

Transformation de la voie professionnelle  
Journée de formation disciplinaire en  
mathématiques – physique-chimie

# Algorithmique et programmation

## Activités

Pascal ASMUSSEN

Audrey BERNARD

Malorie BERTOLOTTI

Evelyne CHOLLET

Clément HARAU

Cassandra LE-MAUFF

Jérôme LENORMAND

Céline PONCET

Max THIEBAUD

Le 26 juin 2019

ACTIVITES	Titre	page	Statistiques à une variable	Fluctuations d'une fréquence, probabilités	problème du premier degré	Fonctions	Géométrie	Algorithmique	Programmation Scratch	Programmation Python
1	Trouver un nombre	2						X		X
2 et 3	Lancer de dés	3		X						X
4	Calcul de volumes	6					X			X
5	Dessiner un patron	8					X			X
6	Année Bissextile	10						X	X	X
7	Le distributeur	11						X		
8	Illusions d'optique	15					X		X	X
8	Équations du premier degré	17			X				X	
10	Réciproque de Pythagore	18					X		X	
11	Pile ou Face	23		X					X	
12	Figures simples	26					X	X	X	X
13	Calcul de l'image	34				X		X		X
14	Statistiques	36	X					X	X	
15	Classification périodique	39						X	X	

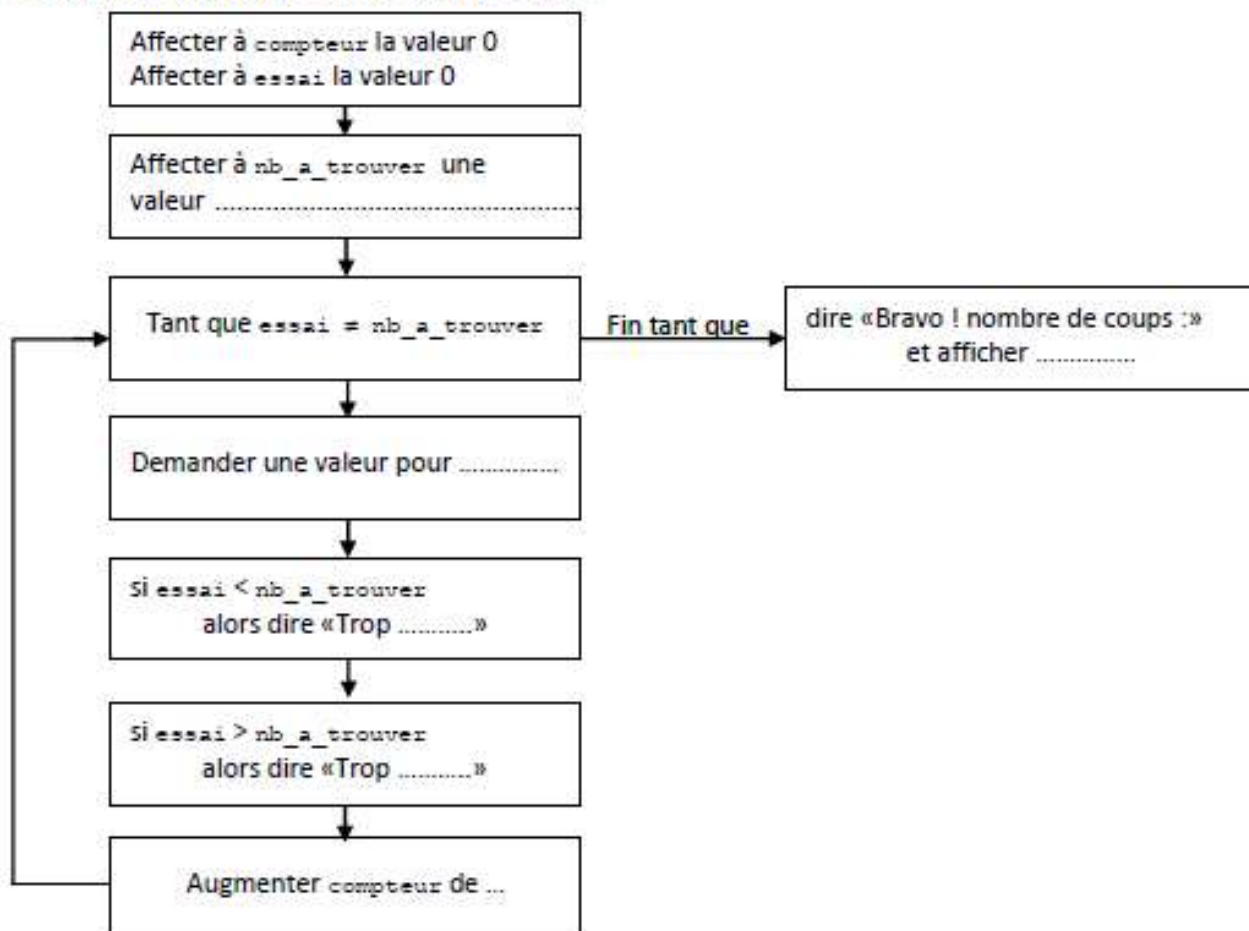


Comment concevoir un programme pour pouvoir jouer tout seul ?

Soient les variables suivantes :

nb_a_trouver	valeur entière choisie aléatoirement par l'ordinateur
compteur	valeur entière permettant de compter le nombre d'essai
essai	valeur entière correspondant à la proposition faite par le joueur

1. Compléter l'algorithme correspondant à ce jeu.



2. Compléter alors le programme `trouver_nombre.py` à l'aide de cet algorithme

## Activité 2

### Simuler le lancer d'un dé avec Python

#### 1. Simulation du lancer d'un dé à 6 faces (non truqué)

Pour simuler le hasard dans Python, il faut commencer par importer le module « random », et plus particulièrement la fonction « randint » qui permet de simuler le tirage d'un entier compris entre 2 bornes (bornes comprises).

On peut par exemple tester le programme suivant :

```
from random import randint
def lancer_un_de() :
    d=randint(1,6)
    return d
```

**Exercice 1 :** En utilisant cette fonction « lancer\_un\_de », créer une fonction intitulée « somme\_trois\_des » qui permet de simuler le lancement de 3 dés, et d'afficher la somme des 3 dés.

#### 2. Simulation du lancer de n dés à 6 faces

On souhaite dans cette seconde partie simuler le lancer simultané de plusieurs dés. On note n le nombre de dés lancés.

Compléter la fonction suivante pour simuler le lancer de n dés à 6 faces, en remplaçant les \* par les bonnes instructions :

```
def lancer_n_des(*):
    liste_des=[]                                # on crée une liste vide dans laquelle
                                                #on placera les résultats
    for i in range(*):
        d=*
        liste_des.append(d)                    #on remplit la liste avec la variable d
    return liste_des
```

**Exercice 2 :** il existe dans le commerce des dés à 8 faces, 12 faces, 20 faces...

Modifier les fonctions précédentes pour pouvoir simuler le lancer de n dés à x faces, où x représente le nombre de faces du dé, et récupérer la liste des n tirages.

#### 3. Calcul des fréquences d'apparition des faces

La fonction « frequences(n) » proposée à la page suivante a pour objectif d'afficher un tableau qui rend compte des fréquences d'apparition de chacune des 6 faces d'un dé classique après n lancers.

Compléter le script (en remplaçant là encore les \* ) pour obtenir le résultat souhaité.

```
def frequences(n):
    frequence_des=[]
    f=lancer_n_des(*)
    print(f)
    for i in range(*):
        frequence_des.append(f.count(*)/*)    #f.count(*) compte le nombre
                                                #de * dans la liste f
    return frequence_des
```

## Activité 3

# Les dés ne mentent jamais

### I. Les dés ne mentent jamais

Dans toute l'activité les dés utilisés ne sont pas truqués

1. Visionner la vidéo suivante jusqu'à 1'20 " :

<https://education.francetv.fr/matiere/mathematiques/cinquieme/video/les-des-ne-mentent-jamais-les-probabilites>

2. Lister tous les résultats qu'Évariste et Manon auraient pu donner :

.....

.....

3. Dire quel résultat, selon vous, Manon aurait dû choisir pour avoir plus de chances de gagner contre Évariste.

.....

.....

4. Expliquer comment on pourrait procéder pour vérifier la conjecture précédente ?

.....

.....

.....

### II. Expérience 1 : On joue

1. Lancer 10 fois les dés et compléter la deuxième ligne du tableau suivant :

Lancer	n°1	n°2	n°3	n°4	n°5	n°6	n°7	n°8	n°9	n°10
Somme obtenue										

2. Rappeler le calcul à effectuer pour calculer la fréquence de chaque résultat : .....

3. Calculer la fréquence de chaque résultat obtenu :

Résultat	2	3	4	5	6	7	8	9	10	11	12
Fréquence											

4. Les résultats permettent-ils de conclure sur le résultat à choisir pour gagner ?

.....

.....

.....

### III. Expérience 2 : On simule

Pour aller plus vite, on souhaite utiliser un programme informatique qui permettra de faire instantanément un grand nombre de simulations.

Voici le programme existant : « lancer\_2\_des.py »

```
from random import randint

def lancer_des():
    de1 = randint(1,6)
    de2 = randint(1,6)
    return (de1+de2)

resultat = lancer_des()
print(resultat)
```

1. Exécuter le programme plusieurs fois. Quel résultat obtient-on ?

.....

.....

2. On désire l'améliorer pour obtenir une liste contenant les résultats de 100 lancers.  
Ajouter à la fin du programme les lignes suivantes :

```
resultat = []  
for i in range(100):  
    resultat.append(lancer_des())  
print(resultat)
```

← `append` permet d'ajouter `lancer_des()` à la liste `resultat`

3. Exécuter le programme et expliquer le résultat affiché :

.....

.....

4. En Python, on peut compter le nombre de fois que le chiffre 2 est présent dans la liste `resultat` en tapant :

```
resultat.count(2)
```

5. Écrire dans le cadre ci-dessous le calcul à effectuer pour calculer la fréquence de 2 obtenus lors des 100 lancers :

```
resultat.count(2)
```

6. Compléter le programme pour que la fonction affiche le nombre de fois qu'on a obtenu chacun des onze résultats possibles.
7. Lancer plusieurs fois le programme.
8. Expliquer si la conjecture énoncée au début de l'activité est ou non confirmée. Conclure en donnant le résultat que Manon aurait dû annoncer pour avoir le plus de chances de gagner contre Évariste.

.....

.....

.....

.....

9. Visionner la fin de la vidéo :

<https://youtu.be/G9VacEB9E4k?t=81>

## IV. Pour aller plus loin

1. Ouvrir le programme « `diagramme_lancer_2_des` » qui permet d'afficher les diagrammes en bâtons des résultats.
2. Changer le code pour effectuer 1 000 lancers.
3. Recommencer avec 10 000 puis 100 000 lancers.
4. En déduire la probabilité d'obtenir chacun des résultats.

.....

.....

