ECG 1 – TP2

On va à partir de maintenant travailler dans la partie « texte » dédiée à la programmation.
On va écrire des scripts (des programmes) que l'on pourra exécuter plusieurs fois sans avoir à « tout retaper ».
N'hésitez pas à utiliser différents onglet en créant à chaque fois un nouveau module python

I) Quelques exemples simples :

Exercice 1 :
Taper le script suivant et dire ce qui se passe.

```
Script 1 :
prenom=input('Quel est ton prénom ?')
print('Bonjour',prenom,', comment vas-tu ?')
```

A quoi sert l'instruction input ?

Exercice 2 : Des nuances qui peuvent influencer l'exécution du programme.
Tapez les scripts suivants, **exécutez**-les et **décrivez** ce qui se passe (vous devez utiliser deux onglets différents)

Script 1 :	**Script 2 :**

`print('longueur du rectangle.')`
`longueur=input()`
`print('largeur du rectangle.')`
`largeur=input()`
`aire=longueur*largeur`
`print("L'aire du rectangle est : ",aire)`
	`print('longueur du rectangle.')`
	`longueur=eval(input())`
	`print('largeur du rectangle.')`
	`largeur=eval(input())`
	`aire=longueur*largeur`
	`print("L'aire du rectangle est :",aire)`

A Retenir :
1) L'instruction input « bloque » l'exécution du script jusqu'à ce que l'utilisateur du script entre un nombre ou une chaine de caractères (= mot ou phrase).
2) Même **si un nombre est entré, Python le considère comme une chaine de caractères**. On ne peut pas utiliser directement la valeur entrée dans un calcul.
3) Pour s'assurer que Python considère que la quantité entrée est bien un nombre, il faut utiliser l'instruction

```
eval( input () )
```

Exercice 3 :
Ecrire un script qui demande 3 notes avec 3 coefficients et qui ressort la moyenne des 3 notes coefficientées.

Exercice 4 :

Ecrire un programme qui demande la rayon d'une sphère et ressort son aire ($4\pi R^2$) et son volume ($\frac{4}{3}\pi R^3$).

Exercice 5 :
Ecrire un programme qui demande d'entrer le prix d'un objet et un pourcentage de remise et qui ressort le prix de l'objet soldé.

Exercice 6 :
Ecrire un programme qui demande d'entrer les coordonnées de 2 points A et B (dans le plan) et qui ressort une équation cartésienne de cette droite (On fera apparaître 3 coefficients a, b et c de l'équation ax + by + c = 0).

II) Instruction for (en français : pour):

L'instruction `for` sert à répéter des instructions un nombre connu de fois . On parle de **boucle bornée**.

Exercice 7 :
Taper les scripts suivants dans la console de programmation. Décrire ce qui se passe.

script 1 :
```
print('On affiche les 10 premiers multiples de 2')
for k in range(1,11):
      print(2*k)
```

Script 2 :
```
print('On affiche les 10 premiers multiples de 2')
for k in range(1,11):
print(2*k)
```

Retenir:

L'instruction for k in range(1,11) : est l'instruction qui permet de faire varier les valeurs de k. Ici k va prendre la valeur 1 puis la valeur 2, puis la valeur 3, …puis la valeur **10**.
La valeur 11 n'est pas atteinte.

La position de l'instruction `print(2*k)` doit **IMPERATIVEMENT** être décalée (on dit **indentée**) par rapport à l'instruction `for` à laquelle elle se rapporte. En écrivant les doubles points l'indentation se fait parfois automatiquement.

Pour **indenter** le code on peut utiliser la touche **TABULATION** .

Le fait de ne pas décaler l'instruction induit une erreur dans l'exécution du programme.

Dans le cas précédent, pour Python, il n'y a rien à faire après la ligne for k in range(1,11). **Ne pas oublier de mettre les doubles points quand on utilise l'instruction for.**

Exercice 8 : Résoudre d'abord ces exercices sans l'ordinateur !

