

# Fiche Guide : Le Jeu de 52 Cartes

Spécialité NSI - Première

## Contexte du projet

L'objectif est de modéliser un jeu de cartes standard, de le mélanger et de gérer une distribution de cartes. Vous allez manipuler des **listes**, des **tuples** et utiliser des **boucles**.

## 1 Les Couleurs

*Objectif : Écrire une instruction Python qui renvoie la liste des 4 couleurs (pique, cœur, carreau, trèfle).*

### 💡 Indication

Il s'agit simplement de définir une variable contenant une liste de chaînes de caractères (**strings**).

```
# Exemple de structure attendue
couleurs = [". . .", ". . .", ". . .", ". . ."]
```

## 2 Les Valeurs

*Objectif : Écrire une instruction qui renvoie la liste des 13 valeurs (As, 2, 3... Roi).*

### 💡 Logique à suivre

Attention, certaines valeurs sont numériques (2 à 10) et d'autres sont des mots ("As", "Valet"...).

- **Option simple** : Tout écrire à la main dans une liste.
- **Option avancée** : Concaténer une liste contenant "As", une liste générée par `range()` pour les nombres, et une liste pour les figures.

## 3 La fonction `creer_jeu()`

*Objectif : Créer une liste de 52 cartes sous forme de tuples ('7', 'pique').*

### 💡 Raisonnement

Pour obtenir toutes les combinaisons possibles (chaque valeur avec chaque couleur), vous devez utiliser deux boucles imbriquées (principe du produit cartésien).

### ⚡ Rappel : Ajouter dans une liste

```
ma_liste = []
# Boucle sur les couleurs
    # Boucle sur les valeurs
        carte = (valeur, couleur) # Création du tuple
        ma_liste.append(carte)     # Ajout à la liste
return ma_liste
```

## 4 La fonction melange(jeu)

Objectif : Renvoyer la liste mélangée du jeu de cartes.

### 💡 Utilisation de modules

Python possède un module dédié à l'aléatoire : `random`. Cherchez la fonction qui permet de mélanger une liste *en place* (in-place).

```
import random

def melange(jeu):
    # Attention : random.shuffle(liste) ne renvoie rien (None),
    # elle modifie directement la liste entre parenthèses.
    # Votre fonction doit :
    # 1. Mélanger le jeu
    # 2. Renvoyer le jeu mélangé
```

## 5 La fonction distribue(jeu, n)

Objectif : Distribuer équitablement les cartes à  $n$  joueurs et renvoyer le "pot" (les cartes restantes).

### 💡 Analyse mathématique

Si vous avez 52 cartes et  $n$  joueurs :

1. Combien de cartes chaque joueur reçoit-il ? (Division entière //)
2. Combien de cartes au total sont distribuées aux joueurs ?
3. Combien de cartes reste-t-il à la fin pour le "pot" ?

### ⚡ Astuce : Le Slicing (Découpage)

Le "pot" correspond aux dernières cartes de la liste qui n'ont pas été distribuées. Si on a distribué  $X$  cartes au total, le pot commence à l'indice  $X$  jusqu'à la fin.

```
nb_cartes_distribuees = ...
pot = jeu[nb_cartes_distribuees : ] # Récupère la fin de la liste
return pot
```