

<b>Stage algorithmique 1</b>  <b>TI-Nspire</b>	<b>Treize</b>
--	---------------

**Le problème :** on se propose de simuler, sur TI-Nspire, le jeu suivant :

On lance un dé jusqu'à ce que le total des points obtenus lors des différents lancers soit supérieur ou égal à 13.

Combien faut-il de lancers de dé ?

Évidemment, il faut au moins trois lancers et au plus treize.

### 1. Description du programme

On utilise :

- Trois variables : une nommée  $s$  pour le total des points, initialisée à 0, une autre nommée  $n$  pour le nombre de lancers, initialisée également à 0 et une nommée  $a$  pour stocker provisoirement le résultat de chaque lancer ;
- Une structure répétitive : placer dans  $a$  un nombre aléatoire compris entre 1 et 6, afficher  $a$ , ajouter  $a$  à  $s$ , augmenter  $n$  de 1, recommencer tant que  $s$  est inférieur strictement à 13 ;
- Un affichage final.

### 2. Le programme

Algorithme	Programme
$a$ , $s$ et $n$ sont déclarées comme variables locales <sup>1</sup> Mettre 0 dans $s$ Mettre 0 dans $n$ Tant que $s$ est strictement inférieur à 13 Mettre dans $a$ un nombre aléatoire entre 1 et 6 Afficher $a$ Ajouter $a$ à $s$ Augmenter $n$ de 1 Fin du tant que Afficher $n$	Define <b>de13</b> ()= Prgm Local $a,s,n$ $s:=0$ $n:=0$ While $s<13$ $a:=randInt(1,6)$ Disp $a$ $s:=s+a$ $n:=n+1$ EndWhile Disp "N=", $n$ EndPrgm

### 3. Prolongement

Écrire un programme qui donne les résultats de 100 répétitions du programme précédent **de13**.

Pour cela, commencer par supprimer les lignes d'affichage du programme **de13** pour en accélérer l'exécution ou le réécrire sans les deux lignes contenant l'instruction Disp. Ce nouveau programme sera appelé **de13b**.

Voir, en annexe, le programme modifié.

<sup>1</sup> Les affectations de leurs valeurs ne sont valables que dans ce programme.

Algorithme	Programme
<p><math>ll</math>, <math>i</math> et <math>x</math> sont déclarées comme variables locales</p> <p>Préparer une liste <math>ll</math> de 13 nombres égaux à 0</p> <p>Mettre 0 dans <math>i</math></p> <p>Tant que <math>i</math> est strictement inférieur à 100</p> <p>    Effectuer le programme <b>de13</b> (sans affichage)</p> <p>    Ajouter 1 au <math>n^{\text{ième}}</math> élément de <math>ll</math></p> <p>    Augmenter <math>i</math> de 1</p> <p>Fin du tant que</p> <p>Afficher <math>ll</math></p>	<pre> Define <b>de100</b>()= Prgm Local <math>ll,i,x</math> <math>ll:=seq(0,x,1,13)</math> <math>i:=0</math> While <math>i&lt;100</math>   <b>de13b</b>()   <math>ll[n]:=ll[n]+1</math>   <math>i:=i+1</math> EndWhile Disp <math>ll</math> EndPrgm </pre>

Remarque :

- Une boucle **For  $i, 1, 100 \dots$  EndFor** remplace avantageusement **While ... EndWhile** puisqu'alors il n'est pas nécessaire d'initialiser ni d'incrémenter la variable  $i$ .

#### 4. Version sans sous-programme utilisant l'instruction For

Algorithme	Programme
<p><math>ll</math>, <math>i</math> et <math>x</math> sont déclarées comme variables locales</p> <p>Préparer une liste <math>ll</math> de 13 nombres égaux à 0</p> <p>Pour <math>i</math> de 1 à 100</p> <p>    Mettre 0 dans <math>s</math></p> <p>    Mettre 0 dans <math>n</math></p> <p>    Tant que <math>s</math> est strictement inférieur à 13</p> <p>        Mettre dans <math>a</math> un nombre aléatoire entre 1 et 6</p> <p>        Ajouter <math>a</math> à <math>s</math></p> <p>        Augmenter <math>n</math> de 1</p> <p>    Fin du tant que</p> <p>    Ajouter 1 au <math>n^{\text{ième}}</math> élément de <math>ll</math></p> <p>Fin du For</p> <p>Afficher <math>ll</math></p>	<pre> Define <b>de100b</b>()= Prgm Local <math>i,s,n,x</math> <math>ll:=seq(0,x,1,13)</math> For <math>i,1,100</math>   <math>s:=0</math>   <math>n:=0</math>   While <math>s&lt;13</math>     <math>a:=randInt(1,6)</math>     <math>s:=s+a</math>     <math>n:=n+1</math>   EndWhile   <math>ll[n]:=ll[n]+1</math> EndFor Disp <math>ll</math> EndPrgm </pre>

Remarque :

On obtient directement l'analyse statistique de l'échantillon en modifiant la fin du programme de la manière suivante :

Se replacer dans le programme ; à la suite de Disp  $ll$ , ajouter les instructions

$l2:=seq(x, x, 1, 13)$

One Var  $l2, ll$

Disp *stats.results*

Après affichage de la liste  $L_1$ , on obtiendra ainsi les paramètres statistiques (voir annexe).

## Annexe

Programme de_13 modifié	Programme avec paramètres statistiques
<pre> Define <b>de13b</b>()= Prgm Local a,s s:=0 n:=0 While s&lt;13 a:=randInt(1,6) s:=s+a n:=n+1 EndWhile EndPrgm </pre>	<pre> Define <b>de100c</b>()= Prgm Local a,i,s,n,x,l1,l2 l1:=seq(0,x,1,13) For i,1,100 s:=0 n:=0 While s&lt;13 a:=randInt(1,6) s:=s+a n:=n+1 EndWhile l1[n]:=l1[n]+1 EndFor Disp l1 l2:=seq(x,x,1,13) OneVar l2,l1 Disp <i>stat.results</i> EndPrgm </pre>
<p><i>On remarquera que la variable n est réutilisée dans le programme <b>de100</b>, ce qui impose de ne plus la déclarer comme variable locale dans le programme <b>de13b</b> « allégé ».</i></p>	