Déterminer la valeur de la variable d après exécution du programme dans chacun des cas ci-dessous.

```
a=1;b=1;c=1;d=0
for k in range(2):
    a=a+b
    b=b+a
    c=c+b
    d=d+c
```

```
a=1;b=1;c=1;d=0
for k in range(2):
    a=a+b
    b=b+a
    c=c+b
d=d+c
```

```
a=1;b=1;c=1;d=0
for k in range(2):
    a=a+b
    b=b+a
c=c+b
d=d+c
```

```
a=1;b=1;c=1;d=0
for k in range(2):
   a=a+b
b=b+a
c=c+b
d=d+c
```

Expliquer !

Exercice 9 : Essayez de deviner ce que vont afficher les scripts avant de les taper et les exécuter

`script 1 :`
`print('Résultat mystère')`
`for k in range(1,1,11):`
`  print(3*k)`
	`script 2 :`
	`print('Résultat mystère')`
	`for k in range(1,11,2):`
	`  print(3*k)`
		`script 3 :`
		`print('Résultat mystère')`
		`for k in range(4,11,2):`
		`  print(3*k)`

`script 4 :`
`print('Résultat mystère')`
`for k in range(11):`
`  print(3*k)`
	`script 5 :`
	`print('Résultat mystère')`
	`for k in range(11,3,-1):`
	`  print(3*k)`
		`script 6 :`
		`print('Résultat mystère')`
		`for k in range(11,5,-2):`
		`  print(3*k)`

Expliquer ce qu'il se passe avec les instruction `range(…,…,...)`

Exercice 10 :

Expliquer ce qui va s'afficher à l'écran puis le vérifier en exécutant le script.

`script 1 :`
`a=1`
`for k in range(1,5):`
`  a=k*a`
`print(a)`
	`script 2 :`
	`a=1`
	`for k in range(1,5):`
	`  a=k*a`
	`  print(a)`

Exercice 11 :

On veut faire afficher le carré des 20 entiers naturels 1, 2 , 3 ,…., 20.

Ecrire un script qui permet cela.

Exercice 12 :

On peut montrer que $\sum_{k=1}^{+\infty} \frac{1}{k^2} = \frac{1}{1^2} + \frac{1}{2^2} + \frac{1}{3^2} + \frac{1}{4^2} + \frac{1}{5^2} + \ldots = \frac{\pi^2}{6} \approx 1.6449\ldots$

Ce résultat est assez surprenant puisque l'on additionne des fractions et que le résultat est irrationnel.

Mathématiquement : $\sum_{k=1}^{+\infty} \frac{1}{k^2} = \lim_{n \to +\infty} \sum_{k=1}^{n} \frac{1}{k^2}$.

On va estimer la valeur de cette somme en calculant $\sum_{k=1}^{n} \frac{1}{k^2}$ pour des valeurs de n « grandes ».

Une personne tape le script suivant.

```
S=0
print('Entrer l'entier naturel  n')
n=eval(input())
for k in range(1, n+1):
S=S+1/k**2
print('La somme vaut environ',S)
```

a) Expliquer la présence de la ligne `for k in range(1, n+1):`

b) Le script comporte une erreur. La trouver et la rectifier. (Si vous ne voyez pas l'erreur sur la version papier, tapez le script à l'ordinateur et essayez de l'exécuter).

Rédiger ci-dessous le script corrigé et le tester sur l'ordinateur avec diverses valeurs de n (n=100, n = 500…).

Exercice 13 :

1) Ecrire des programmes qui calculent les sommes suivantes (valeurs approchées).

a) $\sum_{k=1}^{+\infty} \frac{1}{k^3}$ b) $\sum_{k=1}^{+\infty} \frac{1}{k^4}$ c) $\sum_{k=0}^{+\infty} 0{,}5^k$ d) $\sum_{k=1}^{+\infty} \frac{1}{k(k+1)}$

2) **Aspect math** : Donner la valeur exacte de la somme c).

3) **Aspect math :**

a) Montrer que l'on peut trouver deux entiers relatifs a et b tels que pour tout k non nul, $\frac{1}{k(k+1)} = \frac{a}{k} + \frac{b}{k+1}$.

b) Justifier alors que l'on a pour tout entier n non nul : $\sum_{k=1}^{n} \frac{1}{k(k+1)} = 1 - \frac{1}{n+1}$.

c) Donner la vraie valeur de $\sum_{k=1}^{+\infty} \frac{1}{k(k+1)}$.

