new_file.txt  copy_new_folder  new_file.txt  new_folder
● gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $ rm copy_new_file.txt
● gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $ ls
  commands.txt  copy_new_folder  new_file.txt  new_folder
● gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $ rm -d copy_new_folder
● gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $ ls
  commands.txt  new_file.txt  new_folder
○ gitpod /workspace/linux-for-OSINT-21-day/Day_1 (main) $
```

Enough for now. Today we have learned the most basic commands for working with Linux files. And we tried the simplest options to use them. But all utilities mentioned in this lesson (**cat**, **ls**, **cd**, **rm**) have many different options.

To find out all the options of any Linux command use the **man** (manual) command. For example:

```
man rm
```

```
PROBLEMS  OUTPUT  PORTS  DEBUG CONSOLE  TERMINAL
RM(1)                                           User Commands  RM(1)

NAME
rm - remove files or directories

SYNOPSIS
rm [OPTION]... [FILE]...

DESCRIPTION
This manual page documents the GNU version of rm.  rm removes each specified file.  By default, it does not
remove directories.

If the -I or --interactive=once option is given, and there are more than three files or the -r, -R, or --re-
cursive are given, then rm prompts the user for whether to proceed with the entire operation.  If the response
is not affirmative, the entire command is aborted.

Otherwise, if a file is unwritable, standard input is a terminal, and the -f or --force option is not given,
or the -i or --interactive=always option is given, rm prompts the user for whether to remove the file.  If the
response is not affirmative, the file is skipped.

OPTIONS
Remove (unlink) the FILE(s).

-f, --force
    ignore nonexistent files and arguments, never prompt

-i
    prompt before every removal

-I
    prompt once before removing more than three files, or when removing recursively; less intrusive than
    -i, while still giving protection against most mistakes

--interactive[=WHEN]
    prompt according to WHEN: never, once (-I), or always (-i); without WHEN, prompt always

Manual page rm(1) line 1/92 41% (press h for help or q to quit)
```

Press **q** to exit.

If you did something wrong when typing some commands, always remember that you can just create a new workspace, if you are working on Gitpod.

If you are working on your computer, VM or VPS, delete the folder with the files for this course and then copy it again from Github.

# Day 2. Basic Bash syntax

What is the most important point of using the command line to perform various tasks instead of the GUI?

**The ability to record and repeatedly perform different sequences of actions to automate your work.**

Bash (<https://www.gnu.org/software/bash/>) is a Unix shell and command language written by Brian Fox and first released in 1989. This is what we will use to write commands to automate various tasks in Linux, combining Bash scripts with tools written in other programming languages (Python, Go etc).

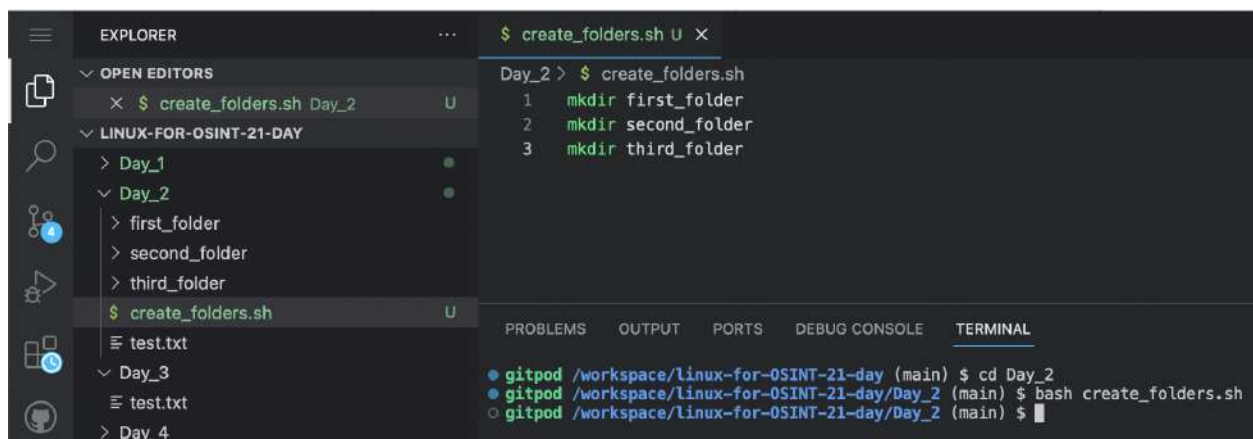
Commands for later execution can be stored in files with the extension **.sh**. Let's take a look at how this works right away.

Go to current lesson folder:

```
cd Day_2
```

And run file create\_folders.sh:

```
bash create_folders.sh
```



The screenshot shows a code editor interface with a terminal window. The terminal window displays the following output:

```
Day_2 > $ create_folders.sh
1 mkdir first_folder
2 mkdir second_folder
3 mkdir third_folder
```

The terminal window also shows the following output:

```
gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_2
gitpod /workspace/linux-for-OSINT-21-day/Day_2 (main) $ bash create_folders.sh
gitpod /workspace/linux-for-OSINT-21-day/Day_2 (main) $
```

If you didn't miss yesterday's lesson, you can see that this script simply creates three directories with different names.

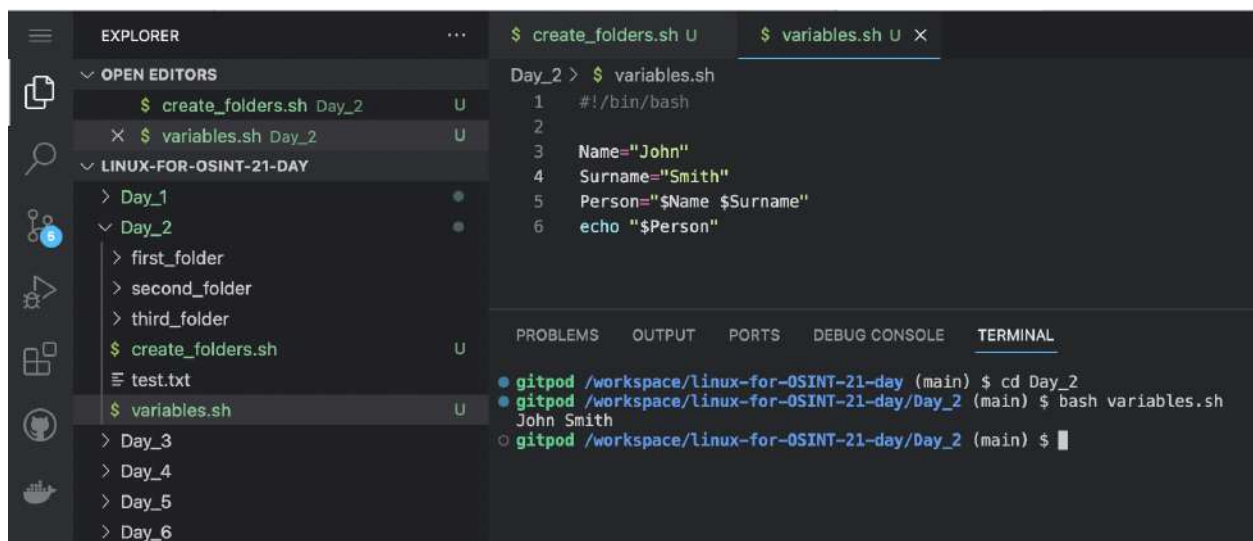
The `.sh` extension is used to store files with code written in the Shell Script language or Bash script language.

Now let's break down the most basic elements of Bash script syntax.

## Variables and string concatenation

Run in command line:

**bash variables.sh**



The screenshot shows a code editor with two tabs: `create_folders.sh` and `variables.sh`. The `variables.sh` tab is active, displaying the following code:

```
Day_2 > $ variables.sh
1  #!/bin/bash
2
3  Name="John"
4  Surname="Smith"
5  Person="$Name $Surname"
6  echo "$Person"
```

Below the code editor is a terminal window showing the execution of the script:

```
gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_2
gitpod /workspace/linux-for-OSINT-21-day/Day_2 (main) $ bash variables.sh
John Smith
gitpod /workspace/linux-for-OSINT-21-day/Day_2 (main) $
```

Comments in the code denote by the symbol `#`.

In most cases this will not affect the scripts, but I recommend that you indicate on the first line of the `.sh` files that it is a Bash script:

**#!/bin/bash**

Create variable contain a string "John":

**Name="John"**

Create variable contain a string "Smith":

```
Surname="Smith"
```

Create variable contain Name variable, space and Surname variable:

```
Person="$Name $Surname"
```

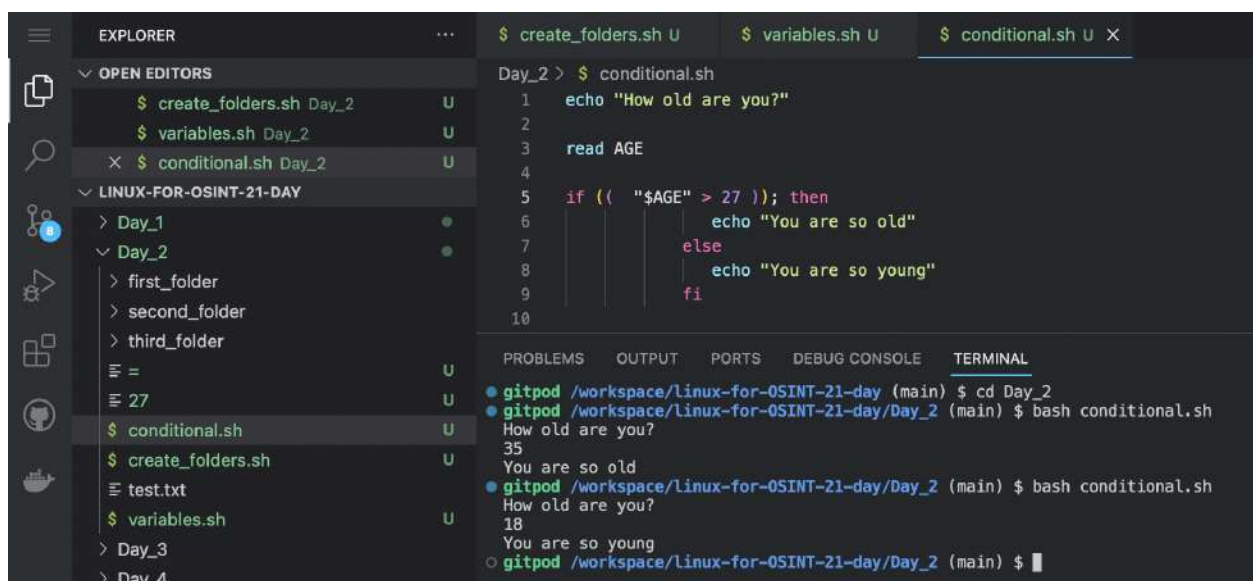
Print Person variable:

```
echo "$Person"
```

## Conditional operator

Run conditional.sh:

```
cd Day_2  
bash conditional.sh
```



The screenshot shows a code editor with three tabs: `create_folders.sh`, `variables.sh`, and `conditional.sh`. The `conditional.sh` tab is active, showing the following script:

```
Day_2 > $ conditional.sh  
1 echo "How old are you?"  
2  
3 read AGE  
4  
5 if (( "$AGE" > 27 )); then  
6     echo "You are so old"  
7 else  
8     echo "You are so young"  
9 fi  
10
```

The terminal output at the bottom shows the script being executed in two instances:

```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL  
● gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_2  
● gitpod /workspace/linux-for-OSINT-21-day/Day_2 (main) $ bash conditional.sh  
How old are you?  
35  
You are so old  
● gitpod /workspace/linux-for-OSINT-21-day/Day_2 (main) $ bash conditional.sh  
How old are you?  
18  
You are so young  
○ gitpod /workspace/linux-for-OSINT-21-day/Day_2 (main) $
```

Ask the user how old he is:

```
echo "How old are you?"
```

Read the value of the AGE variable entered by the user:

```
read AGE
```

If AGE is greater than 27, we tell the user that he is too old:

```
if (( "$AGE" > 27 )); then  
    echo "You are so old"
```

If AGE is less than 27, we tell the user that he is too young:

```
    else  
        echo "You are so young"  
    fi
```

## Loops

There are three basic kinds of loops in a bash script: "while loop", "until loop" and "for loop".

### While loop

Run in command line:

```
bash while_loop.sh
```

The screenshot shows a code editor with a file explorer on the left and a terminal on the right. The file explorer shows a project named 'LINUX-FOR-OSINT-21-DAY' with subfolders 'Day\_1', 'Day\_2', and 'Day\_3'. The 'Day\_2' folder is expanded, showing files like 'conditional.sh', 'create\_folders.sh', 'test.txt', 'variables.sh', and 'while\_loop.sh'. The terminal window shows the execution of 'while\_loop.sh' in the 'Day\_2' directory. The script content is as follows:

```
1 x=1
2 while [ $x -le 5 ]
3 do
4     echo "$x"
5     x=$(( $x + 1 ))
6 done
```

The terminal output shows the script being executed and the variable 'x' being printed and incremented from 1 to 5.

Assign the value 1 to the variable x:

```
x=1
```

As long as the variable x does not equal 5, perform the following action:

```
while [ $x -le 5 ]  
do
```

Display the variable x on the screen and then add 1 to it:

```
echo "$x"  
x=$(( $x + 1 ))  
done
```

## Until loop

Run:

```
bash until_loop.sh
```

```
EXPLORER
... $ until_loop.sh U X
Day_2 > $ until_loop.sh
1 #!/bin/bash
2
3 x=15
4 until [ $x -lt 10 ]
5 do
6     echo "$x"
7     x=$(( $x - 1 ))
8 done

PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_2
gitpod /workspace/linux-for-OSINT-21-day/Day_2 (main) $ bash until_loop.sh
15
14
13
12
11
10
gitpod /workspace/linux-for-OSINT-21-day/Day_2 (main) $
```

Assign the value 15 to the variable x:

```
x=15
```

As long as the variable x greater than 1-, perform the following action:

```
until [ $x -lt 10 ]
```

```
Do
```

Display the variable x on the screen and then subtract 1 to it:

```
echo "$x"
```

```
x=$(( $x - 1 ))
```

```
done
```

For loop

Run:

```
bash for_loop.sh
```

The screenshot shows a code editor with a file explorer on the left and a terminal at the bottom. The file explorer shows a project named 'LINUX-FOR-OSINT-21-DAY' with subfolders 'Day\_1', 'Day\_2', 'Day\_3', 'Day\_4', and 'Day\_5'. The 'Day\_2' folder is expanded, showing files like 'conditional.sh', 'create\_folders.sh', 'for\_loop.sh', 'test.txt', 'until\_loop.sh', 'variables.sh', and 'while\_loop.sh'. The 'for\_loop.sh' file is selected and its content is visible in the editor:

```
Day_2 > $ for_loop.sh
1  #!/bin/bash
2
3  declare -a names=("John", "Paul", "Steven", "Bill")
4
5  for name in ${names[@]};
6  do
7      echo $name;
8  done
```

The terminal at the bottom shows the execution of the script:

```
gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_2
gitpod /workspace/linux-for-OSINT-21-day/Day_2 (main) $ bash for_loop.sh
John,
Paul,
Steven,
Bill
gitpod /workspace/linux-for-OSINT-21-day/Day_2 (main) $
```

Create array with list of names:

```
declare -a names=("John", "Paul", "Steven", "Bill")
```

Go through the elements of the list one by one and do the following for each one:

```
for name in ${names[@]};  
do
```

Display current array element on the screen:

```
echo $name;  
done
```

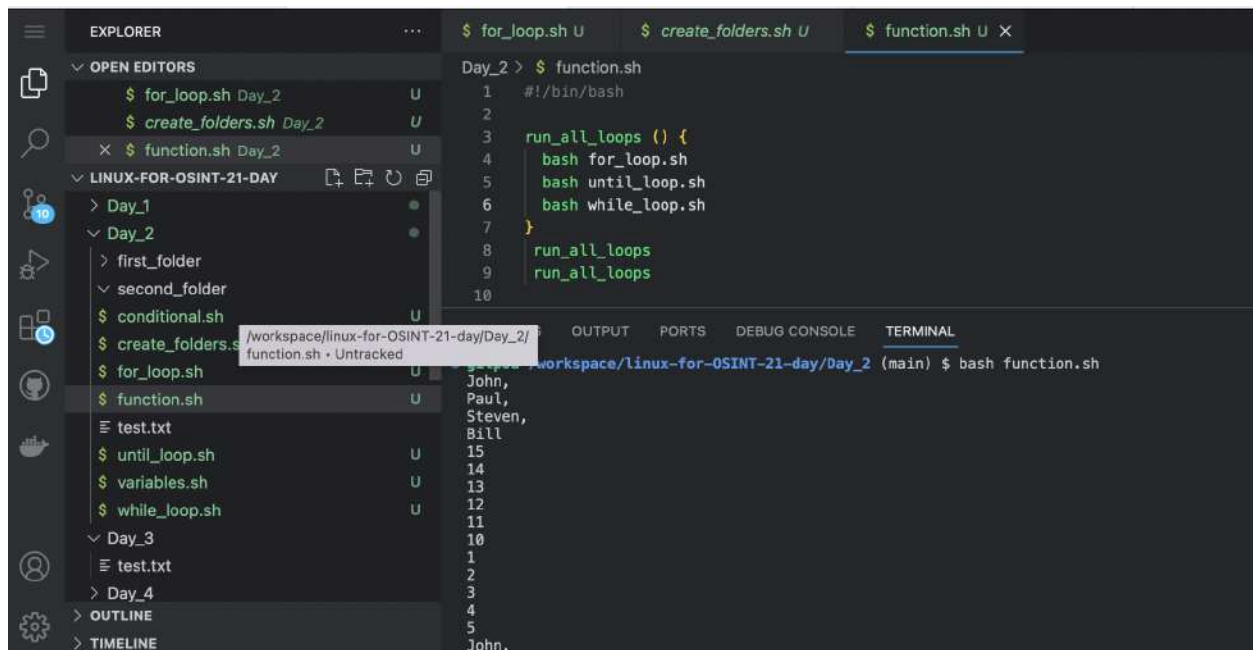
## Functions

Like many other programming languages, bash allows you to combine long programs into functions, so that you can then trigger their execution with a short function name.

Let's look at an example.

Run in command line:

**bash function.sh**



Create a function that runs scripts to demonstrate how loops work:

```
run_all_loops () {  
  bash for_loop.sh  
  bash until_loop.sh  
  bash while_loop.sh  
}
```

Call this function twice:

```
run_all_loops  
Run_all_loops
```

As in other programming languages, bash allows you to use arguments in functions and return the values obtained after processing these arguments. But we will not discuss this point in detail, as it is already one of the longest lessons of this course (yes, it will be easier later).

---

Don't try to learn the basics of bash script syntax right now, we will barely use it in this course (except a little on **Day 4**). This lesson is created primarily to give you an overview of its basic features so that you can stimulate your brain to generate ideas in the future

The most interesting thing about Bash script functions is that you can combine scripts written in different programming languages. For example, combine different Python OSINT tools with a Nuclei scanner written in Go as well as with a wide variety of Linux utilities. Tomorrow we will finally learn how to install them.

# Day 3. Install and run utilities

Linux utility is a command that can be run on the command line or on bash scripts. We have already used some of these in past lessons: **cat**, **bash**, **cd**, **man**, **cp**, **mkdir**. These are utilities that are installed by default in every Linux distribution.

You can see the full list of utilities and applications installed in your own distribution using a special command. Which one depends on the type of your distribution. For example:

Ubuntu, Debian: **dpkg -l**

Fedora, RHEL: **rpm -qa**

OpenBSD, FreeBSD: **pkg\_info**

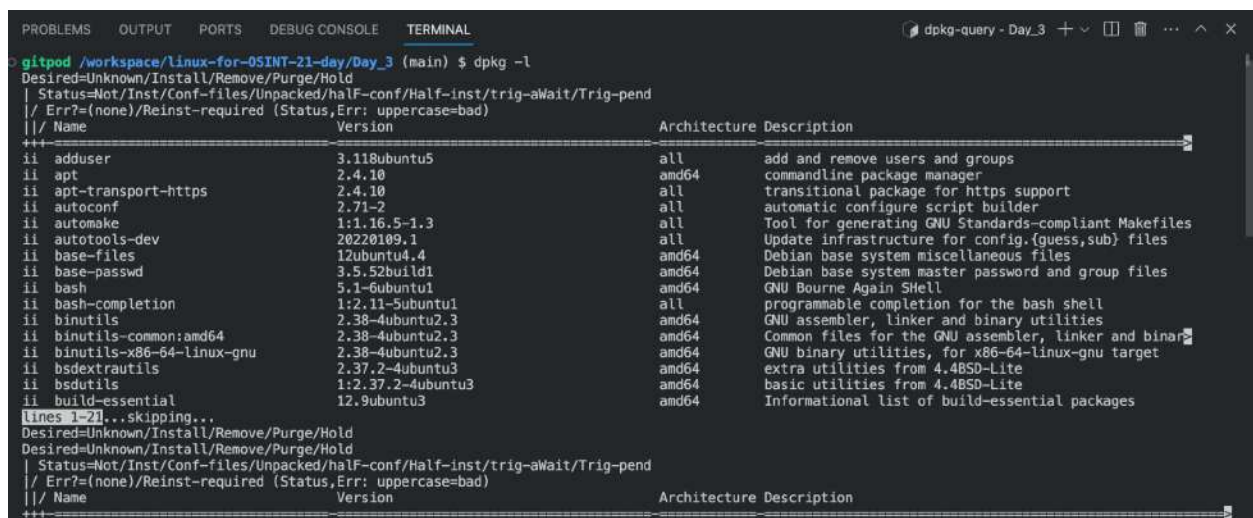
Gentoo: **equery list** or **eix -l**

Arch Linux: **pacman -Q**

If you are using Gitpod, you need to use the commands for Ubuntu.

Run in command line:

**dpkg -l**



```
gitpod /workspace/linux-for-05INT-21-day/Day_3 (main) $ dpkg -l
Desired=Unknown/Install/Remove/Purge/Hold
 | Status=Not/Inst/Conf-files/Unpacked/halF-conf/Half-inst/trig-aWait/Trig-pend
 / Err?=(none)/Reinst-required (Status,Err: uppercase=bad)
++/ Name                    Version                Architecture Description
++-----+-----+-----+-----+
ii adduser                   3.118ubuntu5          all             add and remove users and groups
ii apt                       2.4.10                amd64           commandline package manager
ii apt-transort-https        2.4.10                all            transitional package for https support
ii autoconf                  2.71-2                all            automatic configure script builder
ii automake                   1:1.16.5-1.3         all            Tool for generating GNU Standards-compliant Makefiles
ii autotools-dev             20220109.1           all            Update infrastructure for config.{guess,sub} files
ii base-files                12ubuntu4.4          amd64          Debian base system miscellaneous files
ii base-passwd               3.5.52build1         amd64          Debian base system master password and group files
ii bash                       5.1-6ubuntu1         amd64          GNU Bourne Again SHell
ii bash-completion           1:2.11-5ubuntu1      all            programmable completion for the bash shell
ii binutils                  2.38-4ubuntu2.3      amd64          GNU assembler, linker and binary utilities
ii binutils-common:amd64    2.38-4ubuntu2.3      amd64          Common files for the GNU assembler, linker and binary
ii binutils-x86-64-linux-gnu 2.38-4ubuntu2.3      amd64          GNU binary utilities, for x86-64-linux-gnu target
ii bsdxtrautils              2.37.2-4ubuntu3      amd64          extra utilities from 4.4BSD-Lite
ii bsduutils                 1:2.37.2-4ubuntu3    amd64          basic utilities from 4.4BSD-Lite
ii build-essential           12.9ubuntu3           amd64          Informational list of build-essential packages
lines 1-21... skipping...
Desired=Unknown/Install/Remove/Purge/Hold
Desired=Unknown/Install/Remove/Purge/Hold
 | Status=Not/Inst/Conf-files/Unpacked/halF-conf/Half-inst/trig-aWait/Trig-pend
 / Err?=(none)/Reinst-required (Status,Err: uppercase=bad)
++/ Name                    Version                Architecture Description
++-----+-----+-----+-----+
```

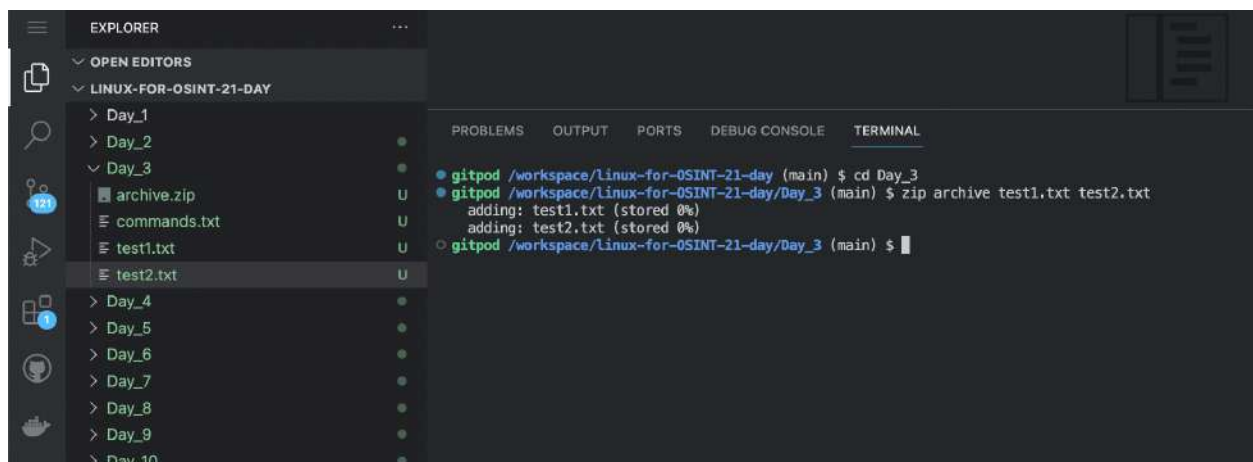
There are also thousands of utilities for a wide variety of purposes that you can install yourself.

As an example, let's install two utilities for working with ZIP archives:

```
sudo apt install zip unzip
```

In this example it is not necessary (we could just use **apt install**), but we add the word **sudo** before the command.

This way we run the command with superuser privileges (maximum permissions to write, read and delete files). In Gitpod, you can use it just like that, but if you are working on your own computer and are not logged in as an administrator, you may have to enter a password.



Now let's try to convert the file into a zip archive:

```
cd Day_3
zip archive test1.txt test2.txt
```

Instead of file names, we could specify directory names or put all txt files from the active directory into the archive using \*.txt instead of listing filenames.

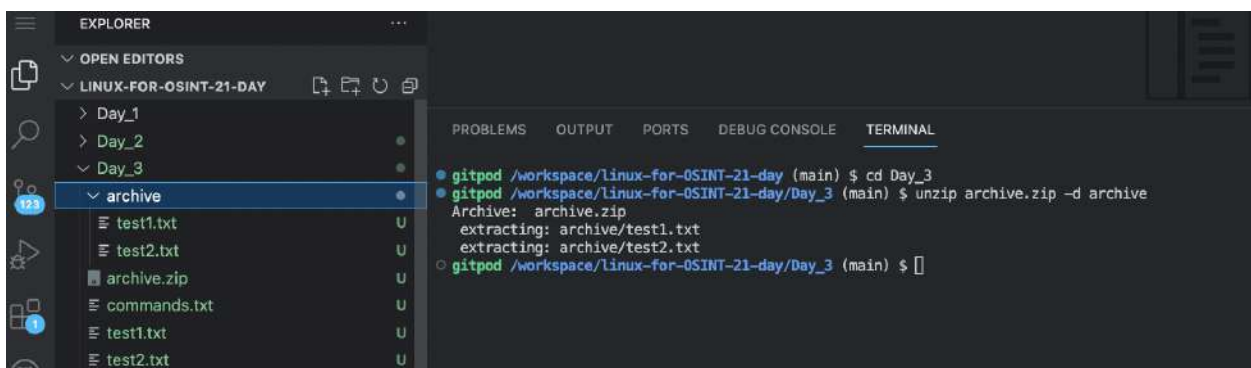
```

● gitpod /workspace/linux-for-OSINT-21-day/Day_3 (main) $ zip -sf archive.zip
Archive contains:
  test1.txt
  test2.txt
Total 2 entries (0 bytes)
○ gitpod /workspace/linux-for-OSINT-21-day/Day_3 (main) $ █

```

Now let's see what files are inside the archive using **-sf** flag:

**zip -sf archive.zip**



You can unzip the archive to a specific folder using the **unzip** command:

**unzip archive.zip -d archive**

You can use the **rar** and **unrar** utilities to work with WinRAR archives.

In the example above, we used apt package manager (<https://ubuntu.com/server/docs/package-management>), which is installed by default in Ubuntu. But there are other package managers that can be used both in Ubuntu and other distributions. Few examples:

PyPi (<https://pypi.org>) - Python Package Index.

**pip3 install ddgr**

Snap Store (<https://snapcraft.io/store>) - software store developed by Canonical.

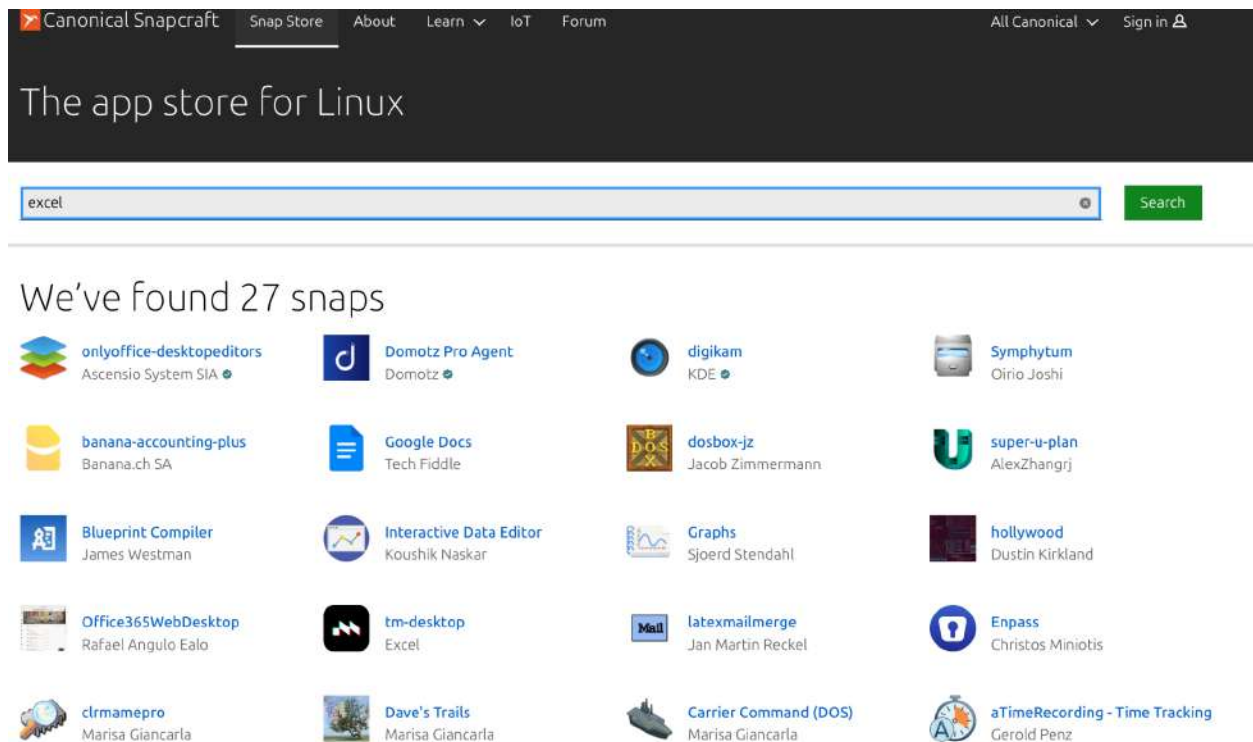
## snap install ddgr

Cast (<https://sourcemage.org>) - package manager for Source Mage Linux distribution.

## cast ddgr

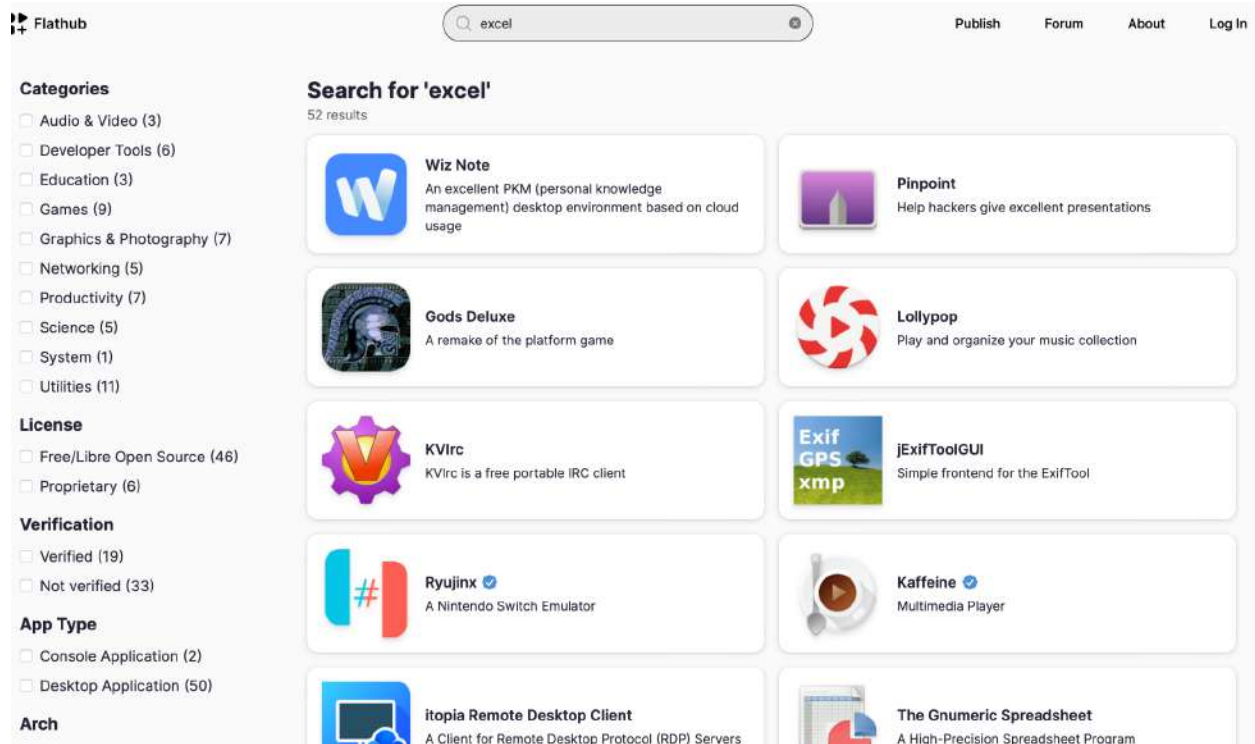
\* (for each package manager, a sample command for installing the ddgr utility is listed, but you do not need to run them. We will install it on **Day 16** of this course)

In this course, we focus on working with the command line. Well for Linux there are a huge number of GUI applications. They are very easy to install through the app stores.



The screenshot shows the Canonical Snapcraft website interface. At the top, there is a navigation bar with links for 'Snap Store', 'About', 'Learn', 'IoT', and 'Forum'. The main heading reads 'The app store for Linux'. Below this is a search bar containing the text 'excel' and a green 'Search' button. The results section is titled 'We've found 27 snaps' and displays a grid of application cards. Each card includes an icon, the application name, and the developer's name. The applications shown include: onlyoffice-desktopeditors (Ascensio System SIA), Domotz Pro Agent (Domotz), digikam (KDE), Symphytum (Oirio Joshi), banana-accounting-plus (Banana.ch SA), Google Docs (Tech Fiddle), dosbox-jz (Jacob Zimmermann), super-u-plan (AlexZhangrj), Blueprint Compiler (James Westman), Interactive Data Editor (Koushik Naskar), Graphs (Sjoerd Stendahl), hollywood (Dustin Kirkland), Office365WebDesktop (Rafael Angulo Ealo), tm-desktop (Excel), latexmailmerge (Jan Martin Reckel), Enpass (Christos Miniotis), clrmamepro (Marisa Giancarla), Dave's Trails (Marisa Giancarla), Carrier Command (DOS) (Marisa Giancarla), and aTimeRecording - Time Tracking (Gerold Penz).

Snap is a software packaging and deployment system developed by Canonical. Snapcraft (<http://snapcraft.io>) contain thousands of snaps (apps) for 41 Linux distributions.



Flathub (<http://flathub.org>) - another app store that contains more apps. But unlike Snap, which allows you to install command line utilities, Flathub only supports GUI applications.

If you use Gitpod, you won't be able to try them right now. Return to this chapter at **Day 21** after you select a VM to continue using Linux with a graphical interface.

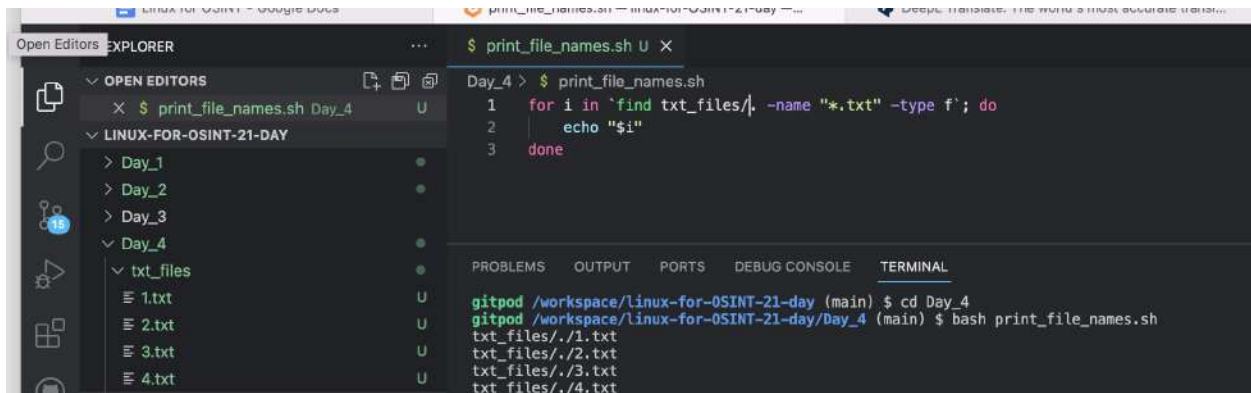
# Day 4. Batch file processing

With Bash scripts you can manipulate thousands of files with a few lines of code.

For example, you can perform the same action on all files with a certain extension located in a certain folder. For this we will use the **Find** utility (<https://man7.org/linux/man-pages/man1/find.1.html>). It is installed in Linux by default.

Please, type in command line:

```
cd Day_4  
bash print_file_names.sh
```



```
Day_4 > $ print_file_names.sh  
1 for i in `find txt_files/. -name "*.txt" -type f`; do  
2     echo "$i"  
3 done  
  
gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_4  
gitpod /workspace/linux-for-OSINT-21-day/Day_4 (main) $ bash print_file_names.sh  
txt_files/./1.txt  
txt_files/./2.txt  
txt_files/./3.txt  
txt_files/./4.txt
```

Source code of print\_file\_names.sh:

```
for i in `find txt_files/. -name "*.txt" -type f`; do  
    echo "$i"  
done
```

You can replace .txt with any other extension. You can also search for files whose names contain a certain word.

In truth, the script above doesn't make much sense on its own, since the file names are displayed simply by running the find command. Try this:

```
find -name "*.txt" -type f
```

```
● gitpod /workspace/linux-for-OSINT-21-day (main) $ find -name "*.txt" -type f
./Day_1/commands.txt
./Day_10/test.txt
./Day_11/test.txt
./Day_12/test.txt
./Day_13/test.txt
./Day_14/example.txt
./Day_15/test.txt
./Day_16/test.txt
./Day_17/test.txt
./Day_18/test.txt
./Day_19/test.txt
./Day_2/test.txt
./Day_3/test.txt
./Day_4/test.txt
./Day_5/test.txt
./Day_6/test.txt
./Day_7/test.txt
./Day_8/test.txt
./Day_9/test.txt
○ gitpod /workspace/linux-for-OSINT-21-day (main) $ █
```

But by going through the file names one by one, we can perform a certain repetitive action on each file (process file with any Linux utility). Just replace “echo” command with something else.

As example, let's try to convert all txt files to csv format.

## Save .txt files as .csv

```
EXPLORER
... $ convert_files.sh U X
OPEN EDITORS
x $ convert_files.sh Day_4 U
LINUX-FOR-OSINT-21-DAY
  Day_2
  Day_3
  Day_4
    txt_files
      1.csv U
      1.txt U
      2.csv U
      2.txt U
      3.csv U
      3.txt U
      4.csv U
      4.txt U
    $ convert_files.sh U
    $ print_file_names.sh U
Day_4 > $ convert_files.sh
1 for i in `find txt_files/. -name "*.txt" -type f`; do
2   sed 's/ \+/,/g' "$i">"${i%.*}.csv";
3 done
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_4
gitpod /workspace/linux-for-OSINT-21-day/Day_4 (main) $ bash convert_files.sh
gitpod /workspace/linux-for-OSINT-21-day/Day_4 (main) $
```

Enter in command line:

```
bash convert_files.sh
```

Source code of `convert_files.sh`:

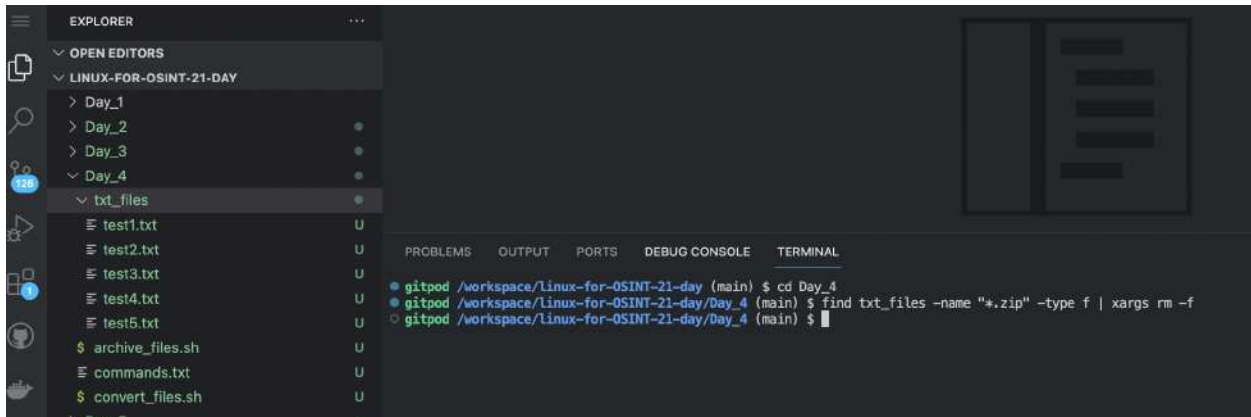
```
for i in `find txt_files/. -name "*.txt" -type f`; do  
  sed 's/ \+/,/g' "$i">"${i%.*}.csv";  
done
```

Note the `sed` (<https://www.gnu.org/software/sed/manual/sed.html>) command. We use to replace spaces with commas and return to it in **Day 7**.

# Xargs

Xargs (<https://man7.org/linux/man-pages/man1/xargs.1.html>) is a utility that allows you to run different Linux commands with certain input parameters. This includes reading these parameters from lists stored in files.

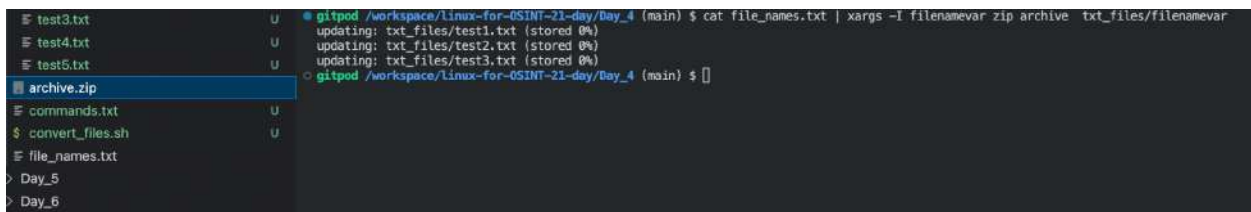
Let's look at examples of how it works.



Let's delete all the archives from the txt\_files folder:

```
find txt_files -name "*.zip" -type f | xargs rm -f
```

Now let's create archives using Xargs:



```
cat file_names.txt | xargs -I filenamevar zip archive  
txt_files/filenamevar
```

As you can see, in the right part of the command, we read the file line by line using the **Cat** command. And on the left side, we run **Xargs** with the **-I** flag to use each line as a **filenamevar** variable and use that variable as the filename in the parameters when we run the zip command.

The material on this day can be combined with almost all of the chapters in our course. But I recommend that you pay special attention to some of them:

**Day 5** - file downloading.

**Day 7** and **Day 10** - text file editing.

**Day 11** - working with video, audio and images.

**Day 12** - working with PDF.

**Day 13** - working with MS Office files.

**Day 14** - working with JSON, CSV and XML files.

After reading the lesson in each of these days, I recommend running at least one command in combination with **Xargs**.

## Day 5. Downloading files and working with APIs. Curl.

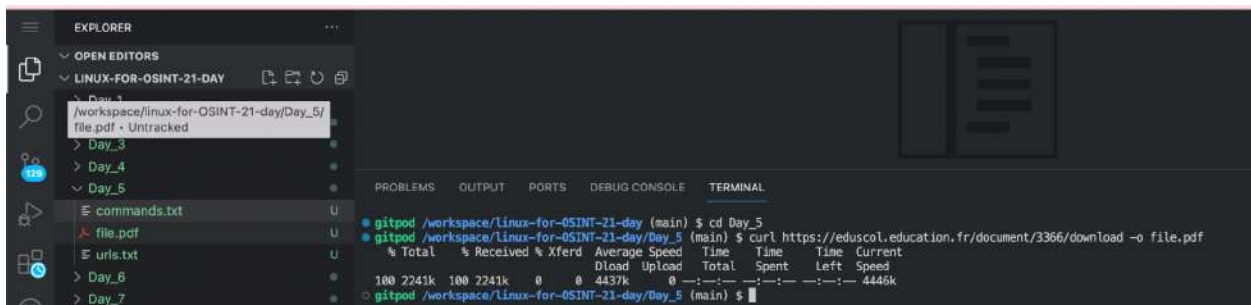
Today we're going to learn one of the most important skills for automating data collection in Linux - downloading files using the **Curl** utility. This comes in handy when processing API requests, scraping websites, downloading data from archives or other sources.

