# TP n°7 - Recherches et récursivité

**TP Noté**

Tous les documents sont autorisés.

# IMPORTANT

À lire *attentivement* avant de commencer

## Comment présenter le fichier à rendre ?

- Créer et enregistrer dans votre dossier personnel un nouveau fichier dans EduPython intitulé

  **NOM1-NOM2-NOM3.py**

  où NOMi est à remplacer par les noms de chaque personne du groupe (en omettant les accents et espaces éventuels qu'il contient).

- Ce fichier sera présenté de la façon suivante :

```
##### NOM1 Prénom1 - NOM2 Prénom2 - NOM3 Prénom3

####Partie 1

##Q.1

"Votre résolution de la question Q1:"

##Q.2

#Q.2.a

"Votre résolution de la question Q2:a)"

etc...

####Partie 2

etc...
```

- Enregistrez ce fichier **régulièrement** - Ctrl+S !
- Une fois votre travail terminé, rendez-vous à l'adresse https://classexo.fr/info puis envoyez votre fichier.

Dans la suite de ce TP, nous aurons parfois besoin des modules `random` et `turtle`. On commencera par importer ces deux modules :

```
from random import *
from turtle import *
```

On rappelle les instructions suivantes concernant le module `turtle` :

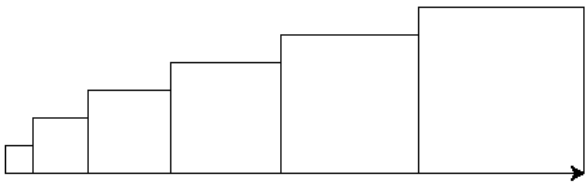
- l'instruction `reset()` permet d'ouvrir une nouvelle fenêtre de dessin lorsqu'il n'y en a pas avec la tortue "de base" ; et de réinitialiser tous les dessins de la tortue de base si la fenêtre est déjà ouverte.
- l'instruction `bye()` permet de fermer la fenêtre de dessin.

## 1. Echauffement

**Q1 :** Écrire une fonction `somme(L)` qui renvoie la somme des valeurs d'une liste `L` de nombres.

**Q2 :** Écrire une fonction `moyenne(L)` qui renvoie la moyenns des valeurs d'une liste `L` de nombres.

**Q3 :** Écrire une fonction `suite_carre(n)` qui fait tracer `n` carrés l'un à côté de l'autre à la tortue du module `turtle` tels que le premier carré est de côté 20 pixels et, à partir du deuxième, chaque carré est de côté +20 pixels par rapport au précédent. On pourra bien-sûr définir d'autre(s) fonction(s) avant `suite_carre(n)` pour plus de facilité !
*Exemple :* l'instruction `suite_carre(6)` doit tracer :



## 2. Recherches séquentielles

Notez bien votre numéro de groupe et rendez-vous sur le page

`https://classexo.fr/info/jeu.php`

pour récupérer la variable `chaine` qui contient une longue chaîne de caractères **en indiquant votre numéro de groupe**. Copiez-collez dans la **console** Python cette variable et appuyer sur "Entrée" pour la définir.
Vous allez devoir récupérer des informations sur le texte contenu dans la variable `chaine` pour obtenir un code secret qui vous permettra d'obtenir une récompense incroyable !

**Instructions :** pour chaque question, vous définirez une fonction (ou plusieurs fonctions) qui vous permettra de récupérer l'information demandée. Avant d'appliquer votre fonction à la variable `chaine`, testez là sur des chaines de caractères plus petites pour vérifier son bon fonctionnement.
*Exemple :*

— *Question :* déterminer le nombre de caractères `'e'` (minuscule et sans accent) dans `chaine` ; on note $a_1$ ce nombre.

— *Méthode :* on crée une fonction `nb_e(chn)` (c'est vous qui choisissez le nom) que l'on teste dans la console sur des petites chaînes pour vérifier le fonctionnement :

```
>>>nb_e('Un test pour voir')
1
>>>nb_e('ça va ?')
0
>>>nb_e('Est-ce que ça fonctionne ? Très bien !')
3
```

Puis on récupère le nombre $a_1$ (si on n'a pas fait d'erreur bien sûr !) :

```
>>>nb_e(chaine)
1232
```

Donc $a_1 = 1232$.

**Q1 :** Déterminer le nombre de caractères (tous confondus : lettres, espaces, ponctuations, etc...) dans `chaine` ; on note $a_1$ ce nombre.

**Q2 :** Déterminer s'il y a un caractère `'y'` (minuscule et sans accent) dans `chaîne`. Si oui, $a_2 = 1$ sinon, $a_2 = 0$.

**Q3 :** Déterminer l'indice du premier `'a'` (minuscule et sans accent) dans `chaine` ; on note $a_3$ ce nombre.

**Q4 :** Déterminer l'indice du dernier `'a'` (minuscule et sans accent) dans `chaine` ; on note $a_4$ ce nombre.

**Q5 :** Déterminer le nombre de caractères `'e'` (minuscule et sans accent) dans `chaine` ; on note $a_5$ ce nombre.

**Q6 :** Déterminer si la sous-chaîne de 2 caractères successifs `'ez'` (minuscule et sans accent) se trouve dans `chaîne`. Si oui, $a_6 = 1$ sinon, $a_6 = 0$.

**Q7 :** Déterminer le nombre de fois qu'apparait la sous-chaîne de 2 caractères successifs `'an'` ( minuscule et sans accent) dans `chaine` ; on note $a_7$ ce nombre.

**Q8 :** Déterminer l'indice du 50-ième `'o'` (il s'agit de la lettre o en minuscule et sans accent) dans `chaine` ; on note $a_8$ ce nombre.

**Q9 :** Déterminer le nombre de sous-chaînes différentes de 2 caractères successifs dans la variable `chaine` ; on note $a_9$ ce nombre.
*Indications :* par exemple, dans `'blabla .'`, il y a $a_9 = 5$ sous-chaînes différentes de 2 caractères successifs : `'bl'` (2 fois), `'la'` (2 fois), `'ab'` (1 fois), `'a '` (1 fois), `' .'` (1 fois).
De plus, si la variable `dico` est un dictionnaire, on peut obtenir le nombre de clés avec l'instruction `len(dico)`.

**Q10 :** Le code secret est obtenu en faisant la somme des neuf $a_i$ plus votre numéro de groupe :

$$\text{Code Secret } = a_1 + a_2 + a_3 + a_4 + a_5 + a_6 + a_7 + a_8 + a_9 + \text{ numéro du groupe}$$

Bonne chance ! *si vous n'avez pas trouvé le code au bout de 1h - 1h15, commencez la partie suivante !*

## 3. Récursivité

**Q1 :** On considère la suite $(u_n)_{n\in\mathbb{N}}$ telle que $\begin{cases} u_0 = 1 \\ u_n = 1 + \frac{1}{u_{n-1}} \quad \text{pour } n \in \mathbb{N}^* \end{cases}$.

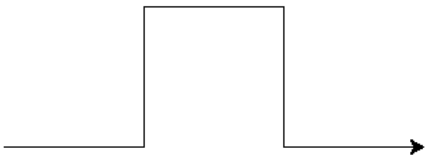
Écrire une fonction **récursive** `u(n)` qui renvoie le terme $u_n$ d'indice $n$ de la suite $(u_n)_{n\in\mathbb{N}}$.
Exemple : l'instrcution `u(500)` doit renvoyer `1.618033988749895`.

Que vaut $u_{2000}$ ? *(s'il y a du "rouge", pensez à regarder le cours sur la récursivité !)*

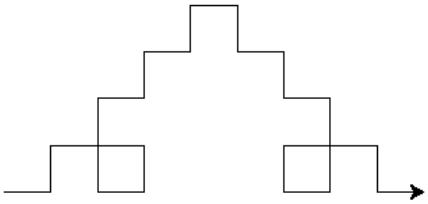
**Q2 :** **La courbe de M. Arnt :** la courbe de M. Arnt est une courbe fractale inspirée de la courbe de Von Koch et que vous allez tracer en vous aidant des travaux de la semaine dernière :
Voici ce que nous allons obtenir :

— Étape 0 :

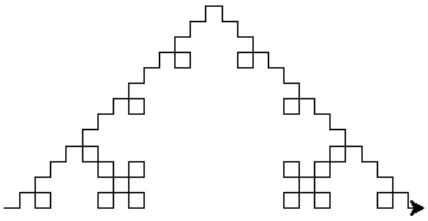
— Étape 1 :



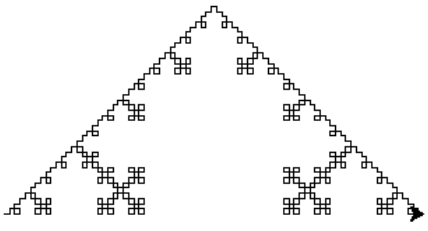
— Étape 2 :



— Étape 3 :



— Étape 4 :



Dans ces images, les longueurs de chaque segment sont égaux et chaque angle est droit.