Remarques:

La première somme vaut ce que l'on appelle la constante d'Apéry (1916-1994). On sait que c'est un nombre irrationnel !

La deuxième somme vaut exactement $\frac{\pi^4}{90}$ (Euler 1707-1783).

Exercice 14 :

1) Ecrire des programmes qui calculent les produits suivants pour des n que l'on se fixe à l'avance :

a) $\prod_{k=1}^{n} \left(1+\frac{1}{k}\right)$ (n ≥ 1) b) $\prod_{k=1}^{n} \left(\frac{k}{k+1}\right)$ (n ≥ 1) c) $\prod_{k=2}^{n} \left(1-\frac{1}{k^2}\right)$, n ≥ 2 d) n!

2) **Aspect math (formule b)**

a) Montrer que le produit a) vaut n + 1.

b) Montrer que le produit b) vaut $\frac{1}{n+1}$. Que vaut alors $\lim_{n \to +\infty} \prod_{k=1}^{n} \left(\frac{k}{k+1}\right)$? Résultat étonnant ?

3)**Aspect math (formule c)**

a) Montrer que $\prod_{k=2}^{n} \left(1-\frac{1}{k^2}\right) = \prod_{k=2}^{n} \left(\frac{k-1}{k}\right) \times \prod_{k=2}^{n} \left(\frac{k+1}{k}\right)$.

b) En déduire que $\prod_{k=2}^{n} \left(1-\frac{1}{k^2}\right) = \frac{n+1}{2n}$.

c) Calculer alors $\lim_{n \to +\infty} \prod_{k=2}^{n} \left(1-\frac{1}{k^2}\right)$.

Exercice 15 :

Dans chaque cas, écrire deux scripts différents qui permettent de calculer les sommes suivantes :

a) 1+3+5+7+….+(2n+1) où n est un entier naturel.

b) 2 + 4 + 6 +….+ 2n où n est un entier naturel.

Exercice 16 :

On admet que $\sum_{k=1}^{+\infty} \frac{1}{k^2} = \frac{1}{1^2} + \frac{1}{2^2} + \frac{1}{3^2} + \frac{1}{4^2} + \frac{1}{5^2} + \ldots = \frac{\pi^2}{6} \approx 1.6449\ldots$.

1) **Aspect math :**

On pose $S_n = \sum_{k=1}^{n} \frac{1}{k^2}$, $P_n = \sum_{k=1}^{n} \frac{1}{(2k)^2}$ et $I_n = \sum_{k=1}^{n} \frac{1}{(2k-1)^2}$.

On pose aussi $S = \sum_{k=1}^{+\infty} \frac{1}{k^2} = \lim_{+\infty} S_n$, $P = \sum_{k=1}^{+\infty} \frac{1}{(2k)^2} = \lim_{+\infty} P_n$ et $I = \sum_{k=1}^{+\infty} \frac{1}{(2k-1)^2} = \lim_{+\infty} I_n$.

a) Justifier que l'on a $P_n = \frac{1}{4} S_n$.

En déduire que $P = \frac{\pi^2}{24}$.

b) Justifier que l'on a $S_{2n} = P_n + I_n$.

En déduire la valeur de I.

2) A l'aide d'un programme Python donner des valeurs approchées de S, P et I.

Exercice 17 :

Ecrire un script qui permet d'afficher la valeur de la somme $\sum_{k=1}^{n} \frac{1}{\sqrt{k}}$ lorsque l'on entre une valeur pour n.

III) Instruction while (tant que) :

L'instruction while sert elle aussi à réaliser des boucles. La boucle tourne tant qu'une condition fixée n'est pas réalisée.

Exemples :

Exemple 1 : Taper le script suivant :

```
t=1
while t<5:
   print("a")
   t=t+1
print('il y a 4 lettres a car la boucle a tourné 4 fois')
```

Dans ce script, on a initialisé t qui prend la valeur 1.

La boucle devra être réalisée tant que t n'atteint pas la valeur 5 (au moins).