```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
● gitpod /workspace/Linux-for-OSINT-21-day (main) $ curl -h
Usage: curl [options...] <url>
  -d, --data <data>           HTTP POST data
  -f, --fail                   Fail silently (no output at all) on HTTP errors
  -h, --help <category>      Get help for commands
  -i, --include                 Include protocol response headers in the output
  -o, --output <file>         Write to file instead of stdout
  -O, --remote-name            Write output to a file named as the remote file
  -s, --silent                 Silent mode
  -T, --upload-file <file>    Transfer local FILE to destination
  -u, --user <user:password>  Server user and password
  -A, --user-agent <name>     Send User-Agent <name> to server
  -v, --verbose                Make the operation more talkative
  -V, --version                Show version number and quit

This is not the full help, this menu is stripped into categories.
Use "--help category" to get an overview of all categories.
For all options use the manual or "--help all".
○ gitpod /workspace/Linux-for-OSINT-21-day (main) $ █
```

**Curl** (<https://curl.se>) is a tool to transfer data from or to a server, using one of the a lot of supported protocols (DICT, FILE, FTP, FTPS, GOPHER, HTTP, HTTPS, IMAP, IMAPS, LDAP, LDAPS, POP3, POP3S, RTMP, RTSP, SCP, SFTP, SMB, SMBS, SMTP, SMTPS, TELNET and TFTP). It does not require installation. It is available in Linux by default.

Let's try to download file. Change original name to file.pdf with **-o** flag:



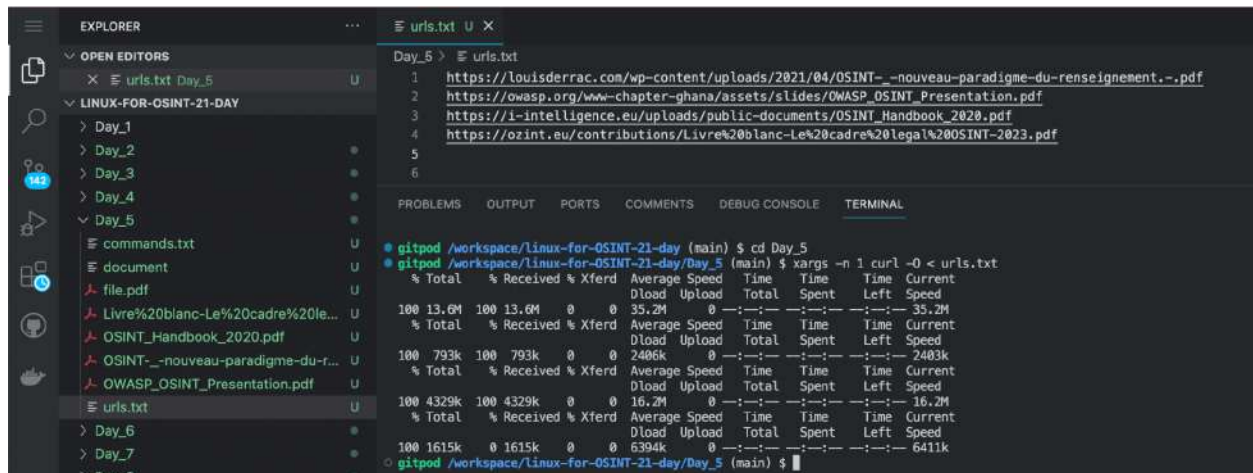
```
gitpod /workspace/Linux-for-OSINT-21-day (main) $ cd Day_5
gitpod /workspace/Linux-for-OSINT-21-day/Day_5 (main) $ curl https://eduscol.education.fr/document/3366/download -o file.pdf
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 2241k 100 2241k 0 0 4437k 0 --:--:-- --:--:-- --:--:-- 4446k
gitpod /workspace/Linux-for-OSINT-21-day/Day_5 (main) $ █
```

Enter in command line:

## Cd Day\_5

```
curl https://eduscol.education.fr/document/3366/download -o file.pdf
```

Now let's try downloading files from the list of links using the **Xargs** command (we talked about it yesterday).



```
gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_5
gitpod /workspace/linux-for-OSINT-21-day/Day_5 (main) $ xargs -n 1 curl -O < urls.txt
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 13.6M 100 13.6M 0 0 35.2M 0 --:--:-- --:--:-- --:--:-- 35.2M
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 793k 100 793k 0 0 2406k 0 --:--:-- --:--:-- --:--:-- 2403k
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 4329k 100 4329k 0 0 16.2M 0 --:--:-- --:--:-- --:--:-- 16.2M
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 1615k 0 1615k 0 0 6394k 0 --:--:-- --:--:-- --:--:-- 6411k
gitpod /workspace/linux-for-OSINT-21-day/Day_5 (main) $
```

```
xargs -n 1 curl -O < urls.txt
```

## API requests

Using the **-H** flag, you can send requests with different http headers. This function is useful when automating the execution of requests to various APIs.

Simple example for Netlas API (<https://github.com/netlas-io/netlas-cookbook>):



Using this query, we retrieved a JSON file with detailed domain information found in Netlas IP search engine by query "lidl.com". You can view it in a convenient format using some online JSON file viewer tools (<https://jsonviewer.stack.hu/>).

On **Day 14**, we'll talk a little more about data in JSON format and learn how to process it using the **jq** utility.

People and documents verification			
Name	Link	Description	Price
Approve.com	<a href="https://approve.co">https://approve.co</a>	Allows you to verify the identities of individuals, businesses, and connect to financial account data across Africa	Paid
Onfido.com	<a href="https://onfido.com">https://onfido.com</a>	Onfido Document Verification lets your users scan a photo ID from any device, before checking it's genuine. Combined with Biometric Verification, it's a seamless way to anchor an account to the real identity of a customer. India	Paid
Superpass.io	<a href="https://surepass.io/passport-id-verification-api/">https://surepass.io/passport-id-verification-api/</a>	Passport, Photo ID and Driver License Verification in India	Paid

Business/Entity search			
Name	Link	Description	Price
Open corporates	<a href="https://api.opencorporates.com">https://api.opencorporates.com</a>	Companies information	Paid, price upon request
Linkedin company search API	<a href="https://docs.microsoft.com/en-us/linkedin/marketing/integrations/community-management/organizations/company-search?context=linkedin%2Fcompliance%2Fcontext&amp;tabs=http">https://docs.microsoft.com/en-us/linkedin/marketing/integrations/community-management/organizations/company-search?context=linkedin%2Fcompliance%2Fcontext&amp;tabs=http</a>	Find companies using keywords, industry, location, and other criteria	FREE
Mattermark	<a href="https://rapidapi.com/raygorodskij/api/Mattermark/">https://rapidapi.com/raygorodskij/api/Mattermark/</a>	Get companies and investor information	free 14-day trial, from \$49 per month

In addition to the Netlas API, which can be used to collect domain or IP address data, there are hundreds of other different APIs that are useful for OSINT.

They can help you collect information about emails, phone numbers, physical addresses, companies, cryptocurrency wallets, flights, and much more. You can find a list of them in the **APIs for OSINT** repository on Github (<https://github.com/cipher387/API-s-for-OSINT>).

Easily generate `curl` command lines to test your new shining API

GET ▾

URL  
`https://github.com/cipher387/python-for-OSINT-21-days/raw/main/Python%20for`

Body

Header key  
Authorization Header value REMOVE

ADD CUSTOM HEADER

JSON Content-Type

Accept self-signed certs

Verbose

here you are!

```
curl -XGET 'https://github.com/cipher387/python-for-OSINT-21-days/raw/main/Python%20for%20OSINT.%2021%20day%20course%20for%20
```

For writing complex queries with header modification, you can use a special online service to generate commands for the **Curl** utility - Curl builder (<https://curlbuilder.com/>).

**Curl** can also be used to view information about web pages. Use **-I** flag to send HEAD request and retrieve only the headers of the response (without the body):

```
gitpod /workspace/Linux-for-OSINT-21-day/Day_5 (main) $ curl -I https://sector035.nl
HTTP/2 200
date: Mon, 25 Dec 2023 22:06:23 GMT
content-type: text/html; charset=utf-8
pragma: no-cache
cache-control: max-age=604800
expires: Mon, 01 Jan 2024 22:06:23 GMT
set-cookie: grav-site-bacle5f=m3pv75gil6n5h1vsvnq45geb16; expires=Mon, 25-Dec-2023 22:36:23 GMT; Max-Age=1800; path=/; domain=sector035.nl; secure; HttpOnly; Same Site=Lax
last-modified: Tue, 11 Jul 2023 05:46:26 GMT
vary: User-Agent
cf-cache-status: DYNAMIC
report-to: {"endpoints":[{"url":"https://a.nel.cloudflare.com/report/v37s=8DexuASPAIFD90B%2B73IWPi%2B6YwPz0yUiGXjjTRHEhKu6v5y80clG9a59SFkzKTQWz%2B%2F1othTtjUmTCZY3xn3GmZkWaRVexXr2FVCijehENYU0f0y1vAgltpeuTfA%3D"}],"group":"cf-nel","max_age":604800}
nel: {"success_fraction":0,"report_to":"cf-nel","max_age":604800}
server: cloudflare
cf-ray: 83b46413db9822ac-CDG
alt-svc: h3=":443"; ma=86400
gitpod /workspace/Linux-for-OSINT-21-day/Day_5 (main) $
```

## `curl -I https://sector035.nl`

This can be used, for example, to safely open short links:

```
gitpod /workspace/linux-for-OSINT-21-day/Day_5 (main) $ curl -sIL https://rb.gy/899j50
HTTP/2 301
date: Mon, 25 Dec 2023 22:20:22 GMT
location: https://cybdetective.com/
cache-control: no-cache, no-store
expires: -1
engine: Rebrandly.redirect, version 2.1
strict-transport-security: max-age=15552000

HTTP/2 200
accept-ranges: bytes
content-type: text/html
date: Mon, 25 Dec 2023 22:20:22 GMT
etag: "5ff7fe41385da1:0"
last-modified: Sun, 22 Oct 2023 22:36:54 GMT
server: Microsoft-IIS/10.0
x-powered-by: ASP.NET
content-length: 9966

gitpod /workspace/linux-for-OSINT-21-day/Day_5 (main) $ curl -sIL https://rb.gy/899j50 | grep ^location
location: https://cybdetective.com/
gitpod /workspace/linux-for-OSINT-21-day/Day_5 (main) $
```

## `curl -sIL https://rb.gy/899j50 | grep ^location`

Note that here we are combining **Curl** with the **Grep** command, using the `|` symbol to cut out the string that starts with "location" from the result. Tomorrow we will talk more about this wonderful utility.

# Day 6. Search in files. Grep

What can you do next with the downloaded files? Lots of different things. But the most important thing an OSINT specialist should be able to do is automatically search and extract different data.

Grep (<https://www.gnu.org/software/grep/>) is one of the oldest UNIX utilities, released in November 1973 (50 years in 2023!). It allows searching for matches to certain text patterns (regular expressions) in files.

The **Grep** utility does not require installation. It is available in Linux by default.

```
● gitpod /workspace/Linux-for-OSINT-21-day (main) $ grep --help
Usage: grep [OPTION]... PATTERNS [FILE]...
Search for PATTERNS in each FILE.
Example: grep -i 'hello world' menu.h main.c
PATTERNS can contain multiple patterns separated by newlines.

Pattern selection and interpretation:
-E, --extended-regexp    PATTERNS are extended regular expressions
-F, --fixed-strings      PATTERNS are strings
-G, --basic-regexp       PATTERNS are basic regular expressions
-P, --perl-regexp        PATTERNS are Perl regular expressions
-e, --regexp=PATTERNS   use PATTERNS for matching
-f, --file=FILE          take PATTERNS from FILE
-i, --ignore-case        ignore case distinctions in patterns and data
    --no-ignore-case    do not ignore case distinctions (default)
-w, --word-regexp        match only whole words
-x, --line-regexp        match only whole lines
-z, --null-data          a data line ends in 0 byte, not newline

Miscellaneous:
-s, --no-messages       suppress error messages
-v, --invert-match       select non-matching lines
-V, --version            display version information and exit
    --help              display this help text and exit

Output control:
-m, --max-count=NUM     stop after NUM selected lines
-b, --byte-offset       print the byte offset with output lines
-n, --line-number       print line number with output lines
    --line-buffered     flush output on every line
-H, --with-filename     print file name with output lines
-h, --no-filename       suppress the file name prefix on output
    --label=LABEL       use LABEL as the standard input file name prefix
-o, --only-matching     show only nonempty parts of lines that match
-q, --quiet, --silent   suppress all normal output
    --binary-files=TYPE assume that binary files are TYPE;
```

Run the help command to appreciate just how many options this seemingly simple utility has.

Let's try to find lines in the text file that include "gmail". Use -F flag for search strings (it's optional, you can miss this flag):

```
example.txt U X
Day_6 > example.txt
1 First string
2 Second string
3 johnsmith@gmail.com
4 103.21.244.25
5 34xp4vRoCGJym3xR7yCVPFHoCNxv4Twseo
6 example@gmail.com
7 bc1qazcm763858nkj2dj986etajv6wquslv8uxwcz
8 23.04.1991
9 173.245.49.202

PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
● gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_6
● gitpod /workspace/linux-for-OSINT-21-day/Day_6 (main) $ grep gmail example.txt
johnsmith@gmail.com
example@gmail.com
● gitpod /workspace/linux-for-OSINT-21-day/Day_6 (main) $ grep '[0-9]' example.txt
103.21.244.25
34xp4vRoCGJym3xR7yCVPFHoCNxv4Twseo
bc1qazcm763858nkj2dj986etajv6wquslv8uxwcz
23.04.1991
173.245.49.202
○ gitpod /workspace/linux-for-OSINT-21-day/Day_6 (main) $
```

**grep -F gmail Day\_6/example.txt**

By default, the **Grep** utility is case sensitive. To disable this option (search for both "gmail" and "GMAIL", "Gmail") add the -i flag.

**grep -F -i gmail Day\_6/example.txt**

Now let's try to find the lines that contain at least one digit (use simple regular expression [0-9]):

**grep '[0-9]' example.txt**

```
example.txt U x  emails.txt U  commands.txt U
Day_6 > example.txt
1 First string
2 Second string
3 johnsmith@gmail.com
4 103.21.244.25
5 34xp4vRoCGJym3xR7yCVPFHoCNxv4Twseo
6 example@gmail.com
7 bc1qazcm763858nkj2dj986etajv6wquslv8uxwcz
8 23.04.1991
9 173.245.49.202

PROBLEMS Focus folder in explorer (cmd + click) OLE TERMINAL
• gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_6
• gitpod /workspace/linux-for-OSINT-21-day/Day_6 (main) $ grep -n ".txt" example.txt
• gitpod /workspace/linux-for-OSINT-21-day/Day_6 (main) $ grep -n "gmail" example.txt
3:johnsmith@gmail.com
6:example@gmail.com
• gitpod /workspace/linux-for-OSINT-21-day/Day_6 (main) $
```

If you want to add line number display to the results, use the -n flag:

```
grep -n "gmail" example.txt
```

```
example.txt U  emails.txt U x  commands.txt U
Day_6 > emails.txt
1 sector035@mail.kz
2 cybdetective@mail.kz
3 jakecreps@mail.kz

PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
• gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_6
• gitpod /workspace/linux-for-OSINT-21-day/Day_6 (main) $ grep -E -o -r "[A-Za-z0-9][A-Za-z0-9_%+~]+@[A-Za-z0-9][A-Za-z0-9.-]+\.[A-Za-z]{2,6}"
example.txt:johnsmith@gmail.com
example.txt:example@gmail.com
emails.txt:sector035@mail.kz
emails.txt:cybdetective@mail.kz
emails.txt:jakecreps@mail.kz
• gitpod /workspace/linux-for-OSINT-21-day/Day_6 (main) $
```

Now the task is a little more difficult. Let's find the strings that contain emails:

```
grep -E -o -r "[A-Za-z0-9][A-Za-z0-9._%+~]+@[A-Za-z0-9][A-Za-z0-9.-]+\.[A-Za-z]{2,6}"
```

We used a few other useful flags in this command:

- E** - use extended regular expressions syntax.
- o** - show only parts of lines that match (NOT all lines).
- r** - search for all files in the current directory, as well as for all files in subdirectories.

The screenshot shows a website titled "Useful Regex Patterns" with a dark theme. It features a grid of 12 cards, each containing a specific regex pattern and a "Test string" input field. The patterns are for: Date in format dd/mm/yyyy, Time in 24-hour format, Date and time in ISO-8601 format, HTML tags, Username, Hex Color Value, URL Slug, Email, SRC of image tag, URL, IPv4 Address, and IPv6 Address. The website also includes a "Post" button, a "Star" button, and a sidebar with a link to "The DNS course for developers" and promotional text for a pre-sale offer.

Patterns to search with **Grep** are called regular expressions. With their help you can search not only numbers and emails, but also URLs, IP addresses, crypto currency wallets, usernames, ZIP codes, code comments and much more.

We will not go into their syntax in this course. For most OSINT purposes, you will just need to copy a regular expressions from Stackoverflow or some online library:

RegexLib (<https://regexlib.com/>)

RegexHub (<https://projects.lukehaas.me/regexhub/>)

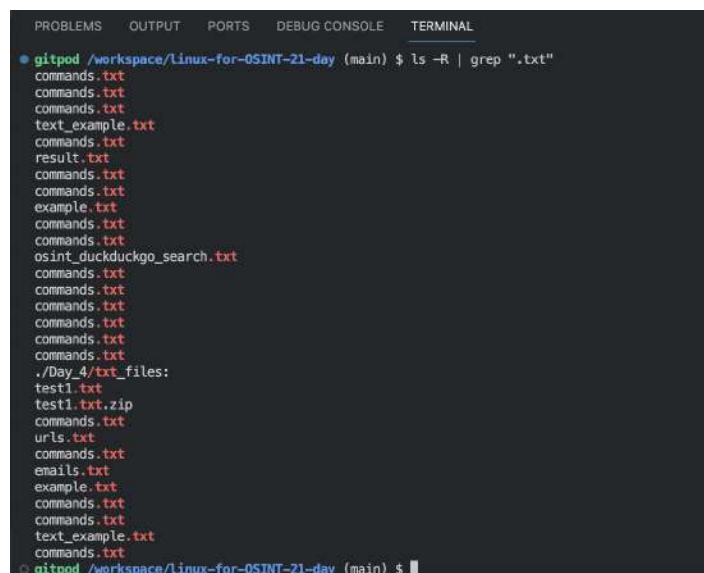
And, of course, you can always ask ChatGPT, Perplexity or Phind to compose a regular expression for you (we'll talk more about helper tools on **Day 20**).

If you are interested in this topic, my Medium blog has a very detailed and long article about it: **How regular expressions can be useful in OSINT. Theory and some practice using Google Sheets** (<https://medium.com/osint-ambition/this-article-consists-of-three-short-parts-31d31efabd5>).

You can also use regular expressions when working with other Linux utilities: **find**, **sed**, **awk**, **tr**, **vi** and others.

If you want to search for files strictly with a certain extension or for files whose name contains a certain string, replace part of the file name with asterisks:

```
grep -n "gmail" *.txt
```

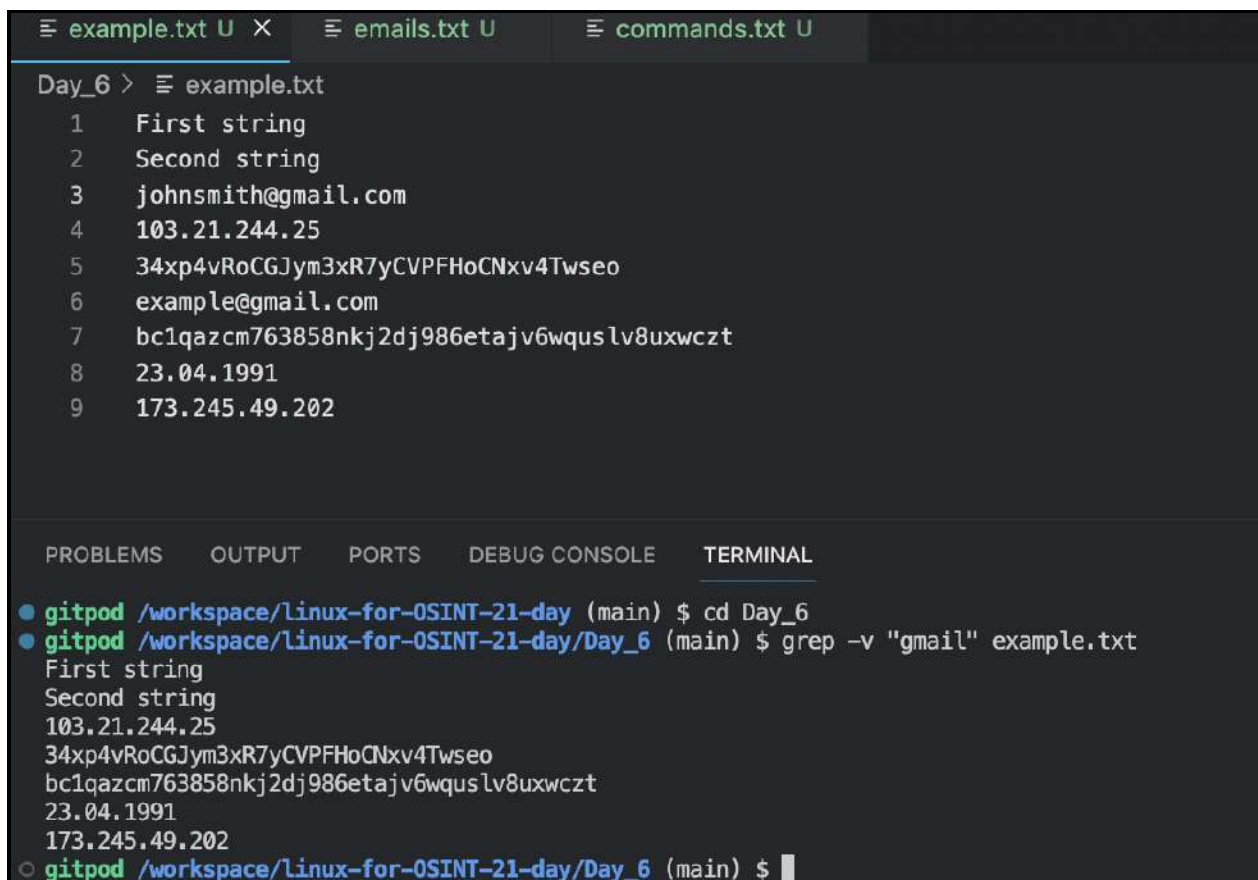


```
gitpod /workspace/linux-for-OSINT-21-day (main) $ ls -R | grep "*.txt"
commands.txt
commands.txt
commands.txt
text_example.txt
commands.txt
result.txt
commands.txt
commands.txt
example.txt
commands.txt
commands.txt
osint_duckduckgo_search.txt
commands.txt
commands.txt
commands.txt
commands.txt
commands.txt
./Day_4/txt_files:
test1.txt
test1.txt.zip
commands.txt
urls.txt
commands.txt
email.txt
example.txt
commands.txt
commands.txt
text_example.txt
commands.txt
gitpod /workspace/linux-for-OSINT-21-day (main) $
```

The real power of **Grep** comes in combination with other commands. For example, you can use it to filter the output of the **ls** command:

```
ls -R | grep ".txt"
```

Note that we used the **ls** command with the **-R** flag to display the list of files in the subdirectories too.



```
Day_6 > ls -R | grep ".txt"
1 First string
2 Second string
3 johnsmith@gmail.com
4 103.21.244.25
5 34xp4vRoCGJym3xR7yCVPFHoCNxv4Twseo
6 example@gmail.com
7 bc1qazcm763858nkj2dj986etajv6wquslv8uxwcz
8 23.04.1991
9 173.245.49.202

gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_6
gitpod /workspace/linux-for-OSINT-21-day/Day_6 (main) $ grep -v "gmail" example.txt
First string
Second string
103.21.244.25
34xp4vRoCGJym3xR7yCVPFHoCNxv4Twseo
bc1qazcm763858nkj2dj986etajv6wquslv8uxwcz
23.04.1991
173.245.49.202
gitpod /workspace/linux-for-OSINT-21-day/Day_6 (main) $
```

Also, **grep** allows you to search for strings that do NOT contain the word you are looking for or match a regular expression. Use the **-v** flag:

```
grep -v "gmail" example.txt
```

```
example.txt U X  emails.txt U  commands.txt U
Day_6 > example.txt
1  First string
2  Second string
3  johnsmith@gmail.com
4  103.21.244.25
5  34xp4vRoCGJym3xR7yCVPFHoCNxv4Twseo
6  example@gmail.com
7  bc1qazcm763858nkj2dj986etajv6wquslv8uxwcz
8  23.04.1991
9  173.245.49.202
10 johnsmith

PROBLEMS  OUTPUT  PORTS  DEBUG CONSOLE  TERMINAL
● gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_6
● gitpod /workspace/linux-for-OSINT-21-day/Day_6 (main) $ grep "johnsmith" example.txt | grep -v "gmail"
johnsmith
○ gitpod /workspace/linux-for-OSINT-21-day/Day_6 (main) $
```

One last, very important note. You can run **Grep** multiple times at the one line. Let's try to find strings that contain "johnsmith" but do not contain "gmail":

```
grep "johnsmith" example.txt | grep -v "gmail"
```

Again, notice how we combine two commands on the same line using the `|`. This is a simple but very important skill that will allow you to get the most out of Linux.

## Alternatives

Name	Description	Link.
RZgrep	"Grep utility that searches through zip,jar,ear,tgz,bz2 in any form of nesting"	<a href="https://github.com/MoserMicha">https://github.com/MoserMicha</a>
XLSgrep	"CLI tool to search text in XLSX, XLS and ODS files. It works similarly to Unix/GNU Linux grep"	<a href="https://github.com/zazuun/xlsx">https://github.com/zazuun/xlsx</a>
Hgrep	"Grep" for searching/replacement in html code	<a href="https://github.com/TUVIMEN/hg">https://github.com/TUVIMEN/hg</a>
BINgrep	Grep for binaries files	<a href="https://github.com/m4b/bingrep">https://github.com/m4b/bingrep</a>
Cgrep	Grep for source codes in C/C++	<a href="https://github.com/awgn/cgrep">https://github.com/awgn/cgrep</a>
Ngrep	PCAP-based tool that allows you to specify an extended regular or hexadecimal expression to match against data payloads of packets. It understands many kinds of protocols, including IPv4/6, TCP, UDP, ICMPv4/6, IGMP and Raw	<a href="https://github.com/jpr5/ngrep">https://github.com/jpr5/ngrep</a>
PHPgrep	Grep for PHP source code	<a href="https://github.com/quaslyte/ph">https://github.com/quaslyte/ph</a>
UCG (Universal Code Grep)	Extremely fast grep-like tool specialized for searching large bodies of source code	<a href="https://github.com/qvansickle/u">https://github.com/qvansickle/u</a>
MP4grep	CLI for transcribing and searching audio/video files	<a href="https://github.com/o-oconnell/n">https://github.com/o-oconnell/n</a>
minigrep	A minimalistic regex search and print out tool implementation as per Ch.12 of The Rust Programming Language	<a href="https://github.com/paul-michell">https://github.com/paul-michell</a>
Hackers Grep	Utility to search for strings in PE executables including imports, exports, and debug symbols	<a href="https://github.com/codyperce/">https://github.com/codyperce/</a>
JVgrep	Grep for Japanese vimmer. You can find text from files that written in another Japanese encodings	<a href="https://github.com/mattn/jvgrep">https://github.com/mattn/jvgrep</a>
LiveGrep	Online grep search in Github repositories	<a href="https://livegrep.com/">https://livegrep.com/</a>
DNgrep	Graphical grep tool for Windows	<a href="https://github.com/dnGrep/dnG">https://github.com/dnGrep/dnG</a>
Greplace.vim	Global search and replace for VI	<a href="https://github.com/skwp/grepla">https://github.com/skwp/grepla</a>
SearchInProject_SublimeText	Use ag, ack, grep and git grep directly from Sublime Text 2 and 3	<a href="https://github.com/leonid-shevtsov/SearchInProject_Subli">https://github.com/leonid-shevtsov/SearchInProject_Subli</a>
ACK3	grep-like Perl search tool optimized for source code	<a href="https://github.com/beyondgrep/">https://github.com/beyondgrep/</a>
The Silver Searcher	Extreme grep-like C+Perl search tool for source code	<a href="https://github.com/ggreer/the_s">https://github.com/ggreer/the_s</a>
Sift	GO fast and powerfull alternative for grep	<a href="https://github.com/svent/sift">https://github.com/svent/sift</a>

The **Grep** utility is designed to search through text files. But it has many analogs that work with other formats: XLSX, ZIP, JAR, MP4, PDF etc. You can find links to them in the Awesome Grep (<https://github.com/cipher387/awesome-grep>) repository.

# Day 7. Sed and Awk

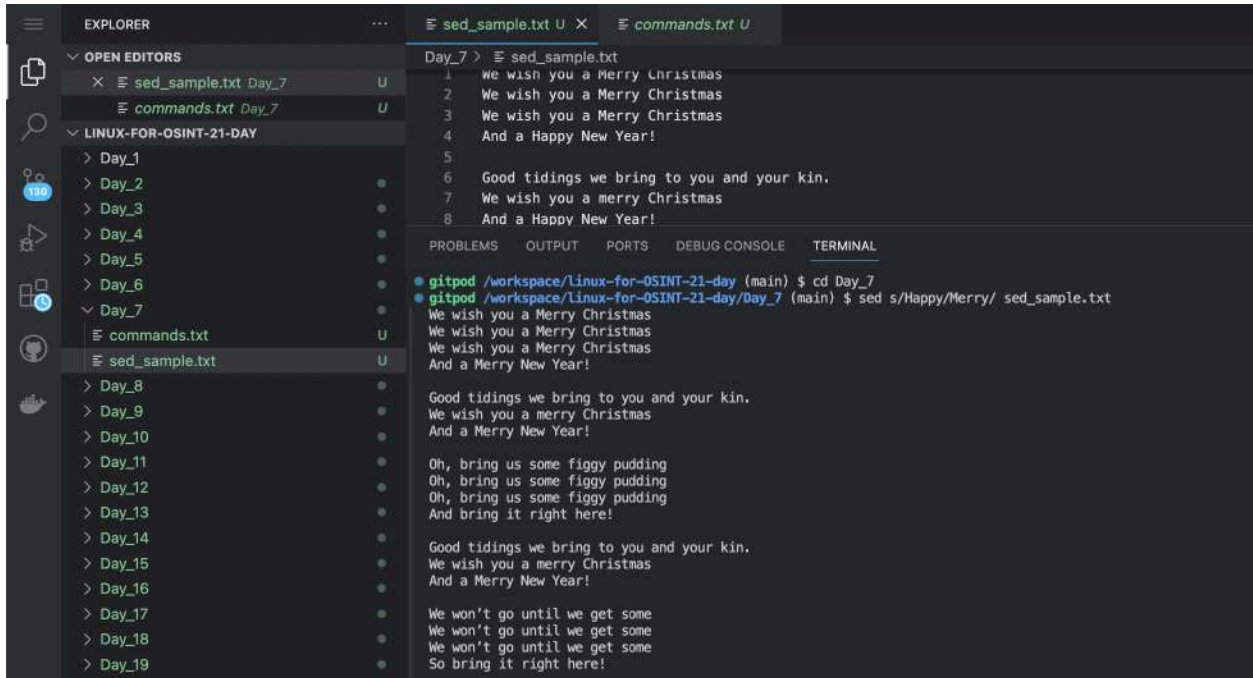
Today is one of the longest lessons with the most examples. Some people will find it the most boring in the whole course. But believe me, from the point of view of learning to save time, it is the most useful day. It is even more useful than the day in which we will talk about AI assistants.

Sed (<https://www.gnu.org/software/sed/>) is a non-interactive command-line text editor (it would be more accurate to say stream editor). In other words, it allows you to make changes to a text file without opening it.

```
Ⓢ gitpod /workspace/Linux-for-OSINT-21-day (main) $ sed -h
sed: invalid option -- 'h'
Usage: sed [OPTION]... {script-only-if-no-other-script} [input-file]...

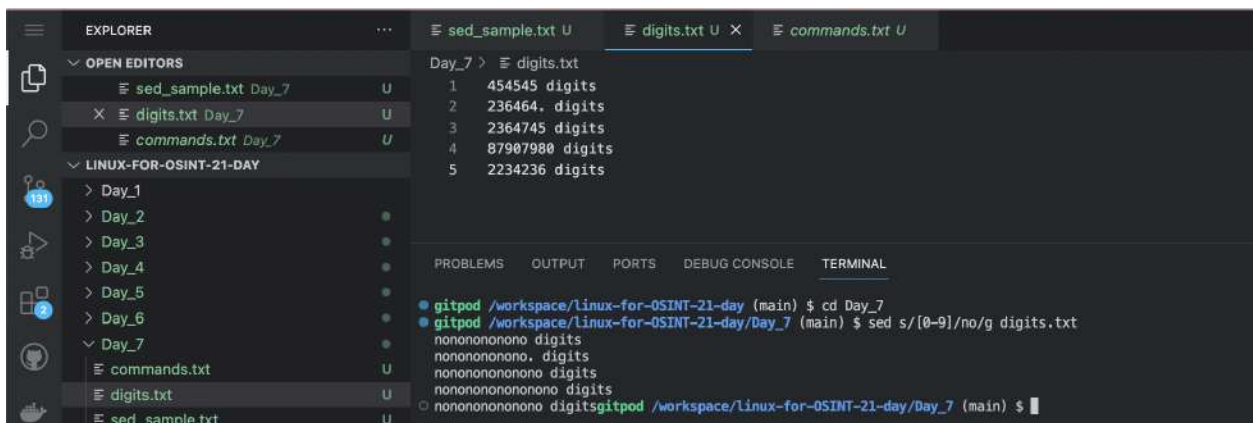
-n, --quiet, --silent
    suppress automatic printing of pattern space
    --debug
    annotate program execution
-e script, --expression=script
    add the script to the commands to be executed
-f script-file, --file=script-file
    add the contents of script-file to the commands to be executed
--follow-symlinks
    follow symlinks when processing in place
-i[SUFFIX], --in-place[=SUFFIX]
    edit files in place (makes backup if SUFFIX supplied)
-l N, --line-length=N
    specify the desired line-wrap length for the 'l' command
--posix
    disable all GNU extensions.
-E, -r, --regexp-extended
    use extended regular expressions in the script
    (for portability use POSIX -E).
-s, --separate
    consider files as separate rather than as a single,
    continuous long stream.
    --sandbox
    operate in sandbox mode (disable e/r/w commands).
-u, --unbuffered
    load minimal amounts of data from the input files and flush
    the output buffers more often
-z, --null-data
    separate lines by NUL characters
--help
    display this help and exit
--version
    output version information and exit
```

First, let's just try to replace one piece of text with another in the file:



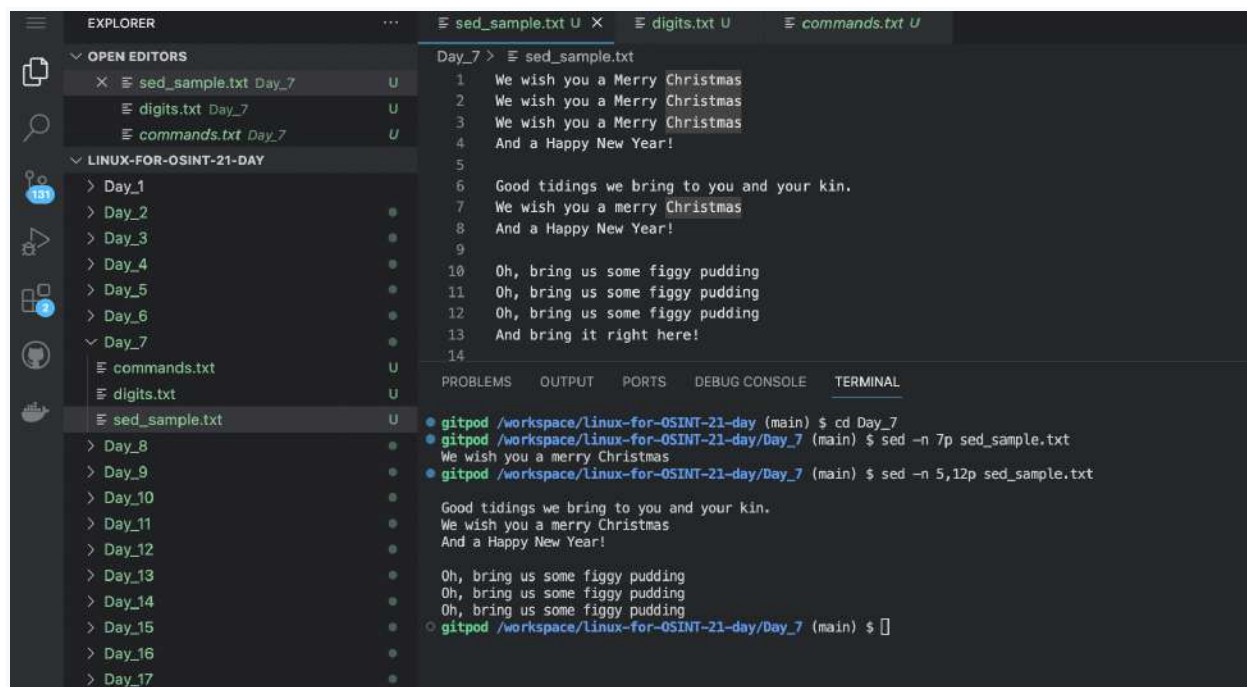
**cd Day\_7**  
**sed s/Happy/Merry/ sed\_sample.txt**

Now let's try to replace the text in the file using a regular expression (replace each digit to word "no"):



**sed s/[0-9]/no/g digits.txt**

The `-n` flag can be used to extract a single line or a range of lines from a file:



```
Day_7 > sed_sample.txt
1 We wish you a Merry Christmas
2 We wish you a Merry Christmas
3 We wish you a Merry Christmas
4 And a Happy New Year!
5
6 Good tidings we bring to you and your kin.
7 We wish you a merry Christmas
8 And a Happy New Year!
9
10 Oh, bring us some figgy pudding
11 Oh, bring us some figgy pudding
12 Oh, bring us some figgy pudding
13 And bring it right here!
14

gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_7
gitpod /workspace/linux-for-OSINT-21-day/Day_7 (main) $ sed -n 7p sed_sample.txt
We wish you a merry Christmas
gitpod /workspace/linux-for-OSINT-21-day/Day_7 (main) $ sed -n 5,12p sed_sample.txt
Good tidings we bring to you and your kin.
We wish you a merry Christmas
And a Happy New Year!
Oh, bring us some figgy pudding
Oh, bring us some figgy pudding
Oh, bring us some figgy pudding
gitpod /workspace/linux-for-OSINT-21-day/Day_7 (main) $
```

**sed -n 7p sed\_sample.txt**

**sed -n 5,12p sed\_sample.txt**

For a file of a few dozen lines, this command seems pointless. Its true power comes out when you have, for example, a CSV file with tens of millions of lines that Excel refuses to open and you need to see what is on lines number 108100, 999588, 10009898 to make sure the data is in the right format.

It is also possible to combine this to function and replace a word or a regular expression in a given range of strings:

```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
● And a Happy New Year!gitpod /workspace/linux-for-OSINT-21-day/Day_7 (main) $ sed '1,30 s/Happy/Merry/g' sed_sample.txt
We wish you a Merry Christmas
We wish you a Merry Christmas
We wish you a Merry Christmas
And a Merry New Year!

