

MODULE

Introduction à l'informatique

Objectifs, contenu et organisation du module

1re année géomatique

proposition pour 2027-2028

Objectif du cours

Consolider ou acquérir les bases informatiques nécessaires aux autres enseignements, en particulier ceux de programmation et les TD utilisant Python.

Demandé dans les autres modules et cours	(re)vu dans ce module
« ouvrez le projet fourni »	travailler dans un éditeur de code, lire une arborescence
« installez Python et numpy »	créer un environnement et le réinstaller ailleurs
« le script lit <code>donnees.csv</code> »	manipuler des fichiers depuis Python
« rendez votre code »	versionner avec git, partager un dépôt, documentation README.md

Objectifs du module liés à la programmation

Maîtriser la forme d'un projet de code : documentation, organisation des fichiers, bibliothèques et environnements.

	ce cours	cours de programmation
Quel algorithme choisir ?		oui
Comment écrire cette boucle ?		oui
Où mettre ce fichier ?	oui	
Comment lancer le script ailleurs ?	oui	

Les trois compétences du module

Éditer

éditeur de code, arborescence de projet

Versionner

git : enregistrer, revenir, partager

Structurer

README, environnement, ligne de commande

Des notions de culture informatique au fil de l'eau, rattachées aux enseignements et projets en cours.

Culture informatique

ordres de grandeur, sécurité informatique, du programme au logiciel

Organisation : sept séances de deux heures

Chaque séance alterne explications courtes et TD sur machine.

	Sujet	Type
1	Logiciels, fichiers, terminal et premier programme	cours
2	Éditeur de code, Markdown et git local	cours
3	Environnements, fichiers en Python, ligne de commande	cours
4	Premier projet : une animation, du notebook au programme	projet
5	Matériel, réseau ; mots de passe, clés, secrets	cours
6	La forge, sur le dépôt du projet 4	cours
7	Second projet : un effet pour l'animation	projet

Attention. Une partie de la séance 1 reprend des notions surement vues au lycée et/ou prépa.

Cours 1

Version 2, proposition de travail

1re année géomatique
2027-2028

Les fichiers du cours

Une archive par séance, un dossier par TD. Aujourd'hui, tout est dans `cours1/`.

```
cours1/          un dossier par TD, dans l'ordre de la séance
├─ 1a_formats/
├─ 1b_terminal/
├─ 1c_programme/
└─ 1d_notebook/
```

Dans chaque dossier, la feuille du TD et son guide détaillé, en PDF.

Contenu de la séance

Trois parties, des logiciels et fichiers jusqu'au notebook.

Partie	Nature	Durée
Logiciels, fichiers et stockage	cours et TD 1a	30'
Terminal et premier programme	cours et TD 1b, 1c	50'
Un notebook et la syntaxe de Markdown	cours et TD 1d	15'

Attention. La première partie de révision avance vite : posez vos questions tout de suite, tout ce qui suit s'appuie dessus.

Durées indicatives.

Logiciels, fichiers et stockage

Ce qu'un logiciel fait, où il s'exécute, le rôle des fichiers et les disques où ils sont enregistrés.

Logiciel : définition et termes courants

Définition : logiciel

Ensemble des programmes, procédés et règles, et éventuellement de la documentation, relatifs au fonctionnement d'un ensemble de traitement de données.

Source : *Journal officiel* du 22/09/2000 (vocabulaire de l'informatique). Le seul de ces mots à avoir une définition officielle.

Autres termes courants, qui désignent chacun un type de logiciel.

application

app

webapp

OS

driver

Les définitions d'usage : Grand dictionnaire terminologique de l'OQLF.

Le système d'exploitation

Un programme ne s'adresse pas directement au matériel : il passe par le système.