La boucle contenant les 2 instructions : print(''a '') et t = t+1 est réalisée 4 fois car t augmente de 1 à chaque passage dans la boucle jusqu'à ce que t atteigne la valeur 5 ce qui annonce l'arrêt de la boucle.

Attention : Après l'instruction while et la condition à réaliser, il faut mettre des doubles points :

Il faut aussi indenter les instructions qui seront à réaliser (les décaler).

Remarque :

1) Attention : si la condition est mal fixée la boucle devient infinie… et peut mener à un BSOD !!

2) Pour stopper le programme, il suffit de taper CTRL + k (k pour ...kill)

Exemple 2 : Taper le script suivant :

```
population=11000
annee=2020
while population<22000:
   population=population*1.004
   print("La population est de",population,"habitants en ",annee)
   annee=annee+1
print('La population a doublé en ',annee-1)
```

Ce script permet d'afficher le nombre d'habitants année après année jusqu'à ce que cette population ait doublé.
Le terme 1,004 indique une augmentation annuelle de la population de 0,4% . Pourquoi ?

Ex 18 :

Taper le script suivant :

```
u=1
while u>0.005:
      u=(u**2)/(u**2+1)
      print(u)
```

1) Que fait ce script ?
2) Modifier le script pour qu'il demande d'entrer un réel x qui permet de faire varier la valeur 0.005.

Ex 19 :

Ecrire un programme qui demande d'entrer un réel x puis qui donne le premier entier naturel n tel que ln(n) ≥ x, en utilisant l'instruction while.

Fonctions utiles du module math :

`math.exp(x)`	Exponentielle	Renvoie exp(x)
`math.log(x)`	Logarithme népérien	Renvoie ln(x)
`math.log2(x)`	Logarithme base 2	Renvoie le log base 2 de x
`math.log10(x)`	Logarithme base 10	Renvoie le log base 10 de x

Ex 20 :

On considère la suite u définie par $u_n = \frac{e^n - n}{e^n + 2n}$.

1) Ecrire un programme qui demande d'entrer un entier naturel n puis affiche u_n , u_{n+1} …u_{n+10}.
On fera un programme avec l'instruction « for » et un autre avec l'instruction « while ».
2) Tester ce programme pour des grandes valeurs de n . Quelle semble être la limite de cette suite ?
3) Aspect math :
Donner la limite de cette suite en justifiant la réponse.

Ex 21 :

a) Que donne le programme suivant ?

```
x = 1
while x < 10:
    print("x a pour valeur", x)
    x = x * 2
print("Fin")
```

b) Et si x = 2 ? et de façon générale ?

Ex 22 :

Exercice 32. En étudiant $f : x \in]-1,+\infty[\mapsto \ln(1+x) - x$ et $g : x \in]-1,+\infty[\mapsto \ln(1+x) - \dfrac{x}{1+x}$ on montre que les suites données par

$$u_n = 1 + \frac{1}{2} + \ldots + \frac{1}{n} - \ln n, \quad v_n = 1 + \frac{1}{2} \ldots + \frac{1}{n} - \ln(n+1)$$

sont adjacentes et convergent vers un réel γ appelé constante d'Euler et de plus $v_n \leq \gamma \leq u_n$ pour tout $n \in \mathbb{N}^*$. On peut prendre $\dfrac{u_n + v_n}{2}$ comme approximation de γ et pour tout $n \in \mathbb{N}^*$, on a

$$\left|\gamma - \frac{u_n + v_n}{2}\right| \leq \ln\left(1 + \frac{1}{n}\right)$$

Ecrire un script qui demande une précision a et ressort la constante d'Euler avec cette précision a.

Ex 23 :

Ecrire un programme qui calcule les sommes suivantes (valeurs approchées) , en utilisant l'instruction « while ».

a) $\sum_{k=1}^{+\infty} \frac{1}{k^3}$ b) $\sum_{k=1}^{+\infty} \frac{1}{k^4}$ c) $\sum_{k=0}^{+\infty} 0.5^k$ d) $\sum_{k=1}^{+\infty} \frac{1}{k(k+1)}$

Ex 24 :

On se donne un réel A. On veut trouver le plus petit entier naturel n tel que 1 + 2 + 3 + …. + n ≥ A.

