Le langage python est préconisé par l'éducation nationale dans l'algorithmique en seconde.

L'apprentissage de l'algorithmique avec ce langage est facilité par la possibilité de tester chaque instruction, dans l'environnement de l'interpréteur. C'est un langage très puissant et assez simple. Il est apte à la récursivité et est orienté objet donc tout à fait adapté à un environnement fenêtré.

Vous pouvez télécharger le langage python sur le site python de l'académie de Grenoble.
Cette page de téléchargement vous explique comment l'installer.

Je reprends dans ce tutoriel la plupart des exemples que vous avez pu trouver dans le didacticiel dédié à **algobox** !
On remarquera que la plupart des exemples sont plus court et plus rapide à mettre en œuvre avec python.
On remarquera aussi que la moindre erreur de syntaxe provoque un arrêt du programme et que nous ne pouvons pas utiliser le *mode pas à pas* avec autant de souplesse qu'avec **algobox.**

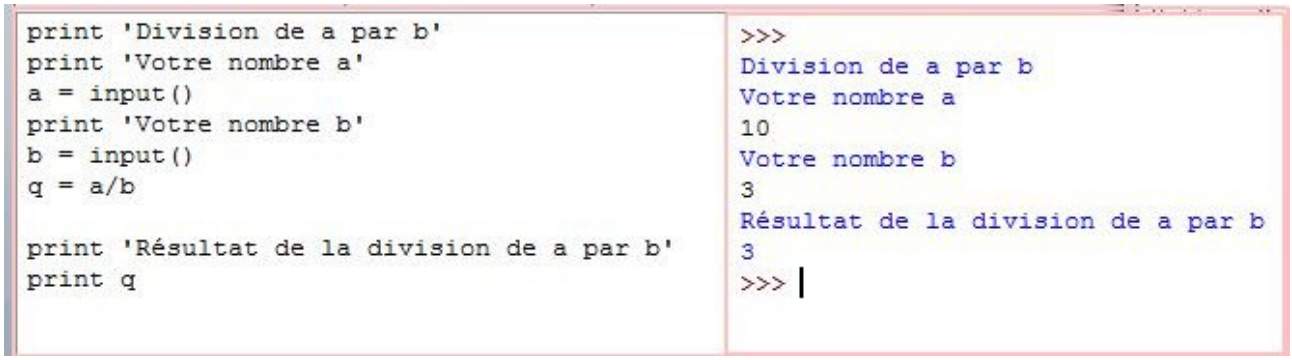
Puisque le **langage Python** permet de manipuler les *listes chaines* avec une grande facilité je propose ici le jeu du *cadavre exquis* en place des *palindromes* sous **algobox**

De même, au niveau du graphisme le module ***turtle*** nous permet de travailler la *récursivité* avec les *fractales* et le *flocon de van koch*

Sommaire

I – La division

Lancer Python en cliquant sous Windows ***sur IDLE (Python GUI).***
Faire ***File – New Window*** et taper le programme ci dessous. Une fois le programme tapé, sauvegardez le avec ***File – Save as*** puis exécutez le en appuyant sur ***F5***



```
print 'Division de a par b'
print 'Votre nombre a'
a = input()
print 'Votre nombre b'
b = input()
q = a/b

print 'Résultat de la division de a par b'
print q
```

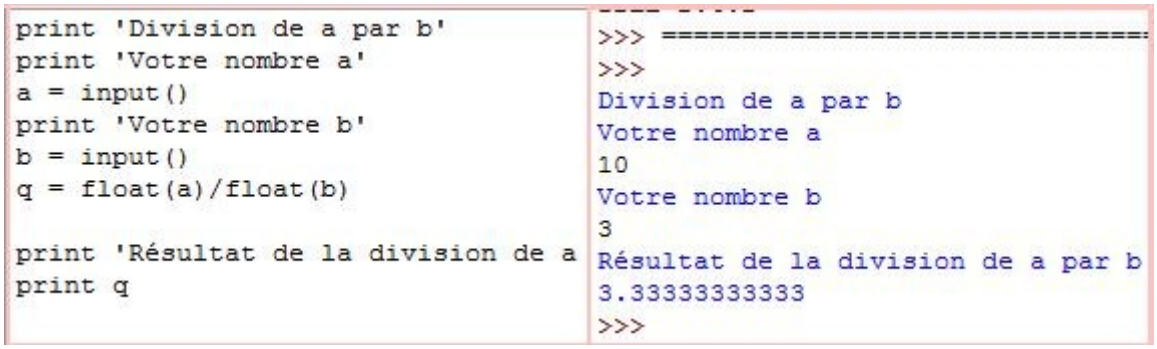
```
>>>
Division de a par b
Votre nombre a
10
Votre nombre b
3
Résultat de la division de a par b
3
>>>
```