Good tidings we bring to you and your kin.
We wish you a merry Christmas
And a Merry New Year!

Oh, bring us some figgy pudding
Oh, bring us some figgy pudding
Oh, bring us some figgy pudding
And bring it right here!

Good tidings we bring to you and your kin.
We wish you a merry Christmas
And a Merry New Year!

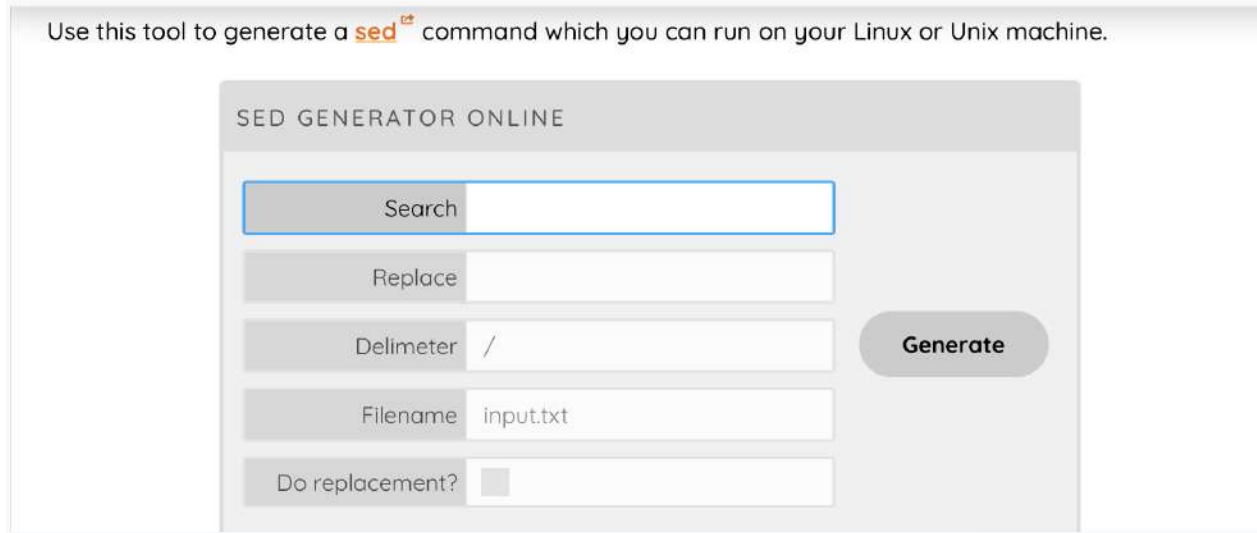
We won't go until we get some
We won't go until we get some
We won't go until we get some
So bring it right here!

Good tidings we bring to you and your kin.
We wish you a merry Christmas
And a Merry New Year!

We all like our figgy pudding
```

**sed '1,30 s/Happy/Merry/g' sed\_sample.txt**

More examples can find in “Useful sed” Github repository (<https://github.com/adrianlarion/useful-sed>) .



Do not try to memorize the syntactic rules for composing **Sed** commands. You can always find an example on StackOverflow, ask some AI code assistant for help, or use a special online service like **Sed command generator** (<https://www.jamiebalfour.scot/devkit/sed/>) .

## AWK

Awk (<https://www.gnu.org/software/gawk/manual/gawk.html>) is a scripting language for text processing. Interesting fact. Awk is not a technical acronym, but simply the initials of the three developers of this language: Alfred V. **A**ho, Peter J. **W**einberger и Brian W. **K**ernighan.

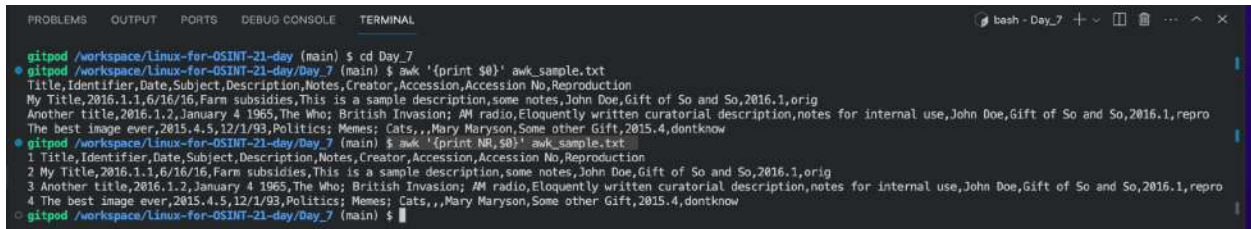
**Awk**, like **Sed** does not require installation as it is included in Linux by default.

First, let's try to display the entire text of the file. It is easier to do this with **Cat**, but this example is necessary to better understand awk syntax:

```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
bash - Day_7 + - [ ] ... ^ x
gitpod /workspace/Linux-for-OSINT-21-day (main) $ cd Day_7
gitpod /workspace/Linux-for-OSINT-21-day/Day_7 (main) $ awk '{print $0}' awk_sample.txt
Title,Identifier,Date,Subject,Description,Notes,Creator,Accession,Accession No,Reproduction
My Title,2016.1.1,6/16/16,Farm subsidies,This is a sample description,some notes,John Doe,Gift of So and So,2016.1,orig
Another title,2016.1.2,January 4 1965,The Who; British Invasion; AM radio,Eloquently written curatorial description,notes for internal use,John Doe,Gift of So and So,2016.1,repr
The best image ever,2015.4.5,12/1/93,Politics; Memes; Cats,,,Mary Maryson,Some other Gift,2015.4,dontknow
gitpod /workspace/Linux-for-OSINT-21-day/Day_7 (main) $
```

## awk '{print \$0}' awk\_sample.txt

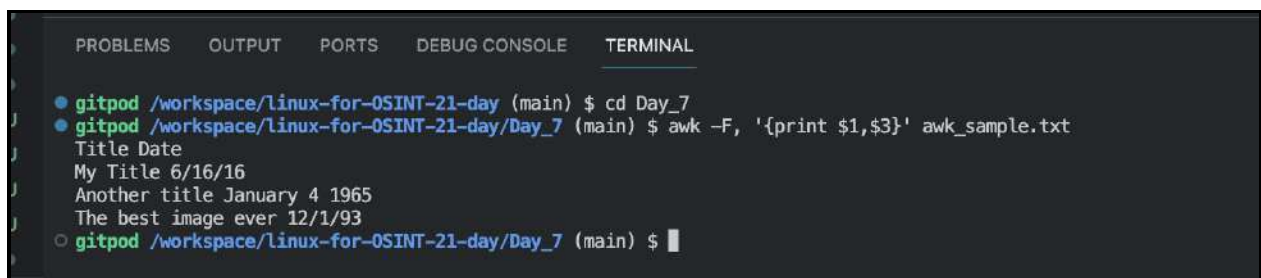
Now let's do the same thing, but with line numbering:



```
gitpod /workspace/linux-for-05INT-21-day (main) $ cd Day_7
gitpod /workspace/linux-for-05INT-21-day/Day_7 (main) $ awk '{print $0}' awk_sample.txt
Title,Identifier,Date,Subject,Description,Notes,Creator,Accession,Accession No,Reproduction
My Title,2016.1.1,6/16/16,Farm subsidies,This is a sample description,some notes,John Doe,Gift of So and So,2016.1,orig
Another title,2016.1.2,January 4 1965,The Who; British Invasion; AM radio,Eloquently written curatorial description,notes for internal use,John Doe,Gift of So and So,2016.1,repo
The best image ever,2015.4.5,12/1/93,Politics; Memes; Cats,,Mary Maryson,Some other Gift,2015.4,dontknow
gitpod /workspace/linux-for-05INT-21-day/Day_7 (main) $ awk '{print NR,$0}' awk_sample.txt
1 Title,Identifier,Date,Subject,Description,Notes,Creator,Accession,Accession No,Reproduction
2 My Title,2016.1.1,6/16/16,Farm subsidies,This is a sample description,some notes,John Doe,Gift of So and So,2016.1,orig
3 Another title,2016.1.2,January 4 1965,The Who; British Invasion; AM radio,Eloquently written curatorial description,notes for internal use,John Doe,Gift of So and So,2016.1,repo
4 The best image ever,2015.4.5,12/1/93,Politics; Memes; Cats,,Mary Maryson,Some other Gift,2015.4,dontknow
gitpod /workspace/linux-for-05INT-21-day/Day_7 (main) $
```

## awk '{print NR,\$0}' awk\_sample.txt

Let's display only the first and third columns of the document (using flag **-F** to indicate that the separator is a comma):



```
gitpod /workspace/linux-for-05INT-21-day (main) $ cd Day_7
gitpod /workspace/linux-for-05INT-21-day/Day_7 (main) $ awk -F, '{print $1,$3}' awk_sample.txt
Title Date
My Title 6/16/16
Another title January 4 1965
The best image ever 12/1/93
gitpod /workspace/linux-for-05INT-21-day/Day_7 (main) $
```

## awk -F, '{print \$1,\$3}' awk\_sample.txt

Now output only the strings that start with a certain character:

```

example.csv U  awk_sample.txt U X
Day_7 > awk_sample.txt
1 Title,Identifier,Date,Subject,Description,Notes,Creator,Accession,Accession No,Reproduction
2 My Title,2016.1.1,6/16/16,Farm subsidies,This is a sample description,some notes,John Doe,Gift of So and So,2016.1,orig
3 Another title,2016.1.2,January 4 1965,The Who; British Invasion; AM radio,Eloquently written curatorial description,notes for internal use,John Doe,Gift of So and So,2016.1,orig
4 The best image ever,2015.4.5,12/1/93,Politics; Memes; Cats,,,Mary Maryson,Some other Gift,2015.4,dontknow
5

PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
gitpod /workspace/Linux-for-OSINT-21-day (main) $ cd Day_7
gitpod /workspace/Linux-for-OSINT-21-day/Day_7 (main) $ awk '/^M/' awk_sample.txt
My Title,2016.1.1,6/16/16,Farm subsidies,This is a sample description,some notes,John Doe,Gift of So and So,2016.1,orig
gitpod /workspace/Linux-for-OSINT-21-day/Day_7 (main) $

```

### awk '/^O/' awk\_sample.txt

Conversely, strings that end with a specific character:

```

example.csv U  awk_sample.txt U X
Day_7 > awk_sample.txt
1 Title,Identifier,Date,Subject,Description,Notes,Creator,Accession,Accession No,Reproduction
2 My Title,2016.1.1,6/16/16,Farm subsidies,This is a sample description,some notes,John Doe,Gift of So and So,2016.1,orig
3 Another title,2016.1.2,January 4 1965,The Who; British Invasion; AM radio,Eloquently written curatorial description,notes for internal use,John Doe,Gift of So and So,2016.1,orig
4 The best image ever,2015.4.5,12/1/93,Politics; Memes; Cats,,,Mary Maryson,Some other Gift,2015.4,dontknow
5

PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
gitpod /workspace/Linux-for-OSINT-21-day (main) $ cd Day_7
gitpod /workspace/Linux-for-OSINT-21-day/Day_7 (main) $ awk '/w$/' awk_sample.txt
The best image ever,2015.4.5,12/1/93,Politics; Memes; Cats,,,Mary Maryson,Some other Gift,2015.4,dontknow
gitpod /workspace/Linux-for-OSINT-21-day/Day_7 (main) $

```

### awk '/w\$/' awk\_sample.txt

And now all the strings that do NOT end with a certain character:

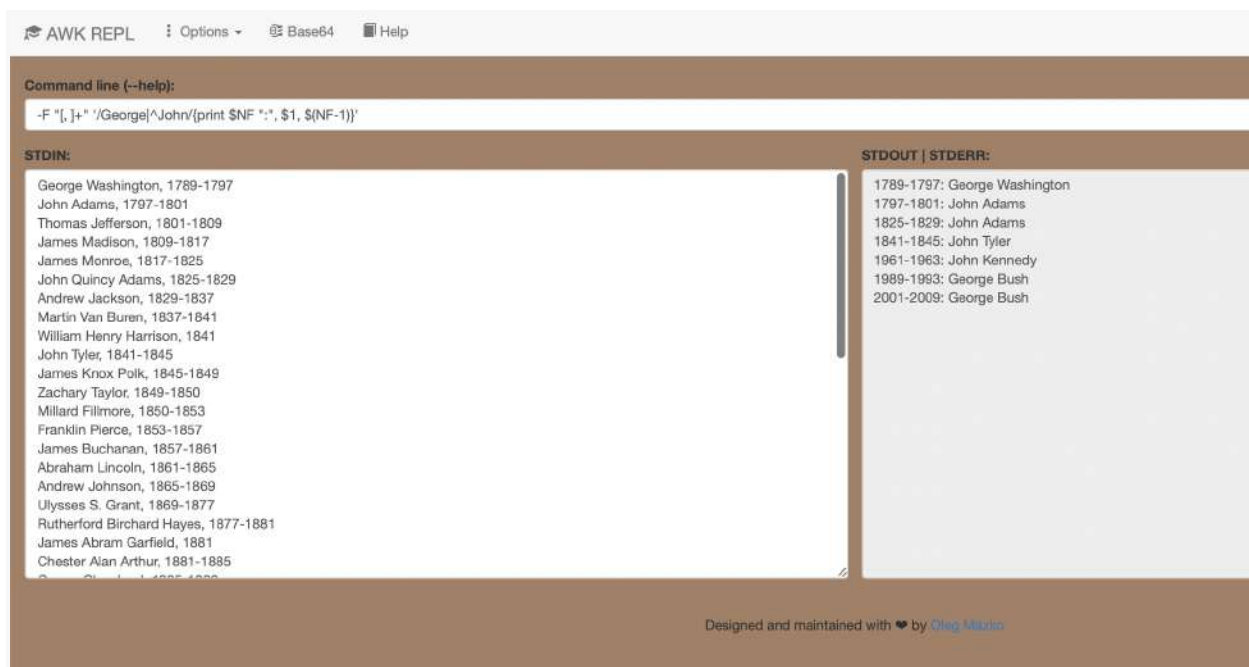
```

PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
gitpod /workspace/Linux-for-OSINT-21-day (main) $ cd Day_7
gitpod /workspace/Linux-for-OSINT-21-day/Day_7 (main) $ awk '/w$/' awk_sample.txt
The best image ever,2015.4.5,12/1/93,Politics; Memes; Cats,,,Mary Maryson,Some other Gift,2015.4,dontknow
gitpod /workspace/Linux-for-OSINT-21-day/Day_7 (main) $ awk '! /w$/' awk_sample.txt
awk: cmd. line:1: ! /w$ /awk_sample.txt
awk: cmd. line:1: ^ syntax error
gitpod /workspace/Linux-for-OSINT-21-day/Day_7 (main) $ awk '! /w$/' awk_sample.txt
Title,Identifier,Date,Subject,Description,Notes,Creator,Accession,Accession No,Reproduction
My Title,2016.1.1,6/16/16,Farm subsidies,This is a sample description,some notes,John Doe,Gift of So and So,2016.1,orig
Another title,2016.1.2,January 4 1965,The Who; British Invasion; AM radio,Eloquently written curatorial description,notes for internal use,John Doe,Gift of So and So,2016.1,repo
gitpod /workspace/Linux-for-OSINT-21-day/Day_7 (main) $

```

### awk '! /w\$/' awk\_sample.txt

More examples can be found in **Simple AWK** Github repository (<https://github.com/adrianlarion/simple-awk>). As with **Sed**, you can ask the AI assistant for help writing commands or use special services.



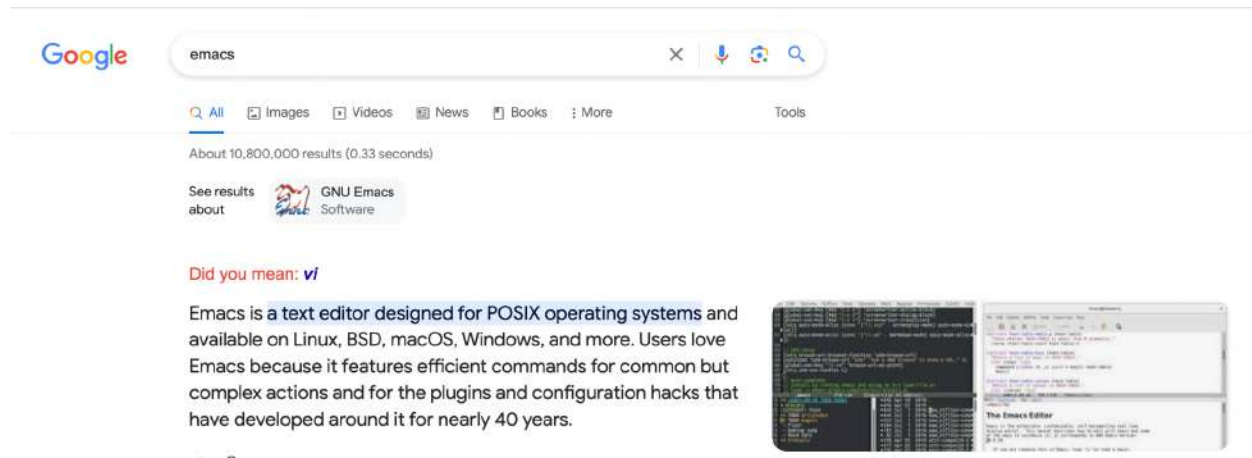
AWK online (<https://awk.js.org>) is an online tool on which it is very convenient to test how various commands are executed on small text fragments.

When reading the additional documentation, you may notice that **Grep**, **Sed** and **Awk** may have common functions and it is not always clear which command is best to use. And more often than not, it doesn't matter.

Yes, 20-30 years ago when computers had modest capabilities it made sense to think deeply about this when working with large files. Today, however, in most situations it is not an important issue at all.

# Day 8. Vim text editor

Interesting fact, if you type "emacs" into Google, the search engine will ask you, "Maybe you mean vi?" (vim is an improved version of the vi text editor, which was introduced first time in 1976). Many developers around the world really love Vim, have been using it for decades and are constantly discovering some amazing new features of this text editor.



Vim (<https://www.vim.org>) - is a universal and highly configurable command line text editor. The first version of it appeared in 1991 and since then it has continued to be popular, despite many competitors in the form of other text editors and IDEs (Integrated Development Environment). Let's run it and see what it's good for.

```
gitpod /workspace/linux-for-OSINT-21-day/Day_14 (main) $ vim --version
VIM - Vi Improved 8.2 (2019 Dec 12, compiled Oct 16 2023 18:15:38)
Included patches: 1-579, 1969, 580-1854, 1857, 1855-1857, 1331, 1858, 1858-1859, 1873, 1860-1969, 1992, 1970-1992, 2010, 1993-3995, 4563, 4646, 4
774, 4895, 4899, 4901, 4919, 213, 1840, 1846-1847
Modified by team+vim@tracker.debian.org
Compiled by team+vim@tracker.debian.org
Huge version without GUI. Features included (+) or not (-):
+acl          +file_in_path    +mouse_urxvt     -tag_any_white
+arabic       +find_in_path    +mouse_xterm     -tcl
+autocmd      +float           +multi_byte      +termguicolors
+autocmdir    +folding         +multi_lang      +terminal
-autoserverna -footer          -mzscheme        +terminfo
+balloon_eva +fork()          +netbeans_intg   +termresponse
+balloon_eva +gettext         +num64           +textobjects
-browse       -hangul_input    +packages        +textprop
++builtin_ter +iconv           +path_extra      +timers
+byte_offset +insert_expand  -perl            +title
+channel      +ipv6           +persistent_und  -toolbar
+cindent      +job            +popupwin        +user_commands
-clientserver +jumplist       +postscript      +vartabs
-clipboard    +keymap         +printer         +vertsplitt
+cmdline_compl +lambda         +profile         +vim9script
+cmdline_hist +langmap        -python          +vminfo
+cmdline_info +libcall        +python3         +virtualedit
+comments     +linebreak      +quickfix        +visual
+conceal      +lispindent     +runtime         +visualextra
+cryptv       +listcmds       +rightleft      +vreplace
+cscope       +localmap       -ruby            +wildignore
+cursorbund   -lua            +scrollbind      +wildmenu
+cursorshape  +menu           +signs           +windows
```

Let's install:

```
sudo apt update
sudo apt install vim
```

```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
gitpod /workspace/linux-for-OSINT-21-day/Day_14 (main) $ sudo apt install vim
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  vim-common vim-runtime
Suggested packages:
  ctags vim-doc vim-scripts
The following packages will be upgraded:
  vim vim-common vim-runtime
3 upgraded, 0 newly installed, 0 to remove and 80 not upgraded.
Need to get 8,650 kB of archives.
After this operation, 7,168 B of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 vim amd64 2:8.2.3995-1ubuntu2.13 [1,734 kB]
Get:2 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 vim-runtime all 2:8.2.3995-1ubuntu2.13 [6,834 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 vim-common all 2:8.2.3995-1ubuntu2.13 [81.5 kB]
Fetched 8,650 kB in 1s (7,913 kB/s)
debconf: delaying package configuration, since apt-utils is not installed
(Reading database ... 35595 files and directories currently installed.)
Preparing to unpack .../vim_2%3a8.2.3995-1ubuntu2.13_amd64.deb ...
Unpacking vim (2:8.2.3995-1ubuntu2.13) over (2:8.2.3995-1ubuntu2.12) ...
Preparing to unpack .../vim-runtime_2%3a8.2.3995-1ubuntu2.13_all.deb ...
Unpacking vim-runtime (2:8.2.3995-1ubuntu2.13) over (2:8.2.3995-1ubuntu2.12) ...
Preparing to unpack .../vim-common_2%3a8.2.3995-1ubuntu2.13_all.deb ...
Unpacking vim-common (2:8.2.3995-1ubuntu2.13) over (2:8.2.3995-1ubuntu2.12) ...
Setting up vim-common (2:8.2.3995-1ubuntu2.13) ...
Setting up vim-runtime (2:8.2.3995-1ubuntu2.13) ...
Setting up vim (2:8.2.3995-1ubuntu2.13) ...
Processing triggers for man-db (2.10.2-1) ...
```

And run:

```
PROBLEMS  OUTPUT  PORTS  DEBUG CONSOLE  TERMINAL
Quod equidem non reprehendo;
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quibus natura iure responderit non esse verum aliunde finem beate vivendi, a se principia rei gerendae peti; Quae enim adhuc protulisti, popularia sunt, ego autem a te elegantiora desidero. Duo Reges: constructio interrete. Tum Lucius: Mihi vero ista valde probata sunt, quod item fratri puto. Bestiarum vero nullum iudicium puto. Nihil enim iam habes, quod ad corpus referas; Deinde prima illa, quae in congressu solemus: Quid tu, inquit, huc? Et homini, qui ceteris animantibus plurimum praestat, praecipue a natura nihil datum esse dicemus?

Iam id ipsum absurdum, maximum malum negligi. Quod ea non occurrence fingunt, vincunt Aristonem; Atqui perspicuum est hominem e corpore animoque constare, cum primae sint animi partes, secundae corporis. Fieri, inquam, Triari, nullo pacto potest, ut non dicas, quid non probes eius, a quo dissentias. Equidem e Cn. An dubium est, quin virtus ita maximam partem optineat in rebus humanis, ut reliqua obruat?

Quis istum dolorem timet?
Summus dolor plures dies manere non potest? Dicet pro me ipsa virtus nec dubitabit isti vestro beato M. Tubulum fuisse, qua illum, cuius is condemnatus est rogatione, P. Quod si ita sit, cur opera philosophiae sit danda nescio.

Ex eorum enim scriptis et institutis cum omnis doctrina liberalis, omnis historia.
Quod si ita est, sequitur id ipsum, quod te velle video, omnes semper beatos esse sapientes. Cum enim fertur quasi torrens oratio, quamvis multa cuiusque modi rapiat, nihil tamen teneas, nihil apprehendas, nusquam orationem rapidam coerceas. Ita redarguitur ipse a sese, convincunturque scripta eius probitate ipsius ac moribus. At quanta conantur! Mundum hunc omnem oppidum esse nostrum! Incenditur eos, qui audiunt, vides. Vide, ne magis, inquam, tuum fuerit, cum re idem tibi, quod mihi, videretur, non nova te rebus nomina imponere. Qui vere falsone, quaerere mittimus-dicitur oculis se privasse; Si ista mala sunt, in quae potest incidere sapiens, sapienter esse non esse ad beate vivendum satis. At vero si ad vitam sensus accesserit, ut appetitum quandam habeat et per se ipsa moveatur, quid facturam putas?

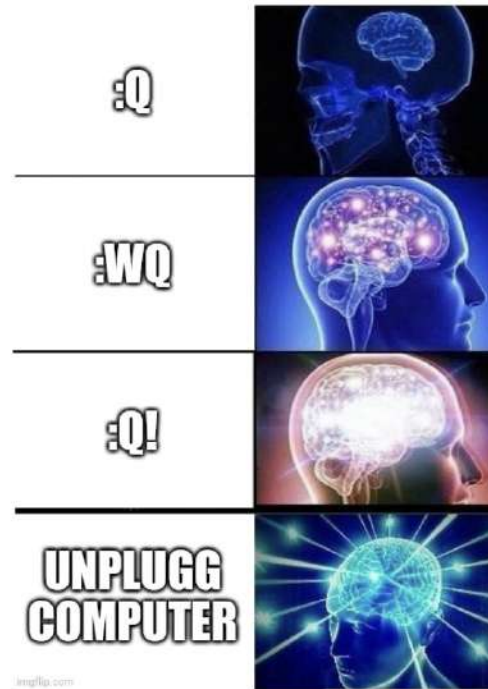
Quem si teneris, non modo meum Ciceronem, sed etiam me ipsum abducas licebit.
Stulti autem malorum memoria torquentur, sapientes bona praeterita grata recordatione renovata delectant.
Esse enim quam vellet iniquus iustus poterat inpune.
Quae autem natura suae primae institutionis oblita est?
Verum tamen cum de rebus grandioribus dicas, ipsae res verba rapiunt;
Hoc est non modo cor non habere, sed ne palatum quidem.
@@@
"text_example.txt" [noeol] 21L, 3541B                                     1,2                               Top
```

**cd Day\_8**

**vim text\_example.txt**

We will do nothing more with this text for now, but go straight to the most important question.

## Exit Vim



This is a very important question that has inspired hundreds of people to create memes (google [vim exit memes](#)). Save these instructions somewhere.

1. Press the **Esc** key to switch to command mode.
2. Type `:q!` and press Enter (or type `:wq!` and press Enter, if you want to save the changes)

In case you forget these instructions remember that you can always just click the right cross and close the terminal (in Gitpod too).

If suddenly, for some reason, the instructions above seem too simple for you, you can try a few dozen other ways to quit vim (<https://github.com/hakluke/how-to-exit-vim>).

## Using Vim

Let's open the file again:

```
vim text_example.txt
```

And let's try to navigate through the text:

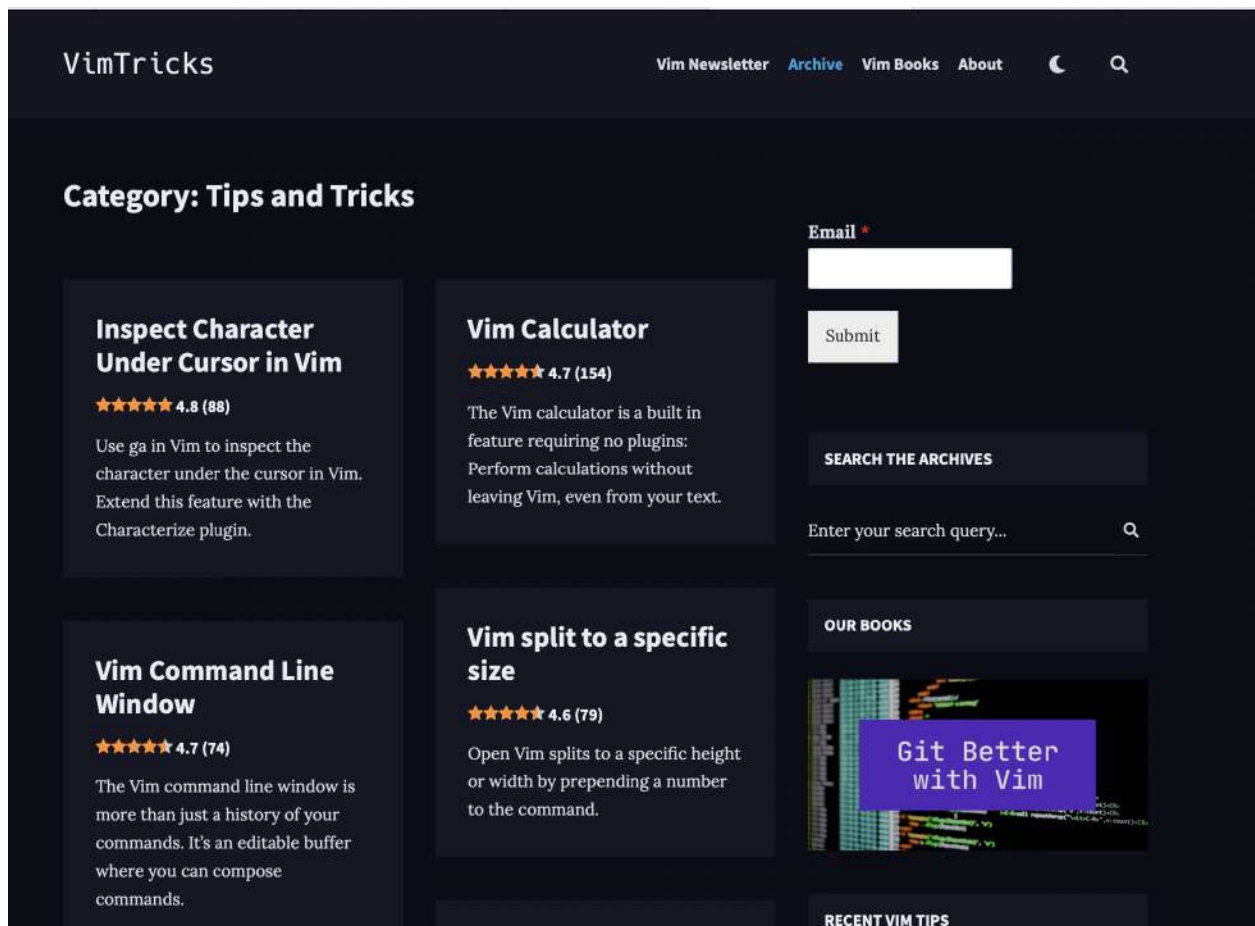
```
Quis istum dolorem timet?  
Quis istum dolorem timet?  
Quis istum dolorem timet?  
Quis istum dolorem timet?  
Quis istum dolorem timet?  
Summus dolor plures dies manere non potest? Dicit pro me ipsa virtus nec dubitabit isti vestro beato M. Tubulum f  
a sit, cur opera philosophiae sit danda nescio.  
  
eptis et institutis cum omnis doctrina liberalis, omnis historia.  
Quod si ita est, sequitur id ipsum, quod te velle video, omnes semper beatos esse sapientes. Cum enim fertur quas  
eneas, nihil apprehendas, nusquam orationem rapidam coerceas. Ita redarguitur ipse a sese, convincunturque script  
omnem oppidum esse nostrum! Incendi igitur eos, qui audiunt, vides. Vide, ne magis, inquam, tuum fuerit, cum re  
Qui-vere falsone, quaerere mittimus-dicitur oculis se privasse; Si ista mala sunt, in quae potest incidere sapie  
vitem sensus accesserit, ut appetitum quendam habeat et per se ipsa moveatur, quid facturam putas?  
  
Quem si teneris, non modo meum Ciceronem, sed etiam me ipsum abducas licebit.  
Stulti autem malorum memoria torquentur, sapientes bona praeterita grata recordatione renovata delectant.  
Esse enim quam vellet iniquus iustus poterat inpune.  
Quae autem natura suae primae institutionis oblita est?  
Verum tamen cum de rebus grandioribus dicas, ipsae res verba rapiunt;  
/omnes
```

- G** - move to the last line.
- gg** - move to the first line
- ^** - move to the first character of current line
- +** - move to the first character of next line
- yyp** - duplicate current line
- dd** - delete current line
- g;** - go back to the last modified location
- x** - delete character or selected
- y** - delete character or selected text
- /** - find text

(If something suddenly doesn't work, press Esc to enter command entry mode).

And the bottom line is that there are over a thousand such quick commands. Yes, over a thousand commands for working with text files. Here's a list of 1,349, but it's probably not complete, since Vim is infinite (<https://pastebin.com/xBRYzAw0>).

If you pick at least a few dozen commands that best suit your goals, you can take your productivity to a whole new level (as well as improve memory and concentration tremendously). This is what hundreds of thousands of **Vim** fans around the world have already done (but I'm just about to do it myself, to be honest).



The screenshot shows the VimTricks website interface. At the top, the site name "VimTricks" is on the left, and navigation links "Vim Newsletter", "Archive", "Vim Books", and "About" are on the right, along with a moon icon and a search icon. Below the navigation, the main heading is "Category: Tips and Tricks".

On the right side, there is an "Email" subscription form with a text input field and a "Submit" button. Below that is a "SEARCH THE ARCHIVES" section with a search input field and a search icon.

The main content area features four tip cards:

- Inspect Character Under Cursor in Vim**: 4.8 (88) stars. Description: "Use ga in Vim to inspect the character under the cursor in Vim. Extend this feature with the Characterize plugin."
- Vim Calculator**: 4.7 (154) stars. Description: "The Vim calculator is a built in feature requiring no plugins: Perform calculations without leaving Vim, even from your text."
- Vim Command Line Window**: 4.7 (74) stars. Description: "The Vim command line window is more than just a history of your commands. It's an editable buffer where you can compose commands."
- Vim split to a specific size**: 4.6 (79) stars. Description: "Open Vim splits to a specific height or width by prepending a number to the command."

At the bottom right, there is a "OUR BOOKS" section with a featured book titled "Git Better with Vim" and a "RECENT VIM TIPS" section.

Learning tricks for **Vim** is a never-ending process. If you want, you can continue it on **Vim tricks newsletter** website (<https://vimtricks.com/>).

**VimAwesome**

AWESOME VIM PLUGINS  
from  
ACROSS THE UNIVERSE

Language  
Completion  
Code display  
Integrations  
Interface  
Commands  
Other

python

Tip: use `/` to search, `⌘/K` to navigate, `N/P` to flip pages

441 PLUGINS

Plugin Name	Author	Stars	Forks
» UltiSnips	by HOLGER RAPP	6602	6272
UltiSnips - The ultimate snippet solution for Vim. Send pull requests to SirVer/ultisnips!			
jedi-vim	by DAVE HALTER	5494	4851
Using the jedi autocompletion library for VIM.			
vim-flake8	by VINCENT DRIESSEN	4049	988
Flake8 plugin for Vim			
taglist.vim	by YEGAPPAN LAKSHMANAN	3302	618
Source code browser (supports C/C++, java, perl, python, tcl, sql, php, etc)			
SimpylFold	by TAYLOR HEDBERG	2995	551
No-BS Python code folding for Vim			
indent/python.vim	by ERIC MC SWEEN	2828	77
An alternative indentation script for python			

If you want to use VIM to its full potential, then install additional plugins suitable for your daily tasks. You can find them at **VimAwesome** (<https://vimawesome.com>).

## Alternatives

Vim is damn good, but I still advise you not to stop at **Vim** and try different good text editors for Linux, which allow you to speed up your work and automate different tasks:

Emacs (<https://www.gnu.org/software/emacs/>)

Neovim (<https://neovim.io>)

Atom (<https://github.com/atom/atom>)

Notepad++ (<https://notepad-plus-plus.org>)

Sublime Text (<https://www.sublimetext.com>)

Many of them have many more plugins and integrations as Vim and can be a complete replacement for the IDE in everyday work.

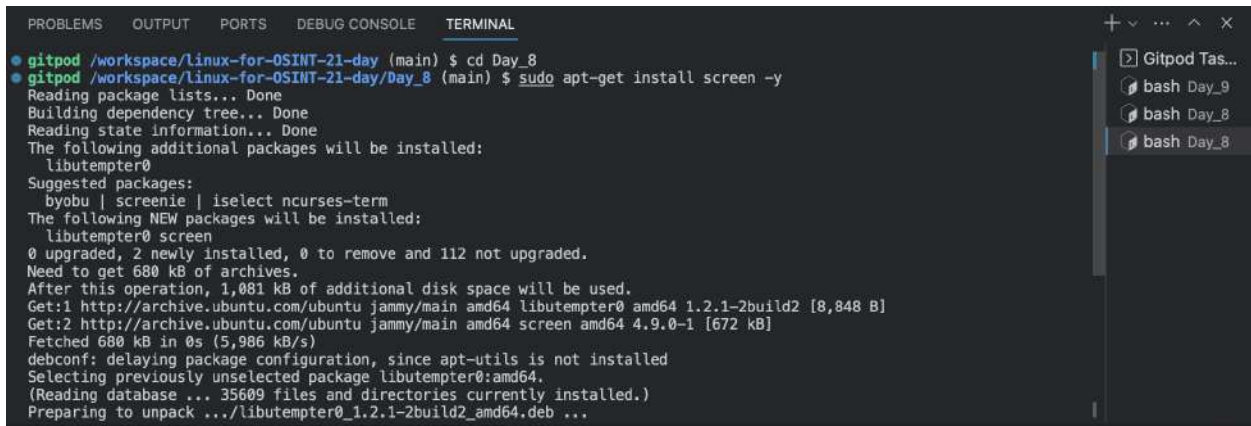
# Day 9. Screen and Crone

Some commands can take a very long time to execute. Sometimes even for several days. For example, if you want to process hundreds of gigabytes of text data on a server with modest technical capacity.

Screen (<https://www.gnu.org/software/screen/>) allows you to perform multiple tasks simultaneously in the background. Roughly speaking, it allows you to make one terminal into several terminals working independently of each other.

Let's start with the installation:

```
sudo apt-get install screen
```



```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_8
gitpod /workspace/linux-for-OSINT-21-day/Day_8 (main) $ sudo apt-get install screen -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libutempter0
Suggested packages:
  byobu | screenie | iselect ncurses-term
The following NEW packages will be installed:
  libutempter0 screen
0 upgraded, 2 newly installed, 0 to remove and 112 not upgraded.
Need to get 680 kB of archives.
After this operation, 1,081 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy/main amd64 libutempter0 amd64 1.2.1-2build2 [8,848 B]
Get:2 http://archive.ubuntu.com/ubuntu jammy/main amd64 screen amd64 4.9.0-1 [672 kB]
Fetched 680 kB in 0s (5,986 kB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package libutempter0:amd64.
(Reading database ... 35609 files and directories currently installed.)
Preparing to unpack .../libutempter0_1.2.1-2build2_amd64.deb ...
```

Create new session with name “demo-screen”:

```
screen -S demo-screen
```

```
PROBLEMS  OUTPUT  PORTS  DEBUG CONSOLE  TERMINAL
$ █
```

Run some commands in it and press **Ctrl+A+D** to exit.

```
PROBLEMS  OUTPUT  PORTS  DEBUG CONSOLE  TERMINAL
$ cat readme.md
cat: readme.md: No such file or directory
$ ls
Day_1  Day_12  Day_15  Day_18  Day_20  Day_5  Day_8  netlas_results.txt  search_osint.txt
Day_10  Day_13  Day_16  Day_19  Day_3  Day_6  Day_9  osint_duckduckgo_search.txt
Day_11  Day_14  Day_17  Day_2  Day_4  Day_7  netlas_results_extracted.txt  README.md
$ cat README.MD
cat: README.MD: No such file or directory
$ cat README.md
# Linux for OSINT. 21 day course for beginners
In this repository you will find sample code files for each day of the course "Linux for OSINT. A 21-day course for beginners".
$ █
```

Now let's get the list of active sessions to see if there is a new session called "demo-screen":

```
screen -ls
```

```
PROBLEMS  OUTPUT  PORTS  DEBUG CONSOLE  TERMINAL
● gitpod /workspace/linux-for-OSINT-21-day (main) $ screen -ls
  There is a screen on:
    9244.demo-screen      (12/16/2023 06:37:02 PM)      (Detached)
  1 Socket in /run/screen/S-gitpod.
○ gitpod /workspace/linux-for-OSINT-21-day (main) $ █
```

To return to a session named "demo-screen" simply type in the command line:

```
screen -r demo-screen
```

And press **Ctrl+A+D** to exit again.

And if you want to delete a session named "demo-screen", type:

```
○ gitpod /workspace/linux-for-OSINT-21-day/Day_9 (main) $
● gitpod /workspace/linux-for-OSINT-21-day/Day_9 (main) $ screen -XS demo-screen quit
⊗ gitpod /workspace/linux-for-OSINT-21-day/Day_9 (main) $ screen -ls
  No Sockets found in /run/screen/S-gitpod.
○ gitpod /workspace/linux-for-OSINT-21-day/Day_9 (main) $ █
```

```
screen -XS demo-screen quit
```

After that, check the list of sessions to make sure the deletion was successful.