Ecrire un script qui permet de trouver cet entier n.

IV) Instruction if (en français : « si ») :

L'instruction if permet d'effectuer des instructions lorsqu'une (et parfois plusieurs) condition est vérifiée.

L'instruction else permet d'avoir une alternative.

Exemples :

1) Taper le script suivant :

```
a=eval(input('Entrer un nombre réel '))
if a==int(a):
  print('Vous avez entré un nombre entier.')
else:
    print("Le nombre n'est pas entier")
```

Expliquer ce qui se passe.

Remarques :

Pensez à mettre les doubles points : après la condition qui suit l'instruction `if`. Il en est de même pour l'instruction `else`.

2) Taper le script suivant :

```
a=eval(input('Entrer un nombre réel '))
if a>2:
   print('a est strictement plus grand que 2.')
if a< -1:
   print('Le nombre est strictement plus petit que - 1.')
if a>=-1 and a<=2:
   print('Le nombre est compris entre -1 et 2 .')
```

Expliquer ce qui se passe.

3) Taper le script suivant :

```
a=eval(input('Entrer un nombre entier. '))
if a==int(a):
    reste=a%2
    if reste==0:
       print("L'entier est pair.")
    if reste==1:
       print("L'entier est impair.")
else:
     print("Vous n'avez pas entré un entier.")
```

Expliquez ce que fait ce script. Bien comprendre les incrémentations.

4) Taper le script suivant :

```
a=eval(input('Entrer un nombre. '))
 if a>=5:
      if a>=12:
         print('Le nombre est plus grand que 12.')
     else:
        print('Le nombre est plus grand que 5 mais inférieur ou égal à 12.')
else:
       print('Le nombre est strictement plus petit que 5.')
```

Expliquer ce qui se passe. Là aussi bien comprendre les incrémentations.

5) Taper le script suivant :

```
a=eval(input('Entrer un nombre. '))
if a>=12:
   print('Le nombre est plus grand que 12.')
elif a>=5:
   print('Le nombre est plus grand que 5 mais inférieur ou égal à 12.')
elif a<5:
   print('Le nombre est strictement plus petit que 5.')
```

L'instruction `elif` signifie « ou alors si ». Elle permet d'éviter l'emploi répétitif de l'instruction `if`.

Ex 25 :

Ecrire un script qui permet de calculer n ! en demandant la valeur de n

Je rappelle que 0 ! = 1 et que n ! = 1×2×3×…×n si n ≥ 1 (n entier naturel).

Ex 26 :

Ecrire un programme qui sort la valeur absolue d'un nombre (sans utiliser la fonction `abs()` !!!)

Rappel : $|a|=\begin{cases} a \text{ si } a\geq 0 \\ -a \text{ si } a\leq 0 \end{cases}$

Ex 27 :

Ecrire un programme qui demande 2 nombres et dit lequel des deux est le plus grand.

Ex 28 :

On considère une équation du type ax + b = 0.

Ecrire un programme qui donne la solution de cette équation. !! Traiter tous les cas (résultat sous forme fractionnaire obligatoire !!

Ex 29 : Pour battre l'ordi….

Taper et exécuter ce script :

```
from random import*
x=int(3*random())
a=input('Penser à : pierre, feuille ou ciseaux et puis taper "enter". ')
if x==0:
       print('Moi je pensais à "pierre".')
else:
       if x==1:
             print('Moi je pensais à "feuille".')
       else:
             print('Moi je pensais à "ciseaux".')
```

Remarques :

1) `From random import*` permet de faire appelle à des fonctions qui génèrent des nombres aléatoires. On en reparlera plus tard.

2) random() permet de générer un nombre aléatoire dans l'intervalle [0 ; 1 [.

Question : Dans notre programme que donne l'instruction `x=int(3*random())` ?

Ex 30 :

On considère les sommes doubles $A=\sum_{1\leq i,j\leq n}(i+j)$, $B=\sum_{1\leq i,j\leq n}(i\times j)$, $C=\sum_{1\leq i,j\leq n}(3i-2j)^2$.

