Explications Algorithme Palindromes Premiers de Lepzulnag

Idée de l'Algorithme

Le principe est de générer tous les palindromes, puis de vérifier s'ils sont premiers ou non.

Génération des Nombres Palindromes

Il est possible de générer les nombres palindromes à partir des nombres entiers. En effet, pour X chiffre quelconque :

1 génère 11 et $\overline{1X1}$,
2 génère 22 et 2X2,
…
9 génère 99 et $\overline{9X9}$,
10 génère 1001 et $\overline{10X01}$,
11 génère 1111 et $\overline{11X11}$,
…

On remarque qu'en obtient les nombres palindromes dans un ordre croissant. Chaque entier génère deux nombres palindromes, l'un pair et l'autre impair.

Test de Primalité par 11

Un test de primalité par 11 permet d'éliminer directement tous les nombres palindromes pairs : un nombre est divisible par 11 si et seulement si la somme des chiffres en position pairs moins la somme des chiffres en position impairs est divisibles par 11. Pour un nombre palindrome possédant un nombre pair de chiffres, la différences des deux sommes étant nulle, et donc le palindrome est non premier.

Cela nous permet de faciliter l'algorithme de génération des palindromes :

1 génère $\overline{1X1}$,
2 génère 2X2,
…
9 génère $\overline{9X9}$,
10 génère $\overline{10X01}$,
11 génère $\overline{11X11}$,
…

Reste maintenant à vérifier quels nombres sont premiers.

Tests de Primalité par 2, 3 et 5

La plupart des palindromes obtenus, quand ils ne sont pas premiers, sont divisibles par 2, 3 ou 5. Il est facile d'éliminer ces cas-là en possédant seulement le générateur du palindrome, ce qui nous permet d'économiser le calcul dudit palindrome.

Si le premier chiffre du générateur est pair, alors le dernier du palindrome l'est aussi, ainsi il est divisible par 2.
Si le premier chiffre du générateur est 5, alors le dernier du palindrome l'est aussi, ainsi il est divisible par 5.
Si la somme des chiffres du palindrome est un multiple de 3, il l'est. Or cette somme est égale à deux fois la somme du générateur plus X.

Description des Variables Utilisées

x : générateur
base : 1, 10, 100, etc...
c : chiffre de x
pm : *Palindrome Milieu*. Soit X dans l'explication ci-dessus.
i : x est le i-ème palindrome prermier
y : variable muette
c1 : premier chiffre de x (et donc p)
p : palindrome généré par x
s : somme des chiffres du palindrome
ps : *Palindrome Suffixe*. C'est x lu à l'envers.

Programme avec commentaires

```
Define primindrome(n)=
Func
If n<1: Return "Entrez un nombre supérieur à 0"
If n≤5: Return {2,3,5,7,11}[n]
n:=n−5

Local i,p,base,x,ps,pm,y,s,c,c1
i:=0
base:=1

Loop

  For x,base,10·base−1          Début boucle principale   x est le générateur
    c1:=ceiling((x+1)/base)−1   Calcul du premier chiffre du générateur
    If mod(c1,2)=0 or c1=5: Cycle   Critères de divisibilité par 2 et 5

    ps:=0
    y:=1
    s:=c1
    While y<base
      c:=mod(ceiling((x+1)/y)−1,10)
      ps:=10·ps+c
      s:=s+c
      y:=y·10
    EndWhile
    ps:=10·ps+c1                Calcul de ps, le suffixe du palindrome, c'est-à-dire x lu à l'envers.

    For pm,0,9                  pm est le chiffre au milieu du palindrome
      If mod(pm+2·s,3)=0: Cycle   Test de primalité par 3
      p:=100·base·x+10·base·pm+ps   Calcul du palindrome qui est égal à x|pm|ps en notation algébrique

      y:=7
      While y·y≤p
        If mod(p,y)=0: Exit
        If mod(p,y+4)=0: Exit
        y:=y+6
      EndWhile                  On vérifie que p soit premier. Alogrithme assez basique. y est un diviseur potentiel de p. En augmentant y de 6 on est certain que ni 2 ni 3 ne diviseront y.

      If y·y>p: i:=i+1          Si p est premier, on incrémente i. p est alors le i-ème palindrome premier. Si i est égal à n, alors on a trouvé et on renvoit p comme solution.
      If i=n: Goto end
    EndFor
  EndFor

  base:=10·base
EndLoop

Lbl end
Return p
EndFunc
```