ENIGMES à résoudre en PYTHON

(Proposées en complément dans le module MA20 par JCY aux fpa)

Cette première énigme sera résolue avec la classe, de manière à poser les bases :

- de l’utilisation de **tkinter** (fenêtre, boutons, Label, Entry, texte (avec scrollbar))

- du **comptage** et du **chronométrage** (remarque : ne pas chronométrer d’affichage)

- de la différence entre la **force brute** (et bête) et un algorithme plus malin (**smart**)

- de **l’affichage des résultats** dans un grand texte (avec scrollbar)

1. **Année carrée et harshad (dimension n)**

L’année 2025 est une année spéciale. Prenons seulement 2 propriétés spéciales :

* 1. C’est un carré (452)
  2. C’est un nombre « harshad », c’est-à-dire qu’ il est divisible par la somme de ses chiffres : 2025 / (2+0+2+5) = 2025 / 9 = 225.

Avant de réaliser un programme qui cherche les nombres qui ont les mêmes propriétés, sauvez-vous trouver le carré harshad juste avant 1000 ?

On se propose de réaliser un programme capable de trouver tous les carrés harshad, par exemple de 1000 à 1'000'000.

Voici le résultat de 2 algorithmes différents, l’un étant 1000 fois plus rapide que l’autre !

Voici le pseudo code du premier :

Pour les années (=a) comprises entre min et max :

Calculer la somme des chiffres (=s) de a

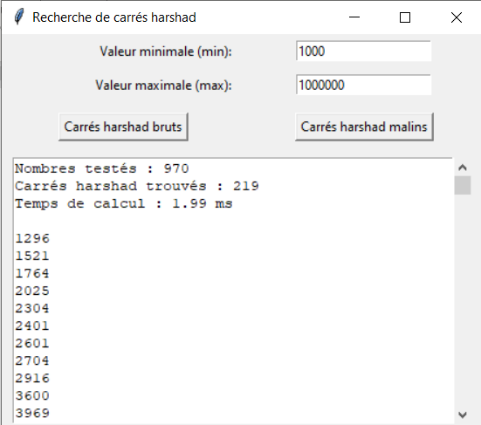
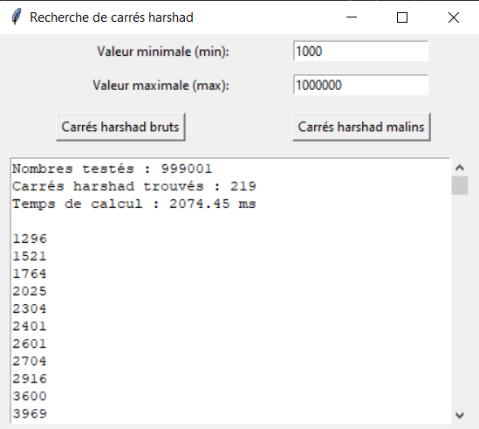
Si a est un carré et qu’il est divisible par s :

Ajouter l’année dans les solutions

Afficher le nombre d’années testées, le nb de solutions, le temps de calcul  
Afficher toutes les solutions

Pour le second, on parcourt les entiers root\_a de racine(min) à racine(max) et on regarde si (root\_a)2 est harshad

Exemple de réalisation :



1. **Triangles rectangles entiers (dimension n2)**

On cherche des triangles entiers pour Pythagore (exemple 3-4-5, ou 5-12-13), jusqu’aux valeurs de 10000 x 10000, en vérifiant que a et b sont premiers entre eux (pour ne pas prendre les solutions multiples d’autres comme 6-8-10).

* 1. Attention à ne pas tester de solutions en double (3-4-5 et 4-3-5)
  2. Compter le nombre de tests faits
  3. Attention aux arrondis (racine (100002 + 12)= racine(100 000 001) =10 000.0000499). La condition suivante risque de générer des solutions fausses : if c==int(c):

Exemple de réalisation pour 2 algorithmes, dernière solution (9592, 9945, 13817) :

Pour tous les a de min à max :

Pour tous les b de a à max :

c2= a2+b2

si pgcd(a,b)==1 :

Si (int(racine(c2)))2 == c2 :

Ajouter (a,b,c) comme solution

Pour tous les c de min à max \* :

Mémoriser c2 dans square[] (liste des carrés)