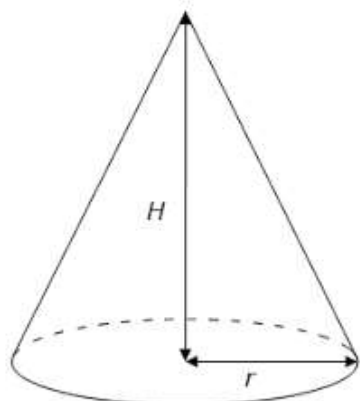
## Activité 4

### Calcul de trois volumes de solides usuels

**Objectif :** écrire un programme permettant de calculer le volume d'un cône de révolution, d'une pyramide à base carrée ou d'une boule.

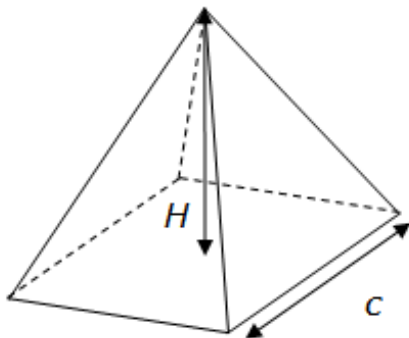
Les formules permettant de calculer les volumes de quelques solides usuels sont données ci-dessous (*source : [www.lememento.fr](http://www.lememento.fr)*) :

*Cône de révolution*



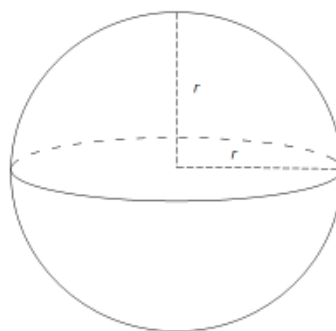
$$V = \frac{\pi \times R^2 \times h}{3}$$

*Pyramide à base carrée*



$$V = \frac{c^2 \times h}{3}$$

*Boule*



$$V = \frac{4}{3} \pi R^3$$

#### 1. Etude du programme pour la pyramide

Le début de programme suivant va permettre de calculer le volume d'une pyramide :

```
figure=int(input("Quelle est la figure choisie ? 1 : Pyramide 2 : Cône 3 : Boule"))
```

```
if figure==1:
```

```
    c=float(input("Quelle est la mesure du côté de la base ?"))
```

```
    h=float(input("Quelle est la mesure de la hauteur de la pyramide ?"))
```

```
    v=_____
```

```
    print("Le volume de la pyramide vaut : ",v)
```

a) Préciser le rôle des instructions `int` et `float` devant les questions posées à l'utilisateur du programme ?

.....  
.....  
.....

b) Après avoir tapé le début sous Edupython, **compléter** le programme pour calculer le volume d'une pyramide.

c) **Vérifier** que le volume de la pyramide du Louvre ( $c = 35,42$  m,  $h = 21,64$  m) vaut environ  $9\,050\text{ m}^3$ .

.....  
.....

## 2. Programme pour le cône et la boule

Pour calculer le volume d'une sphère ou d'une pyramide, il faut utiliser le nombre  $\pi$ , qui n'est pas systématiquement disponible sous Python.

Il convient donc d'intégrer au début du programme la ligne :

```
from math import pi      # permet d'intégrer le nombre pi aux programmes
```

En s'inspirant du programme proposé pour la pyramide, **écrire** les parties du programme qui permet de calculer le volume d'un cône de révolution et d'une boule.

## 3. Amélioration du programme

- a) **Indiquer** comment réagit le programme si la réponse est différente de 1, 2 ou 3 lors du choix du solide ?

.....  
.....  
.....

- b) **Proposer une solution** pour que le programme renvoie un message (par exemple : « Réponse invalide ») et reproposer le choix de figure.



## Activité 5

### Géométrie dans l'espace

**Activité** : Dessiner un patron

**Objectifs** : Comprendre un programme  
Exécuter un programme pas à pas

1. Voici un programme écrit dans le langage Python.

```
1 from random import *
2 from turtle import *
3
4 def carre():
5     for k in range(4):
6         forward(50)
7         left(90)
8
9 for m in range(4):
10     up()
11     goto(-100+50*m,0)
12     down()
13     carre()
14
15 for n in range(2):
16     x=randint(0,3)
17     up()
18     goto(-100+50*x,50-100*n)
19     down()
20     carre()
21
22 exitonclick()
```

1.1. Quel est le rôle du sous-programme carre() ?

.....

.....

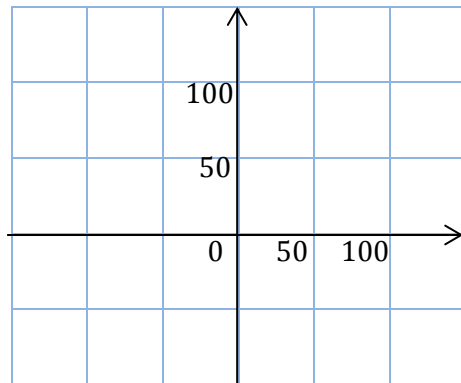
1.2. Quelles valeurs prennent les valeurs m, n, x ?

.....

.....

2. On se propose d'appliquer ce programme et de tracer la figure obtenue dans le repère ci-dessous.

2.1. Tracer la figure correspondant au résultat de la boucle bornée **for m in range (4)** :



2.2. Compléter la figure précédente par le résultat de la boucle bornée **for n in range (2)** :

On supposera que la variable x prend la valeur 0 puis la valeur 2.

2.3. Quel est le rôle de ce programme ?

.....

.....

## Activité 6

### Année bissextile ou non

**Le problème :** Déterminer si une année est bissextile ou non bissextile.

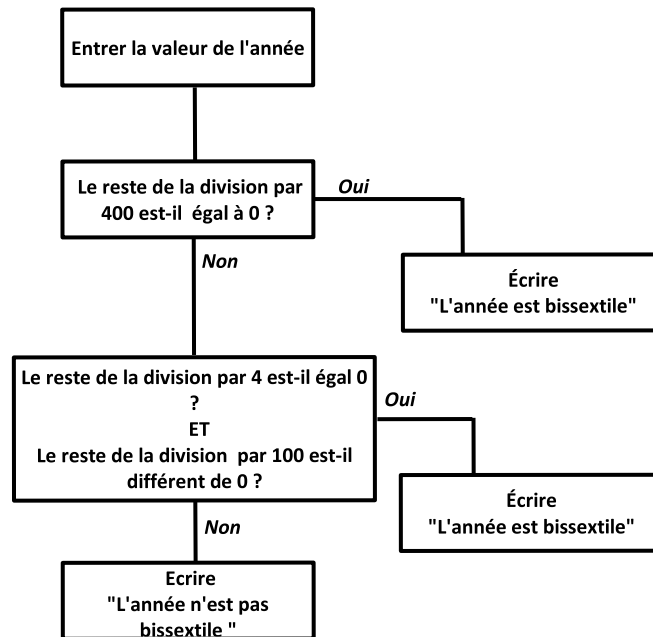
Une année est bissextile si :

▪ sa valeur (ex : 2015) divisée par 400 à pour reste 0

OU

▪ sa valeur divisée par 4 à pour reste 0 ET sa valeur divisée par 100 n'a PAS pour reste 0

**L'algorithme :**



**Les programmes Scratch et Python**



# Ce programme teste si une année est bissextile ou non

```
annee = input("Saisissez une année :")
```

```
annee = int(annee)
```

<https://scratch.mit.edu/projects/306792886/>

**Objectifs** : faire la différence entre un algorithme et un programme

1. **Observer** la vidéo (*sera mise sur le cloud académique*)
2. **Noter** les différentes étapes nécessaires pour obtenir une boisson

---

---

---

---

---

---

---

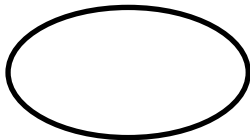
---

---



Qu'est-ce que l'utilisateur fait ?  
Qu'est-ce que la machine fait/indique ?

Pour organiser les différentes étapes les symboles suivant peuvent être utilisés :



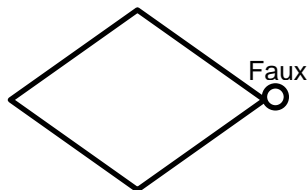
Début ou fin d'une action



Entrée / Sortie  
Lecture / Ecriture, ....

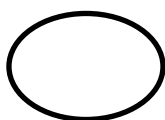


Instruction : ce qui doit être fait  
(calcul, comparaison par exemple)



Signale une condition : Si ... alors, sinon

3. **Associer** chaque consigne à son symbole :



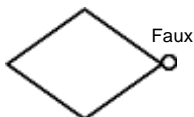
•

- Vérifier si la boisson existe.



•

- Faire tomber la boisson.



•

- Insérer la pièce et lire sa valeur.



•

- Comparer le prix de la boisson et le montant de la pièce.
- Début.
- Si la pièce est supérieur au prix alors...

4. **Schématiser** à l'aide des symboles le fonctionnement du distributeur.

*Distribuer l'algorithme afin que les élèves puissent le construire (cf. la dernière page)*

5. **Ouvrir** le fichier « Distributeur - Algo (Elève).sb3 ».

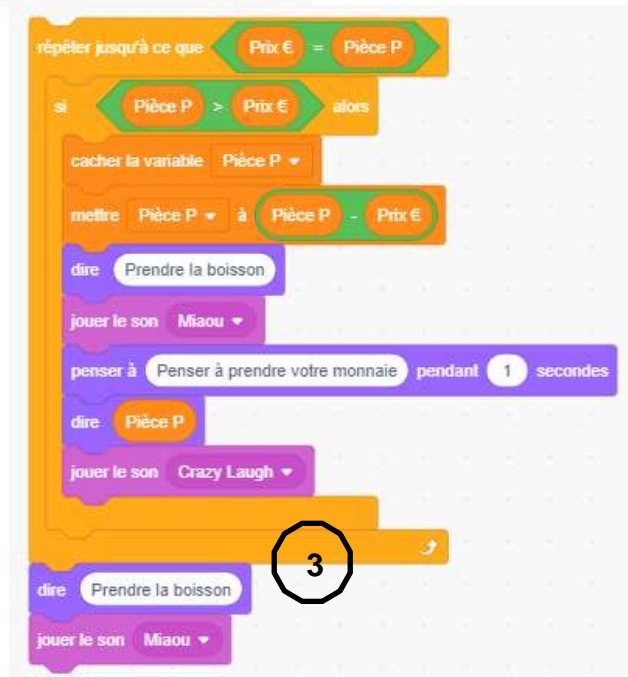
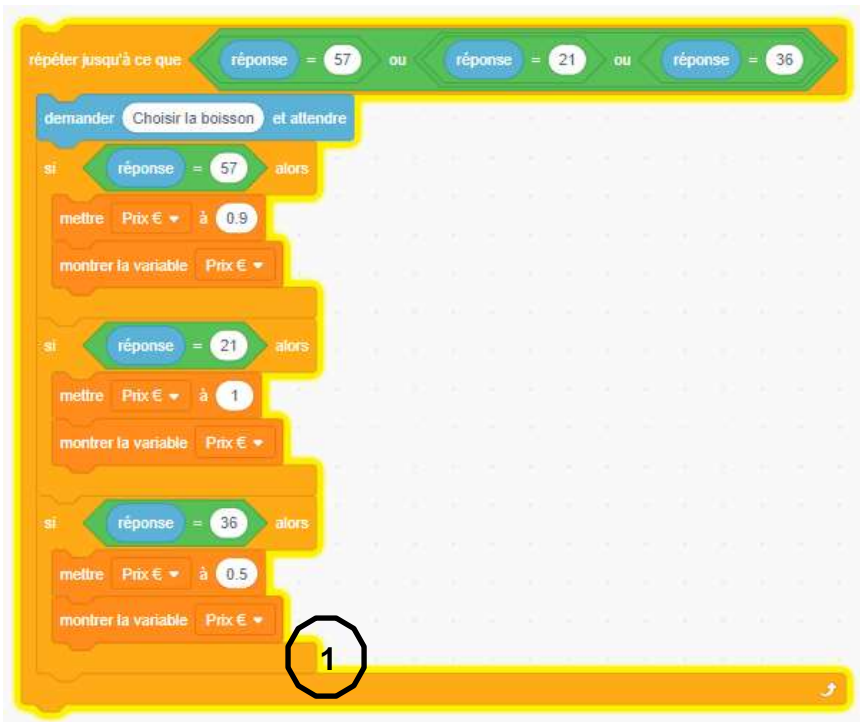
6. **Proposer** un classement pour les morceaux du programme fourni page suivante et **tester** le programme :