à gauche le programme à taper et à droite le résultat à l'exécution.
La fonction ***a = input()*** est équivalente à la commande ***lire a*** dans **algobox**
De même la commande **print 'Votre nombre a '** correspond à **Afficher 'Votre nombre a '** dans **algobox**.

A l'utilisation, on s'aperçoit que ***python*** nous a fait une *division entière* ! Il existent au moins deux possibilités pour palier ce problème :
1* *Taper 3,0* comme diviseur pour lui faire comprendre que l'on en veut un peu plus

```
IDLE 2.6.2
>>> ================================ RESTART ================================
>>>
Division de a par b
Votre nombre a
10
Votre nombre b
3.0
Résultat de la division de a par b
3.33333333333
>>>
```

2* Ou bien *spécifier dans le programme* que l'on veut travailler avec des nombres réels (***float***) comme ceci



```
print 'Division de a par b'
print 'Votre nombre a'
a = input()
print 'Votre nombre b'
b = input()
q = float(a)/float(b)

print 'Résultat de la division de a
print q
```

```
>>> ================================
>>>
Division de a par b
Votre nombre a
10
Votre nombre b
3
Résultat de la division de a par b
3.33333333333
>>>
```

Cette seconde solution est évidemment plus propre car indépendante de l'utilisateur final.

1* Empêcher le diviseur nul

Nous allons utiliser la structure **While** (identique à ***TANT_QUE*** dans ***algobox***) de la façon suivante : nous mettons le diviseur **b** à ***zéro*** et nous demandons une *valeur non nulle* tant que ***b sera nul*** !
Attention à la syntaxe, après le mot **while** il faut placer **une condition** puis **deux points**; !
lorsqu'alors vous appuyez sur entrée ***l'indentation*** se fera automatiquement et, toutes les instructions indentées de même niveau appartiendront au même bloc d'instructions.

```
print 'Division de a par b'
print 'Votre nombre a'
a = input()

b=0
while (b == 0):
    print 'Votre nombre b (non bul)'
    b = input()

q = float(a)/float(b)

print 'Résultat de la division de a par b'
print q
```

```
>>>
Division de a par b
Votre nombre a
10
Votre nombre b (non bul)
0
Votre nombre b (non bul)
3
Résultat de la division de a par b
3.33333333333
>>>
```

Exercice 1 – Transformer le programme en division euclidienne. Les résultats affichés seront le quotient et le reste.
Indication technique : l'opération a%b donne le reste dans la division de a par b

II – Statistiques, moyenne et variance

Les instructions précédées du signe # ne sont que des *commentaires* et ne seront pas exécutées, elles sont là pour donner des indications,,, ce qui est indispensable lorsque l'on doit programmer en équipe par exemple.
Le langage *python* permet de définir *des fonctions* qui retournent une *valeur en sortie* souvent après avoir récupéré une *valeur en entrée*.
Ici nous définissons la fonction stat_moyenne qui permet de calculer la moyenne de la liste de notes nommée echantillon,
Il ne reste plus qu'à passer la liste *echant* à la fonction pour récupérer dans *moy* la moyenne des notes.

```
# -*- coding: cp1252 -*-
# on défini une fonction stat_moyenne qui retourne
# la moyenne de la liste echantillon

def stat_moyenne( echantillon ) :
    taille = len( echantillon )
    moyenne = float(sum( echantillon) ) / taille
    return moyenne

# on récupère une liste de nombre dans echant
print 'vos notes séparées par des virgules SVP'
echant = input()

# on met dans moy la valeur retournée par stat_moyenne
moy = stat_moyenne(echant)

# on écrit cette moyenne
print 'la moyenne de vos notes est '
print float(moy)
```

```
vos notes séparées par des virgules SVP
10,11,12,13,9,8,7,17
la moyenne de vos notes est
10.875
>>>
```