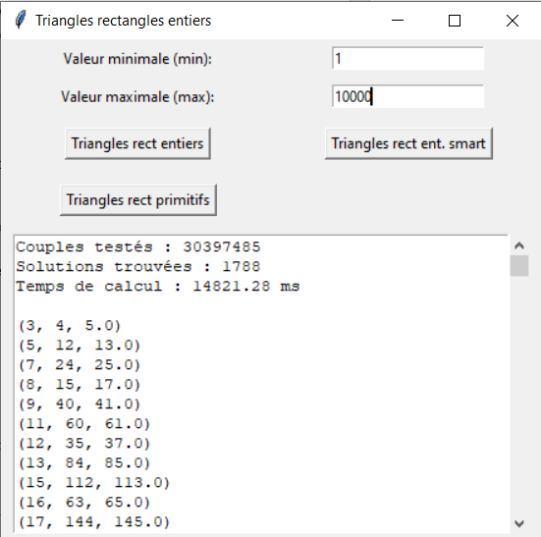
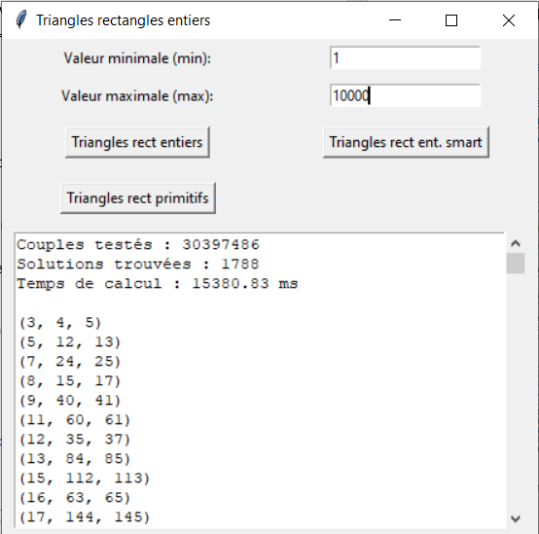
Pour tous les a de min à max :

Pour tous les b de a à max :

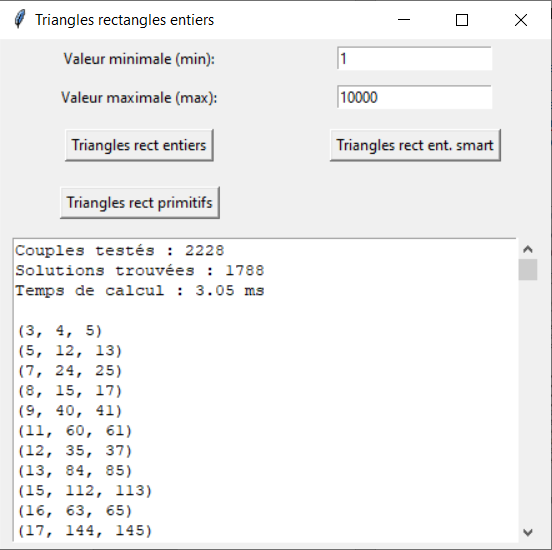
si pgcd(a,b)==1

Si a2+b2 est dans square[]:

Ajouter (a,b,c) comme solution



Mais il existe un algorithme qui permet de trouver 4000x plus rapidement. Comment est-ce possible ?

Algorithme  (chercher autour d’Euclide):

Pour m de 2 à racine (max\* ) :

Pour n de 1 à m-1 :

Si m,n premiers entre eux et de parité différente :

a=2\*m\*n

b= m2-n2

c= m2+n2

si a<=max et b<=max

ajouter solution (a,b,c) si a>b sinon (b,a,c)

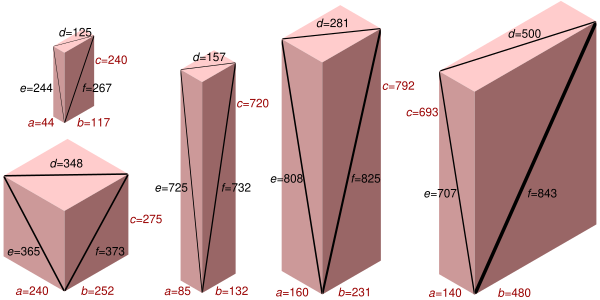
Trier les solutions dans l’ordre croissant de a

1. **Briques d’Euler (dimension n3)**

Une brique d’Euler est un parallélépipède dont les côtés ET les diagonales externes sont des entiers.

Comme il y a 3 dimensions, la force brute est rapidement dépassée (exemple avec 1000x1000x1000, on arrive à 78 secondes (plus de 2 minutes pour 168 millions de valeurs testées). Une solution smart reste sous la seconde avec environ 1 million de valeurs testées (en 2 étapes).

A vous de jouer. On ne demande pas forcément que les nombres soient premiers entre eux.