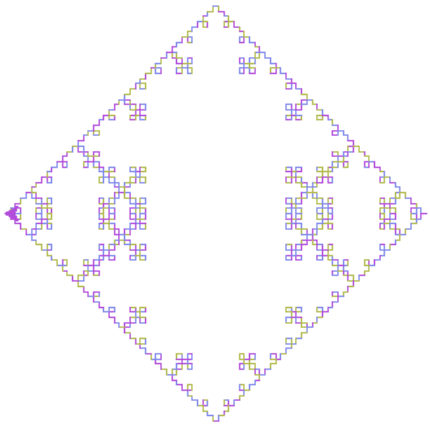
**a)** Écrire une fonction (non récursive ici) `etape1(longueur)` qui trace l'étape 1 de la courbe de M. Arnt de taille `longueur` pixels - il s'agit dela longueur entre le point de départ et le point d'arrivée, pas la longueur totale de la courbe).

**b)** On décrit le principe récursif - multiple - de réalisation des étapes de la courbe de M. Arnt :

> - **Initialisation :** étape 0 : on dessine un trait de la longueur donné en argument ;
> - **Récursion :** étape $n \geq 1$ : on suppose que la tortue sait tracer l'étape $n-1$ sur toute longueur. On veut tracer l'étape $n$ sur une longueur $\ell$, on réalise alors l'étape 1 mais, au lieu d'avancer en ligne droite de $\ell/3$, on réalise chaque fois (donc 5 fois) l'étape $n-1$ sur une longueur $\ell/3$.

Écrire une fonction récursive `marnt(n,longueur)` qui trace l'étape $n$ de la courbe de M. Arnt. On exécutera l'instruction `speed(0)` avant tout test pour s'assurer un tracé le plus rapide possible.

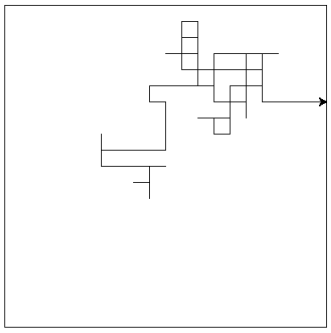
**c)** Utiliser la fonction `marnt(n,longueur)` plusieurs fois pour tracer le *tapis de M. Arnt* à l'étape 4.



## 4. Une marche aléatoire de la tortue

*Remarque : cette partie est tirée du précédent TP noté : vous pouvez repartir de ce que vous aviez fait la fois précédente !*

On souhaite réaliser une marche aléatoire de la tortue sur un quadrillage "imaginaire" de la fenêtre de dessin du module `turtle` dans un carré de côté de longueur 400 pixels. Voici une illustration de ce qui est attendu :



**Q10 :** Quels sont les effets de la fonction suivante ?

```
def initialisation():
    reset()
    tcadre = Turtle()
    tcadre.penup()
    tcadre.forward(200)
    tcadre.left(90)
    tcadre.pendown()
    tcadre.forward(200)
    for k in range(4):
        tcadre.left(90)
        tcadre.forward(400)
    tcadre.hideturtle()
```

**Q11 :** Écrire une fonction `deplacement(r)` qui prend en argument un entier `r` compris entre 0 et 3 et qui, en supposant que la tortue de base est présente dans la fenêtre de dessin et est dirigée horizontalement vers le droite :

- si `r` vaut 0, déplace la tortue de 20 pixels vers la **droite** puis met la tortue dirigée horizontalement vers le droite ;
- si `r` vaut 1, déplace la tortue de 20 pixels vers la **gauche** puis met la tortue dirigée horizontalement vers le droite ;
- si `r` vaut 2, déplace la tortue de 20 pixels vers le **haut** puis met la tortue dirigée horizontalement vers le droite ;
- si `r` vaut 3, déplace la tortue de 20 pixels vers le **bas** puis met la tortue dirigée horizontalement vers le droite ;

**Q12 :** Écrire une fonction `marche()` qui réalise la marche aléatoire de la tortue présentée précédemment avec comme condition d'arrêt, la condition suivante : si la tortue "touche" un bord du carré dessiné

par la fonction initialisation, on arrête la marche. On fera renvoyer à la fonction `marche()` le nombre de pas de la tortue lors de sa marche.
On appellera la fonction `initialisation()` au tout début de la fonction `marche` afin de tracer le cadre de la marche aléatoire.

*Quelques indications :*

- Créer deux variables `h` (pour "horizontal") et `v` (pour "vertical") qui, au fur et à mesure de la marche, enregistrent respectivement la position horizontale et verticale de la tortue (leur but est de nous permettre de savoir quand la tortue touche un bord du carré !)
- On pourra créer une variable `r=randint(0,3)` et l'instruction `deplacement(r)` pour effectuer chaque pas aléatoire.