Exercice 2 : en reprenant les formules de votre cours de maths calculer la moyenne et la variance de la série de notes.
Si possible en définissant une seconde fonction que l'on pourrait appeler ***stat_variance*** *par exemple*

Dans cet exercice vous aurez besoin d'élever au carré et, pour ce faire, il suffit de faire une multiplication de multiplication par exemple :
10**4 = 10000
50**2 = 2500

III - le jeu du c'est plus c'est moins

Pour ce jeu nous avons besoin de tirer un nombre pseudo-aléatoire et nous devons donc commencer ici par importer le module **aléatoire** d'où la première ligne du programme ci-dessous.
From random import randrange.

La fonction randrange(100) génère un nombre pseudo aléatoire entre 0 et 99. Il nous faut donc lui ajouter 1 pour nous placer dans l'intervalle [1; 100]

```
# -*- coding:Latin-1 -*-
from random import randrange

# un nombre pseudo aléatoire entre 1 et 100
choix = randrange(100)+1
print 'J\'ai choisi un nombre entre 1 et 100'

# dans la proposition on met -1 pour être
# différent de la variable choix
proposition = -1

# on demande un nombre tant que
# proposition est différent de choix
while (proposition != choix):
    print 'ta proposition'
    proposition = input()
    if proposition>choix:
        print 'C\'est moins'
    elif proposition<choix:
        print 'C\'est plus'
    else:
        print 'BRAVO'
```

```
>>>
J'ai choisi un nombre entre 1 et 100
ta proposition
50
C'est moins
ta proposition
25
C'est moins
ta proposition
13
C'est moins
ta proposition
7
C'est moins
ta proposition
5
C'est plus
ta proposition
6
BRAVO
>>>
```

Exercice 3 : mettre un petit compteur dans le programme qui indique le nombre de coups

Exercice 4 : en plus du compteur faire intervenir deux bornes ***min*** *et* ***max*** *pour proposer à l'utilisateur l'intervalle dans lequel se trouve le nombre cherché pour afficher des réponses claires du genre* ***proposition 3 le nombre est entre 60 et 100***

Note : il est possible que, pour un affichage propre, vous ayez besoin de la fonction qui transforme une *variable numérique* en *variable chaine de caractères* qui est la fonction ***str*** !
Pour concaténer par exemple la chaine « *proposition numéro* » et le nombre *n* il faut écrire
print 'proposition numéro '+str(n)

IV – Les nombres premiers

En utilisant la fonction ***append*** des *listes* nous allons créer la *liste des diviseurs* d'un nombre entier **n**.
Il suffit pour cela, tout simplement, de diviser **n** par chacun des nombres qui lui sont inférieurs. On vérifiera la divisibilité en utilisant l'opérateur ***a%b*** qui donne *le reste* dans la *division entière* de **a** par **b**,
Si ce reste est **zéro** c'est que **a est divisible par b**.
En fin de programme on affiche la liste des diviseurs de **n**

```
from math import *

# liste des disiseurs d'un naturel n
print 'Ton nombre n '
n = input()

div = []

max = n

while max>0:
    if (n % max == 0):
        div.append(max)
    max = max - 1


# on affiche la liste
print 'la liste des diviseurs de '+str(n)+' est '
print div
```

```
IDLE 2.6.2
>>> ==============================
>>>
Ton nombre n
18
la liste des diviseurs de 18 est
[18, 9, 6, 3, 2, 1]
>>>
```

Exercice 5 : avec très peu d'instructions supplémentaires afficher, en plus de la liste des diviseurs, la primarité du nombre n.
Dans le cas où n est premier on affiche '*Nombre premier*'.

