

Ce projet a pour but de modéliser un jeu d'enfants appelé « le jeu du verger ».

L'objectif est d'estimer la durée moyenne du jeu pour vérifier si elle correspond aux critères classiques pour ce type de jeu (moins de 15 minutes) et de déterminer qui a la situation la plus favorable : les joueurs ou le corbeau.

Vous allez dans un premier temps apprendre à connaître le jeu et le modéliser.

Il pourra être utilisé d'observer des enfants y jouer : nous nous rendrons si possible au primaire (en classe de maternelle a priori).

Une fois le jeu modélisé, vous ferez de nombreuses **simulations** afin d'approcher la valeur moyenne du nombre de coups et la probabilité de victoire de chaque camp. Il est en effet trop compliqué de faire un modèle probabiliste de ce jeu. Ce niveau de complexité nous « oblige » à faire une simulation du jeu qui sera répétée un grand nombre de fois.

Les objectifs sont donc :

- de comprendre le fonctionnement du jeu ;
- de le modéliser par un algorithme ;
- de programmer cet algorithme pour faire de nombreuses simulations ;
- d'utiliser les données issues des simulations pour approcher le nombre de coups moyen d'une partie et la probabilité de victoire de chaque camp ;
- d'étudier l'influence des paramètres du jeu sur ces données.

Un compte-rendu est à rédiger au fur et à mesure des séances. Il contiendra en particulier des copies d'écran des travaux réalisés issus de CaRMetal ou d'autres supports, en suivant les consignes données par cette fiche.

Chaque partie de la fiche correspond plus ou moins à une séance.

1 Présentation

1.1 Le jeu

Faites quelques parties pour maîtriser les différents aspects du jeu.

Le nombre de joueurs est-il un paramètre à prendre en compte ?

Notez tous les paramètres qui interviennent dans le jeu.

1.2 Une première version

Vous allez au fur et à mesure des séances construire un algorithme et un programme qui se rapprocheront du vrai jeu en isolant certaines composantes du jeu. Dans un premier temps, nous allons simplifier la règle du jeu : **nous n'allons tenir compte que des tirages des quatre couleurs, en « oubliant » momentanément les faces « corbeau » et « panier ».**

Travail à faire :

Écrivez un algorithme correspondant à ce jeu (version 1).

1.3 Aide pour la programmation

Vous allez programmer cet algorithme sur le logiciel CaRMetal en lançant un script.

Ici quelques aides pour démarrer :

Pour créer un script (ça veut dire un programme), cliquer sur le menu **Javascript** (c'est le nom du langage), puis sur **Nouveau script dans la construction**. Donner lui le nom que vous voulez.

Une fenêtre s'ouvre dans laquelle vous allez taper votre code.

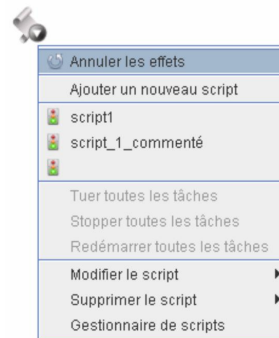
Pour démarrer, on peut taper : `l=Math.ceil(Math.random()*6);`

La partie `Math.ceil(Math.random()*6)` s'obtient en cliquant sur le dé visible dans la fenêtre de droite. Bien penser à mettre un « ; » à la fin de chaque instruction.

On tape ensuite également `Println(l);`. La fonction `Println()` s'obtient en cliquant sur *Affichage 1 ligne* à droite dans la fenêtre du script.

Lancer à présent le script (soit en le fermant et en cliquant sur l'icône  , soit en gardant le script ouvert et en cliquant sur le feu rouge).

Observez ce qui se passe. Pensez à **annuler les effets**. Cela peut se faire depuis le menu suivant :



On atteint aussi cet annulation en haut du script (par la même icône).

Pensez à enregistrer votre document au fur et à mesure.

remarque importante : le script s'enregistre tout seul au fur et à mesure qu'on le construit et qu'on le lance. Cependant, si vous n'enregistrez pas le document en tant que tel, vous perdez tout : le document, **et** le script !