---

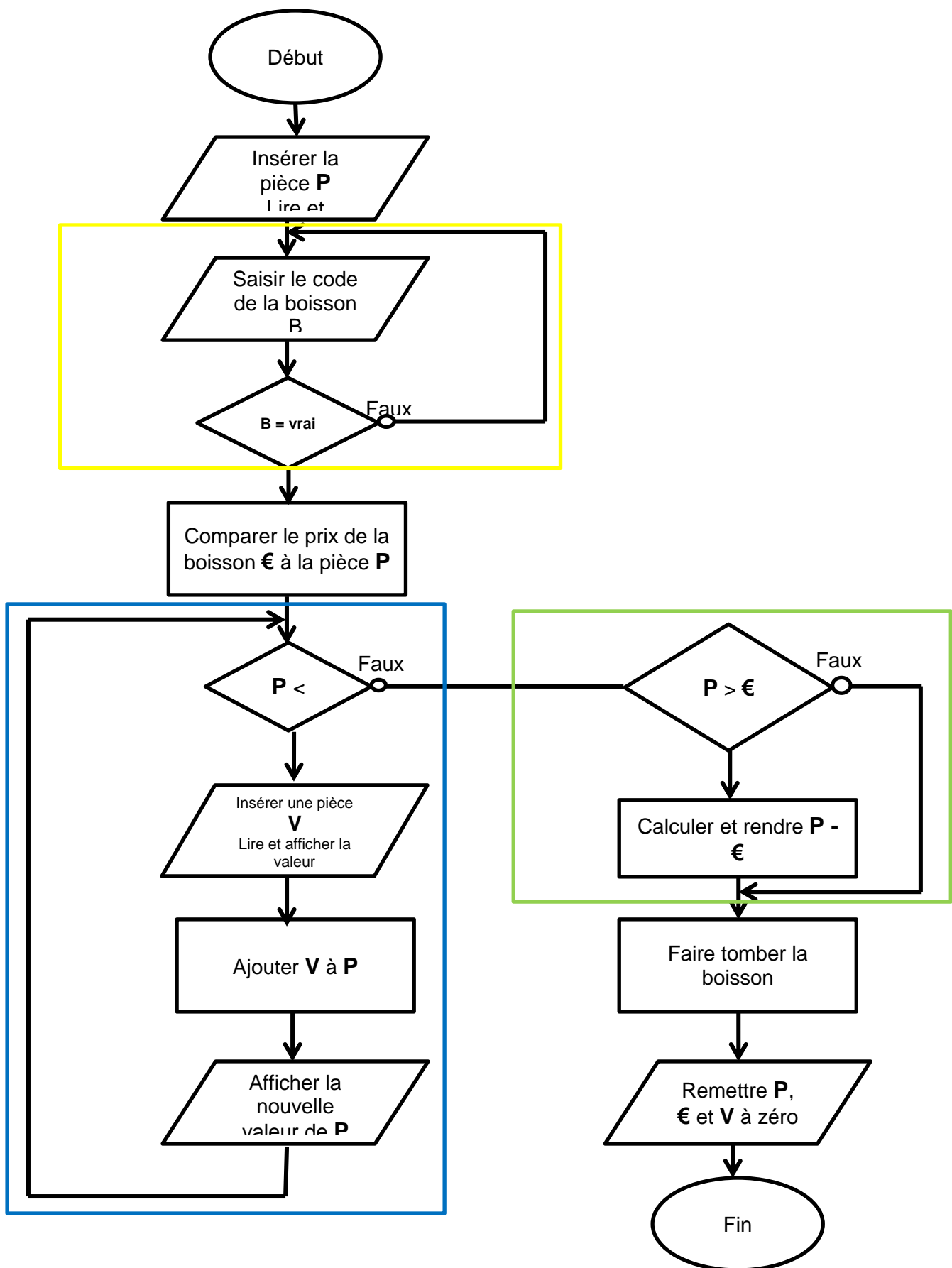
7. **Repérer** avec des couleurs différentes les boucles de l'algorithme (exemple jaune pour la boucle 1, bleu pour la boucle  $P < €$ , rouge pour la boucle  $P \geq €$ ). **Identifier** en coloriant les boucles correspondantes dans le programme.

////////////////////////////////////:  
Un **algorithme** est une suite d'instructions à appliquer dans un ordre logique pour résoudre un problème et obtenir rapidement un résultat. Il est écrit à la main ou à l'aide d'un logiciel dans un langage compréhensible par tous.

////////////////////////////////////:  
L'**algorithme** sert à préparer l'écriture d'un programme.  
////////////////////////////////////:



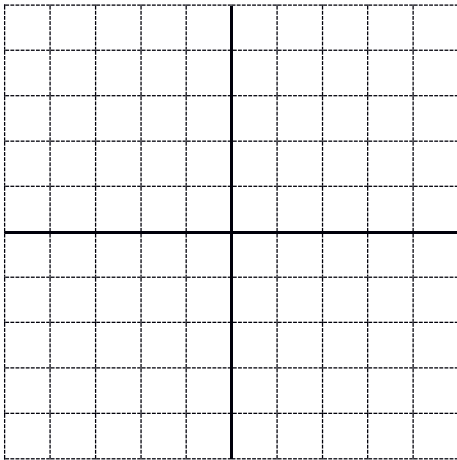
*A proposer aux élèves dans le désordre*




## Activité 8

# Illusion d'optique avec Scratch


On considère le quadrillage ci-dessous pour lequel le centre désigne le centre du repère du plan. L'écart entre deux lignes consécutives est de 20 pas (horizontalement et verticalement).



La forme  symbolise l'objet qui va permettre le tracé.

Cette forme est initialisée avec les instructions suivantes :

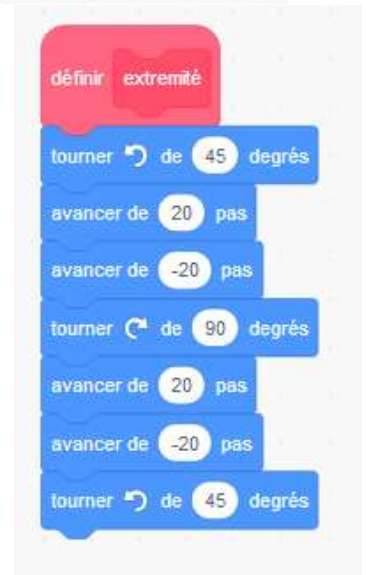


Déterminer la position de départ de la forme  sur le quadrillage précédent à l'aide des instructions ci-dessus.

1. Dans le quadrillage, faire un croquis du tracé obtenu en suivant les instructions données par le bloc `extrémité`
2. Proposer une instruction (à insérer avant le bloc `extrémité`) afin de tracer le symétrique par rapport à un axe vertical de la figure obtenue dans la question précédente. On ne doit pas changer le contenu du bloc `extrémité`.

(Aide : effectuer un demi-tour de cette feuille)

3. Aller à l'adresse <https://huit.re/illusion1-S>  
Déterminer la liste d'instructions permettant de tracer la figure suivante (le segment horizontal a une longueur de 160 pas).



4. Positionner la forme permettant le tracé 40 pas au-dessus de sa position de départ sans faire apparaître un tracé sur le graphique.
5. Proposer une liste d'instructions permettant de tracer la figure ci-dessous (le segment horizontal a une longueur de 160 pas).



6. Effectuer le tracé de ces deux figures sur le même graphique.
7. Les segments horizontaux sont-ils de la même longueur ou de longueurs différentes ?  
Expliquer en quoi cette représentation est une illusion d'optique.



# Illusion d'optique avec Python

1. Ci-dessous la partie de gauche liste les instructions données à `tortue`.

À l'aide de la page 4 du document « Algorithmique et Programmation » compléter la « traduction » de la fonction `extremite` :

Instructions Python	Actions à réaliser
<code>def extremite():</code>	définition de la fonction <code>extremite</code>
<code>tortue.left(45)</code>	tourner vers la .....de .....°
<code>tortue.forward(20)</code>	..... de ..... pas
<code>tortue.backward(20)</code>	reculer de ... ..... pas
<code>tortue.right(90)</code>	.....
<code>tortue.forward(20)</code>	.....
<code>tortue.backward(20)</code>	.....
<code>tortue.left(45)</code>	.....

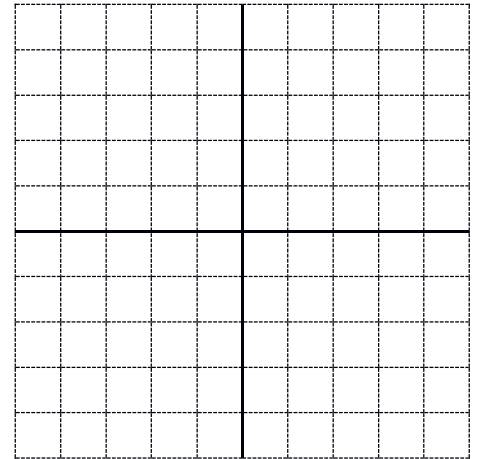
On considère le quadrillage ci-contre pour lequel le centre désigne le centre du repère du plan. L'écart entre deux lignes consécutives est de 20 pas (horizontalement et verticalement).

2. Proposer le point de départ de la tortue correspondant à l'instruction `tortue.goto(-80,-20)` sur le quadrillage ci-contre.

3. À l'aide du quadrillage ci-contre proposer le tracé obtenu en suivant les instructions données par `extremite`.

4. Proposer une instruction à placer avant la fonction `extremite` qui permettra de tracer le symétrique par rapport à un axe vertical de la figure obtenue dans la question précédente.

On ne doit pas changer le contenu de la fonction `extremite`.



(Aide : effectuer un demi-tour de cette feuille)

5. Récupérer le programme à l'adresse <https://huit.re/illusion1-py>, puis ajouter la liste d'instructions permettant de tracer la figure suivante (le segment horizontal a une longueur de 160 pas).