Vos programmes

LibreOffice, un navigateur, votre script

« ouvre releve.csv » ↓ ↑ le contenu



Système d'exploitation

Windows, macOS, Linux

« écris ces octets » ↓ ↑ les octets lus sur le disque



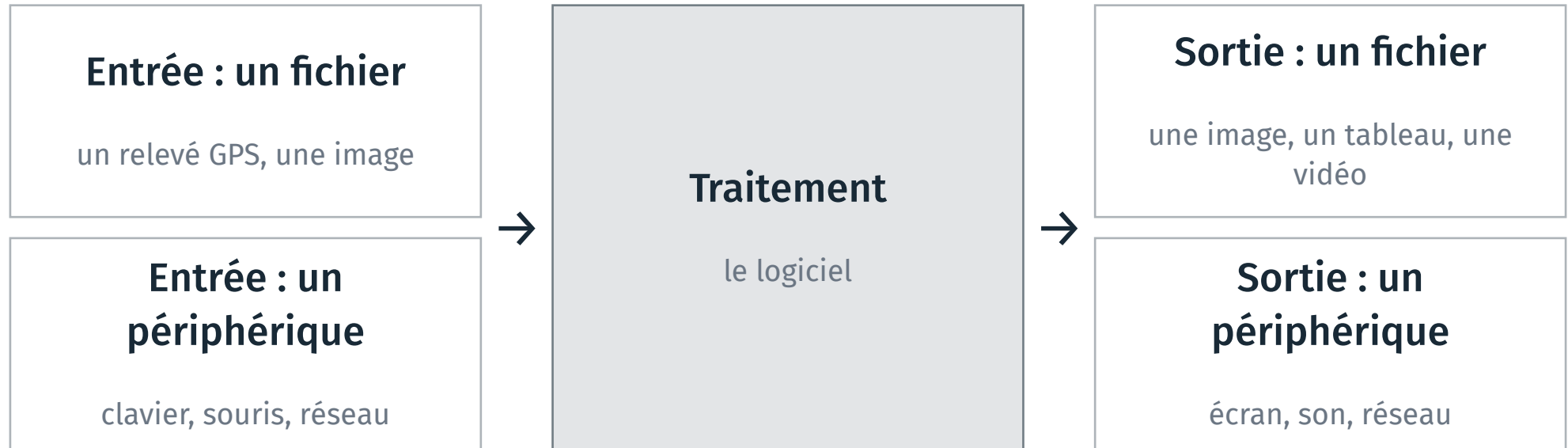
Matériel

processeur, mémoire, disque, réseau

Quel système d'exploitation tourne sur votre téléphone ?

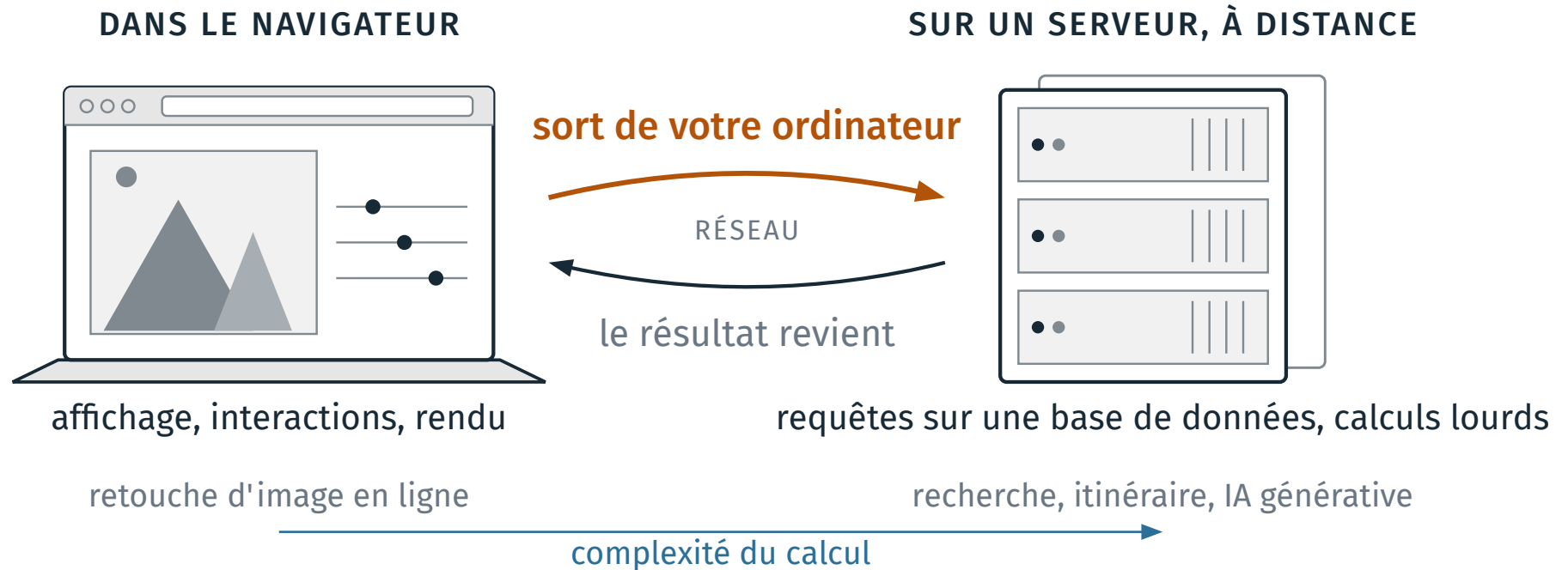
Entrées et sorties d'un logiciel

Ce qu'un logiciel reçoit et ce qu'il produit sont de deux natures : un fichier, qui se conserve, ou un flux vers un périphérique, qui ne garde rien.



Un fichier sert à conserver un résultat et à l'échanger : avec un autre logiciel, avec une autre machine, ou avec quelqu'un d'autre.

Où s'exécute une application web ?



Utilisation / utilité d'un fichier

Un fichier conserve un résultat après l'arrêt du programme : un état de travail à reprendre, ou un document final à transmettre.

Ce qu'il permet	Quand cela vous servira
Conserver un résultat	relire dans une semaine ce que le programme a produit
Passer d'un logiciel à l'autre	le tableau écrit par l'un, ouvert par l'autre
Changer de machine	de votre poste à celui de la salle, et retour
Le remettre à quelqu'un	un rendu, ou le dépôt partagé du cours 6

D'où la suite de cette partie : reconnaître le type d'un fichier, et savoir sur quel disque il est enregistré.