Il est commode d'utiliser des **commentaires** : on les écrit après `//` : ils ne sont pas pris en compte dans l'exécution du programme et peuvent rendre la lecture du programme plus aisée (pensez que d'autres personnes vont lire votre programme).

Par ailleurs, on peut avoir besoin de mettre en oeuvre une ligne à un moment donné et de la « désactiver » plus tard : on utilisera les commentaires. Pour mettre en commentaires une zone du texte, on clique sur la bulle jaune en haut du script après avoir sélectionné la zone à Commenter/Décommenter.

1.4 Premier programme

Après cette entrée en matière, à vous de vous lancer !

Utilisez les fonctions prédéfinies qui se trouvent à droite de la fenêtre Javascript :



Votre objectif est donc de simuler un lancer de dé qui donne en sortie : soit le rouge, soit le bleu, soit le vert, soit le jaune.

Il faut répéter ces lancers et décompter le nombre de fruits au fur et à mesure que le dé est sorti sur la couleur correspondante. Ces lancers sont à effectuer tant qu'il reste des fruits.

On veut également connaître le nombre de coups nécessaires à cette « partie » qui vide tous les fruits des arbres.

2 Deuxième version

2.1 Modèle retenu

On complexifie le modèle précédent : aux quatre couleurs, **on ajoute le corbeau**.

La partie s'arrêtera donc

- soit lorsqu'il n'y a plus de fruits ;
- soit lorsque le corbeau a suffisamment avancé.

Travail à faire :

Écrivez un algorithme correspondant à ce jeu (version 2).

Dans un second temps, vous complétez le script fait précédemment (soit en l'écrasant, soit en le renommant) pour tenir compte du corbeau.

remarque : il faut qu'à la fin de la partie, on sache qui a gagné.

2.2 Plusieurs parties

On souhaite à présent faire plusieurs parties.

Vous pourrez faire deux scripts : un correspondant à une partie et un autre correspondant à plusieurs parties.

Dans ce dernier script, il faut comptabiliser le nombre de victoires du corbeau et des joueurs.

On souhaite avoir en sortie le nombre de coups de chaque partie et la fréquence de victoire du corbeau.

3 Troisième version

3.1 Modèle retenu

On complexifie le modèle précédent : aux quatre couleurs et au corbeau, **on ajoute le panier**.

On va s'intéresser spécifiquement à ce qui se passe quand le panier sort. C'est le seul moment où une « stratégie » est à prendre en compte. Pour un tout petit, la stratégie va consister à retirer les deux fruits qu'il préfère ! Pour un adulte, il faudra retirer les fruits là où ils sont le plus nombreux.

Nous allons choisir un modèle intermédiaire : nous retirerons des fruits « à vue d'oeil », c'est-à-dire grosso modo là où il y en a beaucoup, mais sans les compter.

Pour cela, on peut procéder de la manière suivante : on additionne le nombre de fruits restants pour connaître leur somme.

Par exemple, s'il y a 8 cerises, 4 pommes, 3 prunes et 1 poire, on a 16 fruits.

On va considérer que les fruits sont « rangés » dans cet ordre : 8 cerises/4 pommes/3 prunes/1 poire.

On tire alors au hasard un nombre de 1 à 16 ; ce nombre désignera le type de fruit qui sera choisi.

Reste à répéter le processus deux fois (puisqu'on retire deux fruits).

Travail à faire :

Écrivez un algorithme correspondant à cette partie du jeu.

Dans un second temps, écrivez un nouveau script pour modéliser cette partie du jeu.

Assurez vous que le script fonctionne bien. Pour cela, vous pouvez afficher « l'historique » d'une partie en visualisant pour chaque lancer de dé :

- la face de dé sortie ;
- le cas échéant (si « panier » est sorti), le tirage au sort d'un nombre entre 1 et le nombre total de fruits ;
- le nombre restant de chaque fruit ;
- la position du corbeau ;
- le nombre de coups.

Une fois que le script fonctionne, incorporez le dans le script précédent (ce sera donc la version 3 du script).

3.2 Plusieurs parties

On souhaite à présent faire plusieurs parties.

