# Révision des deux années !

Objectifs :

- Récapituler les notions de base au programme très rapidement
- Exemples / Applications

## 1 Programmation / Syntaxe Python

(en cas de doutes : regarder le document `ressources/info_commune_1a-nup.pdf`)

- **Types simples** : entiers (`int`), flottants (`float`), booléens (`bool`), chaînes de caractères (`str`)
- Affectation
- Opérateurs usuels (division entière `a // b`, modulo `a % b`).
- Distinction entre expression et instruction.
- `if`/`elif`/`else` : donner une exemple simple où l'utilisation de `elif` (et pas `if`) est importante.
- Boucle `for`, boucle `while` : quand utiliser un type de boucle plutôt qu'un autre ?
- **Fonctions**, variables locales, utilisation de `return`.
- Manipulation des chaînes de caractères, et des listes : **slicing**, mutable/non mutable
- Tableaux à plusieurs dimensions : listes de listes ou array `numpy`

## 2 Représentation des nombres

- Nombres flottants : **limites** (test d'égalité, absorption/cancellation)
- Entiers, positifs et négatifs : conversion base 2 / base 10 ; complément à 2

## 3 Algorithmes

Sur les listes python :

1) Chercher un élément dans une liste : parcours de toute la liste en $O(n)$ ou dans une liste triée par **dichotomie** en $O(\log_2(n))$
2) Maximum - **Indice du maximum**. Variante : travailler avec des objets plus compliqués que des entiers.
3) Calcul de moyenne et de variance... d'une liste de nombre

Numérique et Simulation :

(Modules : `math`, `random`, `numpy`, `matplotlib.pyplot`)

4) Intégration : rectangles, trapèzes...
5) Résolution d'équation : **Dichotomie** (sur une fonction), méthod de **Newton**
6) Équations différentiells : **méthode d'Euler** ; **vectorialisation**. Variantes
7) Résolution d'un système linéaire inversible : méthode de **Gauss** avec recherche partielle du pivot.

Sur les chaînes de caractères :

8) Recherche d'un mot dans une chaîne de caractères

## 4 Bases de données

- Relation, Attribut, Domaine (= type de données), **Schéma d'une relation**
- **Clé primaire**, **clé étrangère**

- Union, Intersection, Différence
- **Projection** SELECT ..., **Sélection** (=Restriction) WHERE ... /HAVING ..., Renommage AS, Jointure JOIN ... ON ..., Produit cartésien (et division)
- Agrégation : minimum (MIN), maximum (MAX), somme (SUM), moyenne (AVG), comptage (COUNT)

```
SELECT ...                  -- attributs et fonctions d'agrégation
FROM ...                    -- table(s)
    JOIN ... ON ...
WHERE ...                   -- sélection : condition (booléen) : AND, OR, NOT, ... IN ... , ... IS NONE
GROUP BY ...                -- important !
HAVING ...                  -- sélection : sur le résultat de GROUP BY
ORDER BY ... DESC/ASC
```

# 5 Concepts avancés

- Piles - Type de donnée abstrait : on s'interdit certaines opérations

9) Fonctions récursives : Pensez aux cas de base ; Complexité (cf. `ressources/TP-Recursivite.pdf`)
10) Tris tri par insertion, tri rapide, tri par fusion.

# 6 Algos classiques

11) Pgcd - Algorithme d'Euclide
12) Tester si un nombre est premier

# 7 Complexité

13) Cas simples : boucles, appels de fonction.
14) Diviser pour régner / Récursivité

# 8 Modélisation - Problèmes concrets

Pour représenter une situation concrète :

- Utiliser une liste python (1 dimension)
- Utiliser des matrices : listes de listes ou `array` numpy

On peut représenter : des images, des objets qui se déplacent sur une grille, une matrice, un graphe, des contraintes, ...