Fichier, extension et type de fichier

Le type d'un fichier (texte, vidéo...) est indiqué par son **extension**, la fin du nom après le dernier point. Le système s'en sert pour choisir le logiciel à lancer, mais elle reste une indication sur le contenu, pas une garantie.

releve_2026.csv
le nom, que vous choisissez l'extension

Attention. Windows masque les extensions qu'il connaît : raven.odt s'affiche raven.
Réglage à changer une fois, avant le TD 1a.

L'extension	Ce qu'elle décide
ce qu'elle fait	le système choisit le logiciel à lancer au double-clic
ce qu'elle ne fait pas	elle ne modifie aucun octet du fichier
comment on la change	en renommant le fichier, comme le reste du nom

Reconnaître un format à son extension

Pour chacune de ces extensions, dites de quel type de contenu il s'agit, et si le fichier est lisible dans un éditeur de texte.

.mp3	.mp4	.jpg	.png
.svg	.tif	.pdf	.odt
.xlsx	.csv	.zip	.exe
.py	.md	.json	.yaml

Reconnaître un format à son extension — réponses

.mp3 son, avec perte	.mp4 vidéo, la plus courante	.jpg photo, avec perte	.png image, sans perte
.svg image vectorielle : texte	.tif image, y compris GeoTIFF	.pdf document mis en page	.odt LibreOffice, archive ZIP
.xlsx Excel, archive ZIP	.csv tableau : du texte	.zip archive de fichiers	.exe programme Windows
.py code Python : du texte	.md documentation : du texte	.json données, réglages : texte	.yaml réglages : du texte

Six de ces seize formats sont du texte : ceux qu'on peut ouvrir dans un éditeur, comparer ligne à ligne et versionner. En bleu, les quatre que vous écrirez vous-mêmes dans ce module.

Quizz : vocabulaire associé aux chemins de fichier

Un chemin dit où trouver un fichier dans l'arborescence des dossiers. Plusieurs mots en désignent les parties : dites à quoi chacun correspond dans cet exemple.

`C:\Users\alice\Documents\raven.odt`

Le mot	Ce qu'il désigne dans l'exemple
la racine, ou le disque	
un nom de dossier	
le nom du fichier	
le chemin du fichier	
le dossier parent	

Quizz : vocabulaire associé aux chemins de fichier - Réponse

`C:\Users\alice\Documents\raven.odt`

la racine, ou le disque

trois noms de dossier

le nom du fichier, extension comprise

Le mot	Ce qu'il désigne dans l'exemple
la racine, ou le disque	c:\, le point de départ ; / sous macOS et Linux
un nom de dossier	Users, alice, Documents : trois, du plus large au plus précis
le nom du fichier	raven.odt, extension comprise
le chemin du fichier	tout, de la racine au fichier
le dossier parent	C:\Users\alice\Documents, le dossier qui le contient

Un chemin se lit de gauche à droite, de la racine au fichier ; chaque séparateur descend d'un dossier.

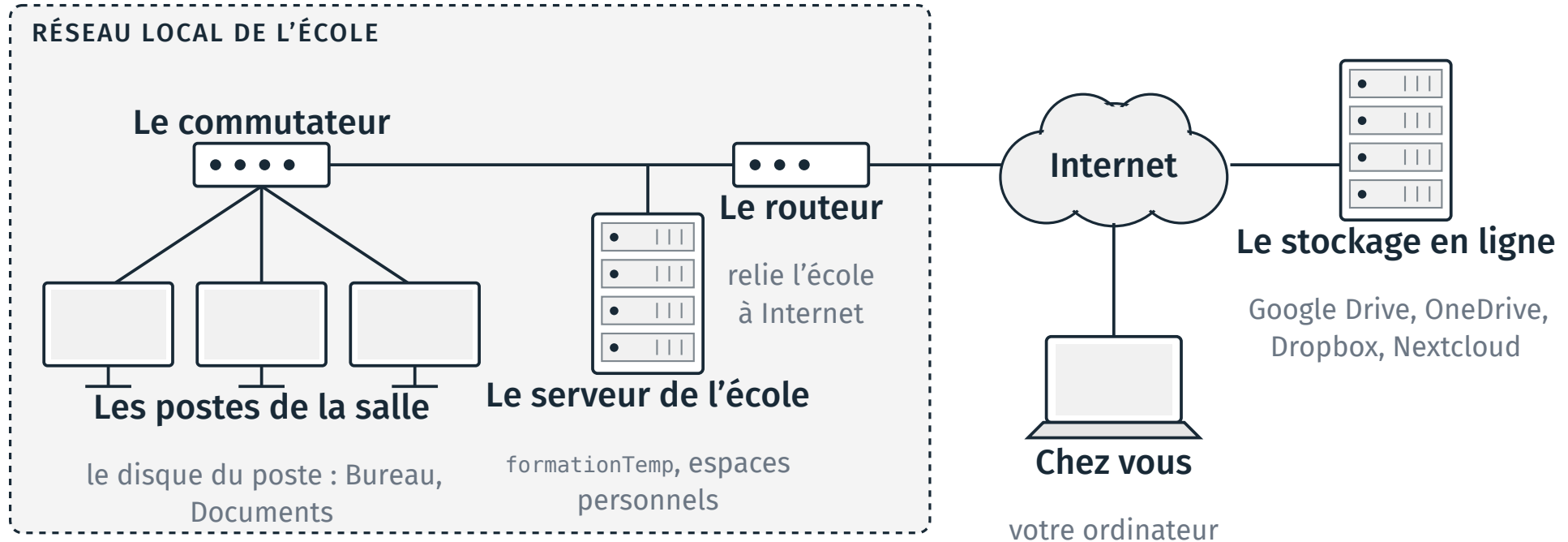
Les emplacements de stockage

Un fichier est enregistré sur un disque : celui du poste, ou celui d'une autre machine que le poste atteint par le réseau. L'explorateur les présente de la même façon.