Exercice 6 : Ajouter une *liste des premiers nombres premiers* que l'on appellera *premiers* et qui commencera donc ainsi
premiers = [2,3,5,7....]
pour pouvoir afficher la liste des *diviseurs premiers* du nombre n

Exercice 7 : essayer de traiter l'appartenance à la liste des nombres premiers *par une fonction* de ce genre par exemple :
def est_premier(qui):
qui retournera 1 si *qui* est premier et zéro sinon.
Comme dans l'exercice précédent on affichera la liste des diviseurs premiers de n

V – Chaînes de caractères, le cadavre exquis

Ce jeu littéraire a été inventé à Paris, au 54 rue du Château, dans une maison où vivaient Marcel Duhamel, Jacques Prévert et Yves Tanguy, Le principe de ce jeu était que chacun des participants écrivent à tour de rôle une partie d'une phrase, dans l'ordre sujet-verbe-complément, sans savoir ce que le précédent a écrit. La première phrase qui résulta et qui donna le nom à ce jeu fut « Le cadavre - exquis - boira - le vin - nouveau. »

Nous pouvons facilement programmer ce genre de jeu où chacun des joueurs tape, sans connaissance de la phrase précédente, une phrase décomposée en sujet – verbe – complément

Nous utilisons ici la structure **for ... in ...** qui permet d'exécuter une série d'instructions un certain nombre de fois.

Dans le cas présent on a écrit **for i in range(combien):**
range(combien) génère une liste [1,2,3,,, combien] qui va être parcourue par la variable i

```
# -*- coding:Latin-1 -*-
from random import randrange

# on initialise l'élément zéro des listes
sujet=[]
verbe=[]
complement=[]

# on demande le nombre de joueurs donc le nombre de phrases
combien = input('Combien de joueurs ? ')

#on lance la boucle de saisie des (combien) phrases

for i in range(combien):
    print 'joueur '+str(i+1)
    sujet.append(raw_input('ton sujet '))
    verbe.append(raw_input('ton verbe '))
    complement.append(raw_input('ton complément '))
    # pour décaler de 20 lignes et cacher la phrase précédente
    for n in range(20):
        print '\n'

# reste à sortir des phrases au hasard

reponse = 'o'
while (reponse<>'n'):
    print 'veux-tu une phrase (o/n) ?'
    reponse = raw_input()
    if (reponse != 'n'):
        #on tire un nombre pseudo aléatoire entre 1 et combien
        s = randrange(combien)
        v = randrange(combien)
        c = randrange(combien)
        phrase = sujet[s]+' '+verbe[v]+' '+complement[c]
        print phrase
    else:
        print 'Au revoir...'
```

*Exercice 8 : plutôt que de partir sur une liste vide pour le sujet par exemple, imposer une liste de sujets prédéfinie comme **sujet = ['les lapins', 'ma mère', 'ta moto', ...]** dans laquelle sera tiré au hasard un des sujets. Ce dernier sera donc proposé au joueur qui n'aura plus qu'à compléter le verbe et le complément.*

VI – Graphisme, la tortue, une étoile

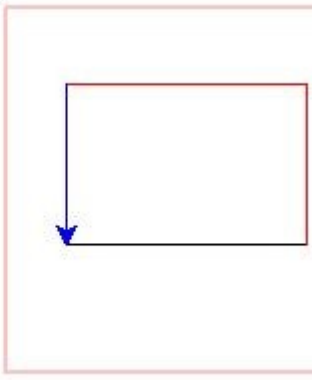
Souvenons nous de la tortue logo pilotée par les élèves de primaire...

Pour nous amuser un peu avec d'autres objets que des nombres, nous allons explorer un module *Python* qui permet de réaliser des « *graphiques tortue* », c'est-à-dire des dessins géométriques correspondant à la piste laissée derrière elle par une petite « *tortue* » virtuelle, dont nous contrôlons les déplacements sur l'écran de l'ordinateur à l'aide d'instructions simples.