Faire calculer ces sommes par python de 2 façons différentes :

a) Avec l'instruction for.

b) Avec l'instruction while.

Ex 31 :

On considère les sommes doubles $A=\sum_{1\le i,j\le n} min\,(i;j)$, $B=\sum_{1\le i,j\le n} max\,(i;j)$, $C=\sum_{1\le i,j\le n} ¿i-j\vee ¿¿$.

1) Faire calculer ces sommes par python de 3 façons différentes :

a) Avec l'instruction for.

b) Avec l'instruction while.

c) En testant ce que vaut min(i, j) en fonction de i et j. (de même pour max et valeur absolue)).

Par exemple min(2 ; 3) = 2 donc avec un test if on peut éviter l'instruction min…

2) Aspect math :

a) Montrer que A = $\frac{n(n+1)(2n+1)}{6}$. Quelle somme usuelle vaut aussi $\frac{n(n+1)(2n+1)}{6}$?

b) Montrer que B = $\frac{n(n+1)(4n-1)}{6}$.

c) En utilisant le fait que | i – j | = max(i, j) – min(i , j) , trouver une formule pour C.

Ex 32 : Recherche de la racine d'une fonction par dichotomie.

Rappel : Soit une fonction continue strictement croissante sur [a,b] telle que f(a) < 0 et f(b) > 0.

Le **théorème des valeurs intermédiaires (TVI)** assure l'existence de c tel que $c \in]a ; b[$ et f(c) = 0.

On a une propriété équivalente si f est strictement décroissante sur [a, b].

L'objectif est de localiser (donner une valeur approchée) c tel que f(c) = 0.

Principe de l'algorithme de dichotomie:

1) Evaluer la fonction en a, b et (a+b)/2 .

2) Si f(a)×f((a+b)/2) < 0, on recherche la solution dans l'intervalle [a,(a + b)/2],

3) sinon on travaille dans l'intervalle [(a + b)/2 , b].

Puis on réitère le processus jusqu'à avoir l'approximation de c choisie

Ecrire un programme qui localise la solution des équations avec une précision entrée par l'utilisateur :

f(x) = x^3 + 3x =5. g(x) = x + exp(x) = 8 h(x) = $\frac{e^x-e^{-x}}{2}$ = 4

Aspect math :

Dans le 3ème cas, trouver par le calcul les solutions exactes.

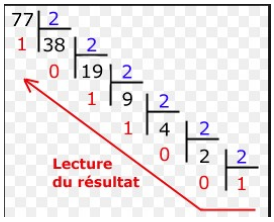
Ex 33 : Ecriture en base 2

Pour écrire un entier en base 2, on effectue les divisions euclidiennes successives de cet entier par 2.

Les restes valent 0 ou 1.

On obtient alors l'écriture en base 2, en remontant la liste des restes.

Par exemple 77 s'écrit en base 2 1001101



Ecrire un programme qui donne l'écriture en base 2 d'un entier qui sera demandé par l'utilisateur.

Ex 34 :

On considère l'expérience aléatoire suivante :

On lance un dé parfait. Si on obtient 6 on gagne. Dans les autres cas on perd.

On répète cette expérience jusqu'à gagner 1 fois.

On note X le nombre de fois que l'on doit lancer le dé pour gagner une première fois.

Ecrire un programme qui simule ces lancers de dés et qui donne le nombre de fois qu'il a fallu jouer pour gagner.

Aide : Utiliser l'instruction random.

Ex 35 :

Dans chaque cas ; écrire un programme qui permet de calculer f(x) lorsque l'on entre une valeur x.

a) f(x) = $\begin{cases} x+7 \; si \; x \geq 1 \\ x^2 \; sinon \end{cases}$

b) f(x) = $\begin{cases} |x-1| \; si \; x<3 \\ \sqrt{x^2+1} \; si \; 3 \leq x<8 \\ \ln(x-6) \; sinon \end{cases}$

c) f(x) = $\begin{cases} x \; si \; x \leftarrow 1 \\ \dfrac{x+5}{x+2} \; si \; -1 \leq x<6 \\ x^2+2x-3 \; sinon \end{cases}$