```
screen -ls
```

Unfortunately, Gitpod is not very suitable for demonstrating the capabilities of **Screen**, because after closing the workspace tab and after the user is inactive for more than half an hour, all commands are stopped.

If you will be using the **Screen** on your own VM, make sure it is running until all the commands you need are executed in all sessions.

The ideal option for using the **Screen** is a VPS (Virtual Private Server) that runs 24/7 and you can just connect via SSH to it from time to time.

## Crontab

Another important advantage of automation is the ability to schedule a certain sequence of actions for a certain time.

Crontab (<https://ss64.com/bash/crontab.html>) is a utility for editing the crontab file, which stores a schedule of commands to be executed by the **cron** daemon (an old UNIX term).

Here is an example from Wikipedia of what a crontab file might look like:

```
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the `crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# m h dom mon dow user  command
17 * * * * root    cd / && run-parts --report /etc/cron.hourly
25 6 * * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6 * * 7 root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6 1 * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
#
```

Gitpod prohibits the use of Crontab utility at all. And we're not going to try any more commands today.

But lastly, I want to tell you about one very good service that will come in handy when you work with Crontab on a VM or VPS.

# crontab guru

The quick and simple editor for cron schedule expressions by [Cronitor](#)

*“At 22:00 on every day-of-week from Monday through Friday.”*

```
next at 2023-05-17 22:00:00
then at 2023-05-18 22:00:00
then at 2023-05-19 22:00:00
then at 2023-05-22 22:00:00
then at 2023-05-23 22:00:00
```

[random](#)

0 22 \* \* 1-5

<u>minute</u>	<u>hour</u>	<u>day</u> (month)	<u>month</u>	<u>day</u> (week)
---------------	-------------	-----------------------	--------------	----------------------

*	any value
---	-----------

value list
------------

'	separator
---	-----------

-	range of values
---	-----------------

/	step values
---	-------------

@yearly	(non-standard)
---------	----------------

@annually	(non-standard)
-----------	----------------

@monthly	(non-standard)
----------	----------------

@weekly	(non-standard)
---------	----------------

@daily	(non-standard)
--------	----------------

@hourly	(non-standard)
---------	----------------

@reboot	(non-standard)
---------	----------------

Crontab Guru (<https://crontab.guru/>) - is an online service that allows you to quickly compose crone schedule expressions.

# Day 10. Text analyzing and editing utilities

A significant part of this course is devoted to working with various text data formats. After all, the most important advantage of Linux is the significant reduction of routine work. Here are some more universal commands that are useful to know. All of them are available in Linux by default.

## Cat

**Cat** (<https://pubs.opengroup.org/onlinepubs/9699919799/utilities/cat.html>) - utility that allows to view the contents of a text file. We have already used it in previous lessons and it may seem very simple, but it is not quite so. Two second examples.

It allows to view the contents of multiple files at once and save them all at once to a single file:



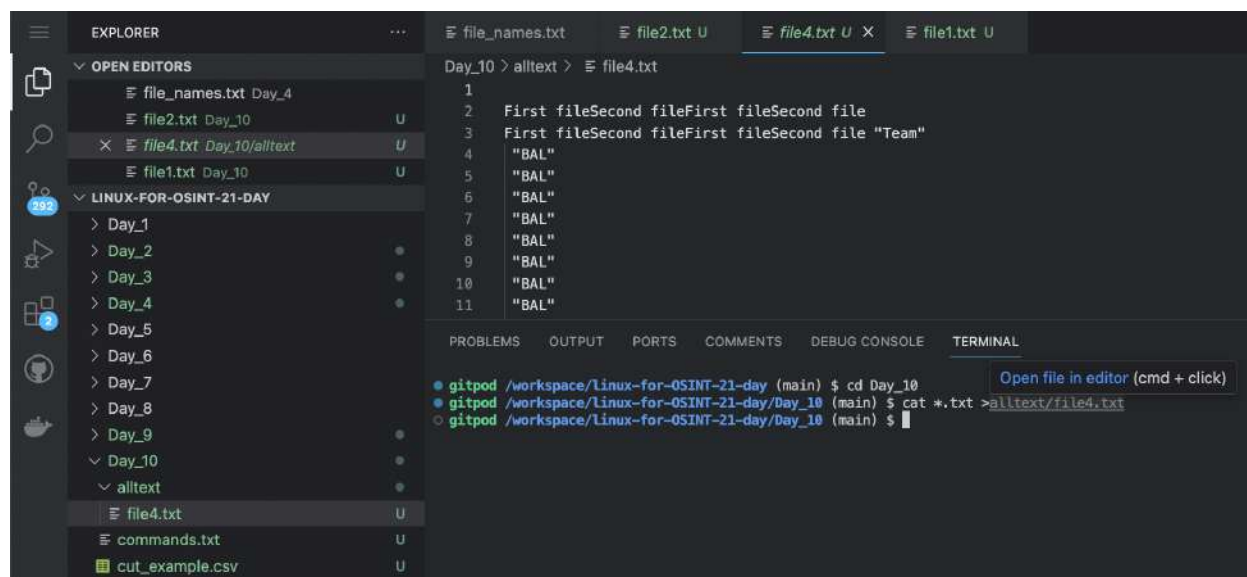
```
file_names.txt file2.txt U file3.txt U X file1.txt U
Day_10 > file3.txt
1 First fileSecond file

PROBLEMS OUTPUT PORTS COMMENTS DEBUG CONSOLE TERMINAL
● gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_10
● gitpod /workspace/linux-for-OSINT-21-day/Day_10 (main) $ cat file1.txt file2.txt >file3.txt
○ gitpod /workspace/linux-for-OSINT-21-day/Day_10 (main) $
```

```
cd Day_10
```

```
cat file1.txt file2.txt >file3.txt
```

You can even use it to collect the content of all files with a certain extension (or containing a certain word in the name) into one file:



```
cat *.txt >alltext/file4.txt
```

## Cut

**Cut** (<https://pubs.opengroup.org/onlinepubs/9699919799/utilities/cut.html>) is utility for cutting sections from each line of a file. Recall that we used the **Sed** utility to cut lines from the file, and the **Awk** utility to cut individual columns (**Day 7**).

Let's try to extract the first character from each line:



```
gitpod /workspace/linux-for-OSINT-21-day/Day_10 (main) $ cut -c 5-10 cut_example.csv
e", "T
m Dona
l Bako
on Her
in Mil
is Gom
an Rob
uel Te
vin Mo
rey Hu
m Ster
f Fior
ddie B
k Mark
ndon F
ey Pat
Payto
Gibbo
k Beda
den Pe
m Loew
gitpod /workspace/linux-for-OSINT-21-day/Day_10 (main) $
```

## cut -c 5-10 cut\_example.csv

Cut can also be used to cut individual bytes or fields with delimiter designation (another method for extracting columns from CSV files).

## Uniq

**Uniq** (<https://pubs.opengroup.org/onlinepubs/9699919799/utilities/uniq.html>) - utility that allows you to remove all duplicate lines from a file.

Let's try:

```
cut_example.csv U  second_column.txt U  uniq_example.csv U X
Day_10 > uniq_example.csv
1  "Erik Bedard", "BAL", "Starting Pitcher", 73, 189, 27.99
2  "Erik Bedard", "BAL", "Starting Pitcher", 73, 189, 27.99
3  "Adam Loewen", "BAL", "Starting Pitcher", 78, 219, 22.89
4  "Adam Loewen", "BAL", "Starting Pitcher", 78, 219, 22.89
5  "Hayden Penn", "BAL", "Starting Pitcher", 75, 185, 22.38
6  "Erik Bedard", "BAL", "Starting Pitcher", 73, 189, 27.99

PROBLEMS  OUTPUT  PORTS  DEBUG CONSOLE  TERMINAL
gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_10
gitpod /workspace/linux-for-OSINT-21-day/Day_10 (main) $ uniq uniq_example.csv
"Erik Bedard", "BAL", "Starting Pitcher", 73, 189, 27.99
"Adam Loewen", "BAL", "Starting Pitcher", 78, 219, 22.89
"Hayden Penn", "BAL", "Starting Pitcher", 75, 185, 22.38
"Erik Bedard", "BAL", "Starting Pitcher", 73, 189, 27.99
gitpod /workspace/linux-for-OSINT-21-day/Day_10 (main) $
```

## uniq uniq\_example.csv

## Sort

**Sort** (<https://pubs.opengroup.org/onlinepubs/9699919799/utilities/sort.html>) - utility that sorts files in alphabetical and reverse alphabetical order, as well as descending and ascending order.

```

Day_10 > cut_example.csv
1 "Name", "Team", "Position", "Height(inches)", "Weight(lbs)", "Age"
2 "Adam Donachie", "BAL", "Catcher", 74, 180, 22.99
3 "Paul Bako", "BAL", "Catcher", 74, 215, 34.69
4 "Ramon Hernandez", "BAL", "Catcher", 72, 210, 30.78
5 "Kevin Millar", "BAL", "First Baseman", 72, 210, 35.43
6 "Chris Gomez", "BAL", "First Baseman", 73, 188, 35.71
7 "Brian Roberts", "BAL", "Second Baseman", 69, 176, 29.39
8 "Miguel Tejada", "BAL", "Shortstop", 69, 209, 30.77
9 "Melvin Mora", "BAL", "Third Baseman", 71, 200, 35.07
10 "Aubrey Huff", "BAL", "Third Baseman", 76, 231, 30.19
11 "Adam Stern", "BAL", "Outfielder", 71, 180, 27.05

gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_10
gitpod /workspace/linux-for-OSINT-21-day/Day_10 (main) $ sort -r cut_example.csv
"Ramon Hernandez", "BAL", "Catcher", 72, 210, 30.78
"Paul Bako", "BAL", "Catcher", 74, 215, 34.69
"Nick Markakis", "BAL", "Outfielder", 74, 185, 23.29
"Name", "Team", "Position", "Height(inches)", "Weight(lbs)", "Age"
"Miguel Tejada", "BAL", "Shortstop", 69, 209, 30.77
"Melvin Mora", "BAL", "Third Baseman", 71, 200, 35.07
"Kevin Millar", "BAL", "First Baseman", 72, 210, 35.43
"Jeff Fiorentino", "BAL", "Outfielder", 73, 188, 23.88
"Jay Payton", "BAL", "Outfielder", 70, 185, 34.27
"Jay Gibbons", "BAL", "Designated Hitter", 72, 197, 30
"Hayden Penn", "BAL", "Starting Pitcher", 75, 185, 22.38
"Freddie Bynum", "BAL", "Outfielder", 73, 180, 26.96
"Erik Bedard", "BAL", "Starting Pitcher", 73, 189, 27.99
"Corey Patterson", "BAL", "Outfielder", 69, 180, 27.55
"Chris Gomez", "BAL", "First Baseman", 73, 188, 35.71
"Brian Roberts", "BAL", "Second Baseman", 69, 176, 29.39
"Brandon Fahey", "BAL", "Outfielder", 74, 160, 26.11
"Aubrey Huff", "BAL", "Third Baseman", 76, 231, 30.19
"Adam Stern", "BAL", "Outfielder", 71, 180, 27.05
"Adam Loewen", "BAL", "Starting Pitcher", 78, 219, 22.89
"Adam Donachie", "BAL", "Catcher", 74, 180, 22.99
gitpod /workspace/linux-for-OSINT-21-day/Day_10 (main) $

```

## sort -r cut\_example.csv

Here we used the `-r` flag to sort the rows in reverse alphabetical order.

## Iconv

Iconv (<https://pubs.opengroup.org/onlinepubs/9699919799/utilities/iconv.html>) utility that convert some text in one encoding into another encoding.

We won't run this command (I don't have a suitable test file at hand right now), but it will be useful for you to know about its existence. If some text suddenly displays obscure characters instead of letters, try saving the file in a different encoding using this command.

## iconv -f UTF-8 -t ASCII//TRANSLIT utf8.txt acii.txt

This is the last chapter in this book about automating text file manipulation in Linux. If you are interested in this topic, I recommend you to check out the repository **Text processing Recipes Linux** (<https://github.com/adrianlarion/text-processing-recipes-linux>).

# Day 11. Video, audio and images

Sometimes in investigations, you have to deal with a large volume of media files that are very difficult to analyze manually. Today I will show you some very simple tools that will help you save time while watching videos, listening to audio and translating text from pictures.

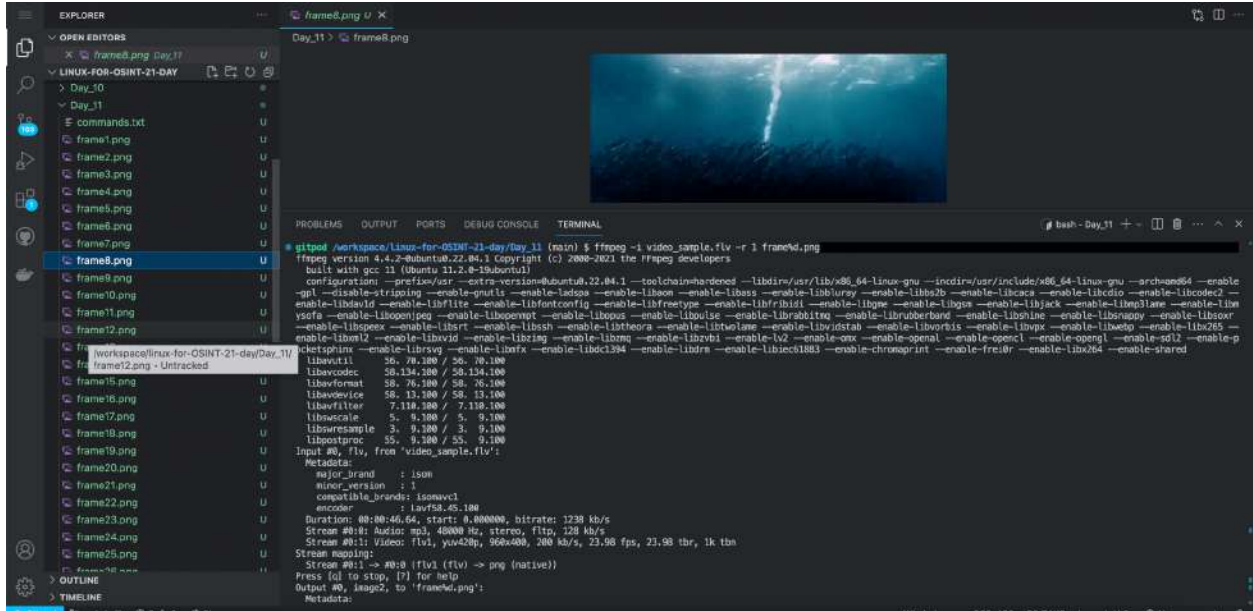
## FFMPEG

FFMPEG (<https://www.ffmpeg.org/>) is a toolkit for handling video, audio, and other multimedia files, capable of hundreds of different operations.

```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
● gitpod /workspace/Linux-for-0SINT-21-day (main) $ sudo apt-get update && sudo apt-get install -y ffmpeg
Get:1 https://download.docker.com/linux/ubuntu jammy InRelease [48.8 kB]
Get:2 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages [26.8 kB]
Get:3 https://ppa.launchpadcontent.net/git-core/ppa/ubuntu jammy InRelease [23.8 kB]
Get:4 https://ppa.launchpadcontent.net/ondrej/apache2/ubuntu jammy InRelease [23.8 kB]
Get:5 https://ppa.launchpadcontent.net/ondrej/nginx-mainline/ubuntu jammy InRelease [23.8 kB]
Get:6 https://ppa.launchpadcontent.net/ondrej/php/ubuntu jammy InRelease [23.9 kB]
Get:7 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Hit:8 http://archive.ubuntu.com/ubuntu jammy InRelease
Get:9 https://ppa.launchpadcontent.net/git-core/ppa/ubuntu jammy/main amd64 Packages [2,961 B]
Get:10 http://archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:11 https://ppa.launchpadcontent.net/ondrej/apache2/ubuntu jammy/main amd64 Packages [3,956 B]
Get:12 https://ppa.launchpadcontent.net/ondrej/nginx-mainline/ubuntu jammy/main amd64 Packages [5,671 B]
Get:13 https://ppa.launchpadcontent.net/ondrej/php/ubuntu jammy/main amd64 Packages [204 kB]
Get:14 http://archive.ubuntu.com/ubuntu jammy-backports InRelease [109 kB]
Get:15 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 Packages [1,572 kB]
Get:17 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [1,305 kB]
Get:16 https://packagecloud.io/github/git-lfs/ubuntu jammy InRelease [28.0 kB]
Get:18 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [1,599 kB]
Get:19 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 Packages [1,046 kB]
Get:20 http://security.ubuntu.com/ubuntu jammy-security/multiverse amd64 Packages [44.0 kB]
Get:21 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [1,326 kB]
Get:22 http://archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 Packages [49.8 kB]
Get:24 http://archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [1,602 kB]
Get:25 http://archive.ubuntu.com/ubuntu jammy-backports/main amd64 Packages [50.4 kB]
Get:26 http://archive.ubuntu.com/ubuntu jammy-backports/universe amd64 Packages [28.1 kB]
Get:23 https://packagecloud.io/github/git-lfs/ubuntu jammy/main amd64 Packages [1,654 B]
Get:27 https://apt.lvm.org/jammy lvm-toolchain-jammy-17 InRelease [6,833 B]
Get:28 https://apt.lvm.org/jammy lvm-toolchain-jammy-17/main amd64 Packages [12.4 kB]
Fetched 9,396 kB in 2s (5,486 kB/s)
Reading package lists... Done
Reading package lists... Done
```

Let's install:

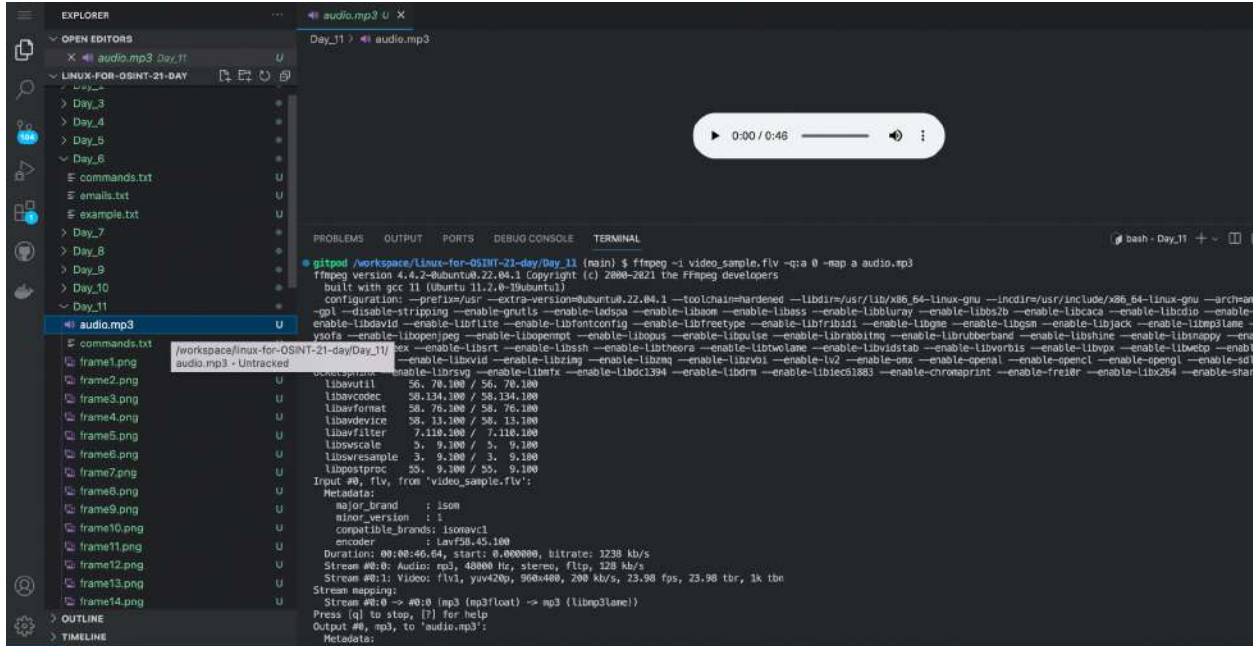
```
sudo apt-get update
sudo apt-get install -y ffmpeg
```



Now let's try to convert the video into a set of frames in png format:

```
cd Day_11
ffmpeg -i video_sample.flv -r 1 frame%d.png
```

Viewing a video frame by frame allows you to quickly understand what the video is about and highlight key episodes in it. A simple but very time-saving trick. Note that if you right-click on a file and folder in Gitpod, it can be downloaded from workspace.



Now let's try to extract the audio from the video with another simple command. You can listen to the result without leaving Gitpod:

```
ffmpeg -i video_sample.flv -q:a 0 -map a audio.mp3
```

As you can guess, audio needs to be extracted from video to turn it into text. This will help you quickly analyze it and find the key semantic parts.

## Transcribe audio

PocketSphinx (<https://github.com/cmuspinx/pocketsphinx>) is a very simple speech recognizer.

```

gitpod /workspace/linux-for-OSINT-21-day/Day_11 (main) $ sudo apt install pocketsphinx pocketsphinx-en-us
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
pocketsphinx-en-us is already the newest version (0.8.0+real5prealpha+1-14ubuntu1).
pocketsphinx-en-us set to manually installed.
The following NEW packages will be installed:
  pocketsphinx
0 upgraded, 1 newly installed, 0 to remove and 87 not upgraded.
Need to get 32.6 kB of archives.
After this operation, 140 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://archive.ubuntu.com/ubuntu/jammy/universe amd64 pocketsphinx amd64 0.8.0+real5prealpha+1-14ubuntu1 [32.6 kB]
Fetched 32.6 kB in 0s (111 kB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package pocketsphinx.
(Reading database ... 36864 files and directories currently installed.)
Preparing to unpack .../pocketsphinx_0.8.0+real5prealpha+1-14ubuntu1_amd64.deb ...
Unpacking pocketsphinx (0.8.0+real5prealpha+1-14ubuntu1) ...
Setting up pocketsphinx (0.8.0+real5prealpha+1-14ubuntu1) ...
Processing triggers for man-db (2.10.2-1) ...
gitpod /workspace/linux-for-OSINT-21-day/Day_11 (main) $

```

Let's install:

**sudo apt install pocketsphinx pocketsphinx-en-us**

If you want to work with audio recordings in other languages, install additional libraries. A complete list can be found [here](#).

PocketSphinx is an easy to use utility that produces quality results. But, unfortunately, it does not work with different audio file formats, but only with wav 16000 Hz.

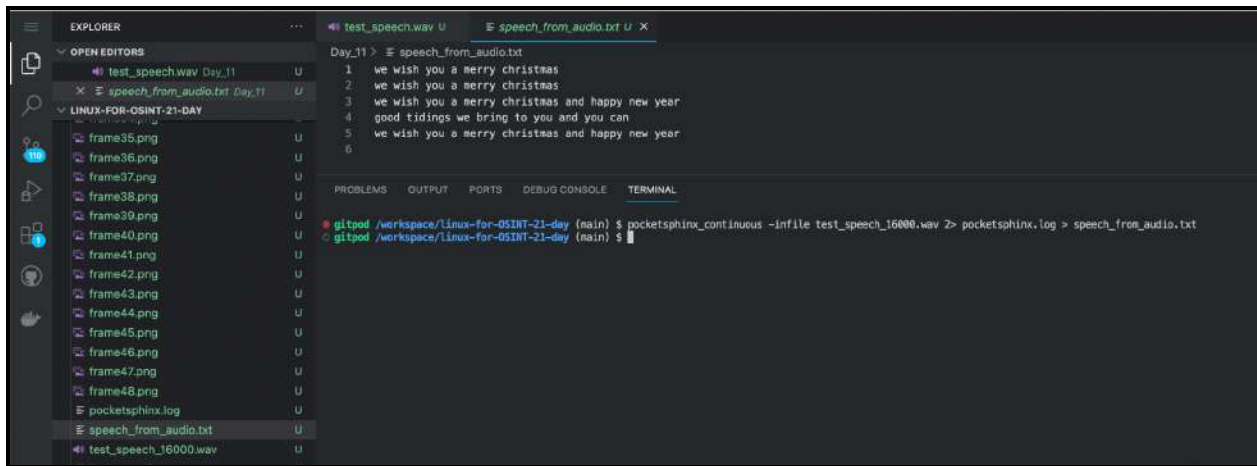
```

gitpod /workspace/linux-for-OSINT-21-day/Day_11 (main) $ ffmpeg -i test_speech.wav -ar 16000 -ac 1 test_speech_16000.wav
ffmpeg version 4.4.2-0ubuntu0.22.04.1 Copyright (c) 2000-2021 the FFmpeg developers
  built with gcc 11 (Ubuntu 11.2.0-19ubuntu1)
  configuration: --prefix=/usr --extra-version=ubuntu0.22.04.1 --toolchain=hardened --libdir=/usr/lib/x86_64-linux-gnu --incdir=/usr/include/x86_64-linux-gnu --arch=amd64 --enable-
  psl --disable-stripping --enable-gnutls --enable-ladspa --enable-libaom --enable-libass --enable-libbluray --enable-libsbs20 --enable-libcaca --enable-libcdio --enable-libcodecs2 --
  enable-libdav1d --enable-libflite --enable-libfontconfig --enable-libfreetype --enable-libfridji --enable-libgme --enable-libgsm --enable-libjack --enable-libmp3lame --enable-libm
  ysofa --enable-libopenjpeg --enable-libopenmpt --enable-libopus --enable-libpulse --enable-librabbitmq --enable-librubberband --enable-libshine --enable-libsnpappy --enable-libsokr
  --enable-libspeex --enable-libsrt --enable-libssh --enable-libtheora --enable-libtwolame --enable-libvidstab --enable-libvorbis --enable-libvpx --enable-libwebp --enable-libx265 --
  enable-libxml2 --enable-libxvid --enable-libzimg --enable-libzmq --enable-libzvi --enable-lv2 --enable-omx --enable-opengl --enable-openssl --enable-opengl --enable-sdl2 --enable-p
  ocketsphinx --enable-libsrt --enable-libmfx --enable-libdc1394 --enable-libdrm --enable-libiec61883 --enable-chromaprint --enable-frei0r --enable-libx264 --enable-shared
libavutil 56. 70.100 / 56. 70.100
libavcodec 58.134.100 / 58.134.100
libavformat 58. 76.100 / 58. 76.100
libavdevice 58. 13.100 / 58. 13.100
libavfilter 7.110.100 / 7.110.100
libswscale 5.  9.100 /  5.  9.100
libswresample 3.  9.100 /  3.  9.100
libpostproc 55.  9.100 / 55.  9.100
Guessed Channel Layout for Input Stream #0.0 : mono
Input #0, wav, from 'test_speech.wav':
  Metadata:
    encoder      : Lavf59.27.100
  Duration: 00:00:14.21, bitrate: 384 kb/s
  Stream #0:0: Audio: pcm_s16le ([1][0][0][0] / 0x0001), 24000 Hz, mono, s16, 384 kb/s
Stream mapping:
  Stream #0:0 -> #0:0 (pcm_s16le (native) -> pcm_s16le (native))
Press [q] to stop, [?] for help
Output #0, wav, to 'test_speech_16000.wav':
  Metadata:
    ISFT        : Lavf58.76.100
  Stream #0:0: Audio: pcm_s16le ([1][0][0][0] / 0x0001), 16000 Hz, mono, s16, 256 kb/s
  Metadata:
    encoder      : Lavc58.134.100 pcm_s16le
  size= 444kB time=00:00:14.20 bitrate= 256.1kbits/s speed=2.86e+03x
video:0kB audio:444kB subtitle:0kB other streams:0kB global headers:0kB muxing overhead: 0.017156%

```

Fortunately, most audio formats can be easily converted to it with ffmpeg. Example:

```
ffmpeg -i test_speech.wav -ar 16000 -ac 1 test_speech_16000.wav
```



And now try convert speech audio to text:

```
pocketsphinx_continuous -infile test_speech_16000.wav 2>
pocketsphinx.log > speech_from_audio.txt
```

If you need very high quality speech recognition for different rare languages, I suggest you take a look at Open AI Whisper API (<https://openai.com/research/whisper>). It is a paid tool, but it is really good.

## Extract text from images

Tesseract OCR (<https://github.com/tesseract-ocr/tesseract>) is a simple Optical Text Recognition utility.

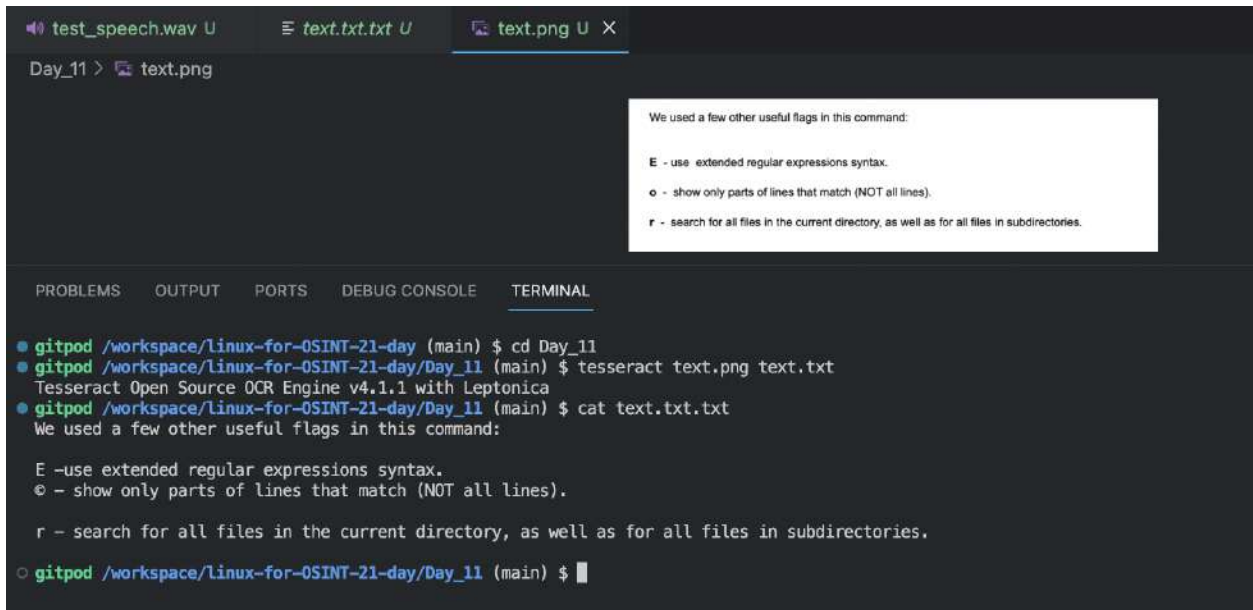
```

gitpod /workspace/Linux-for-OSINT-21-day (main) $ sudo apt install tesseract-ocr
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libarchive13 libgif7 libliblept5 libtesseract4 tesseract-ocr-eng tesseract-ocr-osd
Suggested packages:
  lrzip
The following NEW packages will be installed:
  libarchive13 libgif7 libliblept5 libtesseract4 tesseract-ocr tesseract-ocr-eng tesseract-ocr-osd
0 upgraded, 7 newly installed, 0 to remove and 87 not upgraded.
Need to get 7,634 kB of archives.
After this operation, 22.7 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://archive.ubuntu.com/ubuntu jammy/main amd64 libarchive13 amd64 3.6.0-1ubuntu1 [368 kB]
Get:2 http://archive.ubuntu.com/ubuntu jammy/main amd64 libgif7 amd64 5.1.9-2build2 [33.8 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libliblept5 amd64 1.82.0-3build1 [1,107 kB]
Get:4 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libtesseract4 amd64 4.1.1-2.1build1 [1,308 kB]
Get:5 http://archive.ubuntu.com/ubuntu jammy/universe amd64 tesseract-ocr-eng all 1:4.00~git30-7274cfa-1.1 [1,591 kB]
Get:6 http://archive.ubuntu.com/ubuntu jammy/universe amd64 tesseract-ocr-osd all 1:4.00~git30-7274cfa-1.1 [2,990 kB]
Get:7 http://archive.ubuntu.com/ubuntu jammy/universe amd64 tesseract-ocr amd64 4.1.1-2.1build1 [236 kB]
Fetched 7,634 kB in 1s (7,075 kB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package libarchive13:amd64.
(Reading database ... 36875 files and directories currently installed.)
Preparing to unpack .../0-libarchive13_3.6.0-1ubuntu1_amd64.deb ...
Unpacking libarchive13:amd64 (3.6.0-1ubuntu1) ...
Selecting previously unselected package libgif7:amd64.
Preparing to unpack .../1-libgif7_5.1.9-2build2_amd64.deb ...
Unpacking libgif7:amd64 (5.1.9-2build2) ...
Selecting previously unselected package libliblept5:amd64.
Preparing to unpack .../2-libliblept5_1.82.0-3build1_amd64.deb ...
Unpacking libliblept5:amd64 (1.82.0-3build1) ...
Selecting previously unselected package libtesseract4:amd64.
Preparing to unpack .../3-libtesseract4_4.1.1-2.1build1_amd64.deb ...
Unpacking libtesseract4:amd64 (4.1.1-2.1build1) ...
Selecting previously unselected package tesseract-ocr-eng.

```

## sudo apt install tesseract-ocr

Try to OCR text in image and save to text.txt:



```
test_speech.wav U  text.txt.txt U  text.png U X
Day_11 > text.png

We used a few other useful flags in this command:

E - use extended regular expressions syntax.
o - show only parts of lines that match (NOT all lines).
r - search for all files in the current directory, as well as for all files in subdirectories.

PROBLEMS  OUTPUT  PORTS  DEBUG CONSOLE  TERMINAL
● gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_11
● gitpod /workspace/linux-for-OSINT-21-day/Day_11 (main) $ tesseract text.png text.txt
Tesseract Open Source OCR Engine v4.1.1 with Leptonica
● gitpod /workspace/linux-for-OSINT-21-day/Day_11 (main) $ cat text.txt.txt
We used a few other useful flags in this command:

E -use extended regular expressions syntax.
o - show only parts of lines that match (NOT all lines).

r - search for all files in the current directory, as well as for all files in subdirectories.
○ gitpod /workspace/linux-for-OSINT-21-day/Day_11 (main) $ █
```

## tesseract text.png text.txt

Once we've turned video, audio, and pictures into text, we can translate it! Here is one of the many ways to automate this process.

## Translate text

Translate Shell (<https://github.com/soimort/translate-shell>) is command line translator based on Google Translate, Bing Translator, Yandex.Translate and other translate services.

Let's install:

```
PROBLEMS OUTPUT PORTS COMMENTS DEBUG CONSOLE TERMINAL
gitpod /workspace/Linux-for-OSINT-21-day (main) $ sudo apt install translate-shell
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  aspell aspell-en dictionaries-common gawk libaspell15 libfribidi-bin libtext-iconv-perl rlrwrap
Suggested packages:
  aspell-oc spellutils wordlist gawk-doc mplayer | mpv | mpg123 | espeak
The following NEW packages will be installed:
  aspell aspell-en dictionaries-common gawk libaspell15 libfribidi-bin libtext-iconv-perl rlrwrap translate-shell
0 upgraded, 9 newly installed, 0 to remove and 87 not upgraded.
Need to get 1,523 kB of archives.
After this operation, 6,130 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 gawk amd64 1:5.1.0-1ubuntu0.1 [447 kB]
Get:2 http://archive.ubuntu.com/ubuntu jammy/main amd64 libtext-iconv-perl amd64 1.7-7build3 [14.3 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy/main amd64 libaspell15 amd64 0.60.8-4build1 [325 kB]
Get:4 http://archive.ubuntu.com/ubuntu jammy/main amd64 dictionaries-common all 1.28.14 [185 kB]
Get:5 http://archive.ubuntu.com/ubuntu jammy/main amd64 aspell amd64 0.60.8-4build1 [87.7 kB]
Get:6 http://archive.ubuntu.com/ubuntu jammy/main amd64 aspell-en all 2018.04.16-0-1 [299 kB]
Get:7 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64 libfribidi-bin amd64 1.0.8-2ubuntu3.1 [9,494 B]
Get:8 http://archive.ubuntu.com/ubuntu jammy/universe amd64 rlrwrap amd64 0.43-1build3 [98.2 kB]
Get:9 http://archive.ubuntu.com/ubuntu jammy/multiverse amd64 translate-shell all 0.9.6.12-1 [56.9 kB]
Fetched 1,523 kB in 1s (1,760 kB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package gawk.
(Reading database ... 36331 files and directories currently installed.)
Preparing to unpack .../0-gawk_1:5.1.0-1ubuntu0.1_amd64.deb ...
Unpacking gawk (1:5.1.0-1ubuntu0.1) ...
Selecting previously unselected package libtext-iconv-perl.
Preparing to unpack .../1-libtext-iconv-perl_1.7-7build3_amd64.deb ...
Unpacking libtext-iconv-perl (1.7-7build3) ...
Selecting previously unselected package libaspell15:amd64.
Preparing to unpack .../2-libaspell15_0.60.8-4build1_amd64.deb ...
Unpacking libaspell15:amd64 (0.60.8-4build1) ...
```

## sudo apt install translate-shell

And translate text from file in French:

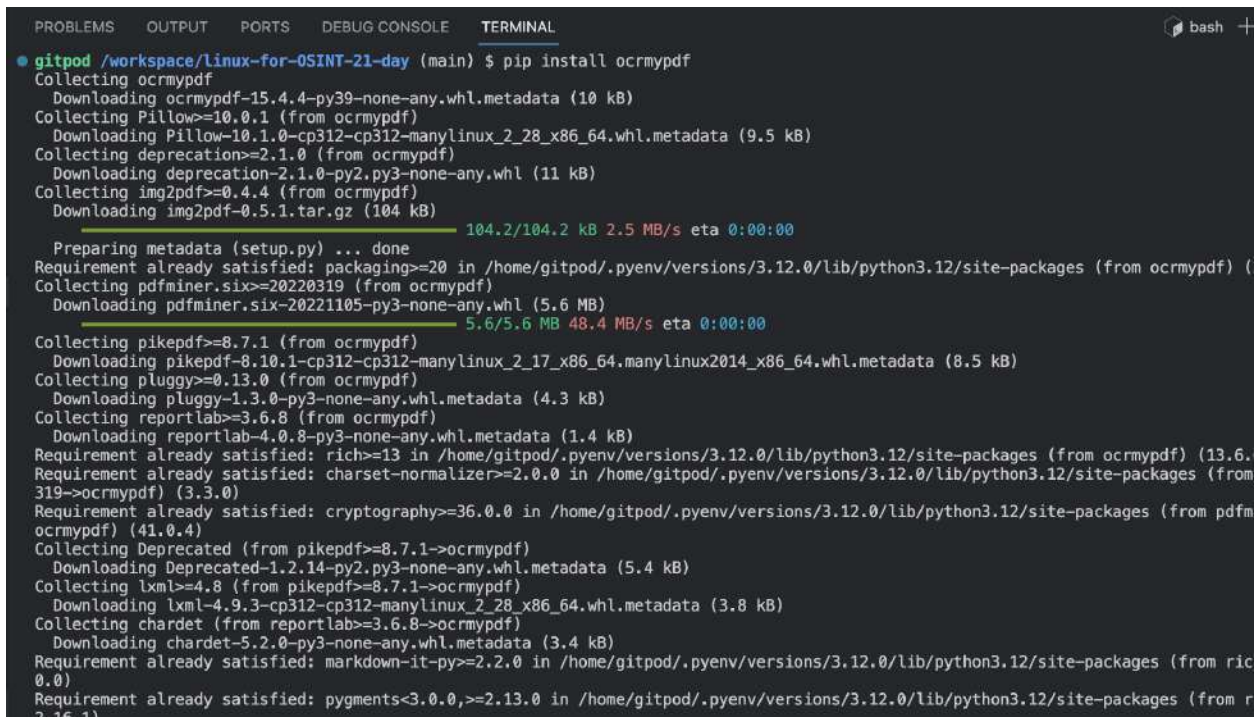
```
cat text.txt.txt | trans :fr
```

Codes for different languages can be found on this page: <https://github.com/soimort/translate-shell/wiki/Languages>.

Today, we explored the easiest-to-use methods for automating media content analysis. But if you have to do it often, I recommend you to learn more about AI tools for image recognition that have appeared in the last six months to a year. Many of them have APIs that can be used to create command line tools.

# Day 12. Analyze PDF

Sometimes investigations have to deal with hundreds of documents and only one of them may contain really important information. Therefore, it is worth trying to automate the analysis of text, images and metadata in PDF.