Algorithme simple :

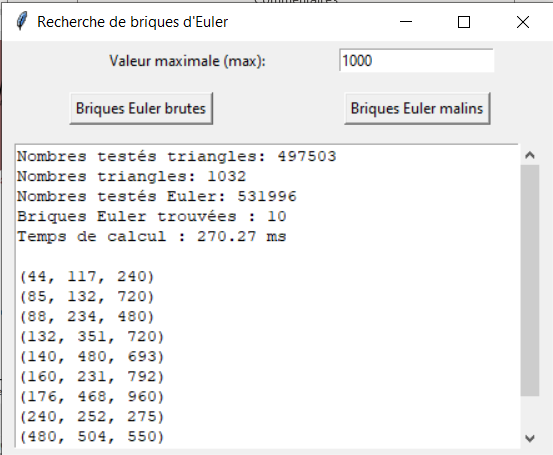
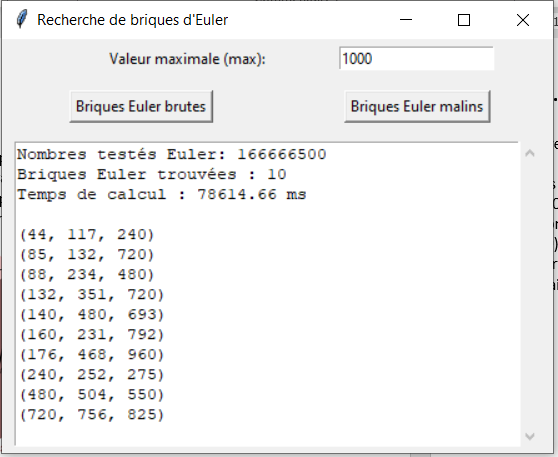
Pour tous les a de 1 à max

Pour tous les b de a à max

Pour tous les c de b à max :

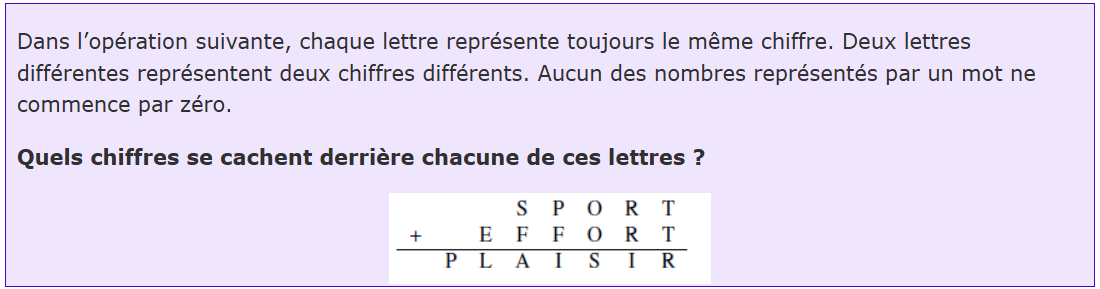
Si racine(a2+b2) est entier et racine(b2+c2) est entier et racine(c2+a2) est entier :

Ajouter (a,b,c) comme solution



1. **SPORT + EFFORT = PLAISIR (dimension 10n ou 10!)**

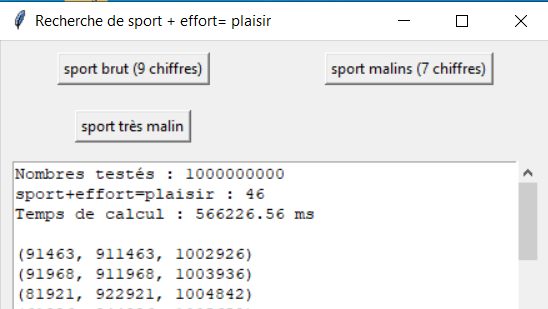
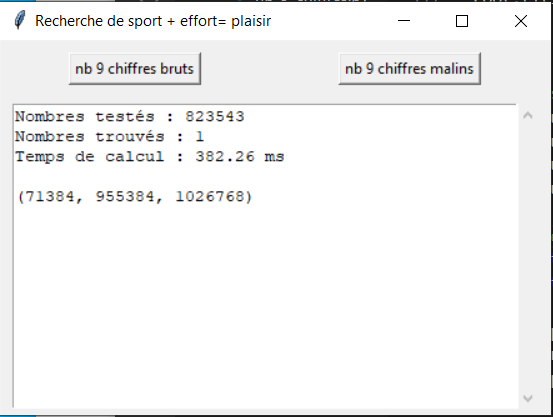
Trouvez la solution de l’énigme suivante (1 seule solution) avec le moins d’essais possibles.