Emplacement	Qui y a accès	D'une séance à l'autre
Bureau, Documents : disque du poste	votre session, sur ce poste	conservés, sur ce poste seulement
formationTemp : serveur de l'école	toute la promotion	modifiable par tous
espace personnel : serveur de l'école	vous, depuis tout poste	conservé
stockage en ligne : serveur hors de l'école	vous, et qui vous invitez	conservé, recopié sur le poste
clé USB	qui a la clé	ce que vous emportez

Les emplacements de stockage sur le réseau

Le serveur de l'école est relié aux postes par le réseau local. Le stockage en ligne est sur des serveurs hors de l'école, atteints par Internet.



Deux élèves dans le même dossier partagé

Un fichier du dossier partagé est le même pour toute la salle. Avec un éditeur de texte, le dernier qui enregistre remplace la version de l'autre.



Attention. Ne pas travailler dans `formationTemp`. Copier les fichiers sur le poste, et travailler dans la copie.

Reconnaître un emplacement réseau à son chemin

Un chemin qui commence par deux barres inversées désigne un dossier d'une autre machine : le nom qui suit est celui du serveur.

Chemin affiché dans la barre d'adresse	Où est le fichier
C:\Users\eleve\Desktop\info01\cours1	sur le disque du poste
\\serveur\formationTemp\info01-cours1.zip	sur le serveur nommé <code>serveur</code> , par le réseau
Z:\info01-cours1.zip	sur un lecteur réseau : un dossier de serveur auquel Windows a donné une lettre

Un lecteur réseau a une icône propre dans « Ce PC », qui distingue z: d'un disque du poste.

TRAVAUX DIRIGÉS

TD 1a Fichiers, formats et extensions
cours1/1a_formats/

≈ 15'

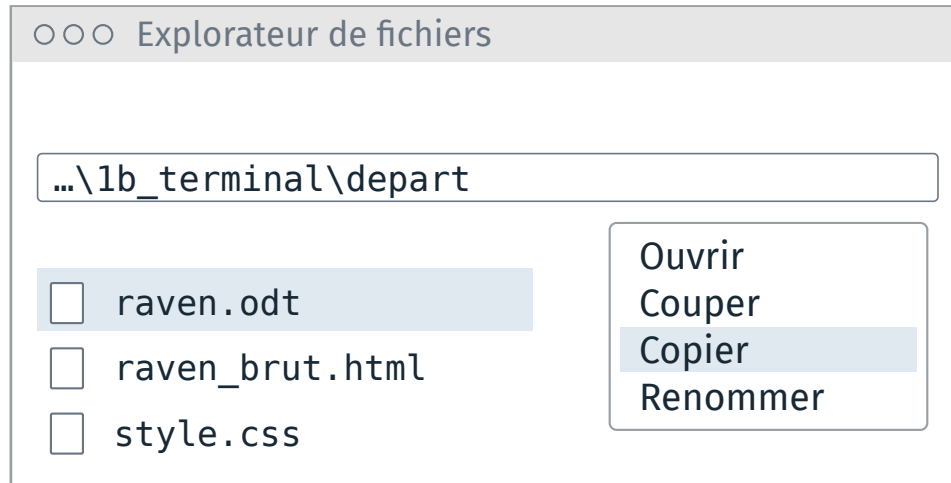
Terminal et premier programme

Se déplacer dans les dossiers et agir sur les fichiers en tapant des commandes, puis lancer un programme Python écrit dans un éditeur de texte.

Interface graphique et ligne de commande

Dans une interface graphique, on choisit l'opération parmi celles que la fenêtre affiche. En ligne de commande, on tape le nom de l'opération, qu'il faut connaître.

Interface graphique : l'explorateur de fichiers



La copie demande ensuite d'ouvrir `travail`, puis de choisir Coller au clic droit.

Ligne de commande : Git Bash

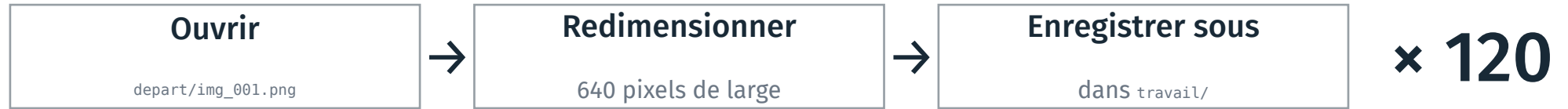


On tape la ligne, puis on appuie sur Entrée. `cp` abrège *copy*.

Une opération sur 120 fichiers

Dans Paint, il faut refaire les mêmes opérations pour chacune des 120 images.