6. Positionner la tortue permettant le tracé, 40 pas au-dessus de sa position de départ sans faire apparaître un tracé sur le graphique.



7. Proposer une liste d'instructions permettant de tracer la figure ci-dessous (le segment horizontal a une longueur de 160 pas).



8. Effectuer le tracé de ces deux figures sur le même graphique.
9. Les segments horizontaux sont-ils de la même longueur ou de longueurs différentes ? Expliquer en quoi cette représentation est une illusion d'optique.

## Activité 9

# ALGORITHMIQUE

## Résolution d'une équation du premier degré

### 1) Résolution d'une équation du type $ax = b$

Créer un algorithme avec Scratch :

- Un lutin dit « Je sais résoudre  $ax = b$ . » quand le drapeau vert est cliqué.
- Quand on clique sur le lutin, il demande la valeur de  $a$ , puis la valeur de  $b$
- Il donne la solution en répondant «  $x = \dots$  »
- Enregistrer votre travail dans votre dossier « maths » sous le nom : equation\_1

En utilisant le programme créé, résoudre les équations :

- |                |       |
|----------------|-------|
| • $5x = 25$    | $x =$ |
| • $25x = 5$    | $x =$ |
| • $2,5x = 5$   | $x =$ |
| • $250x = 6,5$ | $x =$ |

### 2) Résolution d'une équation du type $a + x = b$

Créer un autre lutin dans la même fenêtre :

- Ce lutin dit « Je sais résoudre  $a + x = b$ . » quand le drapeau vert est cliqué.
- Quand on clique sur le lutin, il demande la valeur de  $a$ , puis la valeur de  $b$
- Il donne la solution en répondant «  $x = \dots$  »
- Enregistrer votre travail dans votre dossier « maths » sous le même nom : equation\_1

En utilisant le programme créé, résoudre les équations :

- |                      |       |
|----------------------|-------|
| • $5 + x = 25$       | $x =$ |
| • $25 + x = 5$       | $x =$ |
| • $2,5 + x = 5$      | $x =$ |
| • $9,73 + x = 27,89$ | $x =$ |

### 3) Résolution d'une équation du type $ax + b = c$

Dans une nouvelle fenêtre, créer un programme en s'inspirant des 2 précédents pour résoudre une équation du type  $ax + b = c$ .

Enregistrer votre programme sous le nom : equation\_2

En utilisant votre programme, résoudre les équations :

- |                              |       |
|------------------------------|-------|
| • $5x + 5 = 25$              | $x =$ |
| • $25x + 25 = 375$           | $x =$ |
| • $2,3x + 7,96 = 19$         | $x =$ |
| • $-8,7x + 111,773 = 56,789$ | $x =$ |

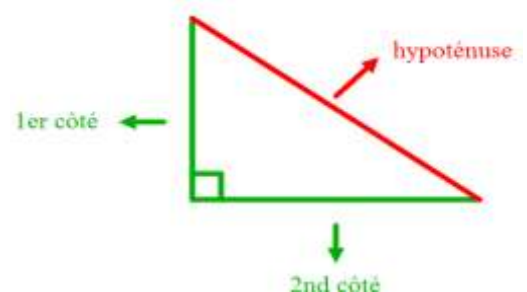
## Objectifs :

- créer un programme pour calculer la 3<sup>e</sup> cote d'un triangle rectangle
- organiser un programme pour vérifier qu'un triangle est rectangle en connaissant la mesure des 3 côtes.

## Activité 1:

On considère dans cette activité que l'on dispose d'un triangle rectangle et de deux de ses mesures. On cherche à calculer la 3<sup>e</sup> cote avec un programme SCRATCH.

Par souci de commodité pour la programmation, on considère le triangle rectangle ci-contre :



## Partie 1 : Calcul de la longueur de l'hypoténuse

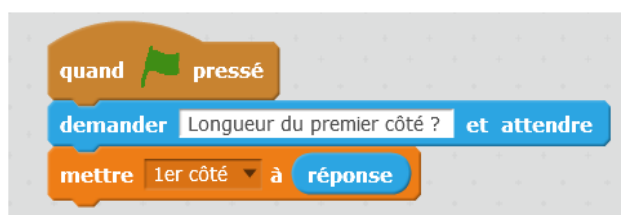
Ouvrir le logiciel SCRATCH.

Nous allons créer un premier programme « Théorème Pythagore » qui demande la valeur des deux côtés de l'angle droit d'un triangle rectangle et qui calcule la longueur de l'hypoténuse de ce triangle.

1. **Créer** 3 variables pour les longueurs des côtés du triangle qu'on appelle : 1er côté , 2nd côté et hypoténuse.



2. On demande alors la mesure du premier côté avec ce début de programme :



3. Faire de même pour la mesure du second côté en s'inspirant des deux derniers blocs de la question précédente.



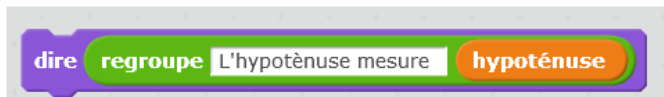
4. On peut maintenant calculer la longueur de l'hypoténuse. D'après le théorème de Pythagore, on sait que :

$$(\text{Hypoténuse})^2 = (1^{\text{er}} \text{ côté})^2 + (2^{\text{nd}} \text{ côté})^2 .$$

On a donc :



5. Il ne nous reste plus qu'à donner la réponse :



6. **Enregistrer** votre programme « Théorème Pythagore » sur le dossier perso / dossier / maths

## **Partie 2 : Tests du programme**

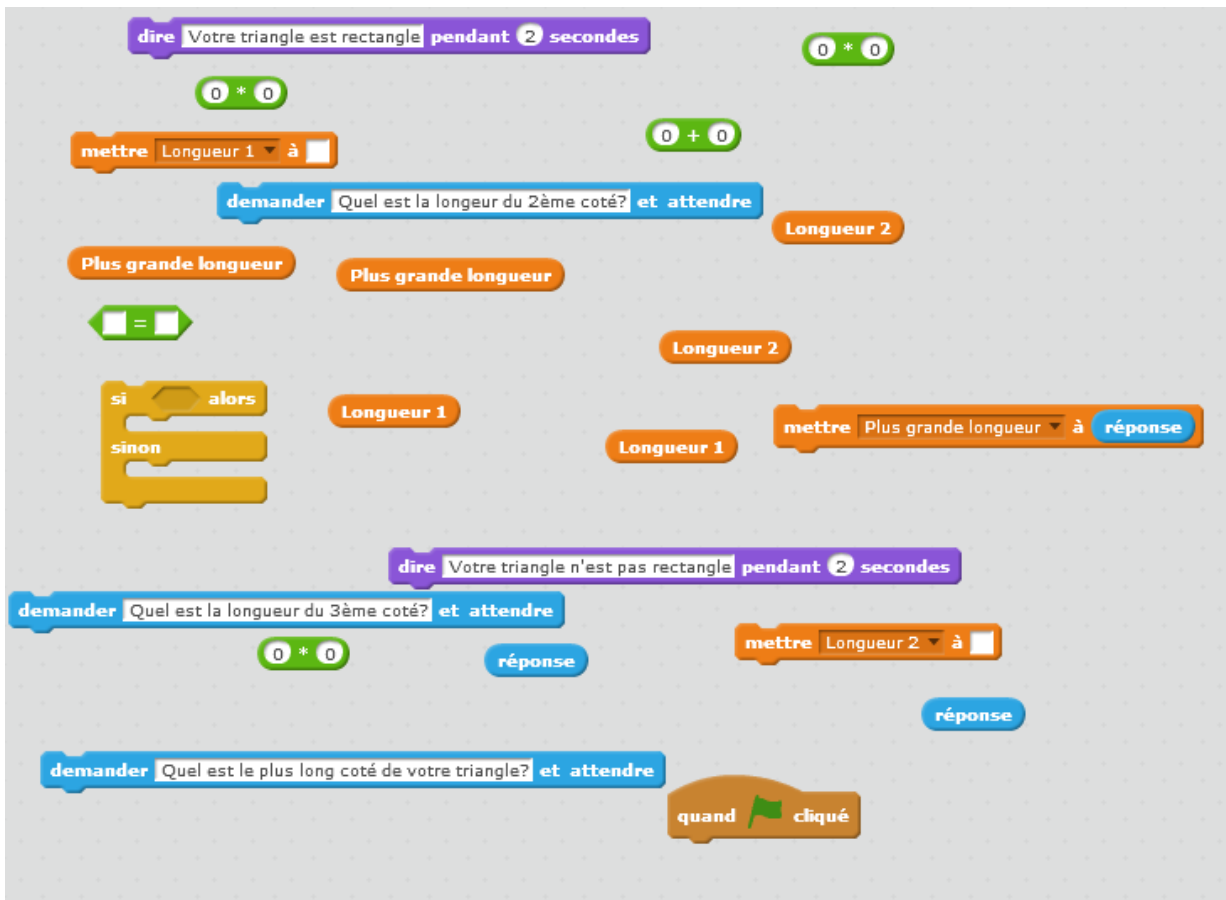
**Tester** les valeurs suivantes et vérifier que la longueur de l'hypoténuse fournie est correcte :

1 <sup>er</sup> côté	2 <sup>nd</sup> côté	Hypoténuse
3 cm	4 cm	5 cm
12 cm	35 cm	37 cm
39 cm	80 cm	89 cm

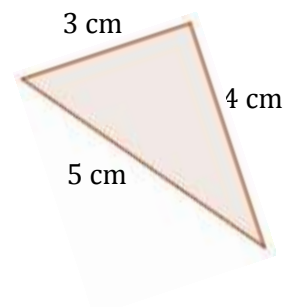
## Activité n°2:

Vous devez créer un programme qui précise à l'utilisateur si un triangle donné est rectangle en utilisant la réciproque du théorème de Pythagore.

Voici les différentes briques utilisées pour la conception de ce programme.



1. **Ouvrir** le fichier « Réciproque Pythagore élève ».
2. **Mettre** les briques dans l'ordre.
3. **Tester** votre programme avec le triangle ci-contre :
4. **Modifier** l'organisation des briques jusqu'à ce que le programme fonctionne.
5. **Enregistrer** votre programme.



