Invasion des uns !! (Sujet n° 6)

Énoncé :

On observe que :

$6^2 - 5^2 = 11$

$56^2 - 45^2 = 1111$

$556^2 - 445^2 = 111111$

$5556^2 - 4445^2 = 11111111$

Peut-on généraliser ?

Solution :

Il semble opportun d'utiliser un raisonnement par récurrence.

Notons P(n) la propriété : $\underbrace{5555\ldots556}_{\text{n fois}}^2 - \underbrace{4444\ldots45}_{\text{n fois}}^2 = \underbrace{111\ldots111}_{\text{2n+2 fois}}$

- P(0) s'écrit $6^2 - 5^2 = 11$ et est vérifiée.
- Posons comme hypothèse de récurrence (HR) : Il existe $p \in \mathbb{N}$ tel P(p) soit vraie.
- Il faut montrer que P(p +1) reste vraie c'est-à-dire que $\underbrace{5555\ldots556}_{\text{p+1 fois}}^2 - \underbrace{4444\ldots45}_{\text{p+1 fois}}^2 = \underbrace{111\ldots111}_{\text{2p+4 fois}}$

On a $D = \underbrace{5555\ldots556}_{\text{p+1 fois}}^2 - \underbrace{4444\ldots45}_{\text{p+1 fois}}^2 = (5\times10^{p+1} + \underbrace{5555\ldots556}_{\text{p fois}})^2 - (4\times10^{p+1} + \underbrace{4444\ldots45}_{\text{p fois}})^2$

$D = (5\times10^{p+1})^2 + 2\times5\times10^{p+1}\times\underbrace{5555\ldots556}_{\text{p fois}} + \underbrace{5555\ldots556}_{\text{p fois}}^2 - (4\times10^{p+1})^2 - 2\times4\times10^{p+1}\times\underbrace{4444\ldots45}_{\text{p fois}} - \underbrace{4444\ldots45}_{\text{p fois}}^2$

$D = 25\times10^{2p+2} - 16\times10^{2p+2} + 2\times10^{p+1}(5\times\underbrace{5555\ldots556}_{\text{p fois}} - 4\times\underbrace{4444\ldots45}_{\text{p fois}}) + \underbrace{5555\ldots556}_{\text{p fois}}^2 - \underbrace{4444\ldots45}_{\text{p fois}}^2$

Ce qui, d'après l'hypothèse de récurrence, donne :

$D = 9\times10^{2p+2} + 2\times10^{p+1}(5\times\underbrace{5555\ldots556}_{\text{p fois}} - 4\times\underbrace{4444\ldots45}_{\text{p fois}}) + \underbrace{111\ldots111}_{\text{2p+2 fois}}$

$D = 9\times10^{2p+2} + 2\times10^{p+1}(5\times(5\times\underbrace{111\ldots111}_{\text{p+1 fois}} + 1) - 4\times(4\times\underbrace{111\ldots111}_{\text{p+1 fois}} + 1)) + \underbrace{111\ldots111}_{\text{2p+2 fois}}$

$D = 9\times10^{2p+2} + 2\times10^{p+1}(25\times\underbrace{111\ldots111}_{\text{p+1 fois}} + 5 - 16\times\underbrace{111\ldots111}_{\text{p+1 fois}} - 4) + \underbrace{111\ldots111}_{\text{2p+2 fois}}$

$D = 9\times10^{2p+2} + 2\times10^{p+1}(9\times\underbrace{111\ldots111}_{\text{p+1 fois}} + 1) + \underbrace{111\ldots111}_{\text{2p+2 fois}}$

$D = 9\times10^{2p+2} + 2\times10^{p+1}(10^{p+1}) + \underbrace{111\ldots111}_{\text{2p+2 fois}}$

$D = 11\times10^{2p+2} + \underbrace{111\ldots111}_{\text{2p+2 fois}}$

$D = \underbrace{111\ldots111}_{\text{2p+4 fois}}$

L'hérédité est donc montrée et le principe de récurrence montre que la formule se généralise à tout $n \in \mathbb{N}$:

Pour tout $n \in \mathbb{N}$, on a $\underbrace{5555\ldots556}_{\text{n fois}}^2 - \underbrace{4444\ldots45}_{\text{n fois}}^2 = \underbrace{111\ldots111}_{\text{2n+2 fois}}$

Olivier Rochoir

Énoncé :
On observe que :
$6^2 - 5^2 = 11$
$56^2 - 45^2 = 1111$
$556^2 - 445^2 = 111111$
$5556^2 - 4445^2 = 11111111$
Peut-on tester avec un programme que ces égalités restent vérifiées en augmentant le nombre de chiffres ?

Solution :

Il n'est bien sûr pas possible sur des exemples de généraliser le résultat précédent (même si les exemples sont très nombreux).
On peut juste émettre une conjecture qu'il faudra vérifier mathématiquement (c'est fait en première page).
On va donc vérifier à l'aide d'un programme que lorsqu'on a n chiffres 5 (et 4) devant le chiffre 6 (et 5), on obtient 2n+2 chiffres 1 comme résultat.