Interface graphique : Paint, pour chaque image



Ligne de commande : Git Bash, pour toutes les images

○○○ Git Bash

```
$ magick mogrify -path travail -resize 640 depart/*.png
```

Le motif `*` désigne les 120 fichiers de `depart/`. Des logiciels graphiques ont aussi un traitement par lots, limité aux opérations qu'ils prévoient.

Comparaison des deux interfaces

L'interface graphique se prend en main plus vite. La ligne de commande répète une opération sur beaucoup de fichiers, et la commande se conserve pour être relancée.

	Interface graphique	Ligne de commande
Trouver une opération	dans les menus affichés	connaître son nom, ou lire l'aide
Voir le résultat	après chaque action	en le demandant
Premières utilisations	peu d'erreurs	des fautes de frappe, des options oubliées
Traiter 120 fichiers	120 fois la même opération	une commande, avec un motif
Refaire le travail plus tard	refaire chaque action	relancer la commande, gardée dans un fichier
Travailler sur un serveur	s'il a un bureau à distance	par <code>ssh</code> , vu au cours 5

Le terminal et l'interpréteur de commandes

Le terminal est la fenêtre : il affiche du texte et transmet ce qu'on tape. L'interpréteur de commandes lit chaque ligne et lance le programme qu'elle nomme.



```
Git Bash

eleve@POSTE-12 MINGW64 ~/Desktop/info01/cours1/1b_terminal
$ ls depart
auld_lang_syne.odt          raven_brut.html
auld_lang_syne_brut.html    raven_style.html
auld_lang_syne_style.html   raven_une_ligne.donnees
auld_lang_syne_une_ligne.donnees raven_une_ligne.txt
auld_lang_syne_une_ligne.txt style.css
raven.odt

eleve@POSTE-12 MINGW64 ~/Desktop/info01/cours1/1b_terminal
$
```

Le mot « terminal » désigne souvent les deux à la fois. Le langage des commandes est celui de l'interpréteur : cmd, PowerShell ou bash.

Les terminaux d'un poste Windows

Plusieurs terminaux sont installés sur les postes de la salle, chacun avec son interpréteur. Le module emploie Git Bash, installé avec git.

Terminal	Où le trouver	Interpréteur	Dans le module
Invite de commandes	menu Démarrer	cmd	celle d'Anaconda, pour Python et JupyterLab
Terminal Windows	menu Démarrer	PowerShell	non employé
Git Bash	clic droit dans un dossier	bash	à toutes les séances

bash est l'interpréteur des terminaux de Linux ; zsh, très proche, celui de macOS. Les commandes du module s'y tapent à l'identique.

Le choix de Git Bash

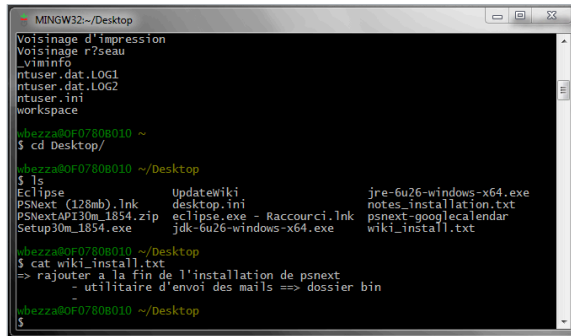
bash n'est pas fourni avec Windows. Le module l'emploie parce qu'il est l'interpréteur de Linux, installé sur la plupart des serveurs, et que Git Bash est installé avec git.

	cmd	PowerShell	bash, dans Git Bash
Sur les postes	avec Windows	avec Windows	avec git
Sous Linux et macOS	absent	à installer	bash ; zsh sous macOS
Statut	conservé pour la compatibilité	recommandé par Microsoft	logiciel libre, GNU
Lister, copier	dir, copy	Get-ChildItem, Copy-Item	ls, cp
Développeurs	non compté	23 %	49 %

Serveurs web : 92 % sous Unix, dont Linux (W3Techs, septembre 2026). Développeurs : enquête Stack Overflow 2025. Les commandes du TD 1b dans les trois interpréteurs sont en annexe de son guide.

Trois terminaux à l'écran

L'invite, au début de chaque ligne, indique quel interpréteur lit la commande.



```
MINGW32-~/Desktop
Voisinage d'impression
Voisinage r7seau
_viminfo
ntuser.dat.LOG1
ntuser.dat.LOG2
ntuser.ini
workspace

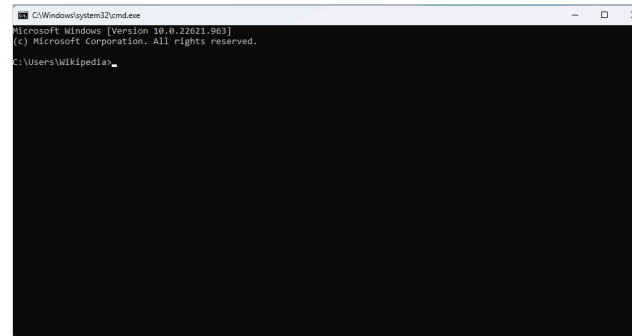
mbezza@MF07808010 ~
$ cd Desktop/

mbezza@MF07808010 ~/Desktop
$ ls
Eclipse                UpdateWiki             jre-6u26-windows-x64.exe
PSNext (128mb).lnk     desktop.ini            notes_installation.txt
PSNextAPI30m_1854.zip  eclipse.exe - Raccourci.lnk  psnext-googlecalendar
Setup30m_1854.exe      jdk-6u26-windows-x64.exe  wiki_install.txt

mbezza@MF07808010 ~/Desktop
$ cat wiki_install.txt
=> rajouter a la fin de l'installation de psnext
- utilitaire d'envoi des mails => dossier bin

mbezza@MF07808010 ~/Desktop
$
```

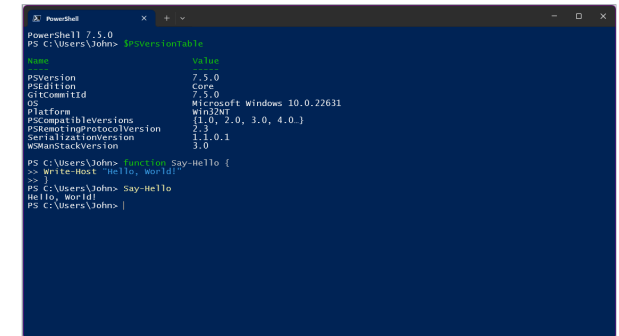
Git Bash :
utilisateur@machine, puis \$



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.22621.963]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Wikipedia>
```

Invite de commandes :
C:\Users\...>



```
PowerShell 7.3.0
PS C:\Users\Johns $PSVersionTable

Name                Value
-----
PSVersion           7.3.0
BuildVersion        7.3.0
GitCommitId         7.3.0
Platform            Microsoft Windows 10.0.22631
OS                  Win32NT
PSCompatibleVersions {1.0, 2.0, 3.0, 4.0}
SerializationVersion 1.1.0.1
WebResourceVersion   3.0

PS C:\Users\Johns function Say-Hello {
>> Write-Host "Hello, World!"
>> }
PS C:\Users\Johns Say-Hello
Hello, World!
PS C:\Users\Johns |
```

PowerShell :
PS C:\Users\...>

L'invite de Git Bash

Avant chaque commande, Git Bash affiche une invite : l'utilisateur, la machine et le dossier courant. La commande se tape après le \$.

```
○○○ MINGW64:/c/Users/eleve/Desktop/info01/cours1

eleve@POSTE-12 MINGW64 ~/Desktop/info01/cours1
$ ls
1a_formats/ 1b_terminal/ 1c_programme/ 1d_notebook/

eleve@POSTE-12 MINGW64 ~/Desktop/info01/cours1
$
```

Dans l'invite	Ce qu'il désigne
eleve	l'utilisateur
POSTE-12	la machine
MINGW64	Git Bash sous Windows
~	le dossier personnel, c: \Users\eleve
~/Desktop/...	le dossier courant
\$	l'attente d'une commande

Le / à la fin d'un nom signale un dossier. Après le TD 1c, une ligne (base) s'ajoute au-dessus : l'environnement conda actif.

La forme d'une commande

Une commande commence par le nom du programme à lancer. Suivent des options, qui commencent par un tiret, et des arguments, souvent des noms de fichiers.

Commande tapée	Programme	Options	Arguments
ls	ls		
ls -a depart	ls	-a	depart
cp depart/raven.odt travail/	cp		depart/raven.odt, travail/
python --version	python	--version	
ls --help	ls	--help	

Une option d'une lettre prend un tiret, une option en toutes lettres en prend deux. `--help` affiche l'aide de la plupart des commandes.

Les commandes de base : se déplacer

Chaque commande de base a son équivalent à la souris dans l'explorateur de fichiers.

Commande	Ce qu'elle fait	Dans l'explorateur
<code>pwd</code>	affiche le dossier courant	la barre d'adresse
<code>ls</code>	liste le dossier courant	la fenêtre ouverte
<code>ls dossier</code>	liste le dossier nommé	ouvrir ce dossier
<code>cd dossier</code>	descend dans le dossier nommé	double-clic sur le dossier
<code>cd ..</code>	remonte au dossier parent	flèche « dossier parent »

Les commandes de base : agir sur les fichiers

Commande	Ce qu'elle fait	Dans l'explorateur
<code>cp source destination</code>	copie un fichier	Ctrl + C, puis Ctrl + V
<code>mv source destination</code>	déplace ou renomme	glisser ; F2
<code>mkdir nom</code>	crée un dossier	Nouveau dossier
<code>rm fichier</code>	supprime, sans passer par la corbeille	Suppr
<code>start fichier</code>	ouvre avec le logiciel associé	double-clic

Le chemin d'un fichier en ligne de commande

En ligne de commande, un fichier se désigne par son chemin, écrit après le nom de la commande. Un chemin **absolu** part de la racine, un chemin **relatif** du dossier où l'on se trouve.

	Windows	macOS et Linux
Le séparateur de dossiers	la barre inversée <code>\</code> , <i>backslash</i>	la barre <code>/</code> , <i>slash</i>

