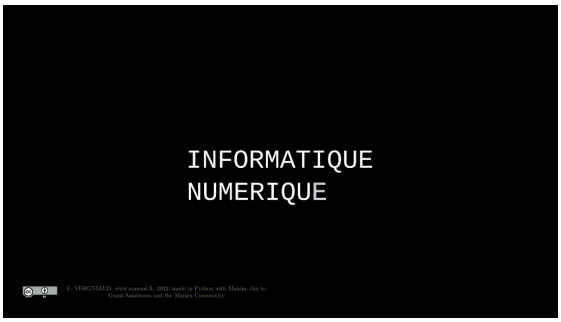
Alignement de séquences

1. Présentation du sujet

On dispose de deux chaînes de caractères : A, qui vaut `INFORMATIQUE`, et B, qui vaut `NUMERIQUE`. On aimerait mettre ces deux chaînes de caractères en correspondance de la manière suivante :

- On place les 2 chaînes l'une en desous de l'autre;
- Si les derniers caractères des deux chaînes coïncident, alors on passe aux caractères suivants;
- Sinon, on va ajouter un trou dans une des deux chaînes, symbolisé par un - et on passe aux caractères suivant.

Voici un exemple d'alignement optimal :



Dans cette situation on a besoin de 9 tirets, pas moins.

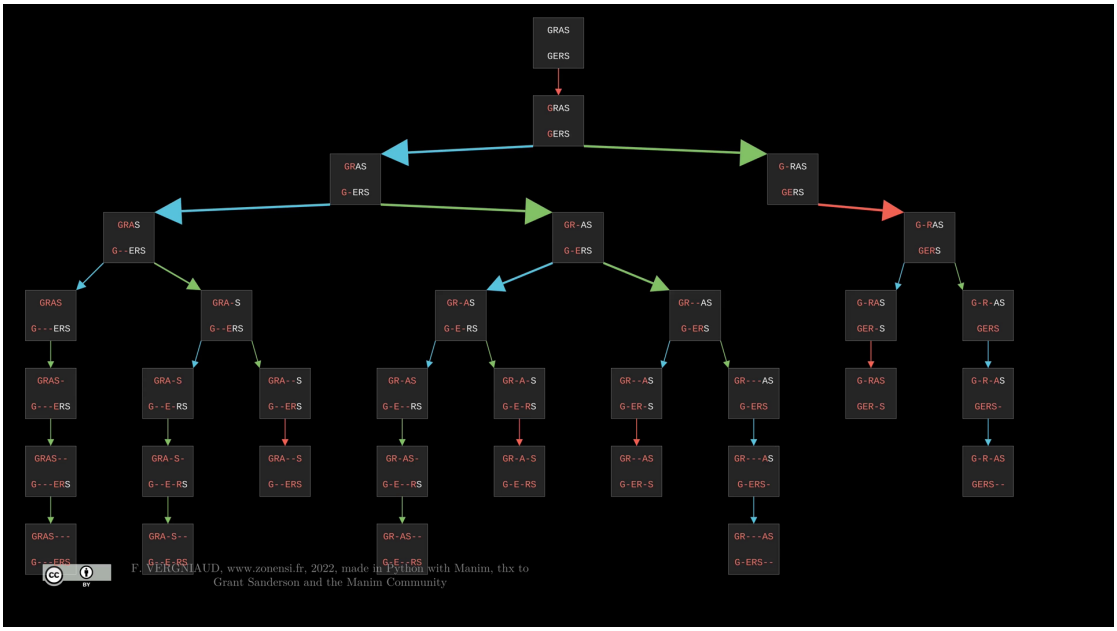
L'objectif est d'aligner le maximum de lettres (donc de mettre le moins de `-` possible). Ce n'est pas un problème simple, surtout quand les chaînes sont longues, comme pour les séquences d'ADN par exemple :

```
TTCACCAGAAAAGA--ACACGGTAGTTA-CGAG--TCCAATATT-GTTAA---ACC--G
TTCA-C-GAAAA-AGTA-ACGG--G---CCGA-TCTCCAATA--AG-T--GCGACCGAG
```

