Mémo : les nombres aléatoires

Pour créer un nombre aléatoire, il faut utiliser deux outils disponibles dans la 9e pile en partant du haut dans l'application Cabri Express ou son extension Smart :

- L'outil "expression" 
- L'outil "évaluation" 

Méthodologie

1. Sélectionner l'outil "expression",
2. Écrire l'expression de son choix (cf. tableau),
3. Sélectionner l'outil "évaluation",
4. Cliquer sur l'expression dans la page, puis cliquer dans la page pour fixer le nombre et pouvoir ainsi le réutiliser.

La fonction rand()

Dès les premières créations systématiques d'activités par des enseignants, le besoin a été ressenti de pouvoir disposer de nombres "tirés au hasard". Pour cela, il est possible de recourir à une fonction. Dans Cabri, une telle fonction rand () est présente. Elle ne nécessite pas d'arguments, cependant la présence de parenthèses est obligatoire (dans la version actuelle). Elle produit des nombres uniformément distribués dans [0,1[.

Expression	→	Intervalle
rand()	→	[0,1[(nombre décimal)
10*rand()	→	[0,10[(nombre décimal)

La fonction floor()

La fonction floor permet d'obtenir des nombres aléatoires entiers.

Expression	→	Intervalle
floor(10*rand())	→	[0,9] (nombre entier)
floor(10*rand())+1	→	[1,10] (nombre entier)

La fraction permet de contrôler le nombre de décimales. Par exemple :

floor(100*rand())/100	→	[0,0,99] (2 décimales)

Généralisation : nombre aléatoire dans [i,j[avec d décimales.

$$\frac{1}{10^d}\left\lfloor (j-i)\times 10^d \text{rand}\,() \right\rfloor + i$$

La fonction choose()

Comment l'écrire ?

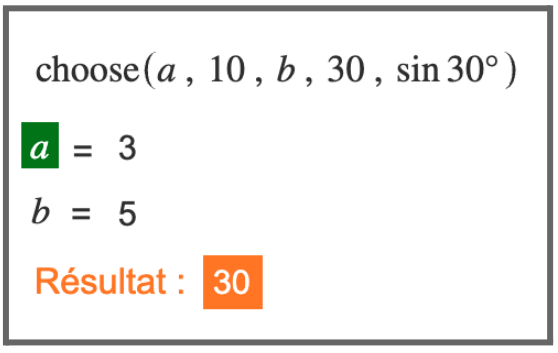
La fonction choose(...) peut s'écrire aussi choix(…) avec ou sans majuscules initiales.

À quoi ça sert ?

La fonction choose() permet de choisir un nombre dans une liste de valeurs numériques. Il s'agit ici du choix du **a-ième** élément dans l'ordre de la liste des arguments suivant le premier. Noter aussi qu'en tapant (choose(1, …) on obtient le 1er élément de la liste ; choose(2, …) donne le second, etc.

Exemple

Ci-dessous, nous avons évalué choose(a , 10, b, 30, sin 30°) pour a = **3** et b = **5**, ce qui donne le résultat **30**.



Dans le cas représenté, la valeur est bien **30**, car il s'agit du 3e élément à droite de l'argument a (le plus à gauche).

Voici comment cet exemple est obtenu : https://cabristeam.fr/videos/evaluation-choose.gif

Les arguments de la fonction choose(...)

La fonction choose(...) **dépend du nombre d'arguments** que nous lui mettons.

Argument	Effet	Exemple
0 argument	La fonction choose sans argument → on obtient une valeur aléatoire uniforme dans **[0,1[**. Autrement dit : « choose() est **synonyme de** rand() ».	
Ci-dessous, on considère qu'il y a au moins $p \geq 1$ argument. Lors d'une évaluation, le premier argument est systématiquement arrondi à l'entier le plus proche (round to even) ⇒ choose(1.5, ...) est calculé comme choose(2, …).		
1 argument	choose(n) **tire un nombre réel uniformément dans [0,n[**.	ex : choose(0) tire dans [0,0[et *a priori* ne tire rien. Par continuité, nous lui donnons la valeur 0. ex : floor(choose(10)) fournit un nombre de 1 chiffre au hasard dans [0, 9].
2 arguments	choose(n, m) **tire un nombre réel uniformément réparti dans [n,m[**. De façon générale, **choose(0,n)** → **choose(n)**.	ex : floor(choose(1,3)) tire 1 ou 2. choose(n,n) est indéfini (non-existant), mais par "continuité" on lui donne la valeur n, ainsi choose(0,0) = 0. choose(0, 3) vaut 0, 1 ou 2. Et donc équivalent à choose(3).
3 arguments et plus	choose(n, a_1, a_2,… , a_i) **choisit l'argument de rang n**. Il s'agit là de l'usage standard correspondant à au moins $p \geq 3$ paramètres comme par exemple dans choose(n, a, b). Si n vaut 0, le résultat est en fait l'un des a_i, tiré uniformément au hasard. Si n n'est pas dans [0, p], • s'il est négatif, on remplace n par 1 et • si $n > p$, on remplace n par p. Pour tirer « au hasard » un élément parmi ceux de rang ≥ 1, on pourra écrire : choose(choose(p), a_1, a_2, …, a_{p-1}) mais le	Exemple avec p = 5 : **choose(0, a_1, a_2, a_3, a_4)** distribuera uniformément les 4 valeurs a_1, a_2, a_3, a_4.

	même résultat s'obtient plus facilement en considérant choose(0, a_1, a_2, …, a_{p-1}).	

La fonction iRand()

La fonction iRand() est centrée sur la production de valeurs aléatoires entières.
Son résultat dépend du nombre de paramètres ainsi que de leurs valeurs.
Les arguments, positifs ou négatifs, sont par ailleurs arrondis à l'entier le plus proche.

Fonction	→	**Intervalle**
iRand()	→	**[0,1]** (nombre)
iRand(n)	→	**[0,n]** (entier)
iRand(n, m)	→	**[n,m]** (entier)