### **Activité n°3:**

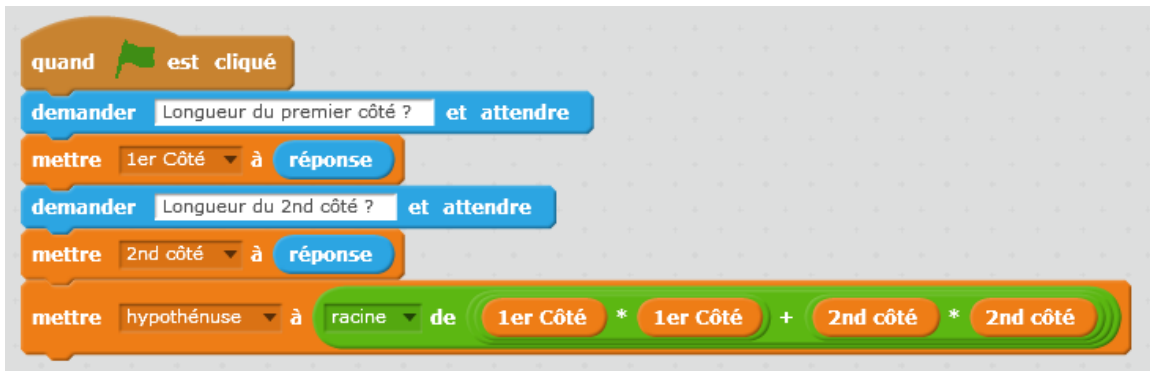
En utilisant le programme de l'activité 1 ou de l'activité 2, **compléter** le tableau suivant :

Programme utilisé	1 <sup>er</sup> côté	2 <sup>nd</sup> côté	Hypoténuse
	16 cm	63 cm	... cm
	... cm	77 cm	85 cm
	48 cm	55 cm	... cm
	9 cm	40 cm	... cm
	24 cm	... cm	25 cm

# Les programmes :

## 1. Activité 1 :

Scratch :



Python :

```
from math import*
```

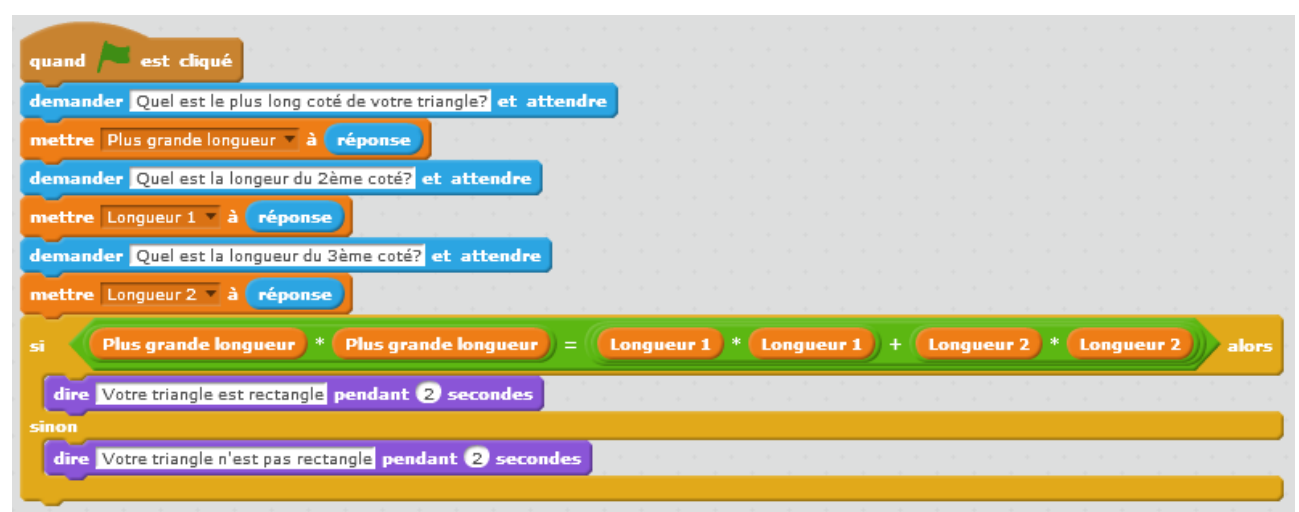
```
print("On cherche l'hypothénuse le triangle est rectangle")
```

```
a= float(input("Longueur du premier côté ?"))
b= float(input("Longueur du deuxième côté ?"))
c = round(sqrt((a**2+b**2)),2)
```

```
print("L'hypothénuse du triangle mesure ", c)
```

## 1. Activité 2 :

Scratch :



Python :

```
a = float(input("Quelle est la longueur du plus grand côté ?"))
b = float(input("Quelle est la longueur du deuxième côté ?"))
c= float(input ("Quelle est la longueur du troisième côté"))
```

```
if a**2 == b**2 + c**2 :
    print("Le triangle est rectangle")
else:
    print("Le triangle n'est pas rectangle")
```

### Jeu de pile ou face Simulation avec Scratch

L'objectif de cette activité est de simuler un jeu de « Pile ou Face ».  
La pièce n'est pas truquée.

Un premier programme vous est proposé. Vous devrez tout d'abord l'analyser, puis le modifier en utilisant les éléments proposés pour pouvoir l'améliorer.

#### **I. Etude de l'algorithme proposé**

L'objectif de cette première partie est d'étudier les grands principes de l'algorithme pour en comprendre les points clés.

1. Ouvrez Scratch en ligne sur votre poste informatique.  
Puis ouvrez le fichier « Pile ou face élève » disponible sur l'espace commun de la classe.  
*Rappel : seule une partie de l'algorithme est active (celle pour laquelle les instructions sont liées entre elles). Les autres éléments seront utilisés dans la deuxième partie.*
2. Testez le programme et vérifiez son bon fonctionnement.
3. Combien y a-t-il de variables actives dans ce programme ? Nommez ces variables.
4. Combien y a-t-il de « sprite » (ou lutins) utilisés ? Quels sont-ils ? Combien de costumes ont-ils chacun ?
5. Quelle est l'instruction qui permet de faire tourner la pièce et d'obtenir un résultat imprévisible ?
6. Comment sont gérés les enchaînements des actions entre chaque sprite ?



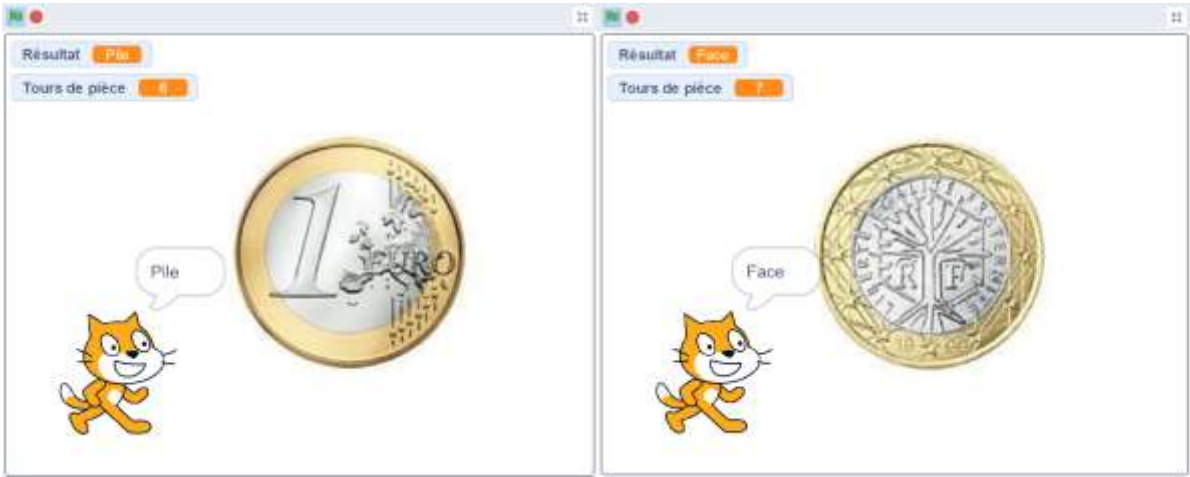
Décrire en quelques lignes le principe de fonctionnement du programme :

.....

.....

.....

.....



**II. Modification de l’algorithme**

Vous allez maintenant devoir modifier l’algorithme pour que le chat enchaîne plusieurs lancers de pièces. Pour cela, le chat va devoir :

- demander à l’utilisateur le nombre de lancers désirés,
- effectuer ces lancers,
- compter, au fur et à mesure, le nombre de « pile » et le nombre de « face » obtenus,
- annoncer le résultat final.

Activez toutes les variables disponibles, puis modifiez le programme en insérant les commandes proposées en marge et en enlevant éventuellement les commandes inutiles.

**III. Exploitation du programme.**

1. En utilisant le programme réalisé (ou celui fourni par le professeur), vous allez maintenant simuler 5 fois 10 lancers de pièce.

Notez les résultats obtenus dans le tableau ci-dessous, et complétez la troisième ligne des fréquences :

Lancer numéro :	1	2	3	4	5
Nombre de « Pile »					
Fréquence de « Pile »					

Tableau n° 1 : 10 lancers de pièce

Que remarquez-vous ?

.....

2. Simulez maintenant 50 lancers, et remplissez le tableau correspondant :

Lancer numéro :	1	2	3	4	5
Nombre de « Pile »					
Fréquence de « Pile »					

*Tableau n° 2 : 50 lancers de pièce*

3. Puis 100 lancers :

Lancer numéro :	1	2	3	4	5
Nombre de « Pile »					
Fréquence de « Pile »					

*Tableau n° 3 : 100 lancers de pièce*

**Conclusion** : comment pouvez-vous interpréter ces résultats ?

### **L'ESSENTIEL**

.....

.....

.....

.....

.....

.....

.....

### PARTIE A : Algorithmie

Paul demande de l'aide à son grand père pour réaliser un triangle équilatéral.

- Grand-père : « C'est pas compliqué, voilà comment j'ai appris. *Tu mets un point sur la feuille, tu traces un trait de 5 cm. À l'autre bout, tu mesures un angle de 60° avec ton rapporteur pour tracer un deuxième trait de 5 cm. Puis, tu mesures à nouveau un angle de 60° avec ton rapporteur pour tracer le troisième coté de 5 cm.* »
- Paul : « Merci papy, mais il faut le faire sur ordinateur avec un programme. »
- Grand-père : « Ah ! Vous les jeunes avec vos écrans ! »

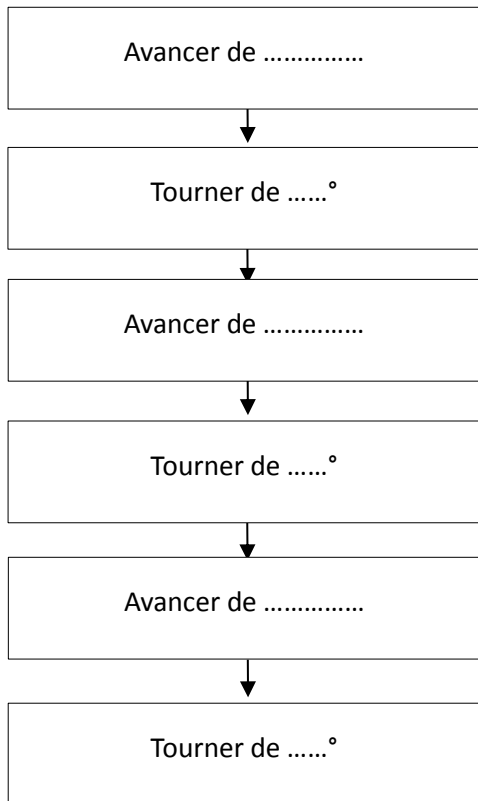
Comment transformer la méthode de construction du grand-père  
et montrer un intérêt des « écrans » ?

Dans la suite de l'énoncé, on considère qu'un trait de 5 cm est réalisé en avançant de 50.

