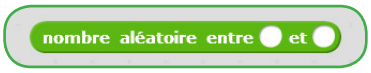
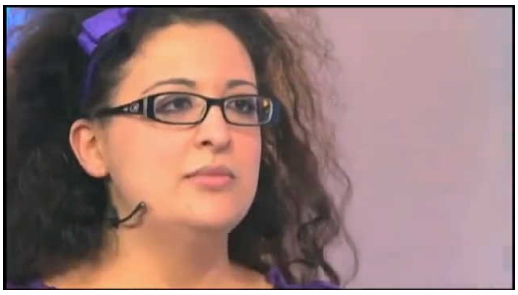
# Le hasard

Scratch offre la possibilité de choisir un nombre entier au hasard entre deux bornes.




Ce bloc a des applications très variées.

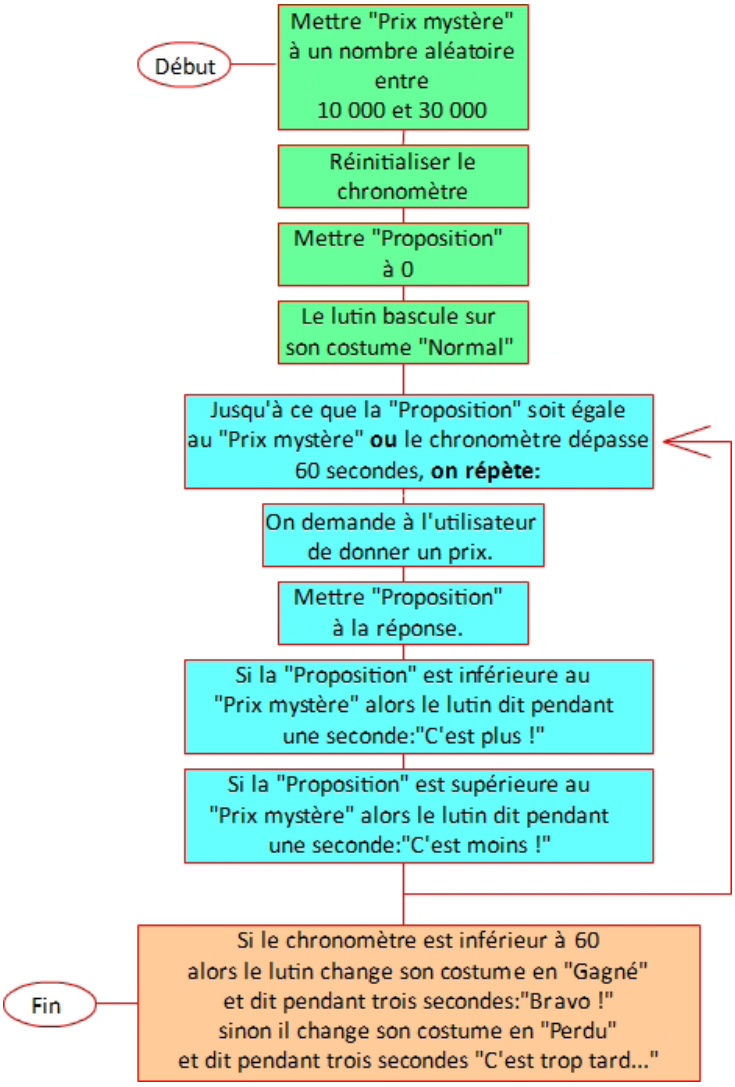
## Exercice 1: “Le juste prix”

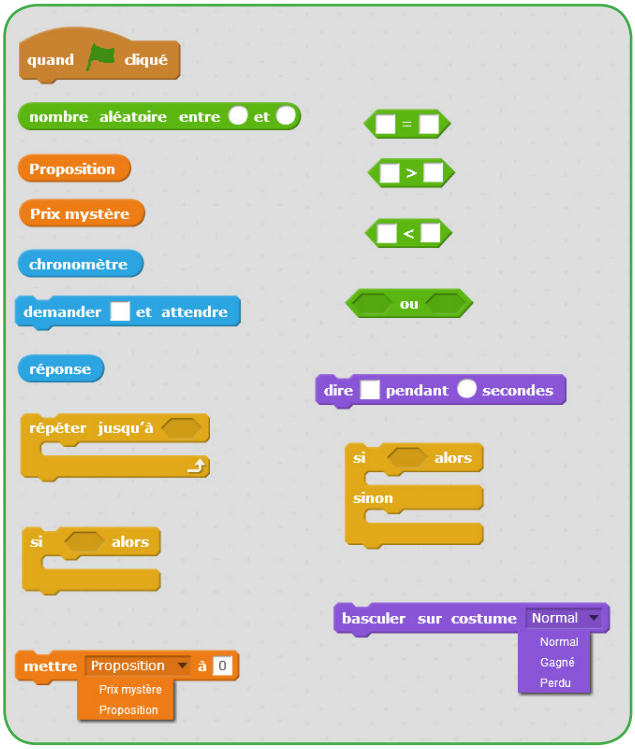


Nous allons simuler le jeu décrit dans la vidéo ci-contre:
Pour cela, nous aurons besoin de trois variables:

- **“Prix mystère”**: le prix à trouver. Il est choisi au hasard entre 10 000 et 30 000 €.
- **“Proposition”**: le prix proposé par le candidat
- **“Chronomètre”** : Cette variable existe déjà et se trouve dans le menu capteur. Il suffit de cocher la case devant son nom pour la faire apparaître dans la scène. On se donnera 60 secondes pour trouver le juste prix.

→ Voici l’algorithme donné sous la forme d’un **arbre**.




→ Ouvrez le programme “Hasard_1.sb2”.
→ Créez les variables nécessaires et utilisez les blocs ci-dessus (parfois en plusieurs exemplaires) pour coder cet algorithme.

## Exercice 2: Placer un lutin au hasard

→ Ouvrez le programme “Hasard_2.sb2”.

Ce programme contient trois lutins: deux maisons et un sapin.
L’objectif est ici de déplacer au hasard le sapin jusqu’à ce qu’il soit à la même distance des deux maisons.
Pour placer un lutin au hasard sur la scène, il suffit de lui choisir une abscisse aléatoire entre -240 et +240 et une ordonnée entre -180 et +180.

→ Utilisez les blocs ci-dessous pour que lorsque vous pressez la barre d’espace, le sapin se place aléatoirement dans la scène.


→ Utilisez une boucle pour qu’à l’appui de la barre d’espace, le sapin se déplace 1000 fois.

On aimerait à présent planter une forêt, c’est à dire que le sapin laisse son empreinte à chaque déplacement.

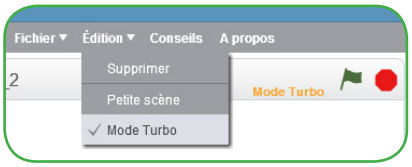
Pour cela, on va utiliser le bloc 

du menu “Stylo”. Ce bloc, tel un tampon, laisse une marque du lutin.

→ Utilisez au début du script un bloc “effacer tout” et après chaque déplacement du sapin le bloc “estampiller” pour créer la forêt de 1000 arbres.

**« Mais c’est trop long ! »**

Il existe dans Scratch un mode “Turbo” qui accélère fortement les boucles.
On le trouve dans le menu “Édition”




A présent, on aimerait ne planter que les arbres situés à <u>égales distances</u> des deux maisons. Pour cela, il va falloir mesurer la distance qui sépare l’arbre des autres lutins. Pour cela, vous allez utiliser le bloc ci-contre.


→ Utilisez un <u>test</u> comparant <u>l’arrondi de la distance</u> de l’arbre à la maison 1, à l’arrondi de la distance de l’arbre à la maison 2. En cas d’égalité, l’arbre laissera une trace. Pour obtenir plus d’arbres, modifier la boucle pour en tester 100 000.

**« Mais pourquoi comparer les arrondis et pas simplement les distances ?»**

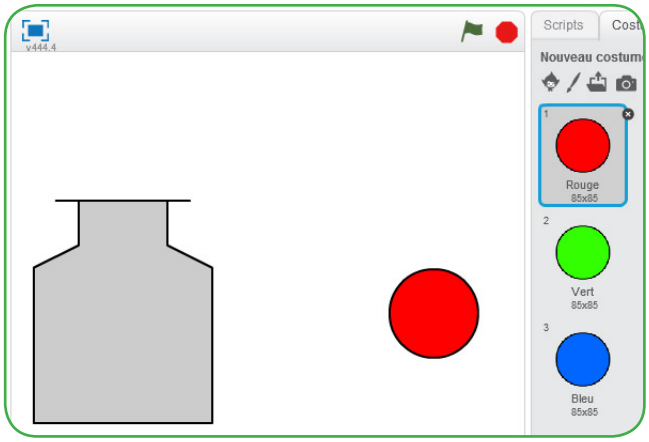
Pour calculer les distances, Scratch utilise le théorème de Pythagore.
Comme vous le savez, ce théorème donne rarement des valeurs exactes, et pour augmenter les chances d’obtenir un arbre “à peu près” à la même distance des deux maisons, on utilisera le bloc “arrondi” qui donne l’arrondi à l’unité d’une valeur ou d’un calcul.


→ Que dire de la position des arbres plantés ?

→ Modifiez le script pour que cette fois, les arbres soient deux fois plus près de la maison 1 que de la maison 2. Quelle forme semble avoir cette forêt ?

## Exercice 3: Modéliser une expérience aléatoire

→ Ouvrez le fichier "Hasard_3.sb2"




Ici, on considère une urne contenant 8 boules: 2 rouges, 5 vertes et 1 bleue.

On tire au hasard une boule dans l'urne et on regarde sa couleur.

Ce programme contient deux lutins:
- Celui de l'urne
- Celui d'une boule qui peut prendre trois aspects.

La modélisation sera la suivante: on numérote chaque boule de 1 à 8. 1 2 3 4 5 6 7 8

Ainsi, **choisir une boule revient à choisir un nombre aléatoire entre 1 et 8.**

→ Au lancement du programme, écrivez un script qui place l'urne aux coordonnées (0;0).
→ Faites de même pour la boule.

**« Mais comment faire croire que la boule est à l'intérieur de l'urne ?»**

Pour cela, il suffit de placer l'urne "devant" la boule.

Il existe pour cela le bloc 


Il suffit alors en début de programme d'envoyer l'urne au premier plan.

→ Créez une variable "Numéro"
→ Complétez le script de la boule de cette façon:
1- Mettre "Numéro" à un nombre aléatoire entre 1 et 8.
2- Si "Numéro" est strictement inférieur à 3, alors la boule prend le costume "Rouge".
3- Si "Numéro" est égale à 8, alors la boule prend le costume "Bleu".
4- Si "Numéro" est strictement supérieur à 2 et strictement inférieur à 8, alors la boule prend le costume "Vert".
5- Faire glisser la boule en 0,5 secondes aux coordonnées (0;130)
6- Puis faire glisser la boule en 0,5 secondes aux coordonnées (0;0)
→ Utilisez une boucle pour répéter ce tirage 100 fois.
(On baissera le temps du glissement de 0,5 à 0,1 seconde)

On aimerait calculer la fréquence d'apparition d'une boule rouge (en %) pour 1000 tirages.
→ Modifiez le script pour qu'il effectue 1000 tirages.
→ Créez une variable "Nombre de boules rouges", et la mettre à 0 en début de programme.
→ Complétez le script en ajoutant 1 à cette variable quand une boule rouge est tirée.
→ Une fois les 1000 tirages effectués, calculez la fréquence de sortie de la boule rouge, faites passer le lutin au premier plan et lui faire dire cette fréquence comme sur la vidéo ci-dessous: