L'objectif de ces séances est de récapituler l'ensemble des savoir-faire de l'année.

Code de partage avec Capytale : d8e9-1683377

1. Les commandes prédéfinies

Ecriture mathématique	π	e	$\ln(5)$	e^3	$\sqrt{11}$	$\lvert -8\rvert$	$\leqslant$	$\geqslant$	$\neq$
Ecriture avec Python									

2. Afficher un résultat

   Si on écrit un programme dans l'éditeur et qu'on l'exécute, il est possible qu'il n'affiche rien (par exemple le calcul de $f(2)$ avec f une fonction prédéfinie). Quelle commande permet l'affichage d'un résultat ?

3. Demander une information

   Ecrire un programme qui demande un entier à l'utilisateur et renvoie son carré.

4. Boucle for

   Ecrire un programme qui affiche le logarithme et la racine carrée des 10 premiers entiers.

5. Listes

   Définir deux listes qui affichent les mêmes résultats que le programme précédent, puis une troisième qui affiche le quotient des termes des deux listes deux à deux.

6. Boucle while

   Ecrire un programme permettant de déterminer le premier entier n tel que $e^{-\sqrt{n}} \leqslant 10^{-4}$

7. Calcul et représentation d'une suite

   Calculer et représenter (avec des points) les 100 premiers termes de la suite u définie par :

$$\begin{cases} u_1 = 1 \\ \forall n \in \mathbb{N}^*, \quad u_{n+1} = 1 - e^{-u_n} \end{cases}$$

   Faire une deuxième représentation à l'aide d'un diagramme en bâtons.

8. Définir une fonction

   Définir avec Python la fonction f, définie sur $\mathbb{R}_+^*$ par : $f(x) = x\ln(x)$

9. Utiliser une condition

   Modifier le programme précédent pour qu'il définisse la fonction f définie sur $\mathbb{R}$ par :

$$\forall x \in \mathbb{R}, \quad f(x) = \begin{cases} x\ln(x) & \text{si} \quad x > 0 \\ x^2 & \text{si} \quad x \leqslant 0 \end{cases}$$

10. Représentation graphique

Représenter la fonction précédente sur l'intervalle $[-10, 10]$, émettre une hypothèse sur la continuité de la fonction.
Python peut présenter une difficulté pour la représentation de ce type de fonction. Auquel cas, on construira plus progressivement la liste des ordonnées.

11. Trouver une valeur approchée à l'aide de la méthode de dichotomie

Sachant que, pour $n \in \mathbb{N}^*$, l'équation $x^n + x - 1 = 0$ admet une unique solution strictement positive notée u_n et que $u_n \in]0, 1[$ pour tout $n \in \mathbb{N}^*$, compléter la fonction Python suivante renvoyant, pour n donné en entrée, une valeur approchée u_n à 10^{-3} près

```
def valeur_approchee(n):
  a=0
  b=1
  while ..... :
    c=(a+b)/2
    if (c**n+c-1)>0 :
      ...
    else :
      ...
  return ...
```

12. Définir et exploiter une matrice

Avec Python définir la matrice $A = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}$

Calculer A^2 et $I_4 + A + A^2 + A^3$ et interpréter les résultats.

13. Générer un nombre aléatoire sur une plage donnée

Ecrire une commande qui renvoie de manière aléatoire et équiprobable un nombre entier entre 1 et 10.

14. Simuler une variable aléatoire suivant une loi de référence

Deux joueurs s'affrontent dans un jeu de PILE ou FACE avec une même pièce donnant PILE avec une probabilité $p \in]0, 1[$ selon le protocole suivant : les deux joueurs lancent la pièce jusqu'à obtenir PILE. Celui qui fait le moins de tirages gagne. Si les deux joueurs ont effectué le même nombre de lancers pour obtenir PILE, ils recommencent, et ce jusqu'à ce qu'un des deux joueurs soit déclaré vainqueur.
Compléter la fonction Python ci-contre permettant de simuler le nombre de lancers nécessaires avant qu'un des deux joueurs ne soit déclaré vainqueur, avec une valeur de p donnée en entrée.

```
import .....

def simul(p):
  x=rd....(p)
  y=rd....(p)
  while x==y:
    x=...
    y=...
  N=min(x, y)
  return ...
```