```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
● gitpod /workspace/linux-for-0SINT-21-day (main) $ pip install ocrmypdf
Collecting ocrmypdf
  Downloading ocrmypdf-15.4.4-py39-none-any.whl.metadata (10 kB)
Collecting Pillow>=10.0.1 (from ocrmypdf)
  Downloading Pillow-10.1.0-cp312-cp312-manylinux_2_28_x86_64.whl.metadata (9.5 kB)
Collecting deprecation>=2.1.0 (from ocrmypdf)
  Downloading deprecation-2.1.0-py2.py3-none-any.whl (11 kB)
Collecting img2pdf>=0.4.4 (from ocrmypdf)
  Downloading img2pdf-0.5.1.tar.gz (104 kB)
  104.2/104.2 kB 2.5 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Requirement already satisfied: packaging>=20 in /home/gitpod/.pyenv/versions/3.12.0/lib/python3.12/site-packages (from ocrmypdf) (
Collecting pdfminer.six>=20220319 (from ocrmypdf)
  Downloading pdfminer.six-20221105-py3-none-any.whl (5.6 MB)
  5.6/5.6 MB 48.4 MB/s eta 0:00:00
Collecting pikepdf>=8.7.1 (from ocrmypdf)
  Downloading pikepdf-8.10.1-cp312-cp312-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (8.5 kB)
Collecting pluggy>=0.13.0 (from ocrmypdf)
  Downloading pluggy-1.3.0-py3-none-any.whl.metadata (4.3 kB)
Collecting reportlab>=3.6.8 (from ocrmypdf)
  Downloading reportlab-4.0.8-py3-none-any.whl.metadata (1.4 kB)
Requirement already satisfied: rich>=13 in /home/gitpod/.pyenv/versions/3.12.0/lib/python3.12/site-packages (from ocrmypdf) (13.6.
Requirement already satisfied: charset-normalizer>=2.0.0 in /home/gitpod/.pyenv/versions/3.12.0/lib/python3.12/site-packages (from
319->ocrmypdf) (3.3.0)
Requirement already satisfied: cryptography>=36.0.0 in /home/gitpod/.pyenv/versions/3.12.0/lib/python3.12/site-packages (from pdfm
ocrmypdf) (41.0.4)
Collecting Deprecated (from pikepdf>=8.7.1->ocrmypdf)
  Downloading Deprecated-1.2.14-py2.py3-none-any.whl.metadata (5.4 kB)
Collecting lxml>=4.8 (from pikepdf>=8.7.1->ocrmypdf)
  Downloading lxml-4.9.3-cp312-cp312-manylinux_2_28_x86_64.whl.metadata (3.8 kB)
Collecting chardet (from reportlab>=3.6.8->ocrmypdf)
  Downloading chardet-5.2.0-py3-none-any.whl.metadata (3.4 kB)
Requirement already satisfied: markdown-it-py>=2.2.0 in /home/gitpod/.pyenv/versions/3.12.0/lib/python3.12/site-packages (from ric
0.0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /home/gitpod/.pyenv/versions/3.12.0/lib/python3.12/site-packages (from r
2.16.1)
```

OCRmyPDF (<https://github.com/ocrmypdf/OCRmyPDF>) - tool that adds an optical character recognition (OCR) text to scanned PDF files. It makes the text searchable and processable.

Let's install:

```
pip install ocrmypdf
```

```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
○ gitpod /workspace/linux-for-0SINT-21-day/Day_12 (main) $ sudo apt update

sudo apt install tesseract-ocr
Get:2 https://download.docker.com/linux/ubuntu jammy InRelease [48.8 kB]
Get:3 https://download.docker.com/linux/ubuntu jammy/stable amd64 Packages [26.8 kB]
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Hit:5 http://archive.ubuntu.com/ubuntu jammy InRelease
Get:6 https://ppa.launchpadcontent.net/git-core/ppa/ubuntu jammy InRelease [23.8 kB]
Get:7 https://ppa.launchpadcontent.net/ondrej/apache2/ubuntu jammy InRelease [23.8 kB]
Get:8 https://ppa.launchpadcontent.net/ondrej/nginx-mainline/ubuntu jammy InRelease [23.8 kB]
Get:9 https://ppa.launchpadcontent.net/ondrej/php/ubuntu jammy InRelease [23.9 kB]
Get:10 http://archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:11 https://ppa.launchpadcontent.net/git-core/ppa/ubuntu jammy/main amd64 Packages [2,961 B]
Get:12 https://ppa.launchpadcontent.net/ondrej/apache2/ubuntu jammy/main amd64 Packages [3,956 B]
Get:13 http://archive.ubuntu.com/ubuntu jammy-backports InRelease [109 kB]
Get:1 https://apt.llvm.org/jammy llvm-toolchain-jammy-17 InRelease [6,833 B]
Get:15 https://ppa.launchpadcontent.net/ondrej/nginx-mainline/ubuntu jammy/main amd64 Packages [5,671 B]
Get:16 https://ppa.launchpadcontent.net/ondrej/php/ubuntu jammy/main amd64 Packages [204 kB]
Get:17 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [1,326 kB]
Get:18 https://apt.llvm.org/jammy llvm-toolchain-jammy-17/main amd64 Packages [12.4 kB]
Get:19 http://archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [1,602 kB]
Get:14 https://packagecloud.io/github/git-lfs/ubuntu jammy InRelease [28.0 kB]
Get:20 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 Packages [1,046 kB]
Get:21 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 Packages [1,572 kB]
Get:22 http://archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 Packages [49.8 kB]
Get:23 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [1,305 kB]
Get:24 http://security.ubuntu.com/ubuntu jammy-security/multiverse amd64 Packages [44.0 kB]
Get:25 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [1,599 kB]
Get:27 http://archive.ubuntu.com/ubuntu jammy-backports/universe amd64 Packages [28.1 kB]
Get:28 http://archive.ubuntu.com/ubuntu jammy-backports/main amd64 Packages [50.4 kB]
Get:26 https://packagecloud.io/github/git-lfs/ubuntu jammy/main amd64 Packages [1,654 B]
Fetched 9,396 kB in 2s (4,285 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

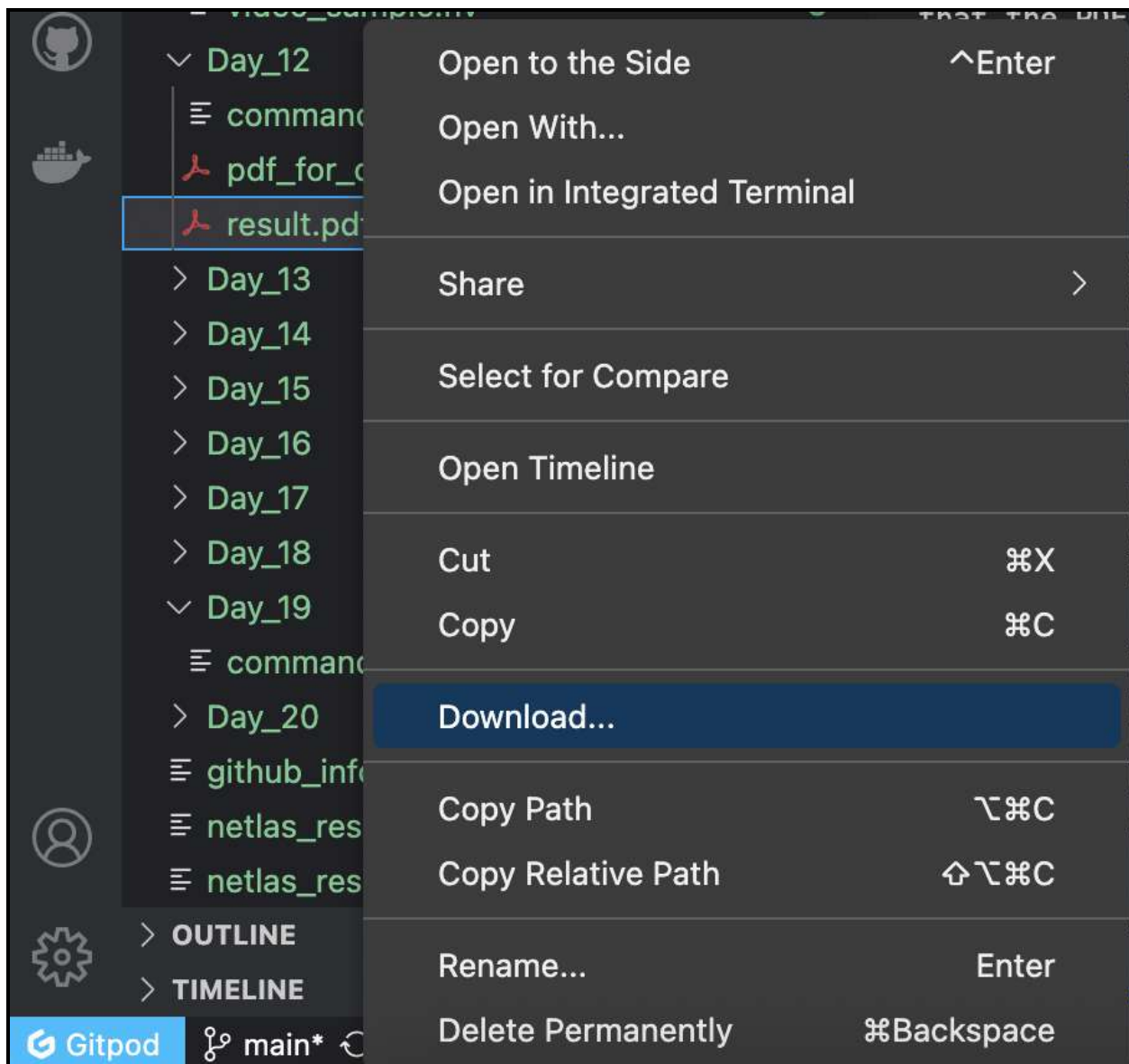
Install Tesseract OCR (<https://github.com/tesseract-ocr/tesseract>), the text recognition utility that OCRMyPDF is based on.

```
sudo apt update
sudo apt install tesseract-ocr
```

Install Ghostscript (<https://www.ghostscript.com>) is an interpreter for the PostScript® language and PDF files:

```
sudo apt install ghostscript
```





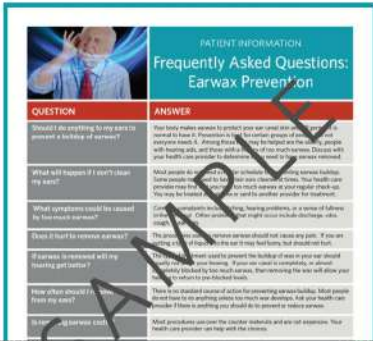
If you use Gitpod, you need to download the PDF document to your computer to view it. Right-click on it and select **Download**.

# Instructions for Adding Your Logo & Address to AAO-HNSF Patient Handouts

CO-BRANDING AAO-HNSF PATIENT HANDOUTS IS AS EASY AS 1-2-3!

## 1 Download & Save Patient Handout

- Download the patient handout  
You will need Adobe Acrobat Reader.  
You can install it for free here:  
[get.adobe.com/reader](http://get.adobe.com/reader)
- Save it to your computer  
Remember where you save it; you will need to access it in the following steps.



## 2 Upload Your Logo & Address

Your image **must** be saved as a .pdf file!

For additional information on how to save your logo/address as a .pdf file, please see the instructions below.

You will not be able to upload your personal information to the patient handout if the file is not saved in the .pdf format.

- Open the saved patient handout document
- Click in the white space at the bottom of the page.
- Within the **Select Icon** window, **Browse** to the .pdf file you would like to **Insert** and then click, **OK**.  
The logo and/or address file will resize appropriately for the box.
- Type your web address in the blue rectangle below the white image field.

## 3 You're Ready to Print!

- Save your file.  
Make sure to save your file before you close the document to ensure your logo and/or address will be there the next time you open up the document.
- You are now ready to print!

Example of a co-branded Patient Handout:



When viewing a file, the text should be available for selection, copying and pasting.

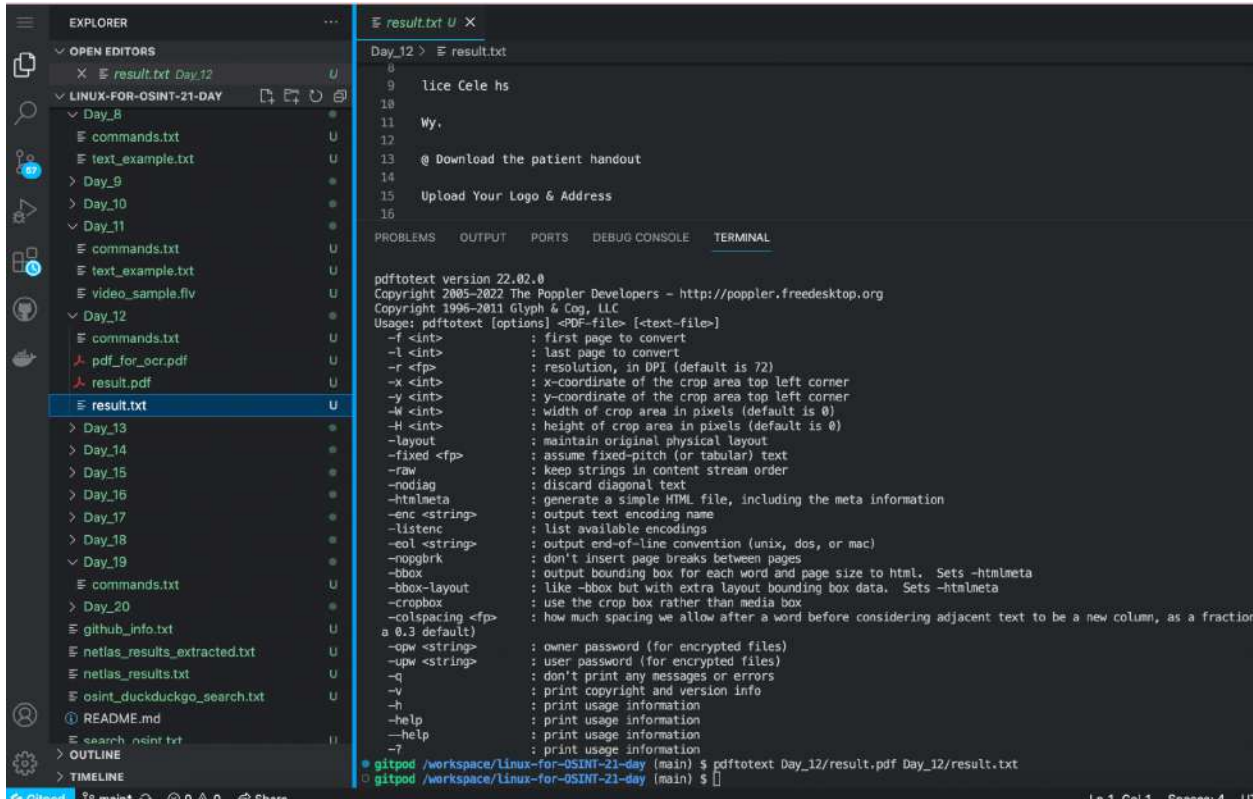
Now let's try to extract the text from the PDF file and save it to a separate file. We will use **Pdftotext** command, part of poppler-utils (<https://popper.js.org/docs/v2/utills/>).

Extract text from PDF

```
PROBLEMS  OUTPUT  PORTS  DEBUG CONSOLE  TERMINAL
Get:27 http://archive.ubuntu.com/ubuntu jammy-backports/universe amd64 Packages [28.1 kB]
Get:25 https://packagecloud.io/github/git-lfs/ubuntu jammy/main amd64 Packages [1,654 B]
Get:19 https://apt.llvm.org/jammy llvm-toolchain-jammy-17 InRelease [6,833 B]
Get:28 https://apt.llvm.org/jammy llvm-toolchain-jammy-17/main amd64 Packages [12.4 kB]
Fetched 9,396 kB in 3s (3,677 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
85 packages can be upgraded. Run 'apt list --upgradable' to see them.
● gitpod /workspace/linux-for-OSINT-21-day (main) $ sudo apt-get install poppler-utils --fix-missing
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libpoppler118
The following NEW packages will be installed:
  libpoppler118 poppler-utils
0 upgraded, 2 newly installed, 0 to remove and 85 not upgraded.
Need to get 1,258 kB of archives.
After this operation, 4,307 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 libpoppler118 amd64 22.02.0-2ubuntu0.3 [1,072 kB]
Get:2 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 poppler-utils amd64 22.02.0-2ubuntu0.3 [186 kB]
Fetched 1,258 kB in 1s (1,533 kB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package libpoppler118:amd64.
(Reading database ... 36155 files and directories currently installed.)
Preparing to unpack .../libpoppler118_22.02.0-2ubuntu0.3_amd64.deb ...
Unpacking libpoppler118:amd64 (22.02.0-2ubuntu0.3) ...
Selecting previously unselected package poppler-utils.
Preparing to unpack .../poppler-utils_22.02.0-2ubuntu0.3_amd64.deb ...
Unpacking poppler-utils (22.02.0-2ubuntu0.3) ...
Setting up libpoppler118:amd64 (22.02.0-2ubuntu0.3) ...
Setting up poppler-utils (22.02.0-2ubuntu0.3) ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for libc-bin (2.35-0ubuntu3.4) ...
```

First, Let's install popper-utils:

```
sudo apt clean
sudo apt autoclean
sudo apt update
sudo apt-get install poppler-utils --fix-missing
```

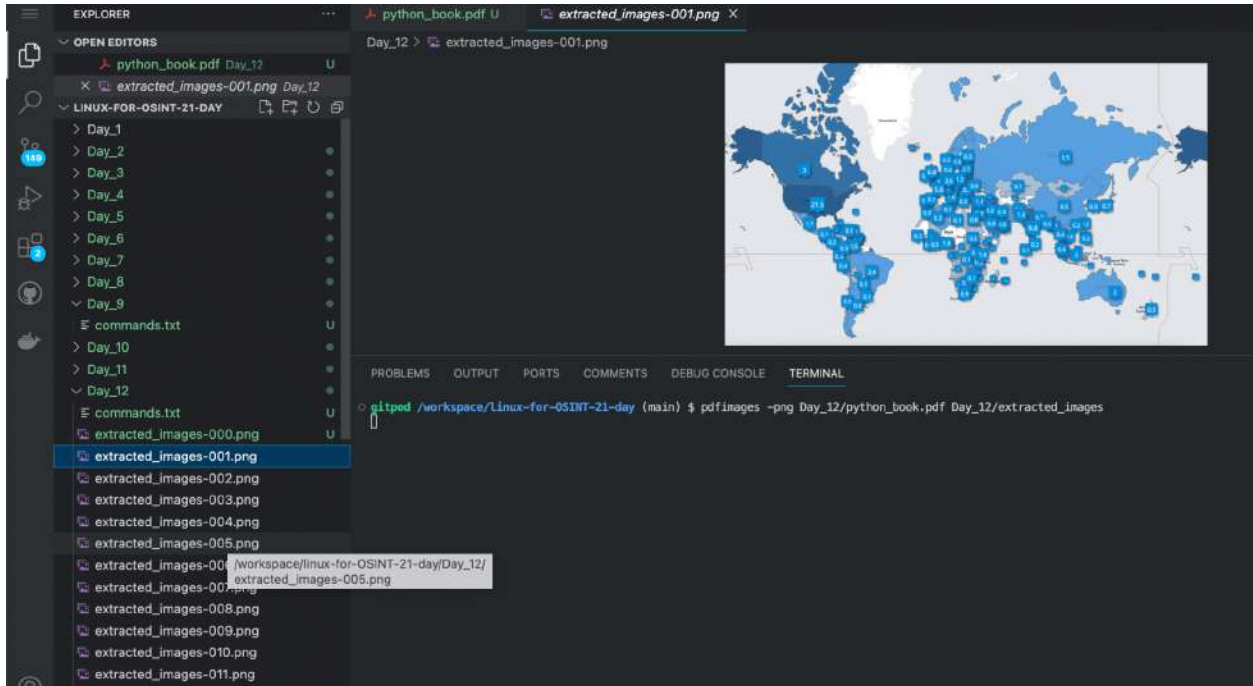


**pdftotext result.pdf result.txt**

You can now return to **Day 6** and **Day 7** to use the commands described there to work with text extracted from PDF files. For some people, it may be more convenient to skip text extraction and search directly in the PDF using PDFGrep (<https://pdfgrep.org/>) utility.

## Extract images from PDF

Also **PopperUtils** include **Pdfimages**, which allows you to extract images from PDF documents. Unfortunately, it works far from perfect, but let's give it a try:



**pdftimages -png python\_book.pdf extracted\_images**

Metadata analyze

```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
● gitpod /workspace/Linux-for-OSINT-21-day (main) $ sudo apt-get update
sudo apt-get install -y libimage-exiftool-perl
Hit:1 https://download.docker.com/linux/ubuntu jammy InRelease
Hit:2 https://ppa.launchpadcontent.net/git-core/ppa/ubuntu jammy InRelease
Hit:3 https://ppa.launchpadcontent.net/ondrej/apache2/ubuntu jammy InRelease
Hit:4 https://ppa.launchpadcontent.net/ondrej/nginx-mainline/ubuntu jammy InRelease
Hit:5 https://ppa.launchpadcontent.net/ondrej/php/ubuntu jammy InRelease
Hit:6 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:7 http://archive.ubuntu.com/ubuntu jammy InRelease
Hit:8 http://archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:10 http://archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:9 https://apt.llvm.org/jammy llvm-toolchain-jammy-17 InRelease
Hit:11 https://packagecloud.io/github/git-lfs/ubuntu jammy InRelease
Reading package lists... Done
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libarchive-zip-perl libmime-charset-perl libsombok3 libunicode-linebreak-perl
Suggested packages:
  libposix-strptime-perl libencode-hanextra-perl libpod2-base-perl
The following NEW packages will be installed:
  libarchive-zip-perl libimage-exiftool-perl libmime-charset-perl libsombok3 libunicode-linebreak-perl
0 upgraded, 5 newly installed, 0 to remove and 85 not upgraded.
Need to get 3,964 kB of archives.
After this operation, 23.5 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy/main amd64 libarchive-zip-perl all 1.68-1 [90.2 kB]
```

ExifTool (<https://exiftool.org/>) is a Perl library for reading, editing and deleting metadata. According to the list on the official website, it supports 209 file formats!

Let's install:

```
sudo apt-get update
sudo apt-get install -y libimage-exiftool-perl
```

```
PROBLEMS  OUTPUT  PORTS  DEBUG CONSOLE  TERMINAL
● gitpod /workspace/linux-for-OSINT-21-day (main) $ exiftool Day_12/result.pdf
ExifTool Version Number      : 12.40
File Name                    : result.pdf
Directory                   : Day_12
File Size                   : 1112 KiB
File Modification Date/Time  : 2023:12:17 13:10:51+00:00
File Access Date/Time       : 2023:12:17 17:31:59+00:00
File Inode Change Date/Time  : 2023:12:17 17:31:59+00:00
File Permissions             : -rw-r--r--
File Type                   : PDF
File Type Extension         : pdf
MIME Type                   : application/pdf
PDF Version                 : 1.7
Linearized                  : Yes
Author                      :
Create Date                 : 2016:12:22 11:43:55-05:00
Modify Date                 : 2023:12:17 13:10:49+00:00
Language                   : en
XMP Toolkit                 : XMP toolkit 2.9.1-13, framework 1.6
Producer                   : pikepdf 8.10.1
Creator Tool                : ocrmypdf 15.4.4 / Tesseract OCR-PDF 4.1.1
Document ID                 : uuid:3f4126da-d4fa-11f9-0000-4575653f8ad2
Format                     : application/pdf
Part                       : 2
Conformance                 : B
Creator                    :
Metadata Date               : 2023:12:17 13:10:49.574827+00:00
Page Count                  : 1
○ gitpod /workspace/linux-for-OSINT-21-day (main) $ █
```

And run :

**exiftool result.pdf**

If you only need certain metadata fields, you can filter them using other Linux utilities such as Grep:

```
PROBLEMS  OUTPUT  PORTS  COMMENTS  DEBUG CONSOLE  TERMINAL
● gitpod /workspace/linux-for-OSINT-21-day (main) $ exiftool Day_12/result.pdf | grep ^Producer
Producer                    : pikepdf 8.10.1
○ gitpod /workspace/linux-for-OSINT-21-day (main) $ █
```

**exiftool result.pdf | grep ^Producer**

You can also try many other tools to automate work with files (PDF and others formats) metadata:

Metadetective (<https://github.com/franckferman/MetaDetective>)

ExifLooter (<https://github.com/aydinyunus/exifLooter>)

MetaFinder (<https://github.com/Josue87/MetaFinder>)

PyMeta (<https://github.com/m8sec/pymeta>)

Hachoir (<https://github.com/vstinner/hachoir>)

# Day 13. MS Office files

LibreOffice (<https://www.libreoffice.org>) is an open source analog of Microsoft Office. Yes, it is far from perfect and lacks some features, but still it allows you to work very successfully with text documents, spreadsheets, presentations and databases.

And all this can be done not only with the GUI, but also with the command line.

```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
gitpod /workspace/linux-for-OSINT-21-day (main) $ sudo apt install libreoffice
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
adwaita-icon-theme apparmor at-spi2-core ca-certificates-java dbus-user-session dconf-gsettings-backend dconf-service default-jre default-jre-headless
dictionaries-common firebird3.0-common firebird3.0-common-doc firebird3.0-server-core firebird3.0-utils fonts-crosextra-caladea fonts-crosextra-carlito
fonts-dejavu fonts-dejavu-extra fonts-liberation fonts-liberation2 fonts-linuxlibertine fonts-noto-core fonts-noto-extra fonts-noto-mono fonts-noto-ui-core
fonts-opensymbol fonts-sil-gentium fonts-sil-gentium-basic gsettings-desktop-schemas gstreamer1.0-gi gstreamer1.0-gtk3 gstreamer1.0-plugins-base
gtk-update-icon-cache humanity-icon-theme hunspell-en-us java-common libabs120210324 libabw-0.1-1 libatk-bridge2.0-0 libatk-wrapper-java
libatk-wrapper-java-jni libatk1.0-0 libatk1.0-data libatspi2.0-0 libauthen-sasl-perl libavahi-client3 libavahi-common-data libavahi-common3
libboost-fsystem1.74.0 libboost-iostreams1.74.0 libboost-locale1.74.0 libboost-thread1.74.0 libbsh-java libcdparanoia0 libcdr-0.1-1 libclone-perl
libclucene-contrib1v5 libclucene-core1v5 libcolamd2 libcolord2 libcups2 libdata-dump-perl libdconf1 libe-book-0.1-1 libegl-mesa0 libegl1 libel-api-java
libencode-locale-perl libeot0 libepoxy0 libepubgen-0.1-1 libetonyek-0.1-1 libexttextcat-2.0-0 libexttextcat-data libfbclient2 libfile-basedir-perl
libfile-desktopentry-perl libfile-listing-perl libfile-mimeinfo-perl libfont-afm-perl libfontenc1 libfreehand-0.1-1 libgpgme11 libgpgmepp6 libgraphene-1.0-0
libgstreamer-gli-0.0 libgstreamer-plugins-base1.0-0 libgtk-3-0 libgtk-3-bin libgtk-3-common libgtk3-0 libgudev-1.0-0 libharfbuzz-icu0 libhsqldb1.8.0-java
libhtml-form-perl libhtml-format-perl libhtml-parser-perl libhtml-tagset-perl libhtml-tree-perl libhttp-cookies-perl libhttp-daemon-perl libhttp-date-perl
libhttp-message-perl libhttp-negotiate-perl libhunspell-1.7-0 libhyphen0 libib-util libio-html-perl libio-socket-ssl-perl libio-stringy-perl
libipc-system-simple-perl libjsp-api-java liblangtag-common liblangtag1 liblibreoffice-java liblvm11 liblwp-mediatypes-perl liblwp-protocol-https-perl
libmailtools-perl libmhash2 libmspub-0.1-1 libmwaw-0.3-3 libmythes-1.2-0 libnet-dbus-perl libnet-http-perl libnet-smtp-ssl-perl libnet-ssleay-perl libnspr4
libnss3 libodfgen-0.1-1 liborc-0.4-0 liborcus-0.17-0 liborcus-parser-0.17-0 libpagemaker-0.0-0 libpaper-utils libpaper1 libpcsc-lite1 libphobos2-ldc-shared98
libpoppler118 libraptor2-0 librasqal3 librdf0 libreoffice-base libreoffice-base-core libreoffice-base-drivers libreoffice-calc libreoffice-common
libreoffice-core libreoffice-draw libreoffice-gnome libreoffice-gtk3 libreoffice-impress libreoffice-java-common libreoffice-math libreoffice-nlpsolver
libreoffice-report-builder libreoffice-report-builder-bin libreoffice-script-provider-bsh libreoffice-script-provider-js libreoffice-script-provider-python
libreoffice-sdbc-firebird libreoffice-sdbc-hsqldb libreoffice-sdbc-mysql libreoffice-sdbc-postgresql libreoffice-style-colibre libreoffice-style-elementary
libpoppler118 libraptor2-0 librasqal3 librdf0 libreoffice-base libreoffice-base-core libreoffice-base-drivers libreoffice-calc libreoffice-common
libreoffice-core libreoffice-draw libreoffice-gnome libreoffice-gtk3 libreoffice-impress libreoffice-java-common libreoffice-math libreoffice-nlpsolver
libreoffice-report-builder libreoffice-report-builder-bin libreoffice-script-provider-bsh libreoffice-script-provider-js libreoffice-script-provider-python
libreoffice-sdbc-firebird libreoffice-sdbc-hsqldb libreoffice-sdbc-mysql libreoffice-sdbc-postgresql libreoffice-style-colibre libreoffice-style-elementary
libtext-iconv-perl libtie-ixhash-perl libtimedate-perl libtommath1 libtry-tiny-perl libuno-cppu3 libuno-cppuhelpergcc3-3 libuno-purpnhelpergcc3-3
libuno-sal3 libuno-salhelpergcc3-3 libuno-loader-java liburi-perl libvisio-0.1-1 libvisual-0.4-0 libvte-2.91-0 libvte-2.91-common libvte3-0
libwebsocket-api-java libwpd-0.10-10 libwpg-0.3-3 libwps-0.4-4 libwww-perl libwww-robotrules-perl libx11-protocol-perl libxaw7 libxcomposite1 libxdamage1
libxft2 libxkbfile1 libxml-parser-perl libxml-twig-perl libxml-xpathengine-perl libxmlsec libxmlsec1-nss libxmu6 libxmuu1 libxpm4 libxtst6 libxxf86dga1
libyajl2 lp-solve openjdk-11-jre openjdk-11-jre-headless perl-openssl-defaults poppler-data python3-uno session-migration tilix tilix-common ubuntu-mono
uno-libs-private ure-ure-java x11-utils x11-xserver-utils xdg-utils
Suggested packages:
apparmor-profiles-extra apparmor-utils ispell | aspell | hunspell wordlist firebird3.0-server firebird3.0-doc gvfs hunspell openoffice.org-hunspell
| openoffice.org-core libdigest-hmac-perl libgssapi-perl colord cups-common libvisual-0.4-plugins java-virtual-machine libhsqldb1.8.0-java-gcj
libcrypt-ssleay-perl pcsd raptor2-utils rasqal-utils librdf-storage-postgresql librdf-storage-mysql librdf-storage-sqlite librdf-storage-virtuoso
redland-utils cups-bsd firefox | firefox-esr | thunderbird ghostscript gpa hyphen-hyphenation-patterns libreoffice-grammarcheck libreoffice-help
```

Let's install:

```
sudo apt install libreoffice
```

```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
● gitpod /workspace/linux-for-OSINT-21-day (main) $ libreoffice -h
LibreOffice 7.3.7.2 30(Build:2)

Usage: soffice [argument...]
        argument - switches, switch parameters and document URIs (filenames).

Using without special arguments:
Opens the start center, if it is used without any arguments.
{file}      Tries to open the file (files) in the components
            suitable for them.
{file} {macro:///Library.Module.MacroName}
            Opens the file and runs specified macros from
            the file.

Getting help and information:
--help | -h | -?  Shows this help and quits.
--helpwriter      Opens built-in or online Help on Writer.
--helpcalc        Opens built-in or online Help on Calc.
--helpdraw        Opens built-in or online Help on Draw.
--helpimpress     Opens built-in or online Help on Impress.
--helpbase        Opens built-in or online Help on Base.
--helpbasic       Opens built-in or online Help on Basic scripting
                language.
--helpmath        Opens built-in or online Help on Math.
--version         Shows the version and quits.
--nstemporarydirectory
                (MacOS X sandbox only) Returns path of the temporary
                directory for the current user and exits. Overrides
                all other arguments.

General arguments:
--quickstart[=no] Activates[Deactivates] the Quickstarter service.
--nolockcheck      Disables check for remote instances using one
                installation.
--infilter={filter} Force an input filter type if possible. For example:
                --infilter="Calc Office Open XML"
                --infilter="Text (encoded):UTF8,LF,,,"
```

Run the help command to verify that the installation is correct. You will appreciate the huge number of functions this utility has.


## libreoffice -h

```
example.docx U example.txt U X
Day_13 > example.txt
1 Sample Document
2 This document was created using accessibility techniques for headings, lists, image alternate text, tables, and columns. It should be completely accessi
3 Headings
4 There are eight section headings in this document. At the beginning, "Sample Document" is a level 1 heading. The main section headings, such as "Heading
5 Lists
6 The following outline of the sections of this document is an ordered (numbered) list with six items. The fifth item, "Tables," contains a nested unorder
7 1. Headings
8 2. Lists

PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
● gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_13
● gitpod /workspace/linux-for-OSINT-21-day/Day_13 (main) $ libreoffice --headless --convert-to txt example.docx
convert /workspace/linux-for-OSINT-21-day/Day_13/example.docx -> /workspace/linux-for-OSINT-21-day/Day_13/example.txt using filter : Text
Overwriting: /workspace/linux-for-OSINT-21-day/Day_13/example.txt
● gitpod /workspace/linux-for-OSINT-21-day/Day_13 (main) $
```

Now let's convert the docx file to a txt file:

## libreoffice --headless --convert-to txt example.docx



```
example.docx U  example.xlsx U  example.csv U X
Day_13 > example.csv
1 Title,Identifier,Date,Subject,Description,Notes,Creator,Accession,Accession No,Reproduction
2 My Title,2016.1.1,6/16/16,Farm subsidies,This is a sample description,some notes,John Doe,Gift of So and So,2016.1,orig
3 Another title,2016.1.2,January 4 1965,The Who; British Invasion; AM radio,Eloquently written curatorial description,notes for internal use,John
4 The best image ever,2015.4.5,12/1/93,Politics; Memes; Cats,,Mary Maryson,Some other Gift,2015.4,dontknow
5

PROBLEMS  OUTPUT  PORTS  DEBUG CONSOLE  TERMINAL
● gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_13
● gitpod /workspace/linux-for-OSINT-21-day/Day_13 (main) $ libreoffice --headless --convert-to txt example.docx
convert /workspace/linux-for-OSINT-21-day/Day_13/example.docx -> /workspace/linux-for-OSINT-21-day/Day_13/example.txt using filter : Text
Overwriting: /workspace/linux-for-OSINT-21-day/Day_13/example.txt
● gitpod /workspace/linux-for-OSINT-21-day/Day_13 (main) $ libreoffice --headless --convert-to csv example.xlsx
convert /workspace/linux-for-OSINT-21-day/Day_13/example.xlsx -> /workspace/linux-for-OSINT-21-day/Day_13/example.csv using filter : Text - txt - csv (StarCalc)
○ gitpod /workspace/linux-for-OSINT-21-day/Day_13 (main) $
```

## libreoffice --headless --convert-to csv example.xlsx

Libre Office includes applications for working with a wide variety of file formats. For example:

- swf, gif
- bmp, jpeg, png, tigg
- svg
- zml, html
- epub, fb2
- wks, wps, wdb
- vsd

And working with all these formats can most likely be automated using the command line. In addition, just as Microsoft allows you to automate documents using VBA or JavaScript, LibreOffice allows you to create automations in LibreOffice Basic (as well as Python).

You can read more about it here:

LibreOffice	Basic	Programming	Guide
<a href="https://wiki.documentfoundation.org/Documentation/BASIC_Guide">https://wiki.documentfoundation.org/Documentation/BASIC_Guide</a>			
LibreOffice	Python	Design	Guide
<a href="https://wiki.documentfoundation.org/Macros/Python_Design_Guide/en">https://wiki.documentfoundation.org/Macros/Python_Design_Guide/en</a>			

If your job involves creating more reporting documentation, it makes sense for you to read the links above in more detail. With the help of document templates and macros, you can simplify many tasks considerably. But there are other tools besides LibreOffice for this, such as Pandoc.

## Creating .docx files

Pandoc (<https://pandoc.org/MANUAL.html>) is a universal utility for analyzing, converting and generating text documents (web pages, ebooks and more).

```
-f FORMAT, -r FORMAT, --from=FORMAT, --read=FORMAT
```

Specify input format. *FORMAT* can be:

- `bibtex` (BibTeX bibliography)
- `biblatex` (BibLaTeX bibliography)
- `bits` (BITS XML, alias for `jats`)
- `commonmark` (CommonMark Markdown)
- `commonmark_x` (CommonMark Markdown with extensions)
- `creole` (Creole 1.0)
- `cs1json` (CSL JSON bibliography)
- `csv` (CSV table)
- `tsv` (TSV table)
- `docbook` (DocBook)
- `docx` (Word docx)
- `dokuwiki` (DokuWiki markup)
- `endnotexml` (EndNote XML bibliography)
- `epub` (EPUB)
- `fb2` (FictionBook2 e-book)
- `gfm` (GitHub-Flavored Markdown), or the deprecated and less accurate `markdown_github`; use `markdown_github` only if you need extensions not supported in `gfm`.
- `haddock` (Haddock markup)
- `html` (HTML)
- `ipynb` (Jupyter notebook)
- `jats` (JATS XML)
- `jira` (Jira/Confluence wiki markup)
- `json` (JSON version of native AST)
- `latex` (LaTeX)
- `markdown` (Pandoc's Markdown)
- `markdown_mmd` (MultiMarkdown)
- `markdown_phpextra` (PHP Markdown Extra)
- `markdown_strict` (original unextended Markdown)
- `mediawiki` (MediaWiki markup)
- `man` (roff man)
- `muse` (Muse)
- `native` (native Haskell)
- `odt` (ODT)
- `opml` (OPML)
- `org` (Emacs Org mode)
- `ris` (RIS bibliography)
- `rtf` (Rich Text Format)
- `rst` (reStructuredText)
- `t2t` (txt2tags)
- `textile` (Textile)
- `tikiwiki` (TikiWiki markup)
- `twiki` (TWiki markup)
- `typst` (typst)
- `vimwiki` (Vimwiki)
- the path of a custom Lua reader, see [Custom readers and writers](#) below

It can work with over 40 different file formats!