Vous pourrez faire deux scripts : un correspondant à une partie et un autre correspondant à plusieurs parties.

Dans ce dernier script, il faut comptabiliser le nombre de victoires du corbeau et des joueurs.

On souhaite avoir en sortie le nombre de coups de chaque partie et la fréquence de victoire du corbeau.

4 Traitement des résultats

A ce stade du travail, vous devez avoir un script qui fonctionne et qui vous donne pour le nombre de simulations désiré **la fréquence de victoire du corbeau** et **le nombre de coups pour une partie**.

Ce sont ces dernières valeurs que nous allons traiter.

4.1 Traitement « manuel »

Vous êtes capable (sur une 100 simulations par exemple) de dresser une liste de 100 valeurs de n : que faire de ces valeurs numériques ?

Vous allez les traiter avec les outils dont vous disposez habituellement (calculatrice, tableur).

4.2 Utilisation du logiciel R

Ce logiciel est particulièrement intéressant pour le traitement statistique d'un grand nombre de données.

Lancez 100 000 simulations du jeu du verger et enregistrez la liste du nombre de coups dans un bloc note (format texte).

Nous allons ensuite utiliser le logiciel R pour faire le même traitement que celui fait de manière plus traditionnel avec les 100 simulations précédentes.

5 Pour aller plus loin dans le graphisme

Le fait d'avoir fait le travail sur un logiciel de Géométrie nous permet « d'habiller » le travail en associant le script et une figure.

5.1 Les fruits

Voici un petit exemple pour comprendre : sur CaRMetal, créez un point rouge bien visible (le mettre en gras et le grossir si besoin). Nommez ce point R1 et cachez le (à l'aide de la gomme).

Faites un script donnant au hasard 1 ou 2 ; on stockera la valeur dans d .

Complétez ensuite ce script de la manière suivante :

```
if (d==1) { SetHide("R1",true) } else { SetHide("R2",false) };
```

Lancez plusieurs fois le script et observez.

Une astuce : si vous voulez faire le même type d'opération pour plusieurs points nommé R1, R2 ... R10, le script comprend l'instruction suivante :

```
if (d==1) { SetHide("R"+k,true) } else { SetHide("R"+k,false) };
```

où k désigne un nombre entre 1 et 10.

On peut ainsi créer des fruits visibles dans un arbre et des fruits cachés dans un panier. Au fur et à mesure que l'on récupère un fruit après un lancer de dé, on cache un fruit dans l'arbre pour faire apparaître ce fruit dans le panier.

On peut ainsi rendre le déroulement du programme plus « amusant ».

5.2 Pause

Il n'est pas question que vous fassiez une pause ! Mais le script lui, oui ...

En effet, pour bien voir les fruits qui descendent de l'arbre pour aller dans le panier, il faut que le script se déroule moins vite.

A vous d'incorporer une pause (en cliquant sur l'icône symbolisant une horloge) et de régler le temps de pause (donner en milli secondes).

Dans un deuxième temps, vous pourrez demander à l'utilisateur de choisir son temps de pause (pour bien voir ce qu'il se passe ou non) en utilisant la fonction *Entrée* située dans la liste des fonctions à droite du script.

5.3 Le corbeau

Il est possible d'importer des images dans CaRMetal (avec le logo correspondant dans la zone de construction).

On peut donc incorporer une image du corbeau et le placer sur le plateau.

On pourra en fait incorporer 9 mêmes images du corbeau, correspondants aux 9 zones potentielles où il peut se trouver au cours du jeu. Par une méthode analogue à celle présentée plus haut, on donnera l'impression que le corbeau avance au fur et à mesure que le dé tombe sur « corbeau ».

6 Conclusions

Répondez à ce type de questions :

- Quelle est la probabilité que le corbeau gagne ?
- Quels paramètres faut-il modifier pour que le jeu soit équilibré ?
- Quel est le temps moyen du jeu ?
- Quel est le nombre minimal de coups ? Quelle est la probabilité que cela arrive ?
- Quel est le nombre maximal de coups ?
- Quelle est la probabilité que le jeu se joue en moins de 30 coups ? En plus de 50 coups ?