Activer cette tortue est un vrai jeu d'enfant. Plutôt que de vous donner de longues explications, nous vous invitons à essayer de suite :

*N'oubliez pas le **from turtle import *** qui charge le module turtle !*

```
from turtle import *

forward(120)
left(90)
color('red')
forward(80)
left(90)
forward(120)
color('blue')
left(90)
forward(80)
```



Commandes du module turtle

reset() On efface tout et on recommence

goto(x, y) Aller à l'endroit de coordonnées x, y

forward(distance) Avancer d'une distance donnée

backward(distance) Reculer

up() Relever le crayon (pour pouvoir avancer sans dessiner)

down() Abaisser le crayon(pour recommencer à dessiner)

color(couleur) <couleur> peut être une chaîne prédéfinie ('red', 'blue', 'green', etc.)

left(angle) Tourner à gauche d'un angle donné (exprimé en degrés)

right(angle) Tourner à droite

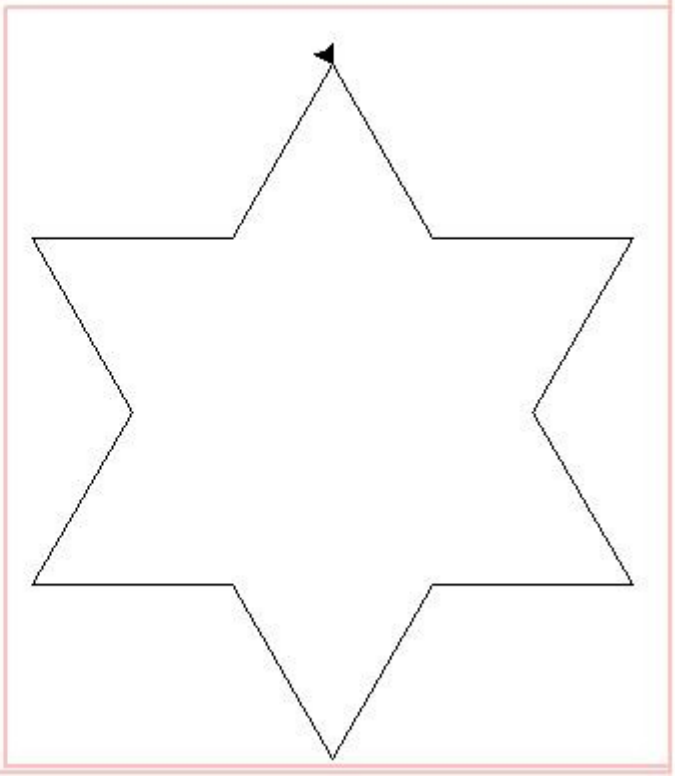
width(épaisseur) Choisir l'épaisseur du tracé

fill(1) Remplir un contour fermé à l'aide de la couleur sélectionnée

write(texte) <texte> doit être une chaîne de caractères délimitée avec des " ou des '

On peut facilement réaliser une étoile ainsi :

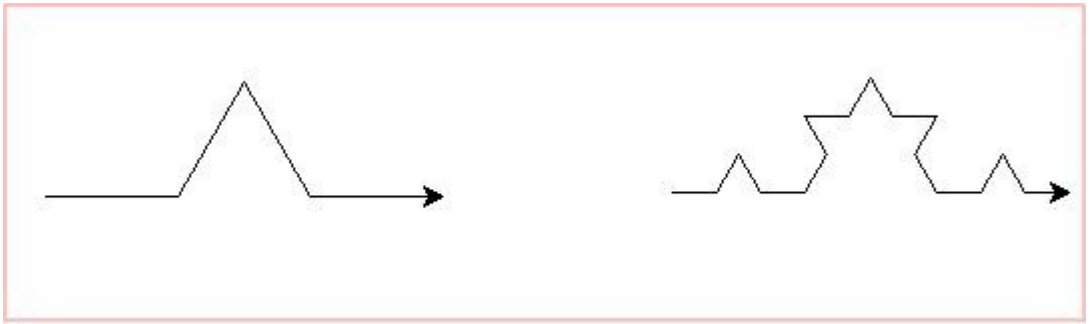
```
from turtle import *

# une page vierge
reset()

# on initialise
a = 0

# et on trace en boucle
while a <7:
    a = a +1
    forward(100)
    left(60)
    forward(100)
    right(120)
```



Exercice 9 : de la même façon réaliser une étoile à cinq branches

VII - Récursivité et fractales, le flocon de van koch

Une procédure récursive est une procédure qui peut s'appeler elle même.