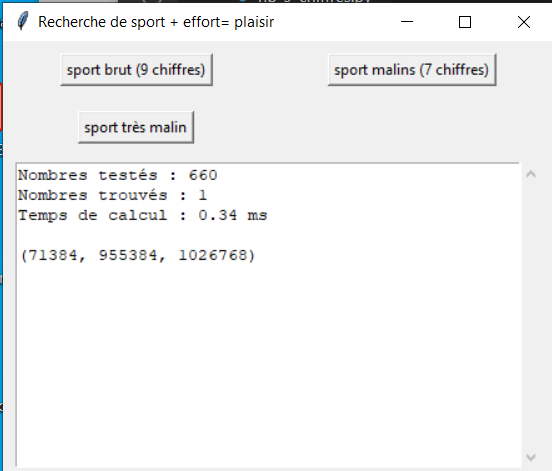


Si on ne prend qu’une seule certitude (P=1), et qu’on teste tous les autres chiffres (0-9 soit range(0,10)), alors le temps de calcul est énorme car on teste 109 solutions (1 milliard = 560’000ms>9 min) Mais on peut faire beaucoup plus simple…

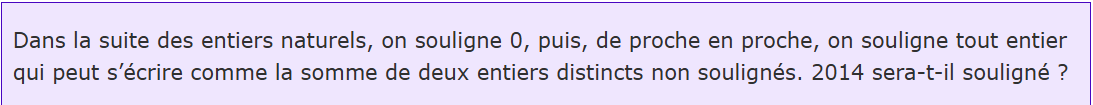
Rien qu’en prenant p,e,l (faciles à deviner) puis en essayant les autres de 2 à 8, on divise par 1500 le temps (ici 0.38s).

Mais on peut encore descendre sous la ms si on sait faire des additions et stopper les boucles improductives !



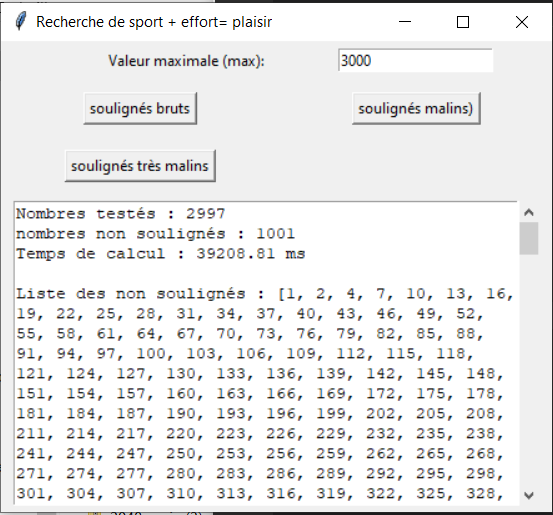
1. **Soulignés ou non ? (n3)**



Début des nombres soulignés :



Ecrire un algorithme (pseudo-code) qui permet d’écrire cette liste par la force brute. Avec la force brute, on augmente avec le cube de la limite, (1s pour 1000, 8s pour 2000, 27s pour 3000)



1. **Nombre à 9 chiffres**

**Un nombre à 9 chiffres**

  On cherche un nombre de neuf chiffres différents de 0 tel que :  
- le nombre formé par le premier chiffre soit divisible par 1  
- le nombre formé par les deux premiers chiffres soit divisible par 2  
- le nombre formé par les trois premiers chiffres soit divisible par 3  
- le nombre formé par les quatre premiers chiffres soit divisible par 4  
- le nombre formé par les cinq premiers chiffres soit divisible par 5  
- le nombre formé par les six premiers chiffres soit divisible par 6  
- le nombre formé par les sept premiers chiffres soit divisible par 7  
- le nombre formé par les huit premiers chiffres soit divisible par 8  
- le nombre formé par les neuf premiers chiffres soit divisible par 9  
  
(on appelle par premier chiffre le chiffre de gauche).  
  Il y en a plusieurs comme 183252321 par exemple. Cependant, il y en a un unique où les neuf chiffres sont distincts, lequel?

1. **Permutation circulaire des chiffres d'un nombre**

  Quel est le plus petit nombre entier tel que si on lui fait subir une permutation circulaire de un rang vers la gauche, il est multiplié par 1.5 ? (par exemple, 45312 --> 53124 lors d'une permutation circulaire de un rang vers la gauche).

1. Sachant que le train Yverdon-St-Croix met 37 minutes pour le trajet, et que l'horaire est affiché sur un affichage digital, trouvez le plus d'horaires possibles qui utiliseraient au maximum 25 éléments lumineux.

(réponse: 16 .solutions)

Exemple: l'horaire ci-dessous utilise:

08H02: 6 + 7 + 6 + 5 = 24 éléments

08h39: 6 + 7 + 5 + 6 ) 24 éléments

soit 48 éléments allumés (beaucoup trop)

Départ

Arrivée

:

:

Représentation des chiffres: