UN ALGORITHME DE CALCUL.

Si c'est simplement la réponse à la question qui nous intéresse, il suffit d'utiliser un ordinateur, d'effectuer une boucle de 0 à 11 111 111 111 et de tester pour chaque nombre si oui ou non l'occurrence « 111 » apparaît… C'est réalisable avec les ordinateurs actuels (6 heures de calculs sur le miens qui a 2 ans) mais que se passerait-il si on voulait compter jusqu'à 111 111 111 111 111 111 111 par exemple ? Le temps pourrait commencer à nous manquer d'où l'intérêt de trouver un algorithme plus rapide et surtout qui le temps de calcul n'augmente pas à cette vitesse.

Voici l'idée : en réalité, nous avons besoin de 3 symboles : le 0, le 1 et * pouvant représenter n'importe quel chiffre entre 2 et 9. Il y a cette fois seulement $3^{11}+3^{10}+\ldots+1=\frac{3^{12}-1}{2}$. Soit 265 720 calculs qui ne prends au total que quelques secondes…

Voici le principe, pour compter les solutions : à chaque fois qu'un nombre générique est trouvé, par exemple 10 111 *** 010, cela représente $8^{\text{nombre d'étoiles}}$ nombres en réalité.

```
var nb:array[1..11] of char;
        i,j,nbt,p:integer;
        N:string;
begin
  for i:=1 to 11 do nb[i]:='0';
  nbt:=0;
  repeat
    N:='';
    For i:=1 to 11 do N:=nb[i]+N;
    if pos('111',N)>0 then begin
      p:=1;
      For j:=1 to 11 do
      if nb[j]='2' then p:=p*8;
      nbt:=nbt+p;
    end;
    inc(nb[1]);
    For i:=1 to 11 do if nb[i]='3' then begin
      nb[i]:='0';
      inc(nb[i+1]);
    end;
  until N='11111111111';
  Showmessage(inttostr(nbt));
End;
```

➥ On déclare les variables, n best un tableau de chiffres (on a remplacé le symbole * par 2)
➥ i,j sont des variables pour les boucles.
➥ p=8 puissance le nombre d'occurrence du 2.
➥ N la chaîne formé des symboles concaténés.
➥ nbt : le nombre de nombre trouvés.

On commence la boucle
➥ On crée la chaîne N.
➥ Si la chaîne N répond à la question,

➥ On calcul combien cela représente de nombres.

➥ On cherche le nombre suivant.

➥ Fin de la boucle
➥ On affiche le résultat.