Let's install:

```
PROBLEMS OUTPUT PORTS COMMENTS DEBUG CONSOLE TERMINAL bash - Day_13
gitpod /workspace/linux-for-05INT-21-day/Day_13 (main) $ sudo apt install pandoc
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libcmark-gfm-extensions0.29.0.gfm.3 libcmark-gfm0.29.0.gfm.3 pandoc-data
Suggested packages:
  texlive-latex-recommended texlive-xetex texlive-luatex pandoc-citeproc texlive-latex-extra context wkhtmltopdf librsvg2-bin groff ghc nodejs php python ruby
  libjs-mathjax libjs-katex citation-style-language-styles
The following NEW packages will be installed:
  libcmark-gfm-extensions0.29.0.gfm.3 libcmark-gfm0.29.0.gfm.3 pandoc pandoc-data
0 upgraded, 4 newly installed, 0 to remove and 0 not upgraded.
Need to get 20.6 MB of archives.
After this operation, 156 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libcmark-gfm0.29.0.gfm.3 amd64 0.29.0.gfm.3-3 [115 kB]
Get:2 http://archive.ubuntu.com/ubuntu jammy/universe amd64 libcmark-gfm-extensions0.29.0.gfm.3 amd64 0.29.0.gfm.3-3 [25.1 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy/universe amd64 pandoc-data all 2.9.2.1-3ubuntu2 [81.8 kB]
Get:4 http://archive.ubuntu.com/ubuntu jammy/universe amd64 pandoc amd64 2.9.2.1-3ubuntu2 [20.3 MB]
Fetched 20.6 MB in 1s (14.4 MB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package libcmark-gfm0.29.0.gfm.3:amd64.
(Reading database ... 35595 files and directories currently installed.)
Preparing to unpack .../libcmark-gfm0.29.0.gfm.3_0.29.0.gfm.3-3_amd64.deb ...
Unpacking libcmark-gfm0.29.0.gfm.3:amd64 (0.29.0.gfm.3-3) ...
Selecting previously unselected package libcmark-gfm-extensions0.29.0.gfm.3:amd64.
Preparing to unpack .../libcmark-gfm-extensions0.29.0.gfm.3_0.29.0.gfm.3-3_amd64.deb ...
Unpacking libcmark-gfm-extensions0.29.0.gfm.3:amd64 (0.29.0.gfm.3-3) ...
Selecting previously unselected package pandoc-data.
Preparing to unpack .../pandoc-data_2.9.2.1-3ubuntu2_all.deb ...
Unpacking pandoc-data (2.9.2.1-3ubuntu2) ...
Selecting previously unselected package pandoc.
Preparing to unpack .../pandoc_2.9.2.1-3ubuntu2_amd64.deb ...
Unpacking pandoc (2.9.2.1-3ubuntu2) ...
Setting up libcmark-gfm0.29.0.gfm.3:amd64 (0.29.0.gfm.3-3) ...
Setting up libcmark-gfm-extensions0.29.0.gfm.3:amd64 (0.29.0.gfm.3-3) ...
Setting up pandoc-data (2.9.2.1-3ubuntu2) ...
Setting up pandoc (2.9.2.1-3ubuntu2) ...
```

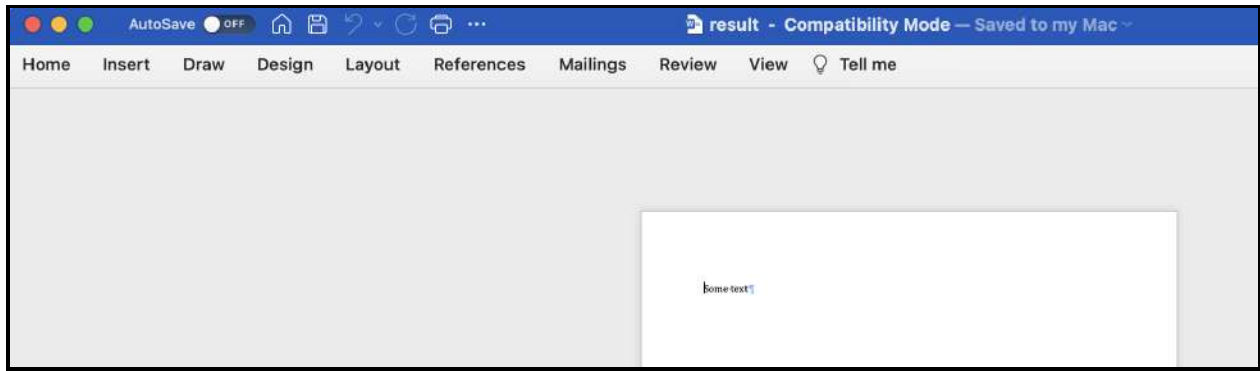
**sudo apt install pandoc**

Now let's write some text into a docx file:

```
> Day_8
> Day_9
> Day_10
> Day_11
> Day_12
v Day_13
  commands.txt U
  example.csv U
  example.docx U
  example.ppt U
  example.txt U
  example.xlsx U
  result.docx U
> Day_14
> Day_15

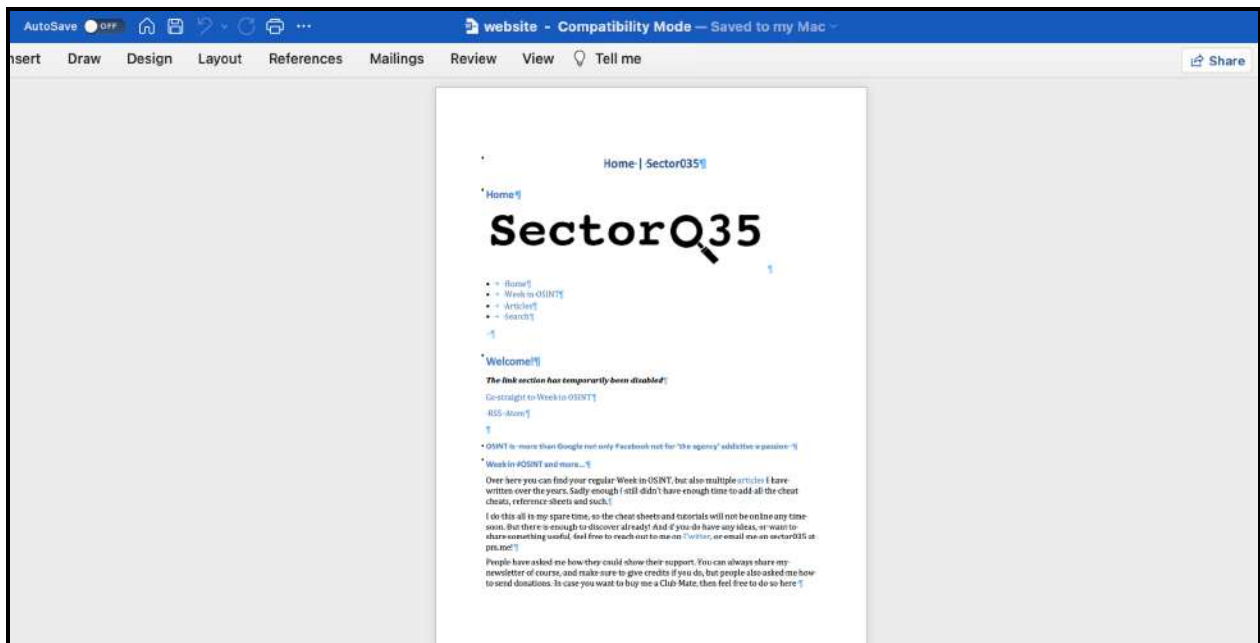
PROBLEMS OUTPUT PORTS COMMENTS DEBUG CONSOLE TERMINAL
gitpod /workspace/linux-for-05INT-21-day (main) $ cd Day_13
gitpod /workspace/linux-for-05INT-21-day/Day_13 (main) $ echo "Some text" | pandoc -o result.docx
gitpod /workspace/linux-for-05INT-21-day/Day_13 (main) $ []
```

**echo "Some text" | pandoc -o result.docx**



Similarly, you can save the result of any command to docx.

Another interesting thing is that Pandoc can directly download files from the internet:



```
pandoc -f html -t docx https://sector035.nl -o website.docx
```

The above shows the simplest examples, but Pandoc also allows you to edit document formatting and structure, read metadata, add tables to documents, and more. Like LibreOffice, it is an incredibly versatile tool.

## Day 14. JSON, XML, CSV

Today we'll talk about the three popular file formats, which are used in various REST-APIs (you can read more about how to make http requests in Day 5) and as formats for exporting/importing data in various tools.

## JSON

```
{ } json_test.json U x
Day_14 > {} json_test.json > ...
1   {
2     "people" : [
3       {
4         "firstName": "Joe",
5         "lastName": "Jackson",
6         "gender": "male",
7         "age": 28,
8         "number": "7349282382"
9       },
10      {
11        "firstName": "James",
12        "lastName": "Smith",
13        "gender": "male",
14        "age": 32,
15        "number": "5678568567"
16      },
17      {
18        "firstName": "Emily",
19        "lastName": "Jones",
20        "gender": "female",
21        "age": 24,
22        "number": "456754675"
23      }
24    ]
25  }
```

JSON is format for representing structured data based on JavaScript object syntax. That is, data in it are described in the form of objects (example on the picture - **people**), each of which has a set of certain properties (example on the picture - **firstName**, **lastName**, **gender**) and each property can have a certain value (example on the picture - "**Joe**", "**Jackson**", "**male**").

JQ (<https://jqlang.github.io/jq/>) is one of most popular command line JSON processor. It is installed by default in most Linux distributions.

```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
● gitpod /workspace/linux-for-OSINT-21-day/Day_13 (main) $ jq -h
jq - commandline JSON processor [version 1.6]

Usage: jq [options] <jq filter> [file...]
      jq [options] --args <jq filter> [strings...]
      jq [options] --jsonargs <jq filter> [JSON_TEXTS...]

jq is a tool for processing JSON inputs, applying the given filter to
its JSON text inputs and producing the filter's results as JSON on
standard output.

The simplest filter is ., which copies jq's input to its output
unmodified (except for formatting, but note that IEEE754 is used
for number representation internally, with all that that implies).

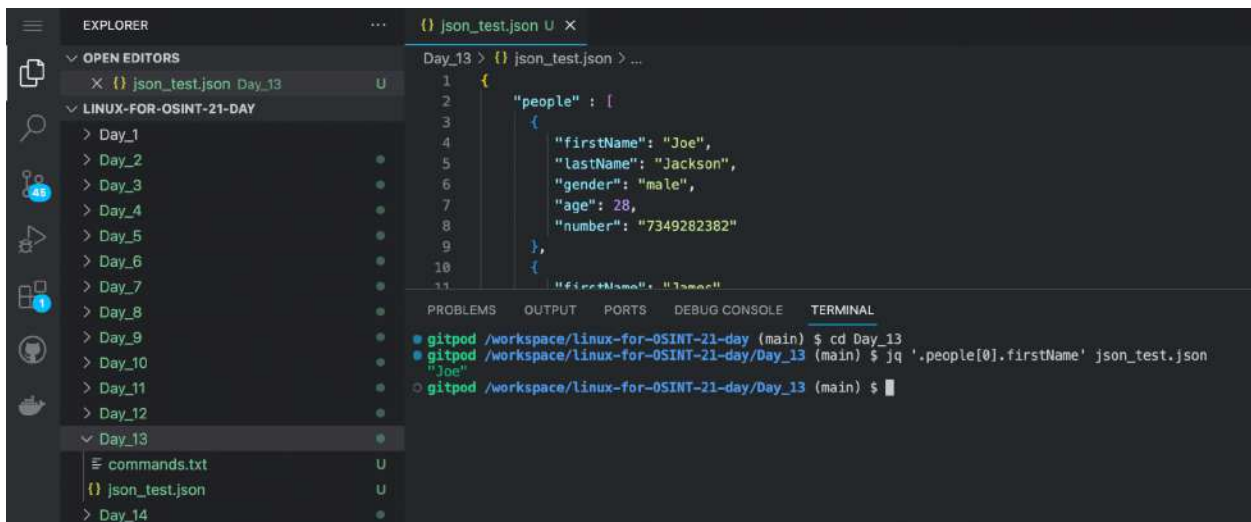
For more advanced filters see the jq(1) manpage ("man jq")
and/or https://stedolan.github.io/jq

Example:

    $ echo '{"foo": 0}' | jq .
    {
      "foo": 0
    }

Some of the options include:
-c          compact instead of pretty-printed output;
-n          use `null` as the single input value;
-e          set the exit status code based on the output;
-s          read (slurp) all inputs into an array; apply filter to it;
-r          output raw strings, not JSON texts;
-R          read raw strings, not JSON texts;
-C          colorize JSON;
-M          monochrome (don't colorize JSON);
-S          sort keys of objects on output;
--tab      use tabs for indentation;
--arg a v  set variable $a to value <v>;
```

Let's try to extract the firstName value of the first people object (key):



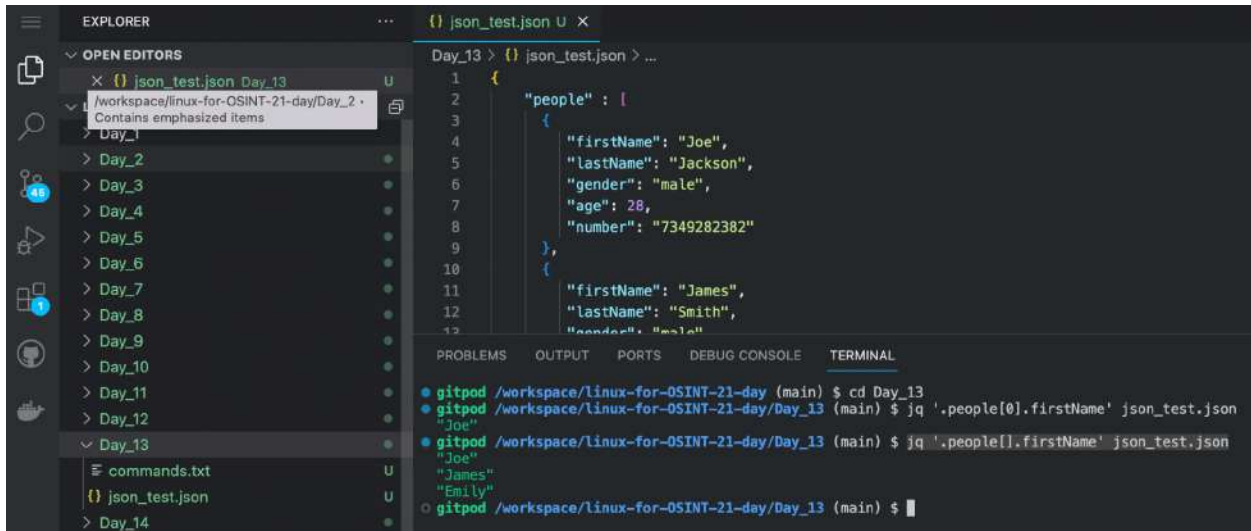
The screenshot shows the VS Code interface. The Explorer sidebar on the left shows a project structure with folders 'Day\_1' through 'Day\_14' and files 'commands.txt' and 'json\_test.json'. The main editor window displays the content of 'json\_test.json', which is a JSON object with a 'people' array containing two objects. The first object has 'firstName': 'Joe', 'lastName': 'Jackson', 'gender': 'male', 'age': 28, and 'number': '7349282382'. The second object has 'firstName': 'James'. The terminal at the bottom shows the following commands and output:

```
gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_13
gitpod /workspace/linux-for-OSINT-21-day/Day_13 (main) $ jq '.people[0].firstName' json_test.json
"Joe"
gitpod /workspace/linux-for-OSINT-21-day/Day_13 (main) $
```

```
cd Day_14
```

```
jq '.people[0].firstName' json_test.json
```

Now let's display the values of the firstName for all "people" keys:



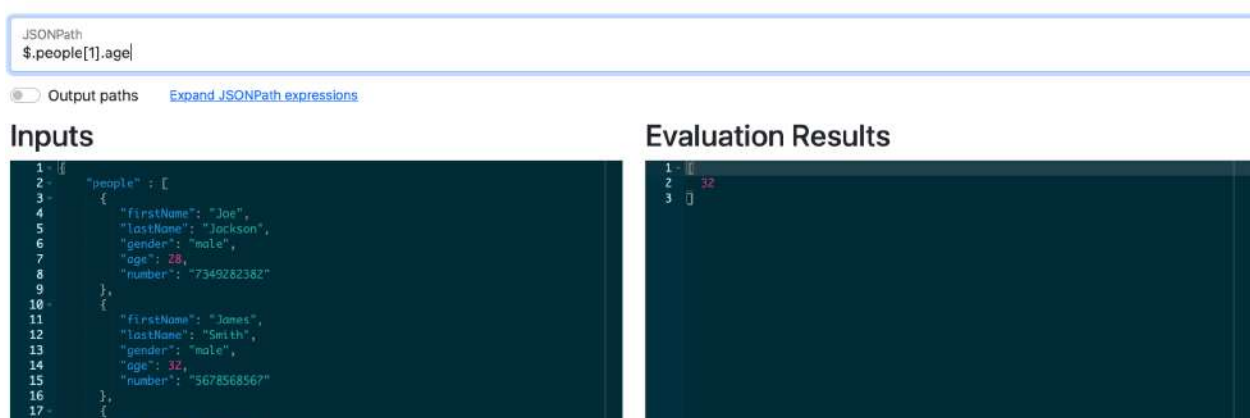
The screenshot shows a VS Code editor with a file explorer on the left and a terminal at the bottom. The file explorer shows a directory structure with files like 'json\_test.json' and 'Day\_1' through 'Day\_14'. The terminal shows the following commands and output:

```
gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_13
gitpod /workspace/linux-for-OSINT-21-day/Day_13 (main) $ jq '.people[0].firstName' json_test.json
"Joe"
gitpod /workspace/linux-for-OSINT-21-day/Day_13 (main) $ jq '.people[].firstName' json_test.json
"Joe"
"James"
"Emily"
gitpod /workspace/linux-for-OSINT-21-day/Day_13 (main) $
```

```
jq '.people[].firstName' json_test.json
```

Similarly, you can access any values of any properties for all keys.

JSONPath Online Evaluator - [jsonpath.com](https://jsonpath.com)



The screenshot shows the JSONPath Online Evaluator website. The input field contains the JSONPath expression `$.people[1].age`. The output field shows the result `32`. The website also has a section for "Inputs" and "Evaluation Results" with a code editor showing the JSON data and the resulting output.

We used a JSON file with a very simple structure as an example, but in the course of your investigation you may well have to deal with files whose structure will include dozens of different levels. Special online services such as **JSONPath Online Evaluator** (<https://jsonpath.com>).

## XML

```
xml_sample.xml U X
Day_14 > xml_sample.xml
1 <CATALOG>
2 <div id="in-page-channel-node-id" data-channel-name="in_page_channel_L8gdhe"/>
3 <CD>
4 <TITLE>Empire Burlesque</TITLE>
5 <ARTIST>Bob Dylan</ARTIST>
6 <COUNTRY>USA</COUNTRY>
7 <COMPANY>Columbia</COMPANY>
8 <PRICE>10.90</PRICE>
9 <YEAR>1985</YEAR>
10 </CD>
11 <CD>
12 <TITLE>Hide your heart</TITLE>
13 <ARTIST>Bonnie Tyler</ARTIST>
14 <COUNTRY>UK</COUNTRY>
15 <COMPANY>CBS Records</COMPANY>
16 <PRICE>9.90</PRICE>
17 <YEAR>1988</YEAR>
18 </CD>
19 <CD>
20 <TITLE>Greatest Hits</TITLE>
21 <ARTIST>Dolly Parton</ARTIST>
22 <COUNTRY>USA</COUNTRY>
23 <COMPANY>RCA</COMPANY>
24 <PRICE>9.90</PRICE>
25 <YEAR>1982</YEAR>
26 </CD>
```

XML (eXtensible Markup Language) is a markup language similar to HTML, but without predefined tags to use (the user can define their own tags).

XMLStarlet (<https://xmlstar.sourceforge.net/doc/UG/xmlstarlet-ug.html>) is a simple cross-platform utility for extracting data from XML files.

Let's install:

```
PROBLEMS  OUTPUT  PORTS  DEBUG CONSOLE  TERMINAL

● gitpod /workspace/linux-for-OSINT-21-day (main) $ sudo apt-get install xmlstarlet
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  xmlstarlet
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 265 kB of archives.
After this operation, 631 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy/universe amd64 xmlstarlet amd64 1.6.1-2.1 [265 kB]
Fetched 265 kB in 0s (3,216 kB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package xmlstarlet.
(Reading database ... 35595 files and directories currently installed.)
Preparing to unpack ../xmlstarlet_1.6.1-2.1_amd64.deb ...
Unpacking xmlstarlet (1.6.1-2.1) ...
Setting up xmlstarlet (1.6.1-2.1) ...
Processing triggers for man-db (2.10.2-1) ...
○ gitpod /workspace/linux-for-OSINT-21-day (main) $ █
```

**sudo apt-get install xmlstarlet**

Now let's display the contents of all <title> tags:

The screenshot shows a VS Code editor with an Explorer sidebar on the left and a main editor area on the right. The Explorer sidebar shows a project structure for 'LINUX-FOR-OSINT-21-DAY' with folders for 'Day\_1' through 'Day\_18'. The main editor area shows the content of 'xml\_sample.xml' with XML tags for CD elements. The terminal at the bottom shows the execution of the 'xmlstarlet' command.

```
155 <CD>
156 <TITLE>Private Dancer</TITLE>
157 <ARTIST>Tina Turner</ARTIST>
158 <COUNTRY>UK</COUNTRY>
159 <COMPANY>Capitol</COMPANY>
160 <PRICE>8.90</PRICE>
161 <YEAR>1983</YEAR>
162 </CD>
163 <CD>
164 <TITLE>Midt om natten</TITLE>
165 <ARTIST>Kim Larsen</ARTIST>
166 <COUNTRY>EU</COUNTRY>
167 <COMPANY>Medley</COMPANY>
168 <PRICE>7.00</PRICE>

gitpod /workspace/Linux-for-OSINT-21-day (main) $ cd Day_14
gitpod /workspace/Linux-for-OSINT-21-day/Day_14 (main) $ xmlstarlet sel -t -m "//TITLE" -v . -n xml_sample.xml
Empire Burlesque
Hide your heart
Greatest Hits
Still got the blues
Eros
One night only
Sylvias Mother
Maggie May
Romanza
When a man loves a woman
Black angel
1999 Grammy Nominees
For the good times
Big Willie style
Tupelo Honey
Soulsville
The very best of
Stop
Bridge of Spies
Private Dancer
Midt om natten
Pavarotti Gala Concert
The dock of the bay
Picture book
Red
Unchain my heart
gitpod /workspace/Linux-for-OSINT-21-day/Day_14 (main) $
```

**xmlstarlet sel -t -m "//TITLE" -v . -n xml\_sample.xml**

XML viewer interface showing an XPath query `//COUNTRY` and its results. The interface includes a search bar, a list of elements found (26), and a preview of the XML document structure.

```

//COUNTRY
Elements found: 26

XML, needs | Format | Save | Copy | Text | Node

<CATALOG>
  <div id="in-page-channel-node-id" data-channel-name="in_page_channel_L8gdhe"/>
  <CD>
    <TITLE>Empire Burlesque
  </TITLE>
  <ARTIST>Bob Dylan
  </ARTIST>
  <COUNTRY>USA
  </COUNTRY>
  <COMPANY>Columbia
  </COMPANY>
  <PRICE>10.90
  </PRICE>
  <YEAR>1985
  </YEAR>
  </CD>
  <CD>
    <TITLE>Hide your heart
  </TITLE>
  </CD>
  </CATALOG>

```

Results list:

- USA
- UK
- USA
- UK
- EU
- UK
- UK
- UK
- EU
- USA
- EU
- USA
- UK
- USA
- UK
- Norway
- UK
- UK
- UK

In order to find the path to specific data in an XML document you can use special online services such as XPather (<http://xpather.com>).

## CSV

If you happened to miss **Day 6** and **Day 7**, I suggest you get back to it. Knowledge of **Grep**, **Sed** and **Awk** is enough to work efficiently with CSV files. Some users prefer to replace grep with CSVGrep (<https://csvkit.readthedocs.io/en/1.0.2/scripts/csvgrep.html>). It's a little more convenient for searching specific columns.

Let's install CSVKit (<https://csvkit.readthedocs.io/en/1.0.2/index.html>):

```

PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
gitpod /workspace/linux-for-OSINT-21-day (main) $ sudo apt-get -y install csvkit
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libimagequant0 libraqm0 mailcap mime-support python-babel-localedata python3-agate python3-agateexcel python3-agateexcel python3-agateexcel python3-attr
  python3-babel python3-bs4 python3-chardet python3-csvkit python3-dbfread python3-et-xmlfile python3-greenlet python3-html5lib python3-icu python3-iniconfig
  python3-isodate python3-jdcal python3-leather python3-lxml python3-olefile python3-openpyxl python3-packaging python3-parsedatetime python3-pil python3-pluggy
  python3-py python3-pygments python3-pytest python3-pytimemachine python3-slugify python3-soupsieve python3-sqlalchemy python3-sqlalchemy-ext python3-toml
  python3-tz python3-unidecode python3-webcodings python3-xlrd
Suggested packages:
  csvkit-doc python3-agate-doc python3-agateexcel-doc python3-agateexcel-doc python3-agateexcel-doc python3-attr-doc python3-dbfread-doc python3-greenlet-dev
  python-greenlet-doc python3-genshi python-leather-doc python-lxml-doc python-pill-doc python-pygments-doc ttf-bitstream-vera python-sqlalchemy-doc python3-fdb
  python3-pymssql python3-mysqldb python3-psycopg2 python3-asyncpg python3-aiosqlite
The following NEW packages will be installed:
  csvkit libimagequant0 libraqm0 mailcap mime-support python-babel-localedata python3-agate python3-agateexcel python3-agateexcel python3-agateexcel python3-attr
  python3-babel python3-bs4 python3-chardet python3-csvkit python3-dbfread python3-et-xmlfile python3-greenlet python3-html5lib python3-icu python3-iniconfig
  python3-isodate python3-jdcal python3-leather python3-lxml python3-olefile python3-openpyxl python3-packaging python3-parsedatetime python3-pil python3-pluggy
  python3-py python3-pygments python3-pytest python3-pytimemachine python3-slugify python3-soupsieve python3-sqlalchemy python3-sqlalchemy-ext python3-toml
  python3-tz python3-unidecode python3-webcodings python3-xlrd
0 upgraded, 44 newly installed, 0 to remove and 0 not upgraded.
Need to get 10.5 MB of archives.
After this operation, 55.4 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy/main amd64 python-babel-localedata all 2.8.0+dfsg.1-7 [4,982 kB]
Get:2 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 python3-tz all 2022.1-1ubuntu0.22.04.1 [30.7 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy/main amd64 python3-babel all 2.8.0+dfsg.1-7 [85.1 kB]
Get:4 http://archive.ubuntu.com/ubuntu jammy/universe amd64 python3-isodate all 0.6.1-1 [24.7 kB]
Get:5 http://archive.ubuntu.com/ubuntu jammy/universe amd64 python3-leather all 0.3.4-1 [19.9 kB]
Get:6 http://archive.ubuntu.com/ubuntu jammy/universe amd64 python3-parsedatetime all 2.6-2 [32.9 kB]

```



```
Day_14 > test_json_to_csv.csv
1 firstName,lastName,gender,age,number
2 Joe,Jackson,male,28,7349282382
3 James,Smith,male,32,5678568567
4 Emily,Jones,female,24,456754675
5

gitpod /workspace/Linux-for-OSINT-21-day (main) $ cd Day_14
gitpod /workspace/Linux-for-OSINT-21-day/Day_14 (main) $ in2csv -k people json_test.json > test_json_to_csv.csv
gitpod /workspace/Linux-for-OSINT-21-day/Day_14 (main) $
```

## in2csv -k people json\_test.json > test\_json\_to\_csv.csv

Csvcut (<https://csvkit.readthedocs.io/en/latest/scripts/csvcut.html>) is a tool for filter and truncate CSV files.

```
Day_14 > cut_csv.csv
1 domain,port
2 "codingtutor.dev,www.pferdepension-wolfgrube.de,www.elite-generalbau.de,addavear.com,daus.online,dominik-peter
3 "codingtutor.dev,www.pferdepension-wolfgrube.de,www.elite-generalbau.de,addavear.com,daus.online,dominik-peter
4 "akp-communications.com,www.alaindelgado.fr,medizinisches-versorgungszentrum.care,stefanhorak.de,wolfs-cave.de,
5 "akp-communications.com,www.alaindelgado.fr,medizinisches-versorgungszentrum.care,stefanhorak.de,wolfs-cave.de,
6 "pactech.mx,product-compliance.net,anima-club.com,holzbau-fischer.com,chip-metropole.de,xn--historische-percke-
7 "ebonyrouse.livesex-secrets.com,arinella.whackcams.com,nicoleheard.livesex-secrets.com,annajensen.livesex-secr
8 "www.northernberkshireropes.com,northernberkshireropes.com,mailleman.com,thebearlovers.com,mail.dumud.net,dumud
9 "www.northernberkshireropes.com,northernberkshireropes.com,mailleman.com,thebearlovers.com,mail.dumud.net,dumud
10 "www.northernberkshireropes.com,northernberkshireropes.com,mailleman.com,thebearlovers.com,mail.dumud.net,dumud
11 "www.northernberkshireropes.com,northernberkshireropes.com,mailleman.com,thebearlovers.com,mail.dumud.net,dumud
12 "www.northernberkshireropes.com,northernberkshireropes.com,mailleman.com,thebearlovers.com,mail.dumud.net,dumud
13 "www.northernberkshireropes.com,northernberkshireropes.com,mailleman.com,thebearlovers.com,mail.dumud.net,dumud
14 "www.northernberkshireropes.com,northernberkshireropes.com,mailleman.com,thebearlovers.com,mail.dumud.net,dumud
15 "www.northernberkshireropes.com,northernberkshireropes.com,mailleman.com,thebearlovers.com,mail.dumud.net,dumud
16 "www.northernberkshireropes.com,northernberkshireropes.com,mailleman.com,thebearlovers.com,mail.dumud.net,dumud
17 "mateuspaduaweb.com.br,www.mateuspaduaweb.com.br",80

gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_14
gitpod /workspace/linux-for-OSINT-21-day/Day_14 (main) $ csvcut -c domain,port example.csv > cut_csv.csv
gitpod /workspace/linux-for-OSINT-21-day/Day_14 (main) $
```

```
csvcut -c domain,port example.csv > cut_csv.csv
```

It is also worth paying attention to:

**Csvjoin** (<https://csvkit.readthedocs.io/en/latest/scripts/csvjoin.html>)

**Csvsort** (<https://csvkit.readthedocs.io/en/latest/scripts/csvsort.html>)

**Csvstat** (<https://csvkit.readthedocs.io/en/latest/scripts/csvstat.html>).

When working with data in XML or JSON format, I recommend that you consider converting it to CSV. In many cases, this format is more convenient to understand and work in the command line (as mentioned above, you can easily do it with **in2csv**).

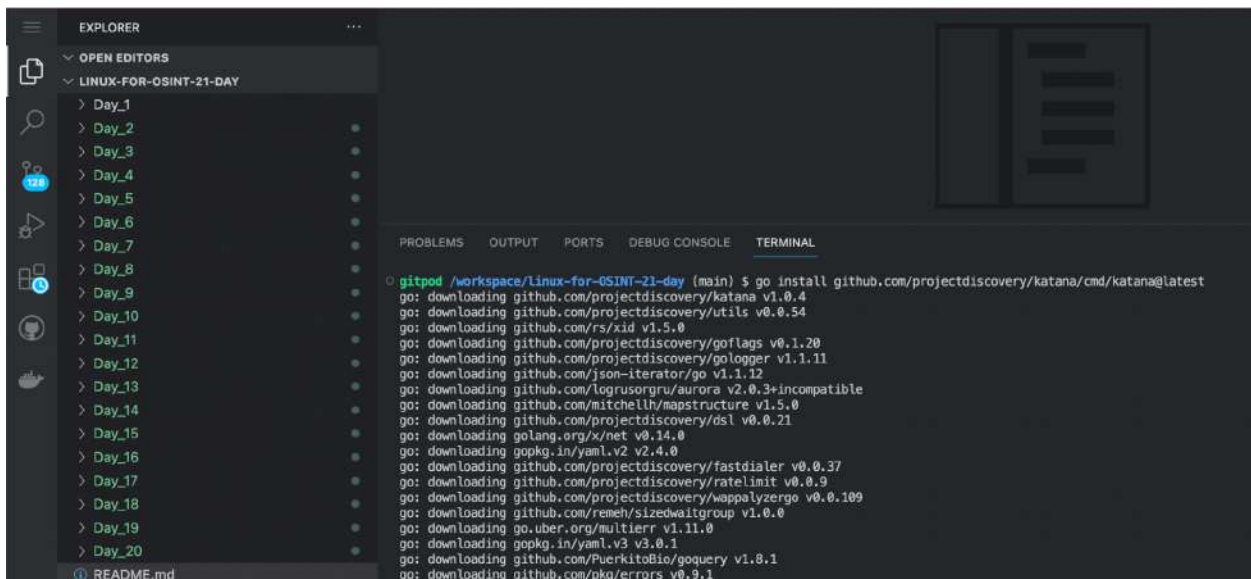
# Day 15. Scraping

Scraping is the process of automatically extracting different data from web pages. If you are good at using **curl**, **grep**, **sed** and **awk** mentioned in the previous days, you can already successfully scrape web pages on Linux. But it's worth exploring a few more tools to be more effective.

## Katana

First of all, let's learn how to make a list of URLs of all pages of the site (or rather, all pages that are linked from other pages of the site, starting with the main page).

This is very easy if you have the wonderful Katana (<https://github.com/projectdiscovery/katana>) tool from Project Discovery (<https://github.com/projectdiscovery>).



```
gitpod /workspace/Linux-for-OSINT-21-day (main) $ go install github.com/projectdiscovery/katana/cmd/katana@latest
go: downloading github.com/projectdiscovery/katana v1.0.4
go: downloading github.com/projectdiscovery/utils v0.0.54
go: downloading github.com/rs/xid v1.5.0
go: downloading github.com/projectdiscovery/goflags v0.1.20
go: downloading github.com/projectdiscovery/gologger v1.1.11
go: downloading github.com/json-iterator/go v1.1.12
go: downloading github.com/logrusorgru/aurora v2.0.3+incompatible
go: downloading github.com/mitchellh/mapstructure v1.5.0
go: downloading github.com/projectdiscovery/dsl v0.0.21
go: downloading golang.org/x/net v0.14.0
go: downloading gopkg.in/yaml.v2 v2.4.0
go: downloading github.com/projectdiscovery/fastdialer v0.0.37
go: downloading github.com/projectdiscovery/ratelimit v0.0.9
go: downloading github.com/projectdiscovery/wappalizergo v0.0.109
go: downloading github.com/remeh/sizedwaitgroup v1.0.0
go: downloading go.uber.org/multierr v1.11.0
go: downloading gopkg.in/yaml.v3 v3.0.1
go: downloading github.com/PuerkitoBio/goquery v1.8.1
go: downloading github.com/pkg/errors v0.9.1
```

Let's install:

```
go install github.com/projectdiscovery/katana/cmd/katana@latest
```

```

> Day_5
> Day_6
> Day_7
> Day_8
> Day_9
> Day_10
> Day_11
> Day_12
> Day_13
> Day_14
> Day_15
> Day_16
> Day_17
> Day_18
> Day_19
> Day_20
README.md
OUTLINE
TIMELINE

PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
gitpod /workspace/linux-for-OSINT-21-day (main) $ katana -u sector035.nl

projectdiscovery.io

[INF] Current katana version v1.0.4 (latest)
[INF] Started standard crawling for => https://sector035.nl
https://sector035.nl
https://sector035.nl/user/plugins/shortcode-ui/js/ui-atext.js
https://sector035.nl/user/themes/quark/js/jquery.treemenu.js
https://sector035.nl/user/plugins/shortcode-ui/css/ui-atext.css
https://sector035.nl/user/themes/quark/css/custom.css
https://sector035.nl/user/themes/quark-open-publishing/js/site.js
https://sector035.nl/user/themes/quark-open-publishing/js/my.js
https://sector035.nl/user/themes/quark/css/line-awesome.min.css
https://sector035.nl/user/plugins/login/css/login.css
https://sector035.nl/user/plugins/highlight/css/default.css
https://sector035.nl/user/plugins/youtube/css/youtube.css
https://sector035.nl/user/themes/quark/css-compiled/spectre.min.css
https://sector035.nl/user/themes/quark/css-compiled/theme.min.css
https://sector035.nl/user/plugins/tntsearch/assets/tntsearch.css
https://sector035.nl/user/plugins/markdown-notices/assets/notice.css
https://sector035.nl/user/plugins/external_links/assets/css/external_links.css
https://sector035.nl/user/plugins/image-captions/css/image-captions.css
https://sector035.nl/user/themes/quark-open-publishing/css/theme.css
https://sector035.nl/user/plugins/form/assets/form-styles.css
https://sector035.nl/user/plugins/highlight/js/highlight.pack.js
https://sector035.nl/system/assets/jquery/jquery-3.x.min.js
https://sector035.nl/user/themes/quark-open-publishing/js/...
https://sector035.nl/search
https://sector035.nl/articles.rss
https://sector035.nl/articles.atom

```

And run:

**katana -u sector035.nl**

```

EXPLORER
OPEN EDITORS
domains.txt Day_15
results.txt Day_15
LINUX-FOR-OSINT-21-DAY
Day_1
Day_2
Day_3
Day_4
Day_5
Day_6
Day_7
Day_8
Day_9
Day_10
Day_11
Day_12
Day_13
Day_14
Day_15
commands.txt
domains.txt
results.txt

domains.txt U
results.txt U X
Day_15 > results.txt
1 https://sector035.nl
2 https://www.osint-jobs.com/
3 https://sector035.nl/user/themes/quark-open-publishing/js/my.js
4 https://sector035.nl/user/themes/quark/css/custom.css
5 https://sector035.nl/user/themes/quark/js/jquery.treemenu.js
6 https://sector035.nl/user/themes/quark-open-publishing/js/site.js
7 https://sector035.nl/user/plugins/shortcode-ui/js/ui-atext.js
8 https://sector035.nl/user/plugins/highlight/css/default.css

PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_15
gitpod /workspace/linux-for-OSINT-21-day/Day_15 (main) $ katana -list domains.txt>results.txt

projectdiscovery.io

[INF] Current katana version v1.0.4 (latest)
[INF] Started standard crawling for => https://osintnewsletter.com/
[INF] Started standard crawling for => https://www.osint-jobs.com/
[INF] Started standard crawling for => https://sector035.nl

```

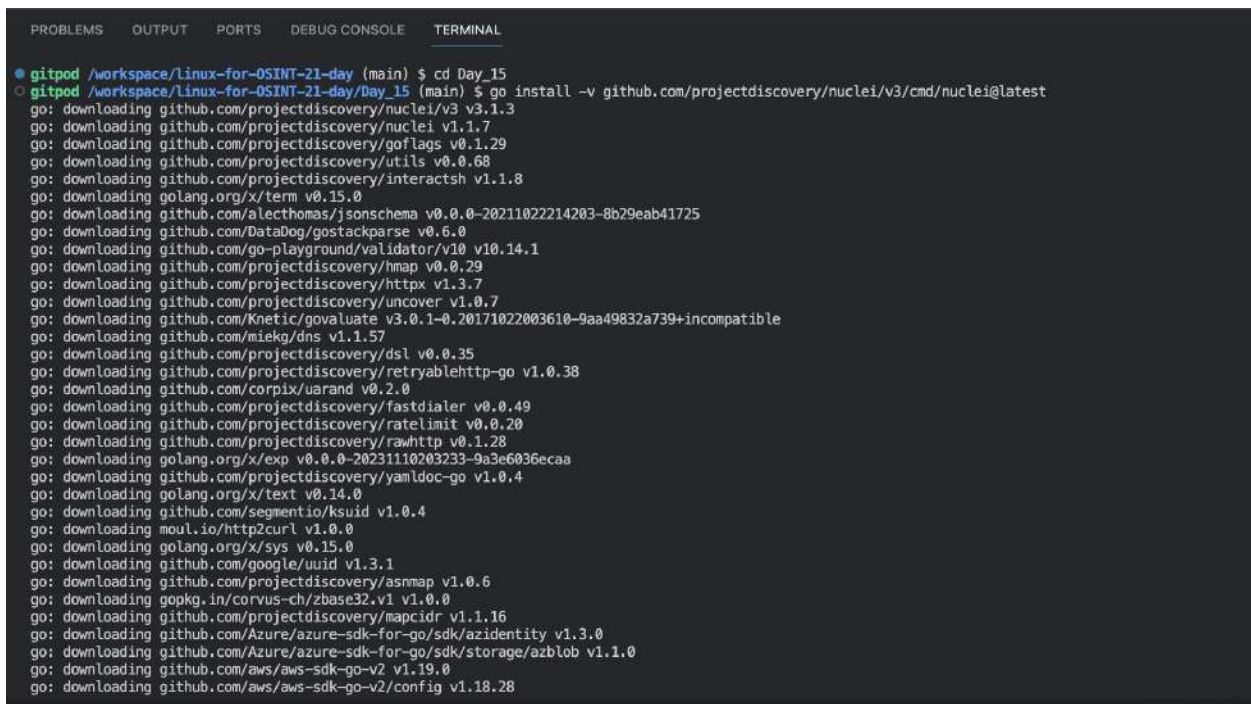
If you want to collect links from a whole list of domains, use the `-list` flag. And let's save the result this time, to a text file, so that we can use it when we run the next command:

```
cd Day_15
katana -list domains.txt>results.txt
```

## NUCLEI

Nuclei (<https://github.com/projectdiscovery/nuclei>) is another great tool from Project Discovery. It is primarily designed to scan sites for vulnerabilities, but it also does an excellent job of collecting various investigative data from web pages.

Let's install:



```
PROBLEMS  OUTPUT  PORTS  DEBUG CONSOLE  TERMINAL
gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_15
gitpod /workspace/linux-for-OSINT-21-day/Day_15 (main) $ go install -v github.com/projectdiscovery/nuclei/v3/cmd/nuclei@latest
go: downloading github.com/projectdiscovery/nuclei/v3 v3.1.3
go: downloading github.com/projectdiscovery/nuclei v1.1.7
go: downloading github.com/projectdiscovery/goflags v0.1.29
go: downloading github.com/projectdiscovery/utils v0.0.68
go: downloading github.com/projectdiscovery/interactsh v1.1.8
go: downloading golang.org/x/term v0.15.0
go: downloading github.com/alecthomas/jsonschema v0.0.0-20211022214203-8b29eab41725
go: downloading github.com/DataDog/gostackparse v0.6.0
go: downloading github.com/go-playground/validator/v10 v10.14.1
go: downloading github.com/projectdiscovery/hmap v0.0.29
go: downloading github.com/projectdiscovery/httpx v1.3.7
go: downloading github.com/projectdiscovery/uncover v1.0.7
go: downloading github.com/Knetic/govaluate v3.0.1-0.20171022003610-9aa49832a739+incompatible
go: downloading github.com/miekg/dns v1.1.57
go: downloading github.com/projectdiscovery/dsl v0.0.35
go: downloading github.com/projectdiscovery/retryablehttp-go v1.0.38
go: downloading github.com/corpix/uarand v0.2.0
go: downloading github.com/projectdiscovery/fastdialer v0.0.49
go: downloading github.com/projectdiscovery/ratelimit v0.0.20
go: downloading github.com/projectdiscovery/rawhttp v0.1.28
go: downloading golang.org/x/exp v0.0.0-20231110203233-9a3e6036ecaa
go: downloading github.com/projectdiscovery/yamldoc-go v1.0.4
go: downloading golang.org/x/text v0.14.0
go: downloading github.com/segmentio/ksuid v1.0.4
go: downloading moull.io/http2curl v1.0.0
go: downloading golang.org/x/sys v0.15.0
go: downloading github.com/google/uuid v1.3.1
go: downloading github.com/projectdiscovery/asnmap v1.0.6
go: downloading gopkg.in/corvus-ch/zbase32.v1 v1.0.0
go: downloading github.com/projectdiscovery/mapcidr v1.1.16
go: downloading github.com/Azure/azure-sdk-for-go/sdk/azidentity v1.3.0
go: downloading github.com/Azure/azure-sdk-for-go/sdk/storage/azblob v1.1.0
go: downloading github.com/aws/aws-sdk-go-v2 v1.19.0
go: downloading github.com/aws/aws-sdk-go-v2/config v1.18.28
```

```
go install -v github.com/projectdiscovery/nuclei/v3/cmd/nuclei@latest
```

Clone repo “Juicy info Nuclei templates” (we’ll talk more about the git command on day 19):

```

gitpod /workspace/linux-for-0SINT-21-day/Day_15 (main) $ git clone https://github.com/cipher387/juicyinfo-nuclei-templates
Cloning into 'juicyinfo-nuclei-templates'...
remote: Enumerating objects: 173, done.
remote: Counting objects: 100% (73/73), done.
remote: Compressing objects: 100% (71/71), done.
remote: Total 173 (delta 41), reused 0 (delta 0), pack-reused 100
Receiving objects: 100% (173/173), 14.55 MiB | 20.96 MiB/s, done.
Resolving deltas: 100% (85/85), done.
gitpod /workspace/linux-for-0SINT-21-day/Day_15 (main) $

```

**git clone https://github.com/cipher387/juicyinfo-nuclei-templates**

Now, let's scan through all the links we've collected using Katana and extract investigation links that contain the word Github in them from the pages (we'll use the github.yaml template to do this):

```

gitpod /workspace/linux-for-0SINT-21-day/Day_15 (main) $ nuclei -t juicyinfo-nuclei-templates/juicy_info/github.yaml -l results.txt >extracted_github_links.txt
[INF] Supplied input was automatically deduplicated (2 removed).
[WARN] Found 1 templates loaded with deprecated protocol syntax, update before v3 for continued support.
[INF] Current nuclei version: v3.1.3 (latest)
[INF] Current nuclei-templates version: v3.7.2 (latest)
[WARN] Scan results upload to cloud is disabled.
[INF] New templates added in latest release: 61
[INF] Templates loaded for current scan: 1
[WARN] Executing 1 unsigned templates, use with caution.
[INF] Targets loaded for current scan: 272
[INF] Running httpx on input host
[INF] Found 3 URL from httpx
gitpod /workspace/linux-for-0SINT-21-day/Day_15 (main) $

```

**nuclei -t juicyinfo-nuclei-templates/juicy\_info/github.yaml -l results.txt >extracted\_github\_links.txt**

Similarly, you can collect links to other social networks, phone numbers, IP addresses, emails, cryptocurrency wallet addresses, etc. (template names are intuitive).

Nuclei also allows you to extract information from html tags. Let's collect the titles (<title> tag) of the same pages:



What if you want to scrape some other tag instead of the <title> tag?

```
! title.yaml X
Day_15 > juicyinfo-nuclei-templates > juicy_info > ! title.yaml > [ms] id
 1 id: extract-titles
 2
 3 info:
 4   name: Titles extractor
 5   author: cipher387
 6   severity: info
 7   description: Extract titles from web page body
 8   tags: osint,juicy-info,scraping
 9
10 requests:
11 - method: GET
12   path:
13     - "{{BaseURL}}"
14
15   matchers:
16     - type: regex
17       part: body
18       regex:
19         - '<title>.*</title>*'
20
21   extractors:
22     - type: regex
23       part: body
24       regex:
25         - '<title>.*</title>*'
26
```

Open the **title.yaml** file and simply replace the title with some other title. If you wish, save the template under a different name and run Nuclei in the above way.

And, of course, you can use whatever regular expressions you can think of for your tasks in the templates.

You could do the same thing with the curl and grep command combinations we talked about on **Days 5** and **6**, but Nuclei is much faster and easier to set up.

## HTML2TEXT

Html2text (<https://alir3z4.github.io/html2text/>) is a very simple Python script that cleans up the text of web pages from html tags and makes it easy to read and automated information retrieval.

```
PROBLEMS  OUTPUT  PORTS  DEBUG CONSOLE  TERMINAL
gitpod /workspace/linux-for-0SINT-21-day (main) $ apt-get install html2text
E: Could not open lock file /var/lib/dpkg/lock-frontent - open (13: Permission denied)
E: Unable to acquire the dpkg frontend lock (/var/lib/dpkg/lock-frontent), are you root?
gitpod /workspace/linux-for-0SINT-21-day (main) $ sudo apt-get install html2text
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  html2text
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 84.2 kB of archives.
After this operation, 279 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy/universe amd64 html2text amd64 1.3.2a-28 [84.2 kB]
Fetched 84.2 kB in 0s (1,037 kB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package html2text.
(Reading database ... 35768 files and directories currently installed.)
Preparing to unpack .../html2text_1.3.2a-28_amd64.deb ...
Unpacking html2text (1.3.2a-28) ...
Setting up html2text (1.3.2a-28) ...
Processing triggers for man-db (2.10.2-1) ...
gitpod /workspace/linux-for-0SINT-21-day (main) $
```

Let's install:

```
sudo apt-get install html2text
```

Let's try to extract the text from the html file contained in the **Day 15** folder:

```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
● gitpod /workspace/linux-for-0SINT-21-day/Day_15 (main) $ html2text example.html

[./example_files/logo3.png]_Filesamples
Categories
Video Audio Document Image Font Ebook Code

[q ]

***** Sample DOCX Files Download *****
DOCX Microsoft Word Open XML Format Document File
A popular word processing file format created as a replacement for DOC files in
Microsoft Word 2007 and above by Microsoft. It uses the Open XML mark-up
language.

Below you will find a selection of sample .docx document files for you to
download. On the right there are some details about the file such as its size
so you can best decide which one will fit your needs.
sample4.docx
DOCX / 13.51 MB

Download
sample3.docx
DOCX / 33.57 KB
```

**html2text example.html**

### Other tools

Scraping is a huge topic with many different nuances that requires a book of its own. Today we just learned how to extract information from html code.

But sometimes collecting data from websites requires automating the opening of the browser and simulating different user actions (because the necessary data is loaded only after executing JavaScript code). In this case, you need to use tools that run browsers (for example, Firefox or Chromium) in the command line.

It should also be said that Linux and Bash are not always the right tools for scraping. For many OSINT-related tasks, any of the many scraping browser plugins that support exporting data in CSV format (pay special attention to AI scraping), which, as you know, is very convenient to work with in Linux, may be sufficient. Few examples:

Webscraper.io (<https://webscraper.io>)

DataMiner (<https://dataminer.io>)

Instant Data Scraper  
(<https://chromewebstore.google.com/detail/instant-data-scraper/ofaokhiedipichpaobibbnahnkdoii>)

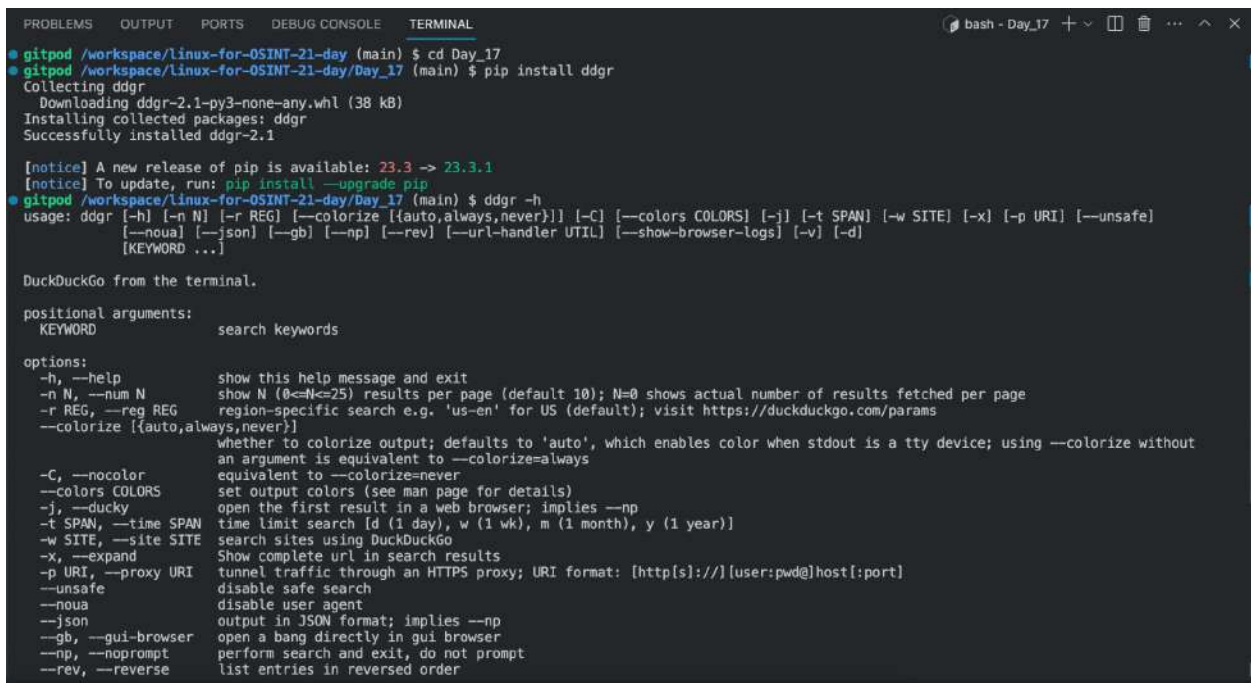
and many many others.

# Day 16. Web search automation tools

Searching for information on Google or another search engine can take up most of an OSINT professional's time. Therefore, even if you forget about all the other chapters in this book and automate only this process, this would already be enough to reduce lost work time through automation.

There are many tools for automating your work with different search engines, but they are all pretty similar to each other. Today, I will show you examples of two types of tools - one that automates a traditional search engine (searches web pages) and one that automates an IP search engine (searches through all the devices connected to the internet).

## DuckDuckGo search automation



```
gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_17
gitpod /workspace/linux-for-OSINT-21-day/Day_17 (main) $ pip install ddgr
Collecting ddgr
  Downloading ddgr-2.1-py3-none-any.whl (38 kB)
Installing collected packages: ddgr
Successfully installed ddgr-2.1

[notice] A new release of pip is available: 23.3 -> 23.3.1
[notice] To update, run: pip install --upgrade pip
gitpod /workspace/linux-for-OSINT-21-day/Day_17 (main) $ ddgr -h
usage: ddgr [-h] [-n N] [-r REG] [--colorize [{auto,always,never}]] [-C] [--colors COLORS] [-j] [-t SPAN] [-w SITE] [-x] [-p URI] [--unsafe]
           [--noui] [--json] [--gb] [--np] [--rev] [--url-handler UTIL] [--show-browser-logs] [--v] [-d]
           [KEYWORD ...]

DuckDuckGo from the terminal.

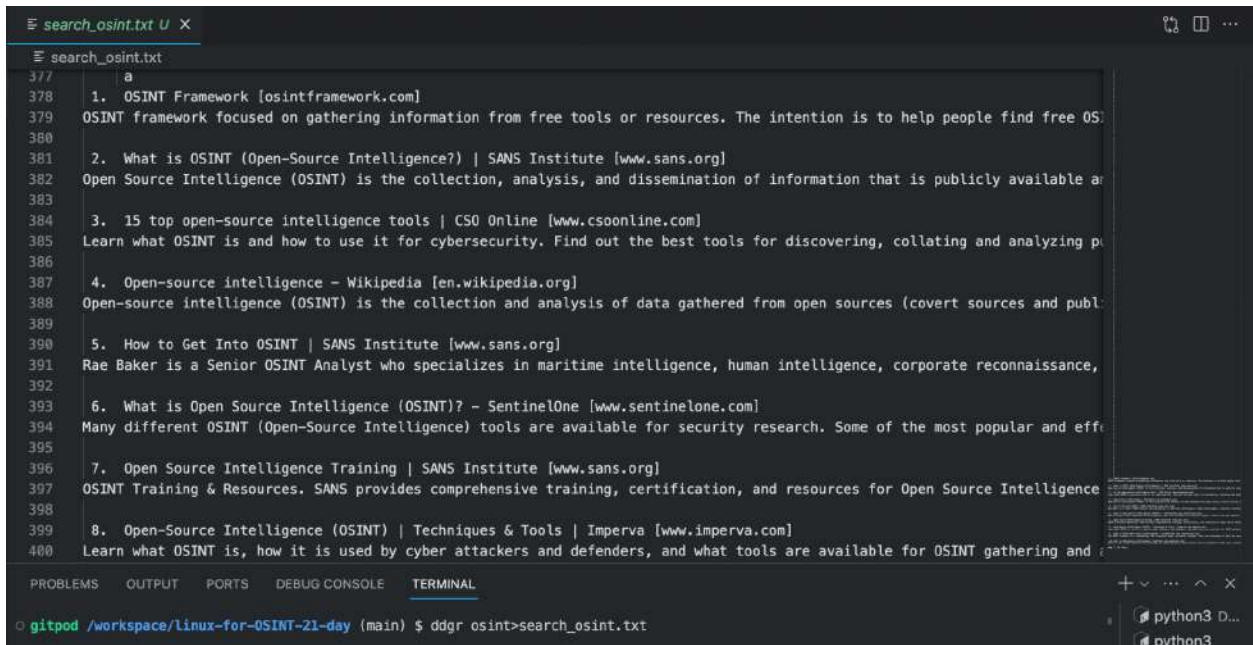
positional arguments:
  KEYWORD                search keywords

options:
  -h, --help            show this help message and exit
  -n N, --num N         show N (0<=N<=25) results per page (default 10); N=0 shows actual number of results fetched per page
  -r REG, --reg REG     region-specific search e.g. 'us-en' for US (default); visit https://duckduckgo.com/params
  --colorize [{auto,always,never}]
                        whether to colorize output; defaults to 'auto', which enables color when stdout is a tty device; using --colorize without
                        an argument is equivalent to --colorize=always
  -C, --nocolor         equivalent to --colorize=never
  --colors COLORS       set output colors (see man page for details)
  -j, --ducky          open the first result in a web browser; implies --np
  -t SPAN, --time SPAN time limit search [d (1 day), w (1 wk), m (1 month), y (1 year)]
  -w SITE, --site SITE search sites using DuckDuckGo
  -x, --expand         Show complete url in search results
  -p URI, --proxy URI  tunnel traffic through an HTTPS proxy; URI format: [http[s]://][user:pwd@]host[:port]
  --unsafe             disable safe search
  --noui              disable user agent
  --json              output in JSON format; implies --np
  --gb, --gui-browser  open a bang directly in gui browser
  --np, --noprompt    perform search and exit, do not prompt
  --rev, --reverse    list entries in reversed order
```

DuckDuckGo search (<https://github.com/jarun/ddgr>) is a command-line utility written in Python that allows you to automate DuckDuckGo searches.

Lets' install from PyPi package manager:

```
pip install ddgr
```



```
search_osint.txt
a
377
378 1. OSINT Framework [osintframework.com]
379 OSINT framework focused on gathering information from free tools or resources. The intention is to help people find free OS
380
381 2. What is OSINT (Open-Source Intelligence?) | SANS Institute [www.sans.org]
382 Open Source Intelligence (OSINT) is the collection, analysis, and dissemination of information that is publicly available a
383
384 3. 15 top open-source intelligence tools | CSO Online [www.csoonline.com]
385 Learn what OSINT is and how to use it for cybersecurity. Find out the best tools for discovering, collating and analyzing p
386
387 4. Open-source intelligence - Wikipedia [en.wikipedia.org]
388 Open-source intelligence (OSINT) is the collection and analysis of data gathered from open sources (covert sources and publ
389
390 5. How to Get Into OSINT | SANS Institute [www.sans.org]
391 Rae Baker is a Senior OSINT Analyst who specializes in maritime intelligence, human intelligence, corporate reconnaissance,
392
393 6. What is Open Source Intelligence (OSINT)? - SentinelOne [www.sentinelone.com]
394 Many different OSINT (Open-Source Intelligence) tools are available for security research. Some of the most popular and eff
395
396 7. Open Source Intelligence Training | SANS Institute [www.sans.org]
397 OSINT Training & Resources. SANS provides comprehensive training, certification, and resources for Open Source Intelligence
398
399 8. Open-Source Intelligence (OSINT) | Techniques & Tools | Imperva [www.imperva.com]
400 Learn what OSINT is, how it is used by cyber attackers and defenders, and what tools are available for OSINT gathering and
```

Now let's try searching for the word "osint" in DuckDuckGo and write the results to a text file:

```
cd Day_16
ddgr osint>search_osint.txt
```

If you want to extract only references from the results file (for example, for automatic scraping), you can use a combination of **Cat** and **Grep** commands:

```
390 5. How to Get Into OSINT | SANS Institute [www.sans.org]
391 Learn how to find OSINT resources and communities, practice techniques, and find a job in OSINT. This blog post shares various metho
392
393 6. What is Open Source Intelligence (OSINT)? – SentinelOne [www.sentinelone.com]
394 OSINT stands for open source intelligence, which refers to legally gathered information about an individual or organization from fre
395
396 7. Open Source Intelligence Training | SANS Institute [www.sans.org]
397 Learn how to collect and analyze publicly available information from various sources to support investigations, security, and decisi
398
399 8. Open-Source Intelligence (OSINT) | Techniques & Tools | Imperva [www.imperva.com]
400 Learn what OSINT is, how it works, and how it can be used by security teams, attackers, and public opinion seekers. Explore the hist
401
402 9. OSINT: What is open source intelligence and how is it used? [portswigger.net]
403 OSINT is intelligence drawn from publicly available material, such as the internet, mass media, and social media. It can be used for
404
405 10. What is Open-Source Intelligence | OpenText [www.opentext.com]
406 OSINT is the insight gained from processing and analyzing public data sources such as TV, radio, social media, and websites. Learn h
407
408 ddgr (? for help):
```

PROBLEMS OUTPUT PORTS COMMENTS DEBUG CONSOLE TERMINAL

```
gitpod /workspace/linux-for-OSINT-21-day/Day_16 (main) $ cat search_osint.txt | grep -o '\[.*\]'
[osintframework.com]
[www.sans.org]
[www.csoonline.com]
[en.wikipedia.org]
[www.sans.org]
[www.sentinelone.com]
[www.sans.org]
[www.imperva.com]
[portswigger.net]
[www.opentext.com]
gitpod /workspace/linux-for-OSINT-21-day/Day_16 (main) $
```

**cat search\_osint.txt | grep -o '\[.\*\]'**

Yes, is not very good that this tool does not collect full URLs but only site names. But the reader will be motivated to try other tools for collecting search results, which I'll talk about below.

Also a good option is to export the JSON results immediately by running the **Ddgr** command with the **--json** flag.

Remember that you can use not only words and phrases in your queries, but also advanced search operators. For example:

**filetype:xlsx**  
**site:facebook.com**  
**intitle:osint**  
**inurl:osint**

If you need to gather search results for a larger list of queries, use proxy servers to avoid having your IP address blocked with **--proxy** flag. These can be found in numerous lists of free proxy servers. For example, <https://github.com/TheSpeedX/PROXY-List>.

To collect the maximum amount of data, I recommend that you collect results not only from DuckDuckGo, but also from other search engines using the tools like:

PyDork (<https://github.com/blacknon/pydork>) - search in Google, Bing, DuckDuckGo, **Baidu** and Yahoo Japan.

Fast Google Dorks Scan (<https://github.com/IvanGlinkin/Fast-Google-Dorks-Scan>)

OxDork (<https://github.com/rly0nheart/oxdork>) - search in Google.

and many others.

Tools for automating the collection of search results for different search engines are plentiful. You can find a list of them in the Dorks collection list repository (<https://github.com/cipher387/Dorks-collections-list>).

In addition, these tools are quite simply and you can easily make your own using APIs, a list of which can be found at “Search engines API” section of API for OSINT repository (<https://github.com/cipher387/API-s-for-OSINT>).

## IP search engines

It is worth talking separately about automating the collection of IP search engines (Shodan, Censys, Fofa, Netlas) search results. This is a bit more complex process, as search results contain much more information than Google or DuckDuckGo search results.

IP search engines differ from “normal” search engines in that they do not index web pages to which they can find links, but all IP addresses connected to the Internet. That is, not only websites, but all other servers and devices.

As usual, I'll use Netlas for an example.

```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
Requirement already satisfied: charset-normalizer<4,>=2 in /home/gitpod/.pyenv/versions/3.12.0/lib/python3.12/site-packages (from requests>=2.12.5->netlas) (3.3.0)
Requirement already satisfied: idna<4,>=2.5 in /home/gitpod/.pyenv/versions/3.12.0/lib/python3.12/site-packages (from requests>=2.12.5->netlas) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in /home/gitpod/.pyenv/versions/3.12.0/lib/python3.12/site-packages (from requests>=2.12.5->netlas) (2.0.6)
Requirement already satisfied: certifi>=2017.4.17 in /home/gitpod/.pyenv/versions/3.12.0/lib/python3.12/site-packages (from requests>=2.12.5->netlas) (2023.7.22)
Downloading netlas-0.4.1-py3-none-any.whl (10 kB)
Downloading click-8.1.7-py3-none-any.whl (97 kB)
 97.9/97.9 kB 38.6 MB/s eta 0:00:00
Downloading tqdm-4.66.1-py3-none-any.whl (78 kB)
 78.3/78.3 kB 19.8 MB/s eta 0:00:00
Building wheels for collected packages: orjson
  Building wheel for orjson (pyproject.toml) ... done
  Created wheel for orjson: filename=orjson-3.9.0-cp312-cp312-linux_x86_64.whl size=735362 sha256=719370238c1c9c98bcd396ddb2932eb023367375ba37d2f4914368bcf0f63766
  Stored in directory: /workspace/.pyenv_mirror/pip_cache/wheels/c4/04/5b/abeb46db205bc1993d473ea3c43bcadae072cd002d798280db
Successfully built orjson
Installing collected packages: appdirs, tqdm, orjson, click, netlas
Successfully installed appdirs-1.4.4 click-8.1.7 netlas-0.4.1 orjson-3.9.0 tqdm-4.66.1

[notice] A new release of pip is available: 23.3 -> 23.3.1
[notice] To update, run: pip install --upgrade pip
gitpod /workspace/linux-for-OSINT-21-day/Day_17 (main) $ netlas -h
Usage: netlas [OPTIONS] COMMAND [ARGS]...

Options:
  -h, --help  Show this message and exit.

Commands:
  count      Calculate count of query results.
  download   Download data.
  host       Host (ip or domain) information.
  indices    Get available data indices.
  profile    Get user profile data.
  savekey    Save API key to the local system.
  search (query) Search query.
  stat       Get statistics for query.
```

First, let's install the Netlas Python package (<https://readthedocs.org/projects/netlas-python/>):

```
pip install netlas
netlas -h
```

```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
gitpod /workspace/linux-for-OSINT-21-day (main) $ netlas -h
Usage: netlas [OPTIONS] COMMAND [ARGS]...

Options:
  -h, --help  Show this message and exit.

Commands:
  count      Calculate count of query results.
  download   Download data.
  host       Host (ip or domain) information.
  indices    Get available data indices.
  profile    Get user profile data.
  savekey    Save API key to the local system.
  search (query) Search query.
  stat       Get statistics for query.
gitpod /workspace/linux-for-OSINT-21-day (main) $
```

Before sending first request, register on Netlas and copy the API key from your personal account (50 request/day FREE):

**netlas savekey YOUR\_API\_KEY**

```
netlas_results.txt U X
netlas_results.txt
1 {"data": {"referer": ["https://217.160.158.203:443/"], "last_updated": "2023-10-22T11:35:21.804Z", "jarm":
"15d3fd16d29d29d00042d43d00000ed1cf37c9a169b41886e27ba8fad60b0", "isp": "1&1 Internet AG", "ip": "217.160.158.203",
"certificate": {"issuer_dn": "CN=R3, O=Let's Encrypt, C=US", "fingerprint_md5": "bc1b541b012aee5b162be25de9b5ca88",
"chain": [{"issuer_dn": "CN=ISRG Root X1, O=Internet Security Research Group, C=US", "fingerprint_md5":
"e829e65d7c4307d6fbc13c179e037a36", "redacted": false, "signature": {"valid": false, "signature_algorithm": {"name":
"SHA256-RSA", "oid": "1.2.840.113549.1.1.11"}, "self_signed": false, "value": "hcx0Rz6j94VEhbzVZ3iyg0tdU0elj0zZXJULYGG6sPt
+CC/X8y3cAC3bjv2XpTe5CCfpu+LsgPnorUWPJH0t005Aud8JYph5mVuP0b02fd0lCvuVM4SvIwnS7jBmC+ir81xkUoit8i4I3sELQj5CFc
+g9kEMwpHIXgJgifDKsibuc5c8mTIwL55wE+0bUQMXpk7LveLE0HoHU0p21kg7W05IfgSjPSTV6AdZQTBciKuEA1Dl6EYH37g4IY3tVqxvTC/h24rKv8hThsFw/
UYL/BerM0luGrwLrw7M7nuS97M/0svhAuGP8BVQzb2a0E2F2q0mdH/pUCnNLfA0GM5NTl1bvK6dsiTAumpS2wXzgwC2b2B+5+3aNQGb0CPXdT
+I55A60KMqd1KkPYVZdyxck098RdTsIUroQw1/KFXxihebnXnCLB+GGk805j9xhcSUqr9/
tJVB5aIuS30XWtePaFdCHbIQhMa6C9fu5q8iJFz3hTOU4Dva9K72WgRT1ds9IKd+WdPi+Fj5W7hIzf5cTxYp/h5VI6/IEbCN6nyTkBcv/ayiCUdGP/DpsLf/
KE1oMtZnXh5po5049Z2LLwvSuk0mbzJXZU0ygd84U4Vdf1lMeq43eSvtc21VhJzcR0xiU4RFBt7M4AVRj+6UlK1E7K15y0W8BzqKu4R8I="}, "subject":
{"country": ["US"], "organization": ["Let's Encrypt"], "common_name": ["R3"], "serial_number":
"192961496339968674994309121183282847578", "version": 3, "issuer": {"country": ["US"], "organization": ["Internet Security
Research Group"], "common_name": ["ISRG Root X1"]}, "fingerprint_sha256":
"67add1166b020ae61b8f5fc96813c04c2aa589960796865572a3c7e737613dfd", "tbs_noct_fingerprint":
"444ebd67bb83f8807b3921e938ac9178b882bd50aadb11231f044cf5f08df7ce", "extensions": {"subject_key_id":
"142eb217b7585fcb350004061f0408b14c2c6", "url_distribution_point": ["http://c1.c.10ncr.org/"], "certificate_policies":
```

Let's try to find servers that have the word "osint" in the body tag of their http response html code:

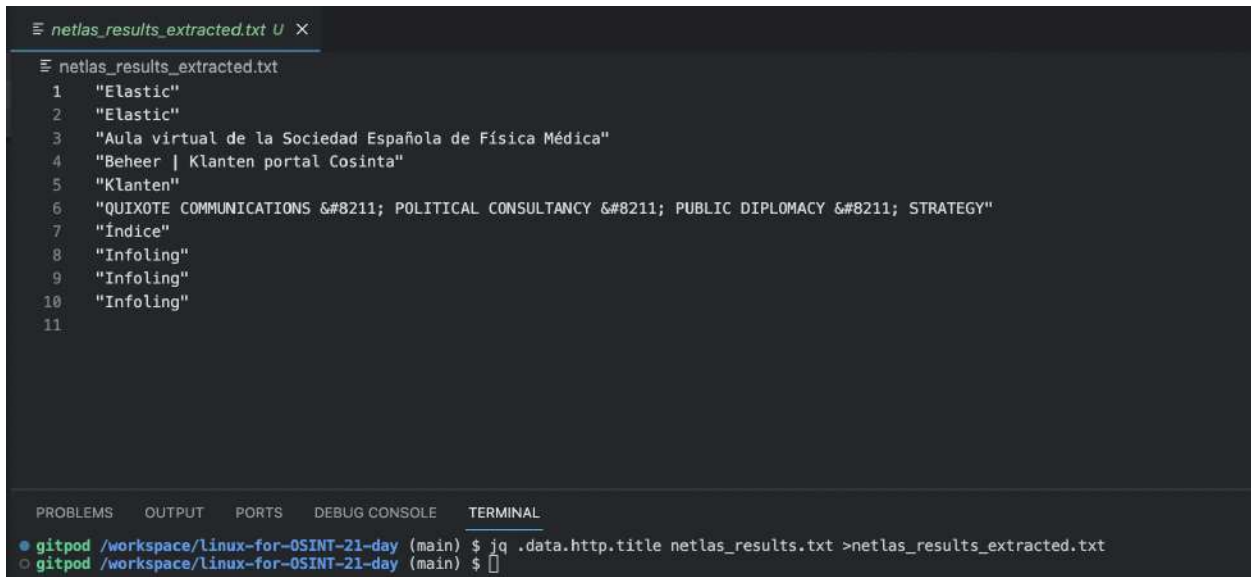
**netlas download "http.body:\*osint\*>netlas\_results.txt"**

Note that in the query, we use the filter for advanced search "http.body" to search only the text of web pages. But Netlas has dozens of other advanced search operators. For example:

- http.title:
- http.meta:
- uri:
- domain:
- port:
- geo.city:
- ip:
- geo.country:

You can extract the any data from the results file using the JQ utility, which is described in detail in **Day 14**:

```
jq .data.http.title netlas_results.txt >netlas_results_extracted.txt
```



```
netlas_results_extracted.txt U X
netlas_results_extracted.txt
1  "Elastic"
2  "Elastic"
3  "Aula virtual de la Sociedad Española de Física Médica"
4  "Beheer | Klanten portal Cosinta"
5  "Klanten"
6  "QUIXOTE COMMUNICATIONS &#8211; POLITICAL CONSULTANCY &#8211; PUBLIC DIPLOMACY &#8211; STRATEGY"
7  "Indice"
8  "Infoling"
9  "Infoling"
10 "Infoling"
11
PROBLEMS  OUTPUT  PORTS  DEBUG CONSOLE  TERMINAL
gitpod /workspace/linux-for-OSINT-21-day (main) $ jq .data.http.title netlas_results.txt >netlas_results_extracted.txt
gitpod /workspace/linux-for-OSINT-21-day (main) $
```

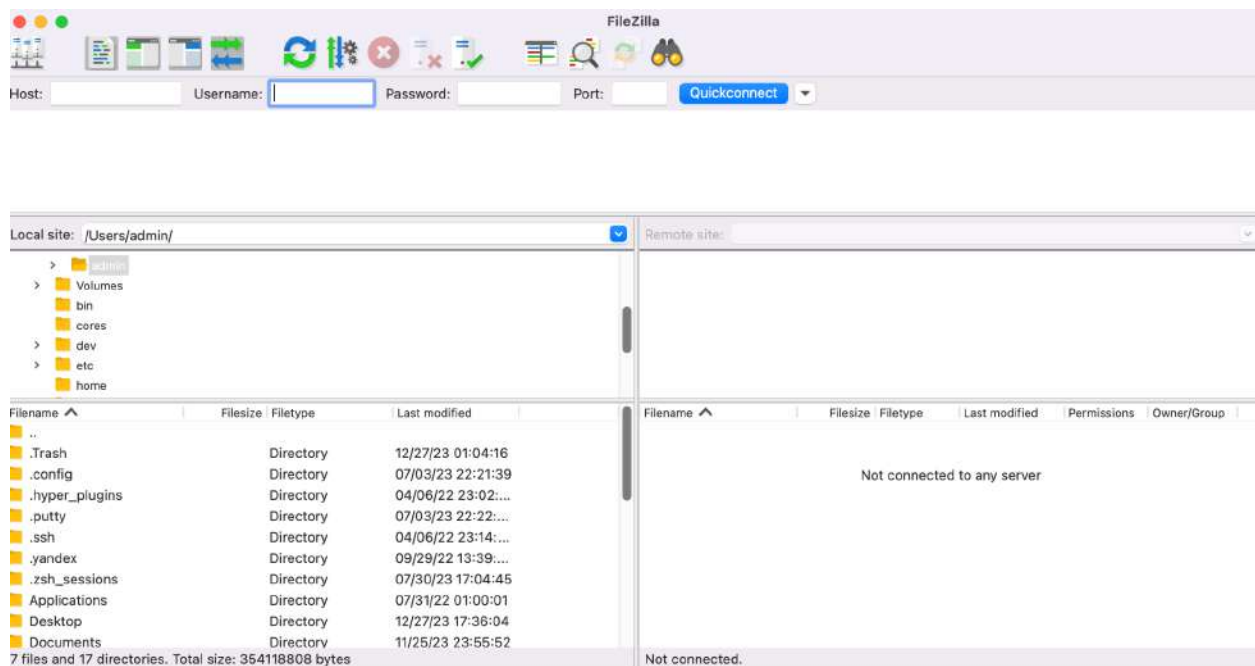
To learn more about how to automate dozens of different OSINT and pentest tasks with Netlas Python library, check out Netlas CookBook (<https://github.com/netlas-io/netlas-cookbook>).

## Day 17. File sharing sites, torrents, FTP

Today we'll return to the topic we started talking about in **Day 5**. It's downloading content (for later automatic analysis). This time we will talk about slightly more complex things, but in a more superficial way: FTP servers, torrents, file sharing and social media.

Today we're not going to run examples of commands, we're just going to talk about different tools that are worth knowing about.

## FTP



On the fifth day, we already talked about the **curl** utility, which supports uploading files through different protocols, including the FTP protocol.

But still, the FTP utility ([https://www.gnu.org/software/inetutils/manual/html\\_node/ftp-invocation.html](https://www.gnu.org/software/inetutils/manual/html_node/ftp-invocation.html)) is a bit more convenient for this task.

If you just need to transfer files, you will probably be more comfortable using a GUI FTP client such as FileZilla ([https://wiki.filezilla-project.org/Main\\_Page](https://wiki.filezilla-project.org/Main_Page)) rather than the command line.

But FileZilla (like many other FTP clients) allows you to automate various actions using the command line ([https://wiki.filezilla-project.org/Command-line\\_arguments\\_\(Client\)](https://wiki.filezilla-project.org/Command-line_arguments_(Client))).

## Torrents

It is illegal to use leaked data files, but many OSINT professionals use them anyway to gather more information. Often the leaked data is stored as huge files on torrents.

For example, I had to deal with one major social network leak that weighed hundreds of gigabytes. Unpacking the archives and extracting the necessary data from it using the JQ utility took more than two days! But thanks to the screen (**Day 9**) I only had to spend about an hour writing commands and check screen sessions a few times to see if everything's going well.

Seedbox is a high-bandwidth remote server used to upload and download digital files from a peer-to-peer (P2P) network ([tech slang](#)).

Here are some examples of popular services for monthly seedbox rentals:

RapidSeedbox (<https://www.rapidseedbox.com>)

CloudBoxes (<https://cloudboxes.io>)

DediSeedBox (<https://dediseedbox.com/>)

Seedboxes is a very handy thing, but you can run command line torrent clients on your VPS as well if you want:

Aria (<https://aria2.github.io>)

Transmission-cli (<https://cli-ck.io/transmission-cli-user-guide/>)

WebTorrent-cli (<https://github.com/webtorrent/webtorrent-cli>)

Stig (<https://github.com/rndusr/stig>)

## File sharing services

When using files that are uploaded to various file sharing sites in investigations, you should also keep in mind that their downloading can be automated. Here are some examples of command line utilities for different services that will help you do this:

Google Drive (<https://github.com/glottlabs/gdrive>)

DropBox (<https://github.com/dropbox/dbxcli>)

Mega.nz (<https://github.com/meganz/MEGAcmd>)

OneDrive (<https://github.com/skillion/onedrive>)

But be very careful when using them and never use accounts where you have important information stored for authorization. **After all, these are unofficial utilities and accounts can be blocked for using them.**

## Social media

When you work with a lot of content that is published on some popular social network, check to see if someone else has made a utility to automate its uploading. For example:

YouTube-dl (<https://youtube-dl.org>)

SlideShare (<https://github.com/mohan3d/slideshare-go>)

Spotify (<https://github.com/spotDL/spotify-downloader>)

You Get (<https://github.com/soimort/you-get>)

And I'll say it again. We are talking about unofficial clients, the use of which can lead to blocked accounts. Be very careful.

# Day 18. Domain investigation

Perhaps when you first saw this book, you thought of it as being mostly about automating the network information gathering. But this book has the goal of teaching OSINT technicians minimal Linux skills.

Therefore, I will cover only the most basic techniques.

## Whois

Whois is a simple utility that displays basic information about a domain: whether it is registered or not, date of registration, end date of the paid holding period, contact details of the owner, etc.

```
PROBLEMS  OUTPUT  PORTS  DEBUG CONSOLE  TERMINAL
● gitpod /workspace/linux-for-OSINT-21-day (main) $ sudo apt-get install whois
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  whois
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 53.4 kB of archives.
After this operation, 279 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy/main amd64 whois amd64 5.5.13 [53.4 kB]
Fetched 53.4 kB in 0s (149 kB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package whois.
(Reading database ... 35595 files and directories currently installed.)
Preparing to unpack .../whois_5.5.13_amd64.deb ...
Unpacking whois (5.5.13) ...
Setting up whois (5.5.13) ...
Processing triggers for man-db (2.10.2-1) ...
○ gitpod /workspace/linux-for-OSINT-21-day (main) $
```

Let's install:

```
sudo apt-get install whois
```

```
PROBLEMS  OUTPUT  PORTS  DEBUG CONSOLE  TERMINAL
● gitpod /workspace/linux-for-OSINT-21-day (main) $ whois github.com
Domain Name: GITHUB.COM
Registry Domain ID: 1264983250_DOMAIN_COM-VRSN
Registrar WHOIS Server: whois.markmonitor.com
Registrar URL: http://www.markmonitor.com
Updated Date: 2022-09-07T09:10:44Z
Creation Date: 2007-10-09T18:20:50Z
Registry Expiry Date: 2024-10-09T18:20:50Z
Registrar: MarkMonitor Inc.
Registrar IANA ID: 292
Registrar Abuse Contact Email: abusecomplaints@markmonitor.com
Registrar Abuse Contact Phone: +1.2086851750
Domain Status: clientDeleteProhibited https://icann.org/epp#clientDeleteProhibited
Domain Status: clientTransferProhibited https://icann.org/epp#clientTransferProhibited
Domain Status: clientUpdateProhibited https://icann.org/epp#clientUpdateProhibited
Name Server: DNS1.P08.NSONE.NET
Name Server: DNS2.P08.NSONE.NET
Name Server: DNS3.P08.NSONE.NET
Name Server: DNS4.P08.NSONE.NET
Name Server: NS-1283.AWSDNS-32.ORG
Name Server: NS-1707.AWSDNS-21.CO.UK
Name Server: NS-421.AWSDNS-52.COM
Name Server: NS-520.AWSDNS-01.NET
DNSSEC: unsigned
URL of the ICANN Whois Inaccuracy Complaint Form: https://www.icann.org/wicf/
>>> Last update of whois database: 2023-12-16T19:51:05Z <<<

For more information on Whois status codes, please visit https://icann.org/epp

NOTICE: The expiration date displayed in this record is the date the
registrar's sponsorship of the domain name registration in the registry is
currently set to expire. This date does not necessarily reflect the expiration
date of the domain name registrant's agreement with the sponsoring
registrar. Users may consult the sponsoring registrar's Whois database to
view the registrar's reported date of expiration for this registration.

TERMS OF USE: You are not authorized to access or query our Whois
```

And let's gather information about the GitHub.com domain:

**whois github.com**

```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
Focus folder in explorer (cmd + click)
● gitpod /workspace/Linux-for-OSINT-21-day (main) $ whois sector035.nl | grep 'Creation'
Creation Date: 2017-05-29
○ gitpod /workspace/Linux-for-OSINT-21-day (main) $ █
```

If you only need certain information from whois, you can extract the necessary strings with grep (remember **Day 6**):

**whois sector035.nl | grep 'Creation'**

## DNS

Dnsutils Debian package (<https://packages.debian.org/en/buster/dnsutils>) includes three commands:

- dig** - query the DNS (Domain Name System) info;
- nslookup** - the older way to do it;
- nsupdate** - perform dynamic updates (See RFC2136).

For the purposes of this course, we will consider only the first.

```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
● gitpod /workspace/linux-for-OSINT-21-day (main) $ sudo apt-get install dnstools
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  bind9-dnssutils bind9-host bind9-libs liblmbd0 libuv1
The following NEW packages will be installed:
  bind9-dnssutils bind9-host bind9-libs dnstools liblmbd0 libuv1
0 upgraded, 6 newly installed, 0 to remove and 0 not upgraded.
Need to get 1,594 kB of archives.
After this operation, 9,892 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://archive.ubuntu.com/ubuntu jammy/main amd64 liblmbd0 amd64 0.9.24-1build2 [47.6 kB]
Get:2 http://archive.ubuntu.com/ubuntu jammy/main amd64 libuv1 amd64 1.43.0-1 [93.1 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 bind9-libs amd64 1:9.18.12-0ubuntu0.22.04.3 [1,240 kB]
Get:4 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 bind9-host amd64 1:9.18.12-0ubuntu0.22.04.3 [52.3 kB]
Get:5 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 bind9-dnssutils amd64 1:9.18.12-0ubuntu0.22.04.3 [157 kB]
Get:6 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64 dnstools all 1:9.18.12-0ubuntu0.22.04.3 [3,924 B]
Fetched 1,594 kB in 0s (13.2 MB/s)
debconf: delaying package configuration, since apt-utils is not installed
Selecting previously unselected package liblmbd0:amd64.
(Reading database ... 35604 files and directories currently installed.)
Preparing to unpack .../0-liblmbd0_0.9.24-1build2_amd64.deb ...
Unpacking liblmbd0:amd64 (0.9.24-1build2) ...
Selecting previously unselected package libuv1:amd64.
Preparing to unpack .../1-libuv1_1.43.0-1_amd64.deb ...
Unpacking libuv1:amd64 (1.43.0-1) ...
Selecting previously unselected package bind9-libs:amd64.
Preparing to unpack .../2-bind9-libs_1%3a9.18.12-0ubuntu0.22.04.3_amd64.deb ...
Unpacking bind9-libs:amd64 (1:9.18.12-0ubuntu0.22.04.3) ...
Selecting previously unselected package bind9-host.
Preparing to unpack .../3-bind9-host_1%3a9.18.12-0ubuntu0.22.04.3_amd64.deb ...
Unpacking bind9-host (1:9.18.12-0ubuntu0.22.04.3) ...
Selecting previously unselected package bind9-dnssutils.
```

Let's install dnstools:

## sudo apt-get install dnstools

```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
● gitpod /workspace/linux-for-OSINT-21-day (main) $ dig google.com

; <<> DiG 9.18.12-0ubuntu0.22.04.3-Ubuntu <<> google.com
;; global options: +cmd
;; Got answer:
;; ->HEADER<<- opcode: QUERY, status: NOERROR, id: 39629
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
;; QUESTION SECTION:
;google.com.                IN      A

;; ANSWER SECTION:
google.com.                122     IN      A      142.250.201.174

;; Query time: 3 msec
;; SERVER: 1.1.1.1#53(1.1.1.1) (UDP)
;; WHEN: Sat Dec 16 19:55:35 UTC 2023
;; MSG SIZE rcvd: 55

○ gitpod /workspace/Linux-for-OSINT-21-day (main) $ 4
```

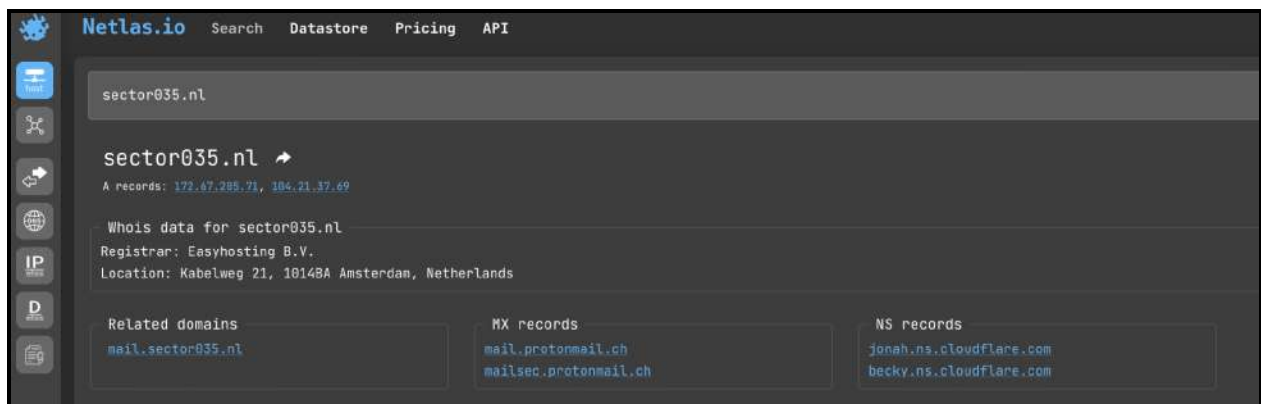
And get information from Domain Name System about google.com:

```
dig google.com
```

## Netlas

As you already know from past lessons, Netlas is a search engine that indexes not only websites, but all IP addresses connected to the Internet. And there is a lot of different information about each address or domain stored there, which can be retrieved with the **host** command.

