Exercice 1: Fonctions On définit les fonctions f et g sur $\mathbb{R}$ par

- $f(x) = x^3$.
- $g(x) = 4x^2 + x - 4$.

1. Calculer $f(-1), g(-1)$ puis $f(2), g(2)$.
2. -1 est-il solution de l'équation $f(x) = g(x)$? 2 est-il solution de $f(x) = g(x)$?
3. Résoudre graphiquement l'équation $f(x) = g(x)$ avec votre calculatrice ou *Géogebra*.
4. Ecrire pour f et g une fonction *Python* prenant en **paramètre** une valeur x et renvoyant la valeur image.

   Exemple:

   ```
   >>>f(4)
   64
   ```

   Tester les valeurs de **retour** de vos fonctions *Python* avec les résultats de la *question 1.*
5. Vérifier dans le **SHELL** le résultat de l'instruction:

   ```
   >>>f(1)==g(1)
   True
   ```

   Expliquer ce résultat.
6. Que renvoie l'expression `f(4) == g(4)` ?
7. **Note :** La pratique qui consiste à tester l'égalité de nombres en programmation n'est correcte que si nous testons des entiers, et c'est le cas dans cet exercice. Vous pouvez le constater en exécutant l'instructions suivante dans le SHELL, Que renvoie cette instruction ?

   ```
   >>> 0.3 == 0.1 + 0.2
   ```

Exercice 2: Boucle *Pour*

1. Taper le code suivant, expliquer l'affichage:

   ```
   for i in range(1, 10):
       print(i)
   ```
2. Rick veut investir dans un commerce et place 5000 euros sur un compte, puis ajoute 300 euros tous les mois. Il désire connaître le montant sur le compte après n mois. Donner ce montant après 3 mois.
3. On utilise l'algorithme ci-contre afin d'obtenir ce montant pour 3 mois.

   Compléter cet algorithme afin de pouvoir répondre à la question. Expliquer le rôle des ligne 2 et 3.

   Algorithme I

1.	$u \longleftarrow$ ....
2.	$n \longleftarrow$ ....
2.	**Pour** i **allant de** 1 à n
3.	$u \longleftarrow u + 300$
4.	**Fin du Pour**
4. Programmer cet algorithme en python et vérifier votre résultat.

5. Rick change d'avis et place toujours 5000 euros au départ mais ajoute 5 % de la somme présente sur le compte chaque mois. Donnez le montant obtenu après 3 mois et modifiez l'algorithme précédent pour obtenir ce résultat.

Algorithme II

1.	$u \longleftarrow$ ....
2.	$n \longleftarrow$ ....
2.	**Pour** i **allant de** 1 à n
3.	$u \longleftarrow$ ....
4.	**Fin du Pour**

Exercice 3: Boucle *Tant Que* Voici un algorithme où a est un nombre réel avec $a > 0$

Algorithme III

1.	$x \longleftarrow 0$
2.	$y \longleftarrow 0$
3.	**TQ** $y \leqslant a$
4.	$x \longleftarrow x + 1$
5.	$y \longleftarrow x^3$
6.	**Fin TQ**

1. Tester pas à pas cet algorithme dans le tableau suivant avec a =100.
2. Traduire cet algorithme en python et vérifier le résultat obtenu à la fin de votre tableau.
3. Expliquer le rôle de cet algorithme.

x	0	1	2								
y											
$y \leqslant a$											

Exercice 4: Dessiner avec le module Turtle de Python Le module *turtle* est un outil du langage *Python* de tracé de figures simples. Un curseur (la tortue) effectue le tracé à l'écran en se déplaçant selon les instructions codées par l'utilisateur. L'entête de votre code devra comporter l'importation de ce module:tapez la ligne *from turtle import **.

- *forward(100)* avance le curseur de 100 pixels.
- *backward(90)* recule le curseur de 90 pixels.
- *left(90)* tourne le curseur de 90° vers la gauche.
- *goto(x,y)* déplace le curseur au point de coordonnées *(x,y)*.
- *color*(couleur) colorie le tracé.
- `http://fr.wikibooks.org/wiki/Programmation_Python/Turtle` vous fournira l'ensemble des commandes disponibles

1. Dessiner un carré, un hexagone puis un décagone. Montrer au professeur votre code.
2. Ecrire une fonction `polygone(n)` prenant en paramètre le nombre de côté et traçant le polygone régulier correspondant.

Née en 1906, **Grace Murray Hopper** étudie la physique et décroche le titre de docteure en mathématiques. Elle entre dans l'armée où elle peut programmer sur l'un des premiers ordinateurs: le Mark 1.

Elle défend l'idée qu'un programme devrait pouvoir être écrit dans un langage proche de l'anglais plutôt que d'être calqué sur le langage machine : c'est pourquoi elle conçoit le premier *compilateur* en 1951.

On lui attribue la naissance du mot **bug** : alors que son programme avait planté pour une raison inconnue, elle découvrit un insecte logé dans une carte perforée: le premier bug était résolu.