

**FICHE PROFESSEUR**

- **1ère Générale ou Technologique**
- **Durée : 1h ou 2h**
- **Matériel à prévoir : des dés et des pions**
  
- **Objectif pédagogique (compétence contextualisée attendue) :**
  - x Représenter une situation par un arbre pondéré de probabilités
  - x Arbres pondérés et calcul de probabilités : règle du produit, de la somme
  - x Comprendre et utiliser une simulation numérique
  - x Concevoir et écrire une instruction conditionnelle
  - x Concevoir et écrire une boucle for
  
- **La situation-problème :** Les élèves découvrent un jeu simple possédant plusieurs stratégies de jeu. Ils expérimentent les différentes stratégies en jouant. Ils doivent ensuite calculer les probabilités de réussite de chacune des stratégies : la valeur exacte en calculant à l'aide d'un arbre ou la valeur approchée en écrivant un programme qui simule un grand nombre de parties de jeu.
  
- **Pré-requis :**
  - x Les élèves doivent avoir vu les calculs sur les arbres pondérés en probabilités
  - x Les élèves doivent être à l'aise avec l'écriture d'un programme Python, notamment les instructions conditionnelles et les boucles for.
  
- **Contexte :** Découverte d'un principe de l'intelligence artificielle.  
Quand on joue à un jeu, on souhaite maximiser sa probabilité de gagner. Si cette probabilité est trop difficile à calculer, on peut obtenir une estimation, en moyennant les résultats d'un grand nombre de parties simulées. Certains programmes d'intelligence artificielle fonctionnent selon ce principe. C'est sur ce principe que l'ordinateur Deep Blue a été programmé et a réussi à gagner face au champion d'échecs Kasparov en 1997. En effet, dans le cas du jeu d'échec, à chaque coup la valeur exacte de la probabilité de gagner est inaccessible, car il y a trop de possibilités à envisager.
  
- **Les consignes et la réalisation attendue :**
  - x Partie A : Travail à faire à la maison
  - x Partie B : Les élèves jouent avec une stratégie différentes chacun.
  - x Partie C : Les élèves calculent les probabilités de gagner des différentes stratégie de jeu.
  - x Partie D : Bilan
  
- **Consignes pour la partie programmation :**  
Les élèves peuvent programmer en local avec un éditeur Python de type Thonny ou Edupython, ou en ligne en utilisant les liens Capytale ci-dessous :
  - x Stratégie des sauts moyens : [fiche énoncé](#) et [corrigé](#)
  - x Stratégie des petits sauts : [fiche énoncé](#) et [corrigé](#)

# TRAVERSÉE DE LA RIVIÈRE

- **Les aides possibles à apporter :**

- ✓ Fournir des arbres de probabilités à compléter
- ✓ Pour aider les élèves à programmer, on peut leur proposer de lire le code des fonctions Python pour « petit saut + grand saut ». Dans les activités en ligne Capytale, on propose aux élèves de commencer par lire et exécuter la fonction `\sauter()` qui simule un petit saut.

- **Différenciation possible :**

- ✓ Programmer la stratégie « Trois petits sauts » est plus dur.

- **Prolongements possibles :**

- ✓ Inventer une stratégie avec une plus grande probabilité de gagner. Par exemple, on peut estimer que la stratégie « saut moyen + grand saut » aura une probabilité de gagner de 0,67.
- ✓ Proposer aux élèves de retirer un des rochers et refaire les calculs.

- **Analyse du dispositif :**

Les élèves ont apprécié le jeu. Les aides à la programmation étaient les bienvenues, et nous conseillons l'utilisation des pages Capytale pour la partie programmation.