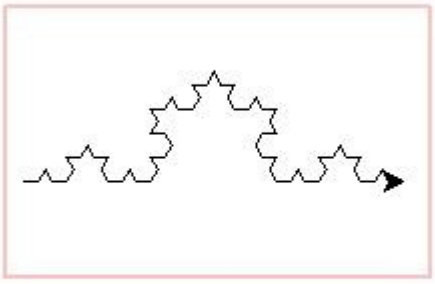
On base le flocon sur la figure de gauche qui va se reproduire elle même sur chacune des branches etc. Comme sur la boîte de fromage de la vache qui rit ! Cette figure de base est constituée de quatre branche de longueur a

On peut dire que la figure de droite est constituée ainsi

- *Un flocon de base a/3*
- Une rotation à gauche de 60°
- *Un flocon de base a/3*
- Une rotation à droite de 120°
- *Un flocon de base a/3*
- Une rotation à gauche de 60°
- *Un flocon de base a/3*

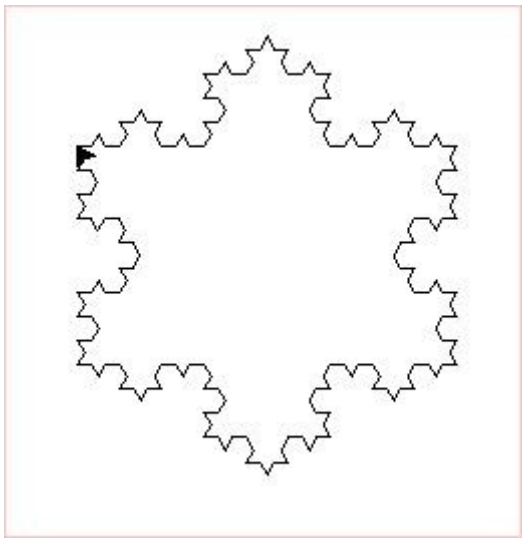
Et on appelle ainsi les flocons les uns dans les autres de façon récursive jusqu'à ce que a/3 ait atteint une limite que l'on s'est fixée (*de **10** comme ci-dessous par exemple*),

```
from turtle import *

#la procédure de travè du flocon
def flocon(a) :
    if (a>10) :
        flocon(a/3)
        left(60)
        flocon(a/3)
        right(120)
        flocon(a/3)
        left(60)
        flocon(a/3)
    else :
        forward(a)


flocon(200)
```



en fait, la procédure flocon empile les instructions les unes sur les autres sans rien faire d'autre et ne commence à tracer que lorsque le paramètre a devient <10 !

Faites des essais en faisant varier la condition (**a>10**)



*Exercice 10 : en imaginant que le flocon complet ci dessous n'est autre que trois **fractales de van koch** comme celle que l'on vient de réaliser au dessus faites sur un triangle équilatéral essayer d'écrire la procédure qui donne ce flocon. Ce qui signifie simplement que chaque côté du triangle est une fractale.*

Note amusante à propos de ce flocon : il a une aire bornée et un périmètre non borné !

VIII – Solution des exercices

Solution de l'exercice 1

```
print 'Division de a par b'
print 'Votre nombre a'
a = input()

b=0
while (b == 0):
    print 'Votre nombre b (non bul)'
    b = input()

q = a/b
r = a % b

print 'Résultat de la division de a par b'
print 'le quotient'
print q
print 'le reste'
print r
```