On utilise le langage Python, ici, car c'est un langage qui (du fait de sa gestion des entiers) peut gérer sans erreur (d'arrondi) les calculs sur les entiers. On peut donc **faire des calculs aussi grand que l'on veut avec des entiers** (la seule limite étant la mémoire vive de l'ordinateur).

Un premier programme :

```
from time import time #utilisé uniquement pour calculer la durée d'exécution du programme

#on définit ci-dessous une fonction permettant de créer les nombres 555...56 et 444...45

def creernombre(n,a,b): #on appellera cette fonction avec a=5 et b=6
    nombre=b #le nombre est initialisé à 6
    for i in range(1,n+1): #une boucle pour ajouter au nombre précédent 50 puis 500 puis 5000....
        nombre=nombre+a*10**i #en python le symbole puissance est **
    return nombre #on renvoie le nombre créé


n=int(input("Entrer la valeur de n : ")) #n sera le nombre de 5 dans le nombre 555...56

t1=time() #démarre le chrono

resultat=creernombre(n,5,6)**2-creernombre(n,4,5)**2 #le calcul !!

print("Nombre de 1 :",len(str(resultat))) #affiche le nombre de 1 du résultat
print("Nombre de chiffres :",str(resultat).count('1')) #affiche le nombre de chiffres du résultat
# Si les deux nombres précédents sont égaux cela signifie qu'il n'y a que des 1 au résultat

#print(resultat)    #Enlever le # devant print pour afficher le résultat
t2=time() #arrete le chrono
print("Effectué en ",t2-t1," secondes")
```

Il est possible de tester ce programme pour différentes valeurs de n et ainsi de vérifier le résultat ci-dessus.
En l'exécutant, on obtient :

```
Entrer la valeur de n : 100
Nombre de 1 : 202
Nombre de chiffres : 202
Effectué en  0.0  secondes
```

Le programme suivant est le même :
Seule la fonction permettant de construire les deux nombres 555…556 et 444…445 diffère.

En effet la fonction « creernombre(n,a,b): » utilise à chaque passage dans la boucle **1 addition et i + 1 multiplications.**
Par exemple, 55556 est obtenu par 6 + 5×10 + 5×10×10 + 5×10×10×10 + 5×10×10×10×10.
Il est possible d'utiliser une autre construction : 55556 = (((((5)×10+5)×10+5)×10+5)×10+5)+1.
Cela veut dire qu'à chaque passage dans la boucle, on effectue **1 addition et <u>1 seule</u> multiplication**.

Cela donne :

```
from time import time #utilisé uniquement pour calculer la durée d'exécution du programme

#on définit ci-dessous une fonction permettant de créer les nombres 555...56 et 444...45

def creernombrebis(n,a,b): #on appellera cette fonction avec a=5 et b=6
    nombre=0 #le nombre est initialisé à 0
    for i in range(1,n+1): #une boucle pour multiplier  par 10 et ajouter 5 à chaque passage...
        nombre=nombre*10+a
    nombre=nombre*10+b #ce calcul est différent pour le dernier chiffre
    return nombre #on renvoie le nombre créé

n=int(input("Entrer la valeur de n : ")) #n sera le nombre de 5 dans le nombre 555...56

t1=time() #démarre le chrono

resultat=creernombrebis(n,5,6)**2-creernombrebis(n,4,5)**2 #le calcul !!

print("Nombre de 1 :",len(str(resultat))) #affiche le nombre de 1 du résultat
print("Nombre de chiffres :",str(resultat).count('1')) #affiche le nombre de chiffres du résultat
# Si les deux nombres précédents sont égaux cela signifie qu'il n'y a que des 1 au résultat

#print(resultat)    #Enlever le # devant print pour afficher le résultat
t2=time() #arrete le chrono
print("Effectué en ",t2-t1," secondes")
```

En l'exécutant, on obtient :

```
Entrer la valeur de n : 100
Nombre de 1 : 202
Nombre de chiffres : 202
Effectué en  0.0  secondes
```

Une comparaison des deux programmes sur la même machine avec différentes valeurs de n :

n =	Nombre de 1	Temps programme 1	Temps programme 2
100	202	0,0 secondes	0,0 secondes
1 000	2 002	0,03 secondes	0,0 secondes
10 000	20 002	6,54 secondes	0,29 secondes
100 000	200 002	**2547,73** secondes	26,9 secondes
1 000 000	2 000 002		2683,5 secondes

La différence est significative dès que n devient grand !!

Une dernière remarque : cette deuxième méthode est utilisée, par exemple, dans le calcul de l'image d'un nombre par un polynôme. Cela évite le calcul de x^n , puis de x^{n-1}, … et économise du temps d'exécution….

Olivier Rochoir