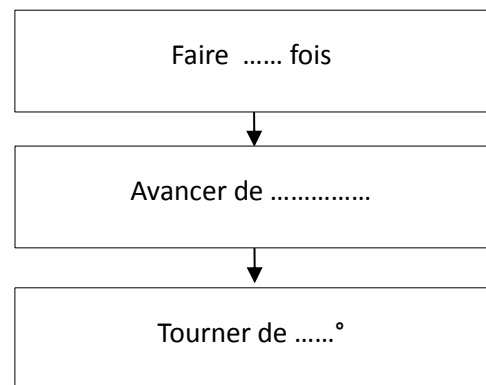
1. À l'aide des informations données par le grand-père, compléter les étapes permettant de tracer un triangle équilatéral.
2. Écrire le programme correspondant.
3. Après avoir testé le programme, que constate-t-on ?  
Proposer une correction permettant de tracer correctement le triangle équilatéral.
4. Des suites d'instructions étant répétées plusieurs fois, proposer une simplification du programme à l'aide d'une boucle.
5. Modifier le programme pour tracer un carré.
6. Modifier le programme pour tracer un pentagone régulier.
7. Modifier le programme pour tracer un hexagone régulier.
8. Proposer un programme qui interroge l'utilisateur sur le nombre de cotés désirés et qui trace le polygone régulier correspondant.

## Aide possible

### Question 1



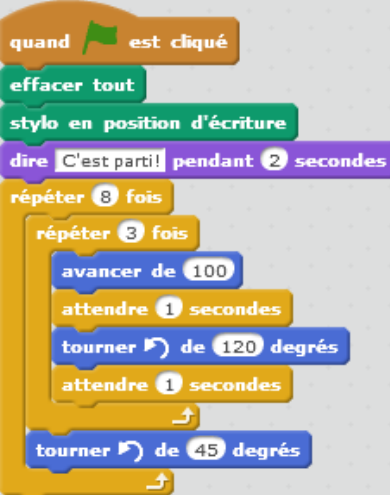


### Question 3



# Les figures sous Scratch

## 1) Activité découverte

Algorithme 1	Algorithme 2	Algorithme 3
		

1) Que fait le chat dans l'algorithme 1 (répondre par une phrase) ?

2) Ouvrir SCRATCH

Recopier l'algorithme 1 puis lancer le programme.

3) La proposition faite en 1) est-elle la bonne ?

4) Que fait le chat dans l'algorithme 2 (répondre par une phrase) ?

5) Recopier l'algorithme 2 dans SCRATCH et lancer le programme.

6) La proposition faite en 4) est-elle la bonne ?

7) Quel est l'avantage de l'algorithme 2 par rapport à l'algorithme 1 ?

8) Recopier l'algorithme 3 dans SCRATCH et lancer le programme.

9) Qu'est ce que l'algorithme 3 fait en plus des précédents ?

### II) Première construction : un carré

En vous inspirant des algorithmes précédents, créer un programme qui permette de tracer un carré.



Le programme doit être le plus court possible !

Copier ci-dessous le dessin et l'algorithme (Outil Capture de Windows)

### III) Deuxième construction : le rectangle

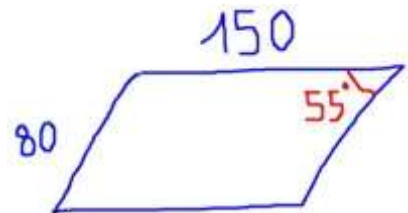
En vous inspirant des algorithmes précédents, créer un programme qui permette de tracer un rectangle.

Copier ci-dessous le dessin obtenu et l'algorithme.

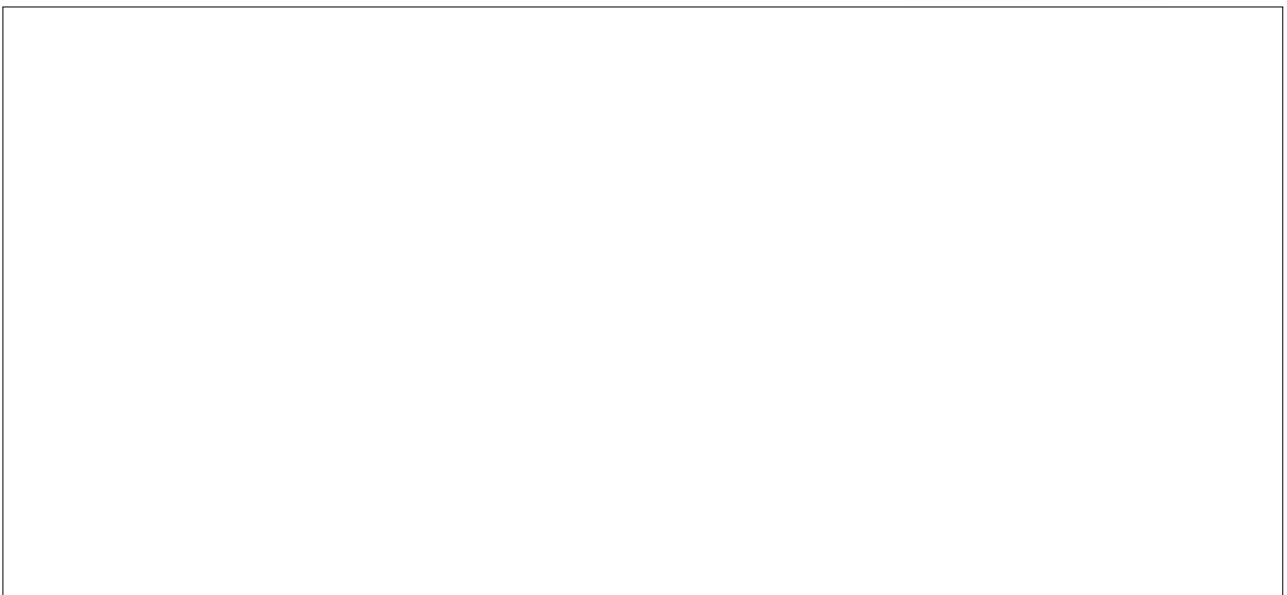


### IV) Troisième construction : le parallélogramme

En vous inspirant des algorithmes précédents, créer un programme qui permette de tracer le parallélogramme suivant :



Copier ci-dessous le dessin obtenu et l'algorithme



### V) Dernière construction : le pentagone (5 côtés)

Créer un programme qui permette de tracer un pentagone.

(Quand on additionne tous les angles à l'intérieur du pentagone, on obtient 360...)

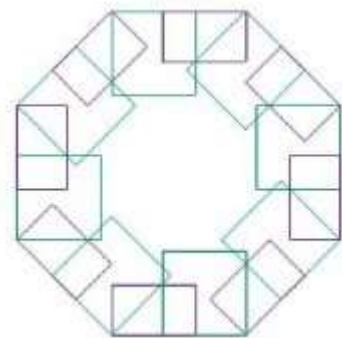
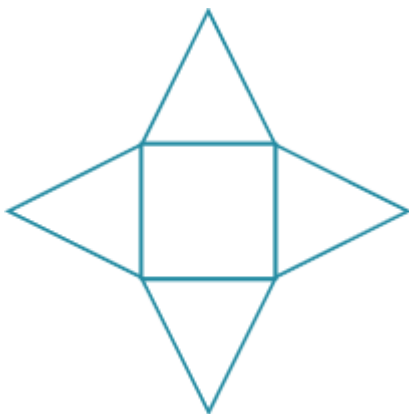


Copier ci-dessous le dessin obtenu et l'algorithme :



### V) Encore mieux

Etes-vous capables de créer des figures plus complexes ?



A vous de jouer !



## PARTIE C : Python

### python

Pour vous familiariser avec le langage Python, essayez ces quelques commandes directement dans l'interpréteur python :

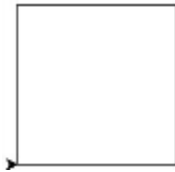
1

```
>>> from turtle import *  
>>> forward(120)  
>>> left(90)  
>>> color("red")  
>>> forward(80)  
>>> reset()
```

Analyser chacune de ces lignes et expliquer ce que fait chacune des commandes.

2 Carré

Tracer un carré de 100 pixels de côté.



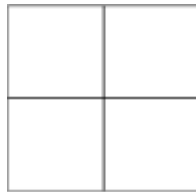
4 Carré ?

Tracer la figure ci-dessous.  
Le carré fait 100 pixels de côté.



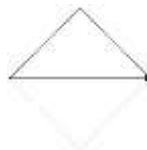
3 Carrés

Tracer la figure ci-dessous.  
Chaque carré fait 100 pixels de côté.



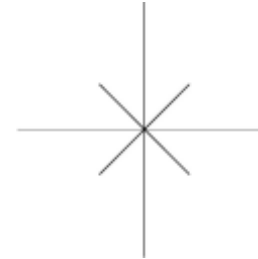
5 Carré ? (2)

Tracer la figure ci-dessous.  
Le carré fait 100 pixels de côté.



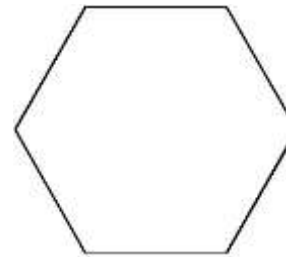
6 Cristal

Tracer le cristal ci-dessous. Les grands segments mesurent 300 pixels, les petits 150.



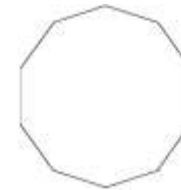
8 Hexagone

Tracer l'hexagone ci-dessous. Les 6 côtés mesurent 100 pixels.



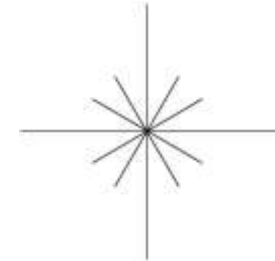
10 (Double) décagone

1. Tracer le décagone ci-dessous.



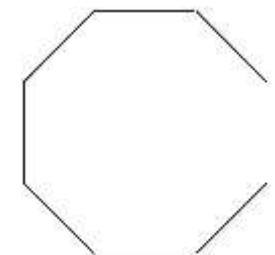
2. Tracer un double décagone

7 Cristal (2) Tracer le cristal ci-dessous. Les grands segments mesurent 300 pixels, les petits 150.



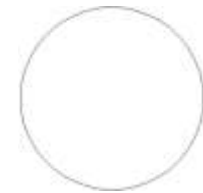
9 Octogone

Tracer le pentagone ci-dessous. Les 8 côtés mesurent 100 pixels.

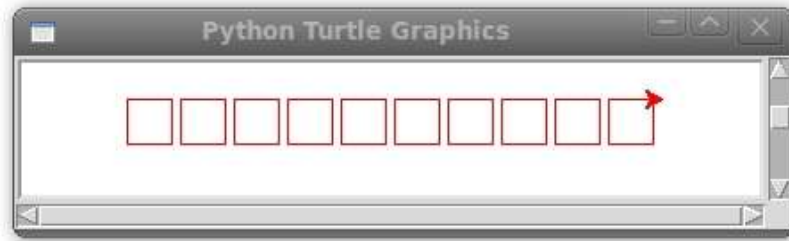


11 Cercle

Tracer le cercle ci-dessous.



- 12 Réaliser le dessin ci-dessous à l'aide du module turtle :



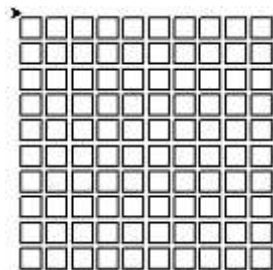
- 13 Réaliser le dessin ci-dessous à l'aide du module turtle :



- 14 Réaliser le dessin ci-dessous à l'aide du module turtle :



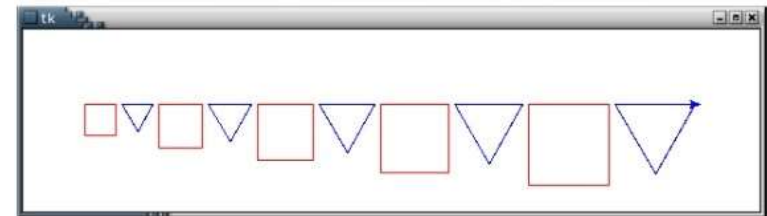
- 15 Réaliser le dessin ci-dessous à l'aide du module turtle :



- 16 Modifier le code du programme précédent pour que le carré se trouvant en ligne 4 colonne 8 soit rouge

- 17 Ecrire un programme qui dessine un carré de côté 100 "plein"

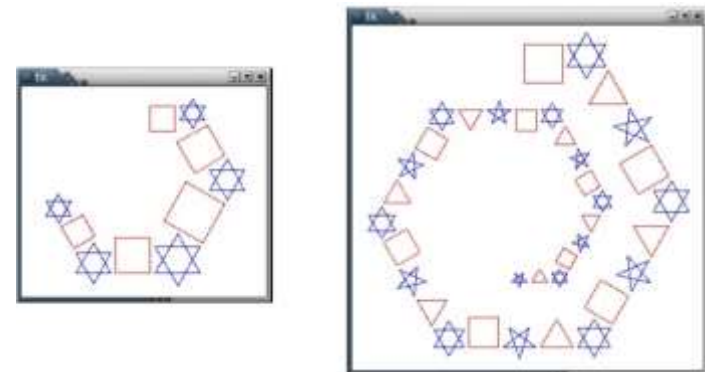
- 18 Réaliser le dessin suivant :



- 19 Dessiner la figure suivante :



- 20 Créer un script qui dessinera une série de ces étoiles :



## Activité 13

### Fonction affine : calcul de l'image

On a créé un programme qui demande de calculer l'image d'une fonction affine et qui vérifie si le résultat est correct ou non.

Page 2, on trouvera l'algorithme et le code en Python correspondant au programme.

#### 1. Compréhension de l'algorithme et du programme

a. `a = randint(-10, 10)`

**Expliquer** quelle valeur prendra la variable a :

.....

.....

b. Si nb = 3, **écrire** ci-dessous la phrase affichée grâce à l'instruction : `print("Donne l'image de", (nb))`

.....

.....

c. **Expliquer** à quoi sert le bloc suivant dans le programme :

```
while not(reponse == (a*nb+b)):  
    print("Ce n'est pas la bonne réponse ")  
    print("Donne l'image de", (nb))  
    reponse =int(input())
```

.....

.....

.....

d. **Lancer** EduPython et **ouvrir** le fichier « Calcul\_image.py »

**Compléter** le programme.

**Vérifier** qu'il fonctionne.

#### 2. Calculs successifs

À chaque fois que la réponse est correcte, on souhaite recommencer automatiquement le programme et cela 10 fois de suite.

a. **Ajouter** sur l'algorithme de la page 2, la boucle permettant de recommencer 10 fois le programme

b. En Python, pour recommencer 10 fois les mêmes instructions il faut écrire :

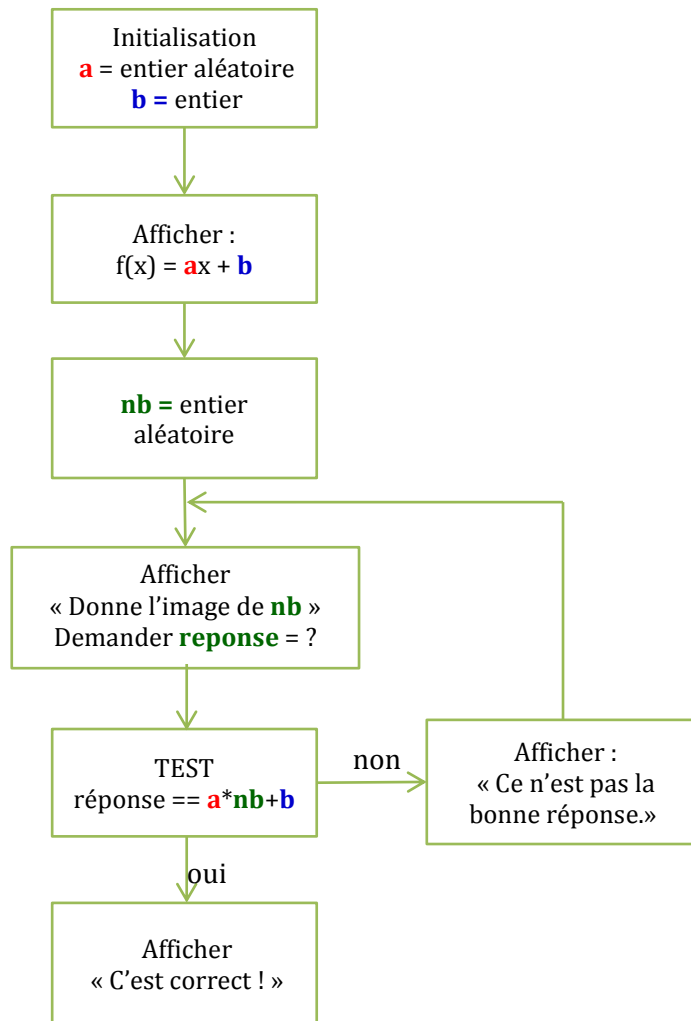
```
for i in range(10) :
```

écrire avec une tabulation le code qui doit s'effectuer 10 fois

**Ajouter**, sur le programme Python, la boucle permettant d'effectuer les calculs 10 fois.

**Attention** : Il faut penser ajouter une tabulation pour les instructions situées dans un bloc.

## Algorithme



## Programme Python

```

from random import randint

a = randint(-10, 10)
b = randint(-10, 10)
print("f(x)=", a, "x + (", b, ")")

nb = randint(-10, 10)

print("Donne l'image de", (nb))
reponse = int(input())

while not(reponse == (a*nb+b)):
    print("Ce n'est pas la bonne réponse.")
    print("Donne l'image de", (nb))
    reponse = int(input())

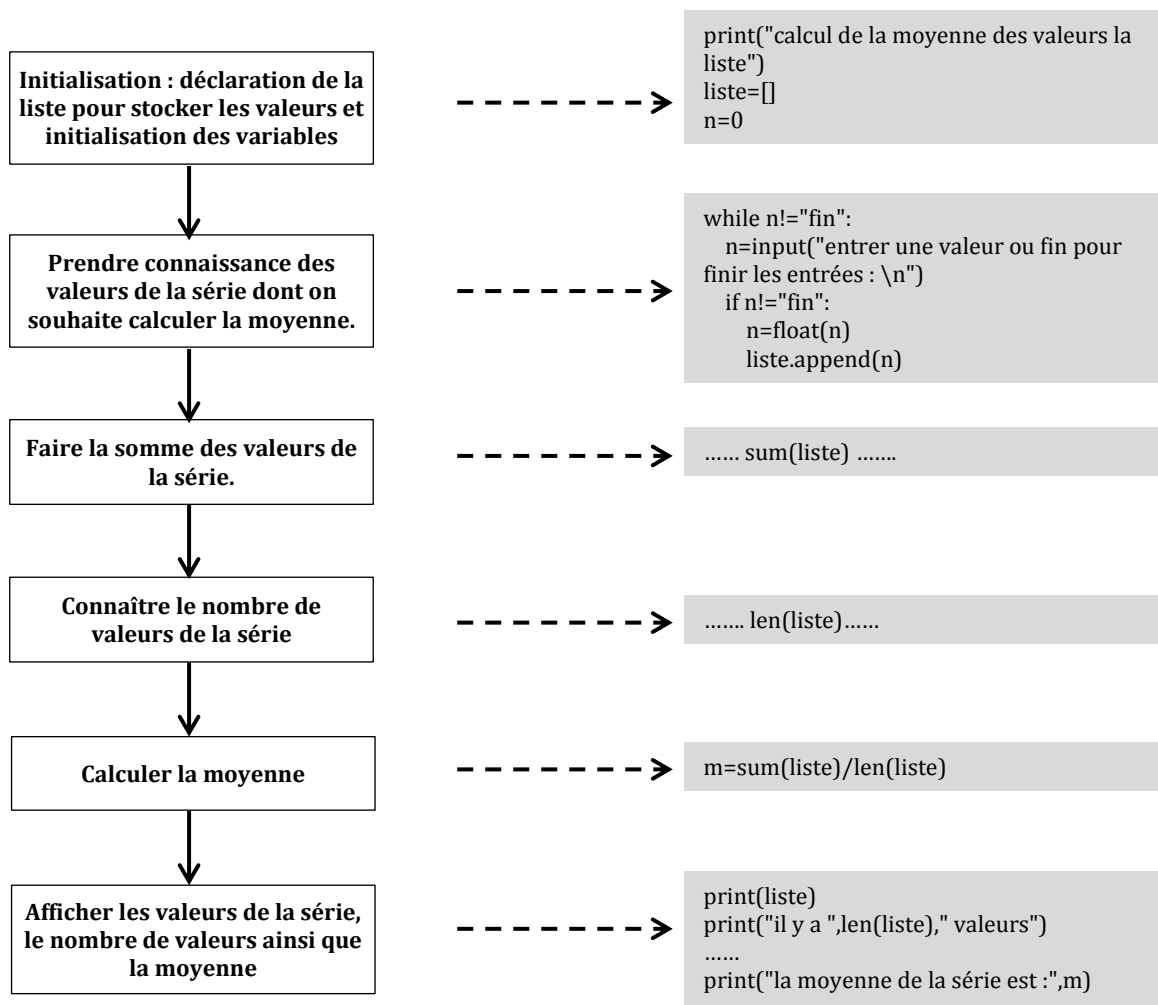
print("C'est correct !")
  
```

### Correspondance entre les langages :

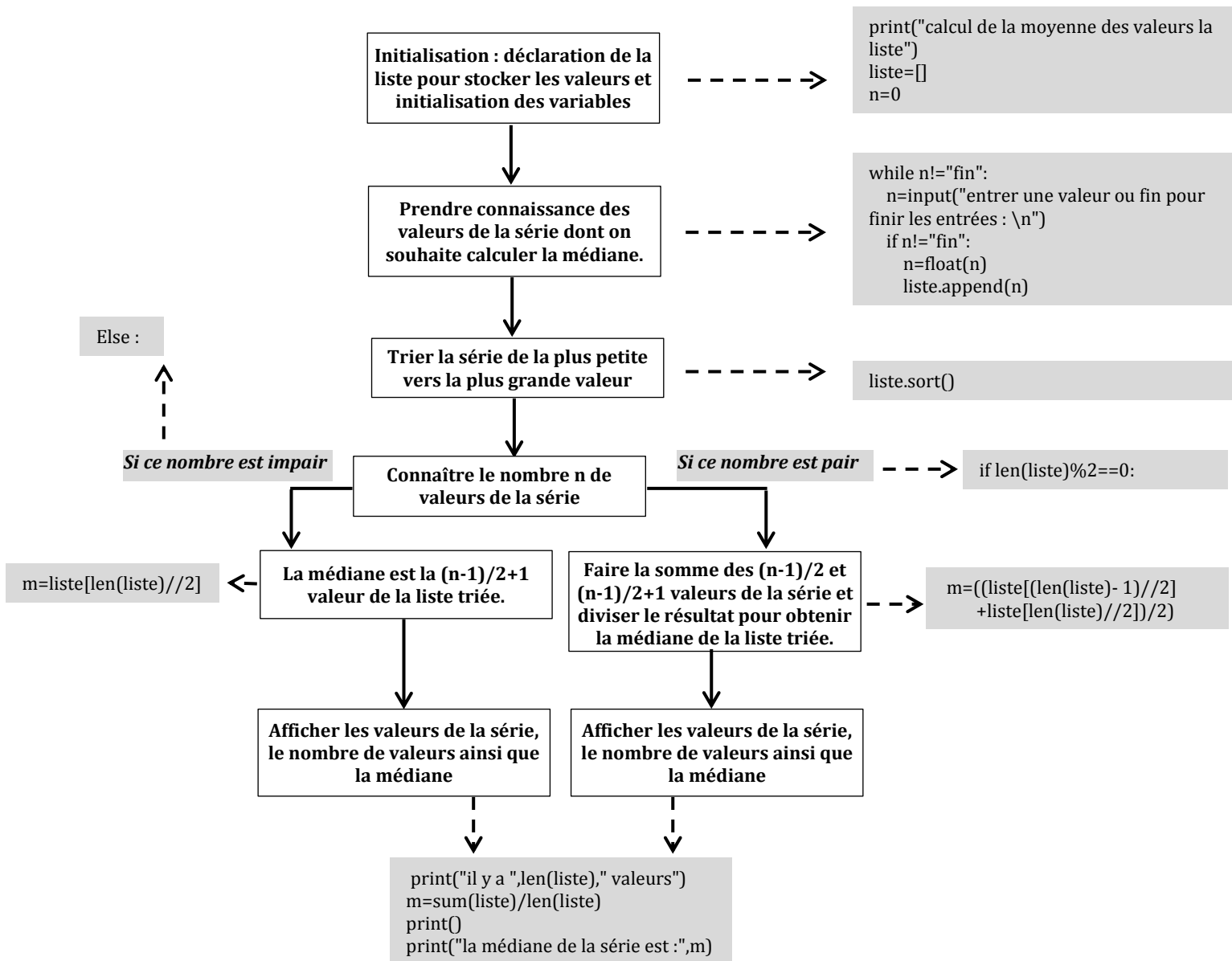
Algorithme	Programme Python
affiche	print()
demande	input()
tant que	while

## Activité 14

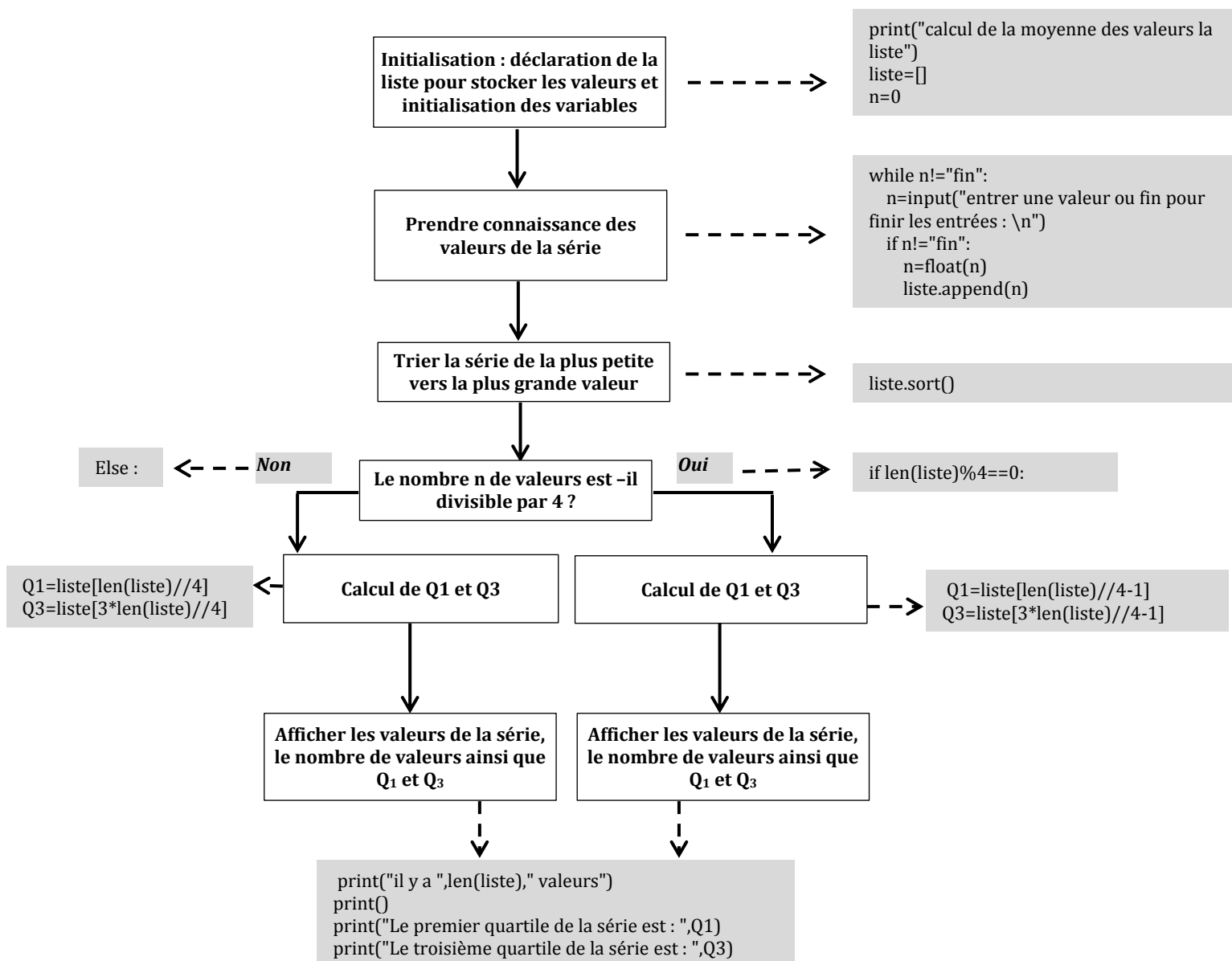
### ① Moyenne



## 2 Médiane



### ③ les quartile $Q_1$ et $Q_3$



## Activité 15

### Le tableau de classification périodique

Exemple d'activité Scratch : le tableau de classification périodique automatique.

Fiche pédagogique

#### 1 – Poser la demande clairement :


*"On souhaite une animation présentant les trois premières lignes du tableau de classification des éléments qui permettent par l'appui sur une touche (la Touche ESPACE par exemple !) de demander le numéro atomique de l'élément que l'on souhaite et d'afficher grâce à la réponse :*

- le nom de l'élément
- le symbole de l'élément
- Le nombre de protons
- le nombre d'électrons
- le nombre de neutrons

*Et avec un affichage agréable. Un nouvel appui sur la touche choisie permet un choisir un nouvel élément"*

#### 2 – A quel niveau l'élève entre-t-il dans l'activité ?

- On peut laisser l'élève faire ces choix tout seul, mais en général il aura des difficultés à entrer dans l'activité.




- On peut lui donner les premières choses à faire pour qu'il puisse commencer sereinement. C'est sûrement la meilleure façon de procéder, et une aide à chaque étape est souvent la clé pour arriver à la fin de l'activité.

- Donner le programme incomplet et avec des commentaires sur les blocks importants demander à l'élève de reconstituer le programme.

Seul désavantage : lire un programme écrit par quelqu'un d'autre est assez difficile.

Par contre pour les algorithmes compliqués c'est utile.



- Fabriquer trois listes : une liste "nom", une liste "Symbole", et une liste "nombre de masse"

*puis compléter ces trois listes avec les éléments du tableau de classification, dans l'ordre de Z croissant*

- Ecrire dans la scène le texte de présentation en laissant des trous pour que l'on puisse y placer les variables

- Créer cinq variables : "A", "nom", "symbole", "p", "e", "n" et les placer dans la scène à la place des trous.



### 3 – Fabriquer l'algorithme

Il est intéressant de fabriquer l'algorithme avec les élèves, c'est souvent le moment de discuter du produit fini.

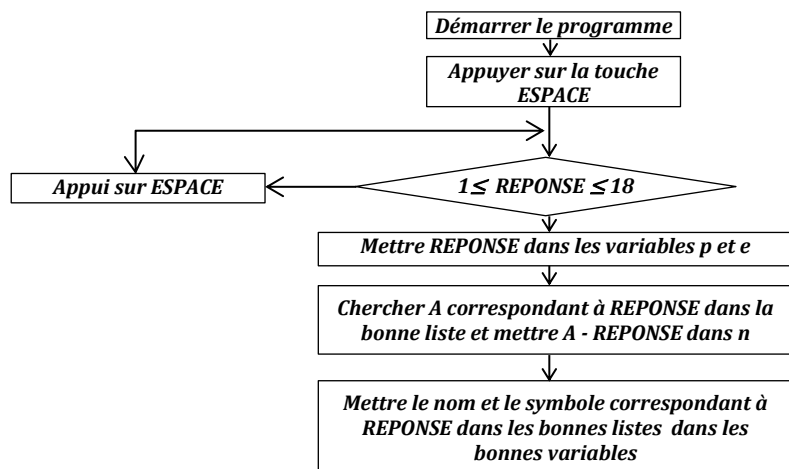


Tableau périodique simplifié des éléments

1 H 1 Hydrogène							4 He 2 Hélium
7 Li 3 Lithium	9 Be 4 Béryllium	11 B 5 Bore	12 C 6 Carbone	14 N 7 Azote	16 O 8 Oxygène	19 F 9 Fluor	20 Ne 10 Néon
23 Na 11 Sodium	24 Mg 12 Magnésium	27 Al 13 Aluminium	28 Si 14 Silicium	31 P 15 Phosphore	32 S 16 Soufre	35 Cl 17 Chlore	40 Ar 18 Argon
39 K 19 Potassium	40 Ca 20 Calcium	...					