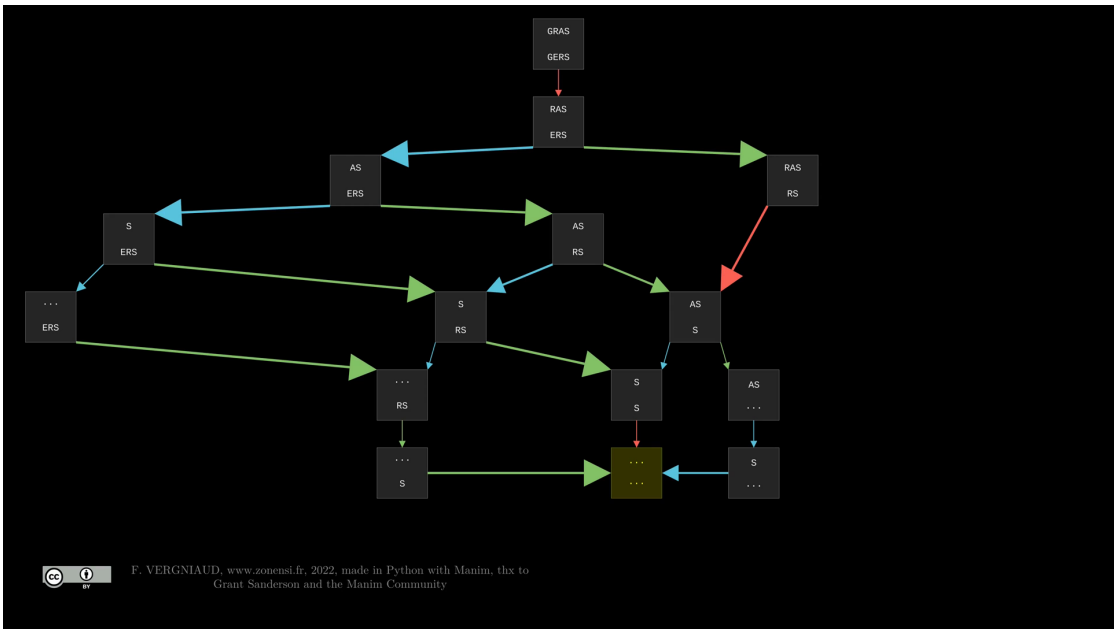
2. Résolution par une méthode récursive

Le principe est présenté dans la vidéo suivante :

Pour les mots `GRAS` et `GERS`, l'arbre obtenu est le suivant :



Après application d'une méthode dynamique (**Top Down**) (c'est-à-dire récursive avec mémoïsation), on obtient le graphe suivant :



3. Résolution par une méthode itérative

3.1. Activité débranchée

Considérons les deux chaînes de caractères `GENOME` et `ENORME`. Afin de chercher le nombre minimal d'insertion à effectuer, nous allons compléter le tableau suivant, avec la convention suivante : à l'intersection de la colonne `N` et de la première ligne `E` se trouve le nombre minimal d'insertion nécessaire pour aligner les chaînes `EN` et `GE`, c'est-à-dire 2 tirets. Par convention la première ligne et la première colonne correspondent à une chaîne vide.

	-	E	N	O	R	M	E
-							
G							
E							
N							
O							
M							
E							

Question

Enoncé

En quoi la méthode ci-dessus est-elle une méthode itérative (**Bottom Up**) ?

Réponse

1. Pour compléter le tableau, on va commencer par compléter la première ligne et la première colonne. Par exemple, la case de la première ligne correspondant à l'intersection de `R` et de `-` doit contenir le nombre minimal de tirets nécessaire pour aligner `ENOR` avec une chaîne vide, c'est-à-dire 4.

	-	E	N	O	R	M	E
-	0	1	2	3	4	5	6
G	1						
E	2						
N	3						
O	4						
M	5						
E	6						

E	N	O	R
-	-	-	-

2. Pour compléter ensuite le reste du tableau, il faut concevoir deux cas différents :

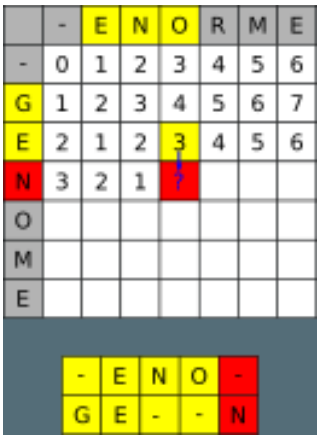
   a. Soit la case correspond aux deux même lettres, comme dans l'exemple ci-dessous :

	-	E	N	O	R	M	E
-	0	1	2	3	4	5	6
G	1	2	3	4	5	6	7
E	2	1	2	3	4	5	6
N	3	2	1	2	3	4	5
O	4	3	2	?			
M							
E							

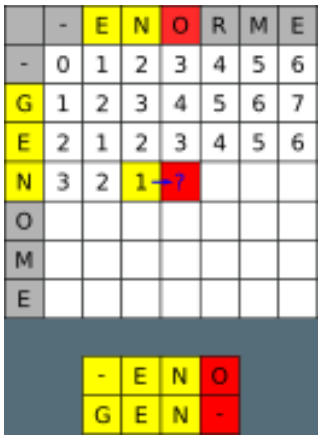
-	E	N	O
G	E	N	O

   Le meilleur alignement de `ENO` et de `GENO` contient autant de tirets que le meilleur alignement de `EN` et de `GEN`, donc ici 1.

   b. Soit les deux lettre sont différentes, et on peut considérer deux situations :



On considère la case comme étant une de l'alignement de `ENO` et `GE`, qui contenait 3 tirets, vers l'alignement de `ENO` et `GEN`, qui en contiendra donc **un de plus**, soit 4.



On considère la case comme étant une de l'alignement de `EN` et `GEN`, qui contenait 1 unique tiret, vers l'alignement de `ENO` et `GEN`, qui en contiendra donc **un de plus**, soit 2.

Comme nous cherchons un alignement minimisant le nombre de tirets, on va alors compléter la case avec la valeur 2, provenant de la situation de droite ci-dessus.

3. On termine alors de compléter le tableau :

	-	E	N	O	R	M	E
-	0	1	2	3	4	5	6
G	1	2	3	4	5	6	7
E	2	1	2	3	4	5	6
N	3	2	1	2	3	4	5
O	4	3	2	1	2	3	4
M	5	4	3	2	3	2	3
E	6	5	4	3	4	3	2

3.2. Application en Python

L'activité est disponible sous la forme d'un notebook capytale