solution de l'exercice 2

```
# -*- coding: cp1252 -*-
# on défini une fonction stat_moyenne qui retourne
# la moyenne de la liste echantillon

def stat_moyenne( echantillon ) :
    taille = len( echantillon )
    moyenne = float(sum( echantillon) ) / taille
    return moyenne

# on défini de même la fonction de la variance

def stat_variance( echantillon ) :
    n = len( echantillon )                # nombre de valeurs
    mq = stat_moyenne( echantillon )**2  # moyenne au carré
    s = sum( [ x**2/n for x in echantillon ] )
    variance = s - mq
    return variance

# on récupère une liste de nombre dans echant
print 'vos notes séparées par des virgules SVP'
echant = input()

moy = stat_moyenne(echant)
vari = stat_variance(echant)

# on écrit cette moyenne et cette variance
print 'la moyenne de vos notes est '
print float(moy)
print 'la variance est '
print vari
```

```
vos notes séparées par des virgules SVP
10,7,11,12,14,8,9
la moyenne de vos notes est
10.1428571429
la variance est
3.12244897959
>>>
```

Solution de l'exercice 3

```
# -*- coding:Latin-1 -*-
from random import randrange

# un nombre pseudo aléatoire entre 1 et 100
choix = randrange(100)+1
print 'J\'ai choisi un nombre entre 1 et 100'

# dans la proposition on met -1 pour être
# différent de la variable choix
proposition = -1

# on met une variable qui va servir de compteur
compt = 0

# on demande un nombre tant que
# proposition est différent de choix
while (proposition != choix):
    compt = compt + 1
    print 'ta proposition numéro '+str(compt)
    proposition = input()
    if proposition>choix:
        print 'C\'est moins'
    elif proposition<choix:
        print 'C\'est plus'
    else:
        print 'BRAVO'
```

```
>>>
J'ai choisi un nombre entre 1 et 100
ta proposition numéro 1
50
C'est plus
ta proposition numéro 2
80
C'est plus
ta proposition numéro 3
90
C'est moins
ta proposition numéro 4
85
C'est plus
ta proposition numéro 5
87
C'est moins
ta proposition numéro 6
86
BRAVO
>>>
```

Solution de l'exercice 4

```
# -*- coding:Latin-1 -*-
from random import randrange

# deux bornes min et max
max, min = 100, 1

# un nombre pseudo aléatoire entre min et max
choix = randrange(max) + min
print 'J\'ai choisi un nombre entre '+str(min)+' et '+str(max)

# dans la proposition on met -1 pour être
# différent de la variable choix
proposition = -1

# on met une variable qui va servir de compteur
compt = 0

# on demande un nombre tant que
# proposition est différent de choix
while (proposition != choix):
    compt = compt + 1
    print 'Proposition ('+str(compt)+') entre '+str(min)+' et '+str(max)
    proposition = input()
    if proposition>choix:
        print 'C\'est moins'
        max = proposition
    elif proposition<choix:
        print 'C\'est plus'
        min = proposition
    else:
        print 'BRAVO'
```

```
J'ai choisi un nombre entre 1 et 100
Proposition (1) entre 1 et 100
50
C'est plus
Proposition (2) entre 50 et 100
75
C'est plus
Proposition (3) entre 75 et 100
87
C'est moins
Proposition (4) entre 75 et 87
79
C'est plus
Proposition (5) entre 79 et 87
84
C'est moins
Proposition (6) entre 79 et 84
82
BRAVO
```

Solution de l'exercice 5

```
from math import *

# liste des disiseurs d'un naturel n
print 'Ton nombre n '
n = input()

div = []

max = n

while max>0:
    if (n % max == 0):
        div.append(max)
    max = max - 1


# on affiche la liste
print 'la liste des diviseurs de '+str(n)+' est '
print div

if len(div)==2 :
    print 'Nombre premier'
```

```
>>>
Ton nombre n
37
la liste des diviseurs de 37 est
[37, 1]
Nombre premier
>>>
```

Solution de l'exercice 6

```
from math import *

# liste des disiseurs premiers d'un naturel n
print 'Ton nombre n '
n = input()

#liste des premiers nombres premiers
premiers = [2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,
            59,61,67,71,73,79,83,89,97]

max = n

div = []

while max>0:
    if (n % max == 0):
        for i in range(len(premiers)):
            if (max == premiers[i]):
                div.append(max)
    max = max - 1


# on affiche la liste
print 'la liste des diviseurs premiers de '+str(n)+' est '
print div
```

```
>>>
Ton nombre n
420
la liste des diviseurs premiers de 420 est
[7, 5, 3, 2]
>>>
```

Solution de l'exercice 7

```
from math import *

# liste des disiseurs premiers d'un naturel n
print 'Ton nombre n '
n = input()

#liste des premiers nombres premiers
premiers = [2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,
            59,61,67,71,73,79,83,89,97]

def est_premier(qui):
    sortie = 0
    for i in range(len(premiers)):
        if (qui == premiers[i]):
            sortie = 1
    return sortie


max = n

div = []

while max>0:
    if (n % max == 0):
        if est_premier(max)==1:
            div.append(max)
    max = max - 1


# on affiche la liste
print 'la liste des diviseurs premiers de '+str(n)+' est '
print div
```

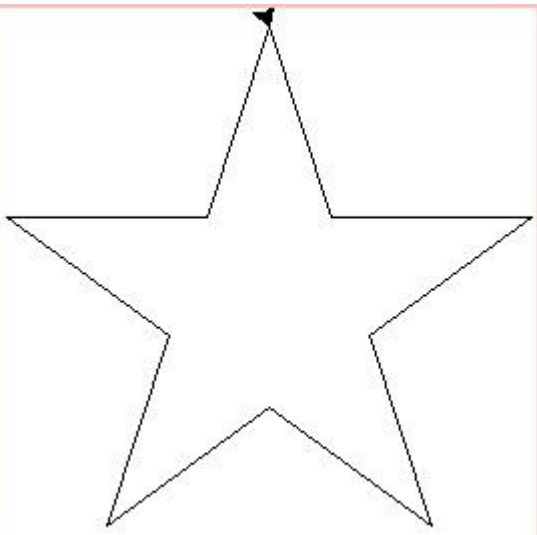
```
>>>
Ton nombre n
83*79*7
la liste des diviseurs premiers de 45899 est
[83, 79, 7]
>>>
```

Solution de l'exercice 8

```
# -*- coding:Latin-1 -*-
from random import randrange
import os

# on initialise l'élément zéro des listes
sujet=['les lapins', 'un arbre','mon chien', 'les oiseaux', 'une baguette', 'la toison', 'ma moto',
       'une montre', 'les monts','une chaîne', 'un brin', 'les roses', 'le diner']
verbe=[]
complement=[]

# on demande le nombre de joueurs donc le nombre de phrases
combien = input('Combien de joueurs ? ')

#on lance la boucle de saisie des phrases

for i in range(combien):
    print 'joueur '+str(i+1)
    print 'le sujet est : < ' +sujet[randrange(len(sujet)+1)]+' >'
    verbe.append(raw_input('ton verbe '))
    complement.append(raw_input('ton complément '))
    # la boucle ci-après permet de ne pas voir la saisie
    # du joueur  précédent
    for n in range(20):
        print '\n'

# reste à sortir des phrases au hasard

reponse = 'o'
while (reponse<>'n'):
    print 'veux-tu une phrase (o/n) ?'
    reponse = raw_input()
    if (reponse != 'n'):
        #on tire un nombre pseudo aléatoire entre 1 et combien
        s = randrange(combien)
        v = randrange(combien)
        c = randrange(combien)
        phrase = sujet[s]+' '+verbe[v]+' '+complement[c]
        print phrase
    else:
        print 'Au revoir...'
```

solution de l'exercice 9

```
# une page vierge
reset()

# on initialise
a = 0

# et on trace en boucle
while a <6:
    a = a +1
    forward(100)
    left(72)
    forward(100)
    right(144)
```



solution de l'exercice 10

```
from turtle import *

#la procédure de travè du flocon
def flocon(a) :
    if (a>10) :
        flocon(a/3)
        left(60)
        flocon(a/3)
        right(120)
        flocon(a/3)
        left(60)
        flocon(a/3)
    else :
        forward(a)

#un triangle équilatère dont les
#trois côtés sont des flocons
flocon(200)
right(120)
flocon(200)
right(120)
flocon(200)
```