Let's try to do it:



If you haven't already done so on **Day 16**, install the Netlas package from PyPi:

```
pip install netlas
```

```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
gitpod /workspace/Linux-for-OSINT-21-day (main) $ netlas host sector035.nl
dns:
  a:
    - 172.67.205.71
    - 104.21.37.69
  level: 2
  mx:
    - mail.protonmail.ch
    - mailsec.protonmail.ch
  ns:
    - jonah.ns.cloudflare.com
    - becky.ns.cloudflare.com
  txt:
    - protonmail-verification=225338d95b2a19240ce2d6c2989cd0a40c5d18cd
    - v=spf1 include:_spf.protonmail.ch mx ~all
    - google-site-verification=nAKZUqZ-WMDmamNaxfTpcQ27E93Z8j8V_2hLE91amxc
  zone: nl
domain: sector035.nl
related_domains:
  - mail.sector035.nl
related_domains_count: 1
related_domains_query: (domain:*.sector035.nl AND NOT domain:sector035.nl)
type: domain
whois:
  created_date: '2017-05-29T00:00:00.000Z'
  domain: sector035.nl
  extension: nl
  extracted_domain: sector035.nl
  last_updated: '2023-03-03T05:03:57.208Z'
  level: 2
  name: sector035
  name_servers:
    - becky.ns.cloudflare.com
    - jonah.ns.cloudflare.com
  punycode: sector035.nl
  registrar:
    name: Easyhosting B.V.
    street: Kabelweg 21, 1014BA Amsterdam, Netherlands
  server: whois.domain-registry.nl
  stats:
    error: no_error
    parser: no_error
    quota_retries: 0
    retries: 0
```

And run:

```
netlas host sector035.nl
```

As you can see, it displays both whois information, dns information, addresses of various servers that are associated with the domain that can be used for further investigation and many other info.

As I warned in the preface to this book, it is not written for pentesters and bug hunters at all, but for OSINT specialists. If you plan to do a job that requires a more in-depth knowledge of computer networks, you should explore many other sources of information.

A good place to start is with the manual for Nmap (<https://nmap.org>), one of the most famous utility for network exploration and security auditing.

All the commands described above can be combined with Xargs (remember **Day 4**) and used to work with domain/IP address lists.

# 20 domain investigation Python packages

Just type in command line: `pip install + package name`

<code>python-whois</code>	<code>virustotal-python</code>
<code>dnstwist</code>	<code>pysnyk</code>
<code>builtwith</code>	<code>pydork</code>
<code>waybackpy</code>	<code>EmailHarvester</code>
<code>netlas</code>	<code>sitemap-generator</code>
<code>pyseoanalyzer</code>	<code>censys</code>
<code>shodan</code>	<code>wapiti3</code>
<code>dns-crawler</code>	<code>dirhunt</code>
<code>oxdork</code>	<code>cloudscraper</code>
<code>urlscanio</code>	<code>metadata_parser</code>



Image from [Twitter](#).

In the picture is a small list with Python packages to gather information about the websites. To install each of them, type **pip install** (package name) at the command line, and then use the `help` command to understand what a particular package is for.

This is the largest homework in this book and you can spend several weeks completing it. We will try some more commands tomorrow, and next lessons will be without command examples, just reading and thinking.

# Day 19. Git and Github

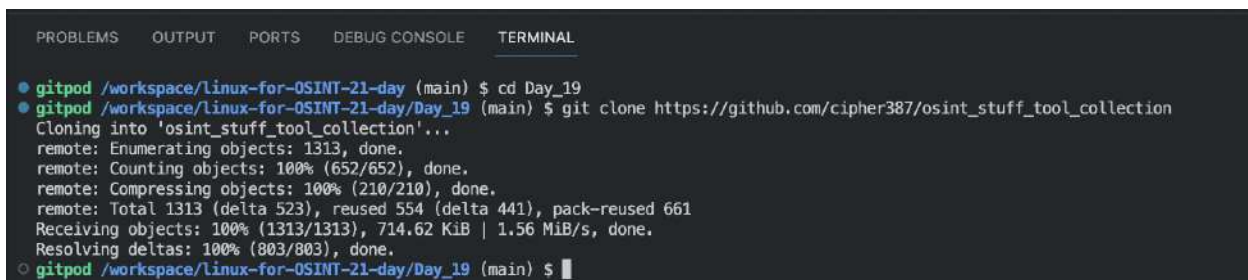
Git (<https://git-scm.com>) is a software version control system created by Linus Torvalds. According to StackOverflow's 2022 survey, more than 93% of developers use it as their primary version control system (<https://survey.stackoverflow.co/2022/#section-version-control-version-control-systems>).

The situation with Git and Linux is that exceptionally rare case where one person managed to invent two such game-changing software in one lifetime (Linux in 1991, Git in 2005).

Even if you don't intend to do any development work, you can still benefit from basic Git skills for OSINT. This comes in handy for running different tools, as well as for analyzing Github repositories that store large amounts of text data (e.g., phishing domain lists or smart contract files).

Gitpod already has Git installed, but in other cases you may need to install it:

```
sudo apt-get update
sudo apt-get install git
```



```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
● gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_19
● gitpod /workspace/linux-for-OSINT-21-day/Day_19 (main) $ git clone https://github.com/cipher387/osint_stuff_tool_collection
Cloning into 'osint_stuff_tool_collection'...
remote: Enumerating objects: 1313, done.
remote: Counting objects: 100% (652/652), done.
remote: Compressing objects: 100% (210/210), done.
remote: Total 1313 (delta 523), reused 554 (delta 441), pack-reused 661
Receiving objects: 100% (1313/1313), 714.62 KiB | 1.56 MiB/s, done.
Resolving deltas: 100% (803/803), done.
○ gitpod /workspace/linux-for-OSINT-21-day/Day_19 (main) $
```

Let's go to <https://github.com/new> and create new empty Github repository.

## Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).

Required fields are marked with an asterisk (\*).

Owner \*  / Repository name \*   
linux\_for\_osint\_20\_day is available.

Great repository names are short and memorable. Need inspiration? How about cuddly-tribble?

Description (optional)

Public  
Anyone on the internet can see this repository. You choose who can commit.

Private  
You choose who can see and commit to this repository.

Initialize this repository with:

Add a README file  
This is where you can write a long description for your project. [Learn more about READMEs](#).

Add .gitignore

Choose which files not to track from a list of templates. [Learn more about ignoring files](#).

Choose a license  
License:

A license tells others what they can and can't do with your code. [Learn more about licenses](#).

This will set `main` as the default branch. Change the default name in your [settings](#).

ⓘ You are creating a private repository in your personal account.

Look at the image above, fill in the repository name and description fields, make the repository private, create a README.MD file and click on **Create repository**.

Now go back to Gitpod and type in the command line:

```
PROBLEMS  OUTPUT  PORTS  DEBUG CONSOLE  TERMINAL
● gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_19
● gitpod /workspace/linux-for-OSINT-21-day/Day_19 (main) $ git clone https://github.com/cipher387/linux_for_osint_20_day
Cloning into 'linux_for_osint_20_day'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
○ gitpod /workspace/linux-for-OSINT-21-day/Day_19 (main) $
```

**git clone https://github.com/cipher387/linux\_for\_osint\_20\_day**

Replace cipher387 with your Github login and if your repository name is different, replace that too.

You can only copy private repositories from your own account, or from the accounts of people who have given you access to one or more private repositories.

Public repositories can be cloned from any account.

Now let's make sure that the repository has been cloned correctly:

```
PROBLEMS  OUTPUT  PORTS  DEBUG CONSOLE  TERMINAL
● gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_19
● gitpod /workspace/linux-for-OSINT-21-day/Day_19 (main) $ git clone https://github.com/cipher387/linux_for_osint_20_day
Cloning into 'linux_for_osint_20_day'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.
● gitpod /workspace/linux-for-OSINT-21-day/Day_19 (main) $ cd linux_for_osint_20_day
● gitpod /workspace/linux-for-OSINT-21-day/Day_19/linux_for_osint_20_day (main) $ ls
README.md
● gitpod /workspace/linux-for-OSINT-21-day/Day_19/linux_for_osint_20_day (main) $ cat README.md
# linux_for_osint_20_day
Test repository for book
○ gitpod /workspace/linux-for-OSINT-21-day/Day_19/linux_for_osint_20_day (main) $
```

```
cd linux_for_osint_20_day
```

```
ls
```

```
cat README.md
```

Let's create a new file in the repository folder:

```
ls >test.txt
```

```
cat test.txt
```

```
● gitpod /workspace/linux-for-OSINT-21-day/Day_19/linux_for_osint_20_day (main) $ ls >test.txt
● gitpod /workspace/linux-for-OSINT-21-day/Day_19/linux_for_osint_20_day (main) $ cat ls
cat: ls: No such file or directory
● gitpod /workspace/linux-for-OSINT-21-day/Day_19/linux_for_osint_20_day (main) $ cat test.txt
README.md
test.txt
● gitpod /workspace/linux-for-OSINT-21-day/Day_19/linux_for_osint_20_day (main) $ git add test.txt
● gitpod /workspace/linux-for-OSINT-21-day/Day_19/linux_for_osint_20_day (main) $ git commit
[main 5fac56b] test commit
1 file changed, 2 insertions(+)
create mode 100644 test.txt
● gitpod /workspace/linux-for-OSINT-21-day/Day_19/linux_for_osint_20_day (main) $ git push
Enumerating objects: 4, done.
Counting objects: 100% (4/4), done.
Delta compression using up to 16 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 299 bytes | 299.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/cipher387/linux_for_osint_20_day
37d73f3..5fac56b main -> main
○ gitpod /workspace/linux-for-OSINT-21-day/Day_19/linux_for_osint_20_day (main) $
```

And add it to a new commit:

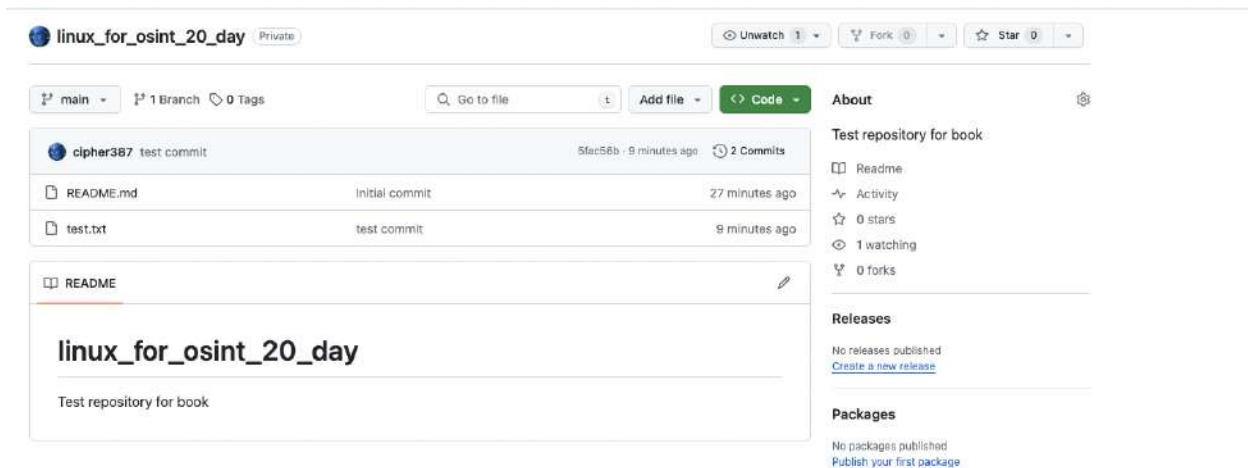
```
git add test.txt
```

```
git commit
```

Write some comment on the commit, close the text editor and type.

```
git push
```

Now open Github and verify the changes you have made:



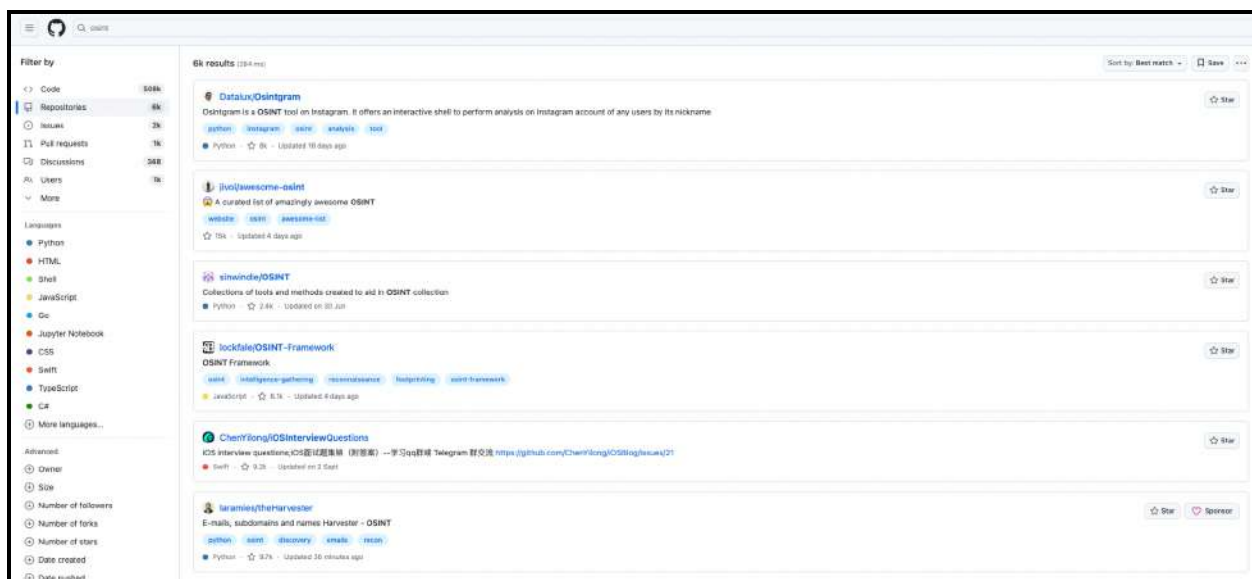
If you are in doubt about whether you have made the latest changes, check the status of the repository with this command:

```
PROBLEMS OUTPUT PORTS COMMENTS DEBUG CONSOLE TERMINAL
gitpod /workspace/linux-for-OSINT-21-day (main) $ cd Day_19
gitpod /workspace/linux-for-OSINT-21-day/Day_19 (main) $ cd linux_for_osint_20_day
gitpod /workspace/linux-for-OSINT-21-day/Day_19/linux_for_osint_20_day (main) $ git status
On branch main
Your branch is up to date with 'origin/main'.

nothing to commit, working tree clean
gitpod /workspace/linux-for-OSINT-21-day/Day_19/linux_for_osint_20_day (main) $
```

## git status

There are about a dozen different git commands. But the most important thing you should realize from this tutorial is that working with Github repositories using the command line is easy and convenient.



Github is not only a service for organizing collaboration on software development. It is also a huge platform for free distribution of open source software and technical knowledge.

You can also find an active OSINT-enthusiast community there, including thousands of users and repositories.

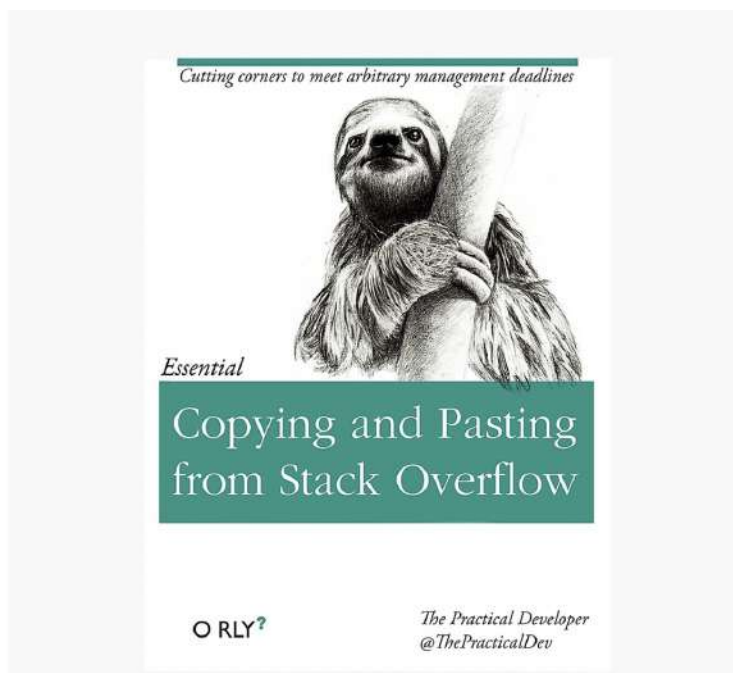
You've probably already created a Github profile to authorize Gitpod. Feel free to use it to interact with the global OSINT community:

- give stars to repositories you like. Pay special attention to good and little-known projects that have very few stars.
- If you encounter any errors while running any tools, feel free to write about them in Issues.
- If you have something useful, even if it's just a list of tools or a cheat sheet with useful commands, feel free to put it in the public repository.
- and, of course, don't forget to subscribe to my profile (<https://github.com/cipher387>).

# Day 20. Tools to make Linux easier to use

In this book, I have already written about tools that can help you write commands in Linux. For example, in the chapters on **Sed** and **Awk**, JSON and XML. Today we will talk about more universal tools.

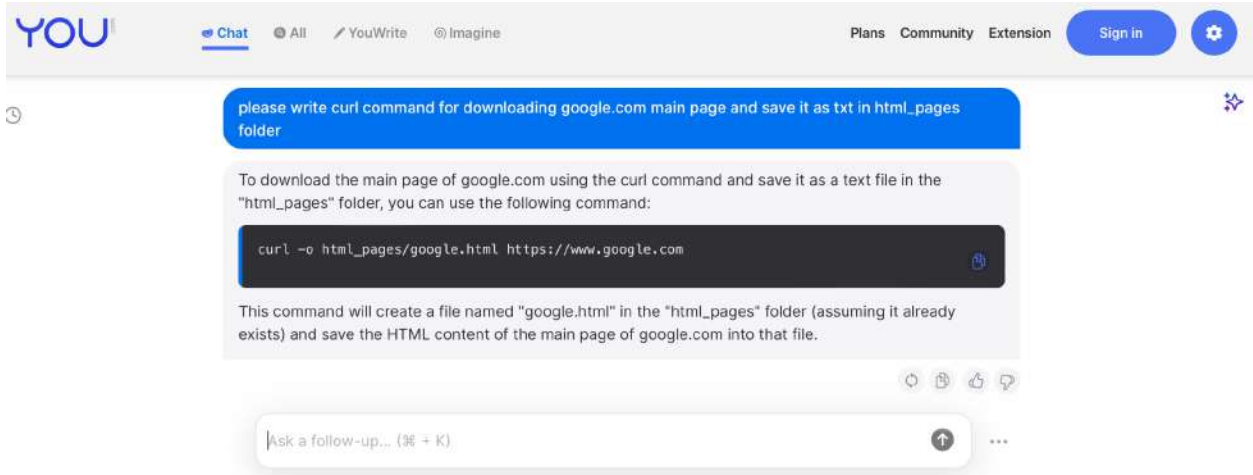
**The most important thing you should know about the Linux command line is that you need to know almost nothing about it to successfully solve your problems using it.**



If you have read at least 1-2 of the past lessons (days) and understand where to put files and where to enter commands, you can do a lot with the help of Google and StackOverflow. Really a lot.

## AI tools

Using AI assistants to write code has become very popular in the last year and a half. Many popular chatbots can generate both individual commands and full-fledged bash scripts:



ChatGPT (<https://chat.openai.com>)

Perplexity.ai (<https://www.perplexity.ai>)

Google Bard (<https://bard.google.com>)

You.com (<https://you.com/>)

They are exceptional at saving time on simple tasks, but when dealing with more complex issues, they can make mistakes and lead down the wrong path. Be careful.

showing all, navigate: ← explain crontab(1) → explain shell syntax

```
for user in $(cut -f1 -d: /etc/passwd); do  
    crontab(1) -u $user -l  
done
```

**for name [ [ in [ word ... ] ] ; ] do list ; done**

The list of words following **in** is expanded, generating a list of items. The variable **name** is set to each element of this list in turn, and **list** is executed each time. If the **in word** is omitted, the **for** command executes **list** once for each positional parameter that is set (see **PARAMETERS** below). The return status is the exit status of the last command that executes. If the expansion of the items following **in** results in an empty list, no commands are executed, and the return status is 0.

maintain crontab files for individual users (Vixie Cron)

Before a command is executed, its input and output may be redirected using a special notation interpreted by the shell. Redirection may also be used to open and close files for the current shell execution environment. The following redirection operators may precede or appear anywhere within a simple command or may follow a command. Redirections are processed in the order they appear, from left to right.

#### Redirecting Output

Redirection of output causes the file whose name results from the expansion of word to be opened for writing on file descriptor n, or the standard output (file descriptor 1) if n is not specified. If the file does not exist it is created; if it does exist it is truncated to zero size

And don't forget that you can not only ask AI assistants to write code or a command from scratch, but explain what some command or script found on Stackoverflow does. You can also use the Explain Shell (<https://explainshell.com/>) for this purpose.

## write curl command for download google com main page

## ANSWER | PHIND V8 MODEL






To download the main page of Google.com using the `curl` command, you can use the following command:

```
curl https://www.google.com > google_main_page.html
```

Here's a step-by-step explanation of the command:

1. `curl` is the command-line tool used for transferring data with URLs. It supports various protocols including HTTP, HTTPS, FTP, FTPS, SCP, SFTP, and more <sup>1</sup>.
2. `https://www.google.com` is the URL of the webpage you want to download. This is the main page of Google.com <sup>2</sup>.
3. `>` is a redirection operator in Unix-like operating systems. It is used to redirect the output of a command to a file. In this case, it redirects the output of the `curl` command (which is the HTML content of the webpage) to a file named `google_main_page.html` <sup>2</sup>.
4. `google_main_page.html` is the name of the file where the HTML content of the webpage will be saved. If this file does not exist, it will be created. If it does exist, it will be overwritten <sup>2</sup>.

After running this command, you will have a file named `google_main_page.html` in your current directory. This file contains the HTML content of the Google.com main page. You can open this file in a web browser to view the webpage, or you can open it in a text editor to view the HTML source code <sup>2</sup>.


Rewrite     

## SUGGESTIONS

Can I download a specific section of the Google.com main page using 'curl'?


Ask a followup question

SOURCES  


1  stackoverflow.com > questions > 48133080 > how-to-d...  
How to download a Google Drive url via curl or wget  
13 Answers Sorted by: 118 How about this method? When the file is such large size, Google returns a code for downloading the file. You.

2  www.networkworld.com > article > 957290 > using-curl...  
Using curl and wget commands to download pages from web sites

Probably the most obvious thing to do with the curl command is to download a page from a web site for review on the command.

3  linuxize.com > post > curl-command-examples

Curl Command in Linux with Examples | Linuxize  
The syntax for the curl command is as follows: curl [options] [URL...]  
In its simplest form, when invoked without any option, curl displays the specified.

4  www.digitalocean.com > community > tutorials > workfL...

How to Download Files with cURL | DigitalOcean  
Step 1 — Fetching remote files Out of the box, without any command-line arguments, the curl command will fetch a file and display its contents.

1 2 3 4 5

Separately, I would like to pay your attention to the AI assistant Phind (<https://www.phind.com/>), which not only write commands, but also provides explanations with sources on various reference sites like StackOverflow.

## Code checkers

StackOverflow users, AI assistants and everyone who writes code sometimes make mistakes. Special tools will help you to understand where exactly the error is.

# ShellCheck

finds bugs in your shell scripts.

You can cabal, apt, dnf, pkg or brew install it locally right now.

Paste a script to try it out:

```
#!/bin/bash
## Example: ShellCheck can detect some higher level semantic problems
while getopts "nf:" param
do
  case "$param" in
    f) file="$OPTARG" ;;
    v) set -x ;;
    *) ;;
  esac
done
case "$file" in
  *.gz) gzip -d "$file" ;;
  *) ;;
esac
```

```
$ shellcheck myscript

Line 6:
case "$param" in
^-- SC2213 (warning): getopts specified -n, but it's not handled by this 'case'.
^-- SC2220 (warning): Invalid flags are not handled. Add a *) case.

Line 8:
v) set -x ;;
^-- SC2214 (warning): This case is not specified by getopts.

Line 13:
*.gz) gzip -d "$file" ;;
^-- SC2221 (warning): This pattern always overrides a later one on line 15.
```

ShellCheck (<https://www.shellcheck.net/#>) - simple online tool that help to find bugs in shell scripts. You can also run it locally using the command line.

```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
-W NUM          --wiki-link-count=NUM  The number of wiki links to show, when applicable
-x             --external-sources     Allow 'source' outside of FILES
              --help      Show this usage summary and exit

@ gitpod /workspace/Linux-for-OSINT-21-day (main) $ shellcheck Day_20/wrong_code.sh

In Day_20/wrong_code.sh line 6:
case "$param" in
^-- SC2213 (warning): getopts specified -n, but it's not handled by this 'case'.
^-- SC2220 (warning): Invalid flags are not handled. Add a *) case.

In Day_20/wrong_code.sh line 8:
v) set -x ;;
^-- SC2214 (warning): This case is not specified by getopts.

In Day_20/wrong_code.sh line 13:
*.gz) gzip -d "$file" ;;
^-- SC2221 (warning): This pattern always overrides a later one on line 15.

In Day_20/wrong_code.sh line 15:
*.tar.gz) tar xzf "$file" ;;
^-- SC2222 (warning): This pattern never matches because of a previous pattern on line 13.

In Day_20/wrong_code.sh line 19:
if [[ "$(uname)" == "Linux" ]]
^-- SC2193 (warning): The arguments to this comparison can never be equal. Make sure your syntax is correct.

For more information:
https://www.shellcheck.net/wiki/SC2193 -- The arguments to this comparison ...
https://www.shellcheck.net/wiki/SC2213 -- getopts specified -n, but it's no...
https://www.shellcheck.net/wiki/SC2214 -- This case is not specified by get...

@ gitpod /workspace/Linux-for-OSINT-21-day (main) $
```

Install:

```
sudo apt-get install shellcheck
```

And run the test:

```
shellcheck Day_20/wrong_code.sh
```

The most important reason for the loss of efficiency in Linux (especially among beginners) is syntax errors when entering commands. TheFuck (<https://github.com/nvbn/thefuck>) is a utility that corrects errors in the command you entered before. Let's install it and see how it works.



```
python 76x18
git: 'brnch' is not a git command. See 'git --help'.

Did you mean this?
    branch
→ thefuck git:(master) X fuck
git branch [enter/ctrl+c]
  diezcami-java
* master
  mcarton-git-aliases
  mreder-master
  scorpus-issue-221-tf-alias
  tthorner-sudo-rule-fix
  underscorejho-master
→ thefuck git:(master) X puthon
No command 'puthon' found, did you mean:
  Command 'python' from package 'python3' (main)
  Command 'python' from package 'python-minimal' (main)
```

Unfortunately, at the moment this tool failed to install on Gitpod (due to certain issues with Python customizations) and I'm showing you a picture from the official documentation.

## Bash history

Linux has an command that allows you to see a history of what you've typed in the terminal.

```
PROBLEMS OUTPUT PORTS DEBUG CONSOLE TERMINAL
● gitpod /workspace/linux-for-0SINT-21-day (main) $ history
1 cd Day_3
2 cd Day_14
3 sudo apt update
4 sudo apt install vim
5 vim --version
6 find txt_files/ -name "*.txt" -type f
7 find -name "*.txt" -type f
8 cd Day_17
9 pip install ddgr
10 ddgr -h
11 pip install netlas
12 netlas -h
13 cd ..
14 cd Day_4
15 bash archive_files.sh
16 cd Day_13
17 jq -h
18 jq .items[].data.http.title json_test.json
19 jq .items[].data.http.title json_test.json
20 jq data.http.title json_test.json
21 jq [].data.http.title json_test.json
22 jq [0].data.http.title json_test.json
23 jq data[0] json_test.json
24 jq [].data[0] json_test.json
25 cd Day_13
26 jq [].data[0] json_test.json
27 [0].data jq [].data[0] json_test.json
28 jq [0].data json_test.json
29 jq [] json_test.json
30 jq [[]] json_test.json
31 jq [[]].data json_test.json
32 jq valid json_test.json
33 jq [].valid json_test.json
34 apt-get install shellcheck
35 sudo apt-get install shellcheck
```

Just type at the command line:

**history**

Don't forget that you can save the results of running the command to a file and do text searches with Grep.

That's it for today, but there are many more different tools to make working with the Linux command line and writing bash scripts easier. But, unfortunately, the format of this book does not allow to cover all of them. In fact, such a conclusion can be written for all the previous chapters.

# Day 21. Which Linux distribution is better to use?

The short answer is the one that you are most comfortable working with and the one that you put the most effort into setting up. Try different distributions. But I would still recommend that you start with the ones **that are actively developing at the moment and have a large community** of users around them.

Here are some examples of distributions that were created for cybersecurity researchers:

**Tsurugi Linux Acquire** ([https://tsurugi-linux.org/tsurugi\\_acquire.php](https://tsurugi-linux.org/tsurugi_acquire.php)) - lightweight distribution (the installation file is just over 1 GB, there is a 32-bit version, very good for running from a USB device) with many pre-installed tools for OSINT.

**Kali Linux** (<https://www.kali.org/get-kali/>) - the world's most popular distribution for pentest and ethical hacking.

**CSI Linux** (<https://csilinux.com>) - complete "cyber forensic" platform based on Ubuntu.

**BlackArch** (<https://blackarch.org>) - lightweight (Slim ISO 5.5 Gb) expansion to Arch Linux for penetration testers.

Here are a couple more long lists of Linux distributions for cybersecurity professionals:

Distro Forensics Github repository (<https://github.com/CScorza/DistroForensics>)

Linux for OSINT Twitter list ([https://twitter.com/cyb\\_detective/status/1441686068173033473](https://twitter.com/cyb_detective/status/1441686068173033473))

**But if you work in cybersecurity, that doesn't mean you really need to use distributions that are designed for cybersecurity professionals. It's not necessary at all.**

And there is absolutely no way to say that one distribution is better than another. After all, the answer to that question is different for each person and changes over time.



## Latest VM Images

<b>Fedora 39</b> VirtualBox Image VMware Image	<b>Ubuntu 23.10</b> VirtualBox Image VMware Image	<b>Kali Linux 2023.4</b> VirtualBox Image VMware Image
<b>Debian 12.0.0</b> VirtualBox Image VMware Image	<b>Lubuntu 23.04</b> VirtualBox Image VMware Image	<b>Ubuntu 23.04</b> VirtualBox Image VMware Image
<b>Manjaro 22.0</b> VirtualBox Image VMware Image	<b>Linux Mint 21</b> VirtualBox Image VMware Image	<b>Rocky Linux 9.0</b> VirtualBox Image VMware Image

You can use Linux as a virtual machine. Download images of different distributions in LinuxVMImages website (<https://www.linuxvmimages.com/>).

But think about other forms of usage such as VPS (virtual private server). The cost of renting them starts at a few dollars a month (few [examples](#) from \$3.99 to \$15) or bootable USB drive (example of instruction for Ubuntu and Windows <https://ubuntu.com/tutorials/create-a-usb-stick-on-windows#1-overview>).

```
Pause Reset Exit Send Ctrl-Alt-Del Send Alt-Tab Get CD-ROM image Insert floppy image Save State Load State Memory Dump Capture network traffic Disable mouse Lock mouse Go fullscreen
Take screenshot Mate Scale: 100 %

trapno=0000000d err=00000000 EIP=0805379c
INTO exception:
si_signo=11 si_errno=0 si_code=128 si_addr=0x00000000
trapno=00000004 err=00000000 EIP=080537c2
OUTB exception:
si_signo=11 si_errno=0 si_code=128 si_addr=0x00000000
trapno=0000000d err=00000000 EIP=080537e3
INB exception:
si_signo=11 si_errno=0 si_code=128 si_addr=0x00000000
trapno=0000000d err=00000000 EIP=08053805
REP OUTSB exception:
si_signo=11 si_errno=0 si_code=128 si_addr=0x00000000
trapno=0000000d err=00000000 EIP=08053835
REP INSB exception:
si_signo=11 si_errno=0 si_code=128 si_addr=0x00000000
trapno=0000000d err=00000000 EIP=08053862
HLT exception:
si_signo=11 si_errno=0 si_code=128 si_addr=0x00000000
trapno=0000000d err=00000000 EIP=08053880
tests% ls
reference.test test-1386
tests% cd ..
/root% ls
test.lua tests
/root% _

Running: 2m 11s
Speed: 0.4 mIPS
Avg speed: 2.0 mIPS

IDE device (CD-ROM)
Sectors read: 3025
Bytes read: 6195200
Sectors written: 0
Bytes written: 0
Status: Idle

VGA
Mode: Text
Resolution: -
BPP: -

Mouse: No

This is the serial console. Whatever you type or paste here will be sent to com1
```

Maybe after reading this book, Gitpod or even just browser-based versions of Linux will be enough for you:

<https://copy.sh/v86/?profile=linux26>

<https://www.masswerk.at/jsuix/index.html>

<https://bellard.org/jslinux/vm.html?cpu=riscv64&url=fedora33-riscv.cfg&mem=>

You might be comfortable using Linux on an Android smartphone using terminal emulators such as Termux (<https://termux.dev/en/>).

You will never know what's best for you until you try to solve your personal everyday problems.

## Does it make sense to use Linux as your primary operating system

To be honest, I've only worked on a MacBook for 9 years now. And I do much of my work in macOS.

I use Linux several times a week to automate various tasks. But there are so many people who use it all the time and there are several reasons why this is a good practice.



Opendesktop.org

## Libre Cloud Services.

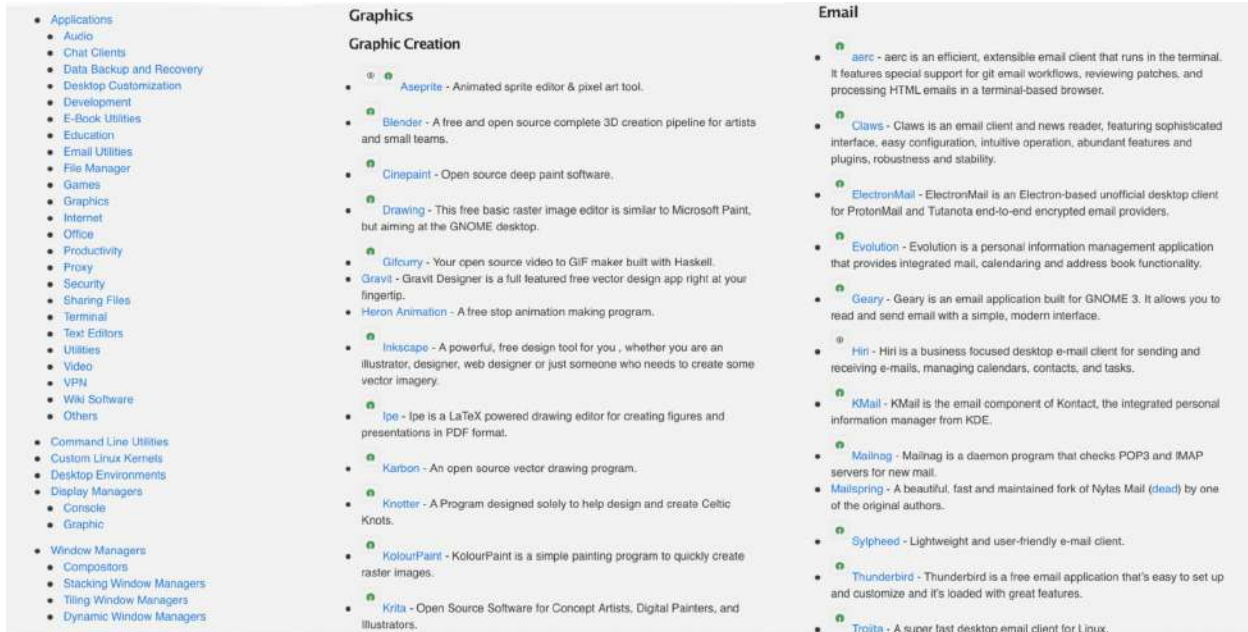
Opendesktop.org is a libre platform providing free cloud storage, online office editing, contacts & calendar tools, personal chat and messaging, as well as project development and product publishing to anyone who values freedom and openness.

Get a free account to enjoy true freedom with all the services below:



Firstly, it is an opportunity to significantly reduce the influence of global corporations on your life. If this idea inspires you, you can also opt out of Microsoft and Google online services like calendar, maps, cloud file storages, as well as streaming services and social networks. You can find decentralised open source analogues for all this on OpenDesktop (<https://www.opendesktop.org>).

(But keep in mind that you won't get rid of the influence of corporations on your life until the end, as the leading technology companies spend a lot of money to support Linux foundation and Open Source community).



Secondly, it's a huge selection of FREE, yet LEGAL software for a wide variety of purposes. Check out the **Awesome Linux Software** (<https://github.com/luong-komorebi/Awesome-Linux-Software/>) list, which has thousands of links.

By the way, many of these apps also have versions for Windows and macOS.

README CC0-1.0 license

# Awesome Shell

A curated list of awesome command-line frameworks, toolkits, guides and gizmos. Inspired by awesome-php. This awesome collection is also available on [Unix-Shell.ZEEF.com](http://Unix-Shell.ZEEF.com).

- [Shells](#)
- [Command-Line Productivity](#)
  - [Directory Navigation](#)
- [Customization](#)
- [For Developers](#)
- [System Utilities](#)
- [Downloading and Serving](#)
- [Multimedia and File Formats](#)
- [Applications](#)
- [Games](#)
- [Shell Package Management](#)
- [Shell Script Development](#)
- [Guides](#)
- [Awesome Zsh](#)
- [Awesome Fish](#)
- [Other Awesome Lists](#)

29.4k stars  
 778 watching  
 2.4k forks  
 Report repository

**Releases**  
 No releases published

**Packages**  
 No packages published

**Contributors** 266  
 + 252 contributors

You can see what a huge number of command line utilities exist if you open the repository Awesome Shell (<https://github.com/alebcay/awesome-shell>). It includes almost 400 links (and these are only the most useful and well-known utilities).

Third, if you do need some applications that only exist on Windows, you can still run them on Linux using launchers such as:

Wine (<http://winehq.org>)

CrossOver ( <http://codeweavers.com/crossover>)

Unfortunately, they are not always convenient and often produce errors. For example, I recently failed to run a programme for viewing dental images on Wine. But in many cases, they do work.

Fourth, Linux allows you maximum flexibility in customizing your desktop GUI (incomparable to Windows and Mac). You can start by trying one of the popular Linux Desktop Environments:

Gnome 3 (<https://www.gnome.org/gnome-3/>)

KDE (<https://www.kde.org/>)

Cinnamon (<http://developer.linuxmint.com/projects.html>)

Mate (<http://mate-desktop.com/>)

Xfce (<http://www.xfce.org>)

## What to do next?

Practice as much as possible. Keep doing OSINT investigations and try to automate tasks with Linux utilities. Don't be afraid to make mistakes and don't hesitate to use Stackoverflow and AI tools described in **Day 20**.

Don't forget to subscribe to me on Twitter, Mastodon, Telegram, BlueSky, Substack, Medium to get regular news about new OSINT tools, and not to miss new books. All links can be found at [cybdetective.com](https://cybdetective.com).

Once you've had at least a little bit of practice and you want to learn something new again, you can also read "Python for OSINT. A 21-day course for beginners" (<https://github.com/cipher387/python-for-OSINT-21-days>). It's also distributed freely (donate what you want and if you want).

Also remember that the global open source community is growing due to the small efforts of each of the tens of millions of enthusiasts around the world.

Don't forget to write about open source tools in your social networks, support developers with good reviews and donations, and if you like programming, don't hesitate to contribute to various projects on Github. Even if you don't know much yet, you can help with testing or writing documentation.

## Application. Is it possible to do the same thing on Windows?

Yes, it is possible and plenty of people do it. But as mentioned above, it's easier to run Linux on VM or VPS than to think about how to do the same thing on Windows.

The most justifiable reason to run Linux utilities on Windows - it is to combine some complex automations created with PowerShell or VBA with automations based on Linux utilities. In that case, here are some instructions on how to do it:

How to install bash in Windows from HackerNoon

(<https://hackernoon.com/how-to-install-bash-on-windows-10-lqb73yj3>)

How to run Linux software on Windows from GeeksForGeeks

(<https://www.geeksforgeeks.org/how-to-run-linux-software-on-windows/>)