```
C:\Users\alice\  
└─ cours1\  
  └─ 1a_formats\  
    └─ depart\  
      └─ raven.odt  
    └─ 1c_programme\  
      └─ altitudes.py
```

	Le chemin de raven.odt	Depuis
Absolu	C:\Users\alice\cours1\1a_formats\depart\raven.odt	la racine
Relatif	1a_formats\depart\raven.odt	cours1
Relatif qui remonte	..\1a_formats\depart\raven.odt	1c_programme

Sous macOS et Linux, l'autre séparateur : `/home/alice/cours1/1a_formats/depart/raven.odt`. Un projet qui n'écrit que des chemins relatifs se copie, se déplace et s'envoie sans rien changer : `C:\Users\alice` n'existe que sur un poste.

Les chemins dans Git Bash

Git Bash écrit les chemins à la façon de Linux : des barres obliques, et le disque `c:` devient `/c`.

Dans l'explorateur	Dans Git Bash	
C:\Users\eleve\Desktop\info01	/c/Users/eleve/Desktop/info01	chemin absolu
C:\Users\eleve	~	le dossier personnel
le dossier ouvert	.	le dossier courant
le dossier au-dessus	..	le dossier parent
..\1a_formats\depart	../1a_formats/depart	chemin relatif

Un chemin copié depuis la barre d'adresse de l'explorateur garde ses `\` : dans Git Bash, il s'écrit entre guillemets.

Fichiers cachés et motif *

Un nom qui commence par un point désigne un fichier caché : `ls` ne l'affiche pas, `ls -a` l'affiche.

```
$ ls travail
$ ls -a travail
./ ../
```

`.` et `..`, le dossier courant et son parent, sont les deux entrées cachées de tout dossier.

Dans un nom de fichier, `*` remplace n'importe quelle suite de caractères.

Commande	Noms désignés dans depart/
<code>ls depart/*.txt</code>	les deux fichiers <code>.txt</code>
<code>ls depart/raven*</code>	les cinq fichiers de The Raven
<code>cp depart/*.html travail/</code>	les quatre pages web

Git Bash remplace le motif par la liste des noms qui correspondent, puis lance la commande. Le TD 1c crée un fichier caché, `.bash_profile`, dans le dossier personnel ; le cours 2 un dossier caché, `.git`.

TRAVAUX DIRIGÉS

TD 1b Les fichiers du TD 1a en ligne de commande
`cours1/1b_terminal/`

≈ 15'

REPRISE DU COURS

Premier programme

Un programme Python est un fichier texte, que l'interpréteur python exécute.

Python, un programme en ligne de commande

`python` se lance dans un terminal, suivi du nom du fichier à exécuter. Il n'ouvre pas de fenêtre et écrit ses résultats dans le terminal. Tout terminal où Python est disponible convient.

Dans Git Bash

○○○ Git Bash

```
$ cd ~/Desktop/info01/cours1/lc_programme
$ python altitudes.py
moyenne : 129.0 m
```

Dans l'invite de commandes d'Anaconda

○○○ Anaconda Prompt

```
(base) ...>cd Desktop\info01\cours1\lc_programme
(base) ...\\lc_programme>python altitudes.py
moyenne : 129.0 m
```

L'invite de commandes d'Anaconda est `cmd`, avec `conda` déjà activé : `python` y lance le Python d'Anaconda sans réglage. Comme `ls` ou `cp`, `python` est une commande, qu'un script peut lancer.

Une commande désigne un fichier exécutable

`python` est le nom d'un fichier, `python.exe`. `bash` le cherche dans une liste de dossiers, la variable `PATH`, et lance le premier qu'il trouve.

Sans conda : le premier `python.exe` trouvé, surligné, est lancé

/usr/bin	ls, cp, bash...
/c/Windows/system32	cmd.exe...
/c/Python27	python.exe, Python 2.7

Après conda activate : les dossiers d'Anaconda passent en tête

/c/ProgramData/anaconda3	python.exe, Python 3
/usr/bin	ls, cp, bash...
/c/Windows/system32	cmd.exe...
/c/Python27	python.exe, Python 2.7

Un nom qui contient un `/` désigne un fichier précis, sans recherche : `/c/ProgramData/anaconda3/python.exe` lance le Python d'Anaconda. `type -a python` affiche les fichiers trouvés, dans l'ordre.

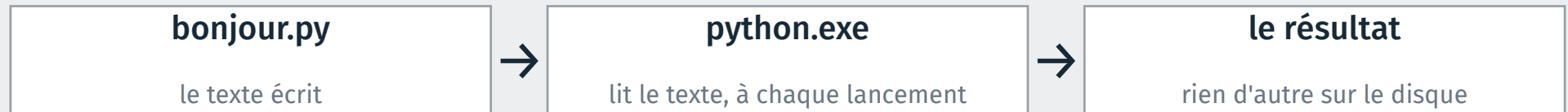
Compiler ou interpréter

Le processeur n'exécute que des instructions machine. Un compilateur traduit le programme une fois, un interpréteur lit son texte à chaque lancement.

Compilé : C, C++



Interprété : Python



L'interpréteur est lui-même un programme compilé : `python.exe` est écrit en C.

Compilé ou interprété : ce que chacun apporte

Le module emploie Python. Il se relance sans étape de compilation, s'essaie ligne à ligne, et ses bibliothèques de calcul font leurs calculs en code compilé.

	Compilé : C, C++	Interprété : Python
Après une modification	recompiler, puis lancer	relancer
Essayer une ligne	l'écrire dans un programme complet	la taper dans une session interactive
Ce qu'on donne à un autre	l'exécutable, qui se lance seul	le fichier .py, et Python installé
Sur un autre système	recompiler pour ce système	le même fichier
Un calcul en boucle	la référence	environ 50 fois plus lent
Une faute de frappe	signalée à la compilation	signalée au lancement

Vitesse : deux mesures publiées, dans les notes. numpy calcule en code compilé, « at near-C speeds » selon sa documentation.

Un fichier ou une session interactive

L'interpréteur python exécute un fichier enregistré, ou les lignes tapées une à une.

Un programme, écrit dans un éditeur de texte

Une session interactive, dans le terminal



```
Git Bash

$ python
Python 3.12.14 (main, Sep  2 2026, 23:27:36) [GCC 15.3.0] on
linux
>>> 128.4 + 131.0
259.4
>>> exit()
```

Le programme se relance à l'identique, la session ne garde rien sur le disque.

Un script de commandes

Un script est un fichier texte qui contient des commandes. `bash` les exécute dans l'ordre, comme si on les tapait. Le travail se refait en une commande, aussi souvent qu'il le faut.

`commandes.sh`, dans Notepad++

```
# Commandes du TD 1b, puis le programme du TD 1c.  
# Lancer depuis 1c_programme : bash commandes.sh  
mkdir -p copies  
cp ../1b_terminal/depart/*.html copies/  
ls copies  
python altitudes.py
```

Lancé dans Git Bash

Git Bash

```
$ bash commandes.sh  
auld_lang_syne_brut.html    raven_brut.html  
auld_lang_syne_style.html  raven_style.html  
moyenne : 129.0 m
```

Une ligne qui commence par `#` est un commentaire. `python` y est une commande comme les autres. L'équivalent pour l'invite de commandes est un fichier `.bat`, en annexe du guide du TD 1c.

TRAVAUX DIRIGÉS

TD 1c Écrire et lancer un programme
`cours1/1c_programme/`

≈ 20'

Un notebook et la syntaxe de Markdown

Un document qui réunit du texte, du code et ses résultats ;
le texte s'y écrit en Markdown.

Programmation littérale

Le texte, le code et son résultat tiennent dans un seul document. Les cours et les TD de Python du module emploient des notebooks, qu'il faut donc savoir lancer et exécuter.



Du texte, une cellule de code et sa sortie, puis du texte. Le noyau, un programme Python lancé par JupyterLab, exécute les cellules et garde leurs variables. Le terme « programmation littérale » est de Donald Knuth, 1984.

La syntaxe minimale de Markdown

Une cellule de texte s'écrit en Markdown : du texte brut, où quelques signes indiquent la mise en forme.

Ce qu'on tape	Ce qui s'affiche
# Altitudes, ## Les données	Altitudes Les données
deux lignes séparées par une ligne vide	deux paragraphes
du <i>*texte*</i> en **gras**	du <i>texte</i> en gras
- une puce	• une puce
1. une étape	1. une étape
[le site](https://jupyter.org)	le site
la variable <code>total</code>	la variable <code>total</code>

Le texte tapé reste lisible sans être affiché mis en forme. Le cours 2 y ajoute les tableaux et les images.

Lancer JupyterLab

JupyterLab est installé avec Anaconda, et se lance dans un terminal où conda est actif. Le module donne deux terminaux, pour ne pas être bloqué si le réglage de Git Bash manque sur un poste.

Depuis l'invite de commandes d'Anaconda

```
(base) ...>cd Desktop\info01\cours1\1d_notebook  
(base) ...>jupyter lab
```

Menu Démarrer, « Anaconda Prompt ». conda y est actif sans réglage : à essayer d'abord.

Le navigateur s'ouvre sur `localhost:8888`, le poste lui-même : le serveur de JupyterLab tourne sur le poste, et le terminal reste occupé tant qu'il tourne. `Ctrl + C` l'arrête. Le cours 5 y revient.

Depuis Git Bash

```
$ cd ~/Desktop/info01/cours1/1d_notebook  
$ jupyter lab
```

Le terminal du module, après le réglage du TD 1c.

TRAVAUX DIRIGÉS

TD 1d Un notebook dans JupyterLab
[cours1/1d_notebook/](#)

≈ 12'

À retenir

Un logiciel	transforme une entrée en sortie ; un fichier conserve le résultat
Une extension	nomme le type du fichier, elle ne modifie pas son contenu
Le disque du poste	garde vos fichiers ; le dossier partagé est à toute la salle
Le terminal	transmet des commandes à un interpréteur ; au module, Git Bash
Un chemin relatif	part du dossier courant ; .. remonte d'un dossier
Une commande	désigne un fichier exécutable, cherché dans <code>PATH</code>
L'interpréteur python	exécute le fichier enregistré sur le disque
Automatiser	un logiciel pour ce qu'il prévoit, un script de commandes pour répéter, un programme pour le reste
Un notebook	texte, code et résultats ; le noyau retient les variables