# Mini-projet
# Choixpeau magique

# I. Le choixpeau magique dans le roman Harry Potter

Le Choixpeau magique (Sorting Hat) est un artefact doué d'une âme qui détermine de façon magique dans laquelle des quatre maisons de Poudlard chaque nouvel élève doit être envoyé. Ces quatre maisons ont pour nom : Gryffondor (Gryffindor), Poufsouffle (Hufflepuff), Serdaigle (Ravenclaw) et Serpentard (Slytherin).

Le Choixpeau est investi de l'intelligence des fondateurs et a le don de Legilimancie, lui permettant de lire dans les pensées de celui qui le met sur sa tête et de ressentir ses aptitudes ou ses humeurs et peut également répondre aux pensées de celui qui le porte.

Il est utilisé à chaque rentrée le 1er septembre lors de la cérémonie de répartition des élèves de première année, et ce depuis plusieurs siècles. Le Choixpeau choisit la maison adéquate pour chaque élève en lisant dans ses pensées, et déclare à haute voix sa décision finale prise en fonction des vertus, des projets et des centres d'intérêts de celui-ci.

https://harrypotter.fandom.com/fr/wiki/Choixpeau_magique

# II. Le choixpeau magique en NSI

L’objectif de ce mini-projet est de **modéliser un choixpeau magique numérique**.

Votre création numérique devra donc être en mesure de déterminer la maison la mieux adaptée à un profil d’« élève cible ».

Vous devrez donc :

- **Construire un modèle, basé sur l’algorithme des k plus proches voisins**, capable d’attribuer une maison lorsqu’on lui fournit un profil d’élève cible et des profils d’élèves en référence.
- **Construire une procédure permettant de créer ce profil d’élève cible**, en élaborant un QCM pertinent (2nde partie).
- **Construire une interface graphique ergonomique**, permettant la saisie du QCM (2nde partie), puis l’affichage du verdict du choixpeau magique.

# III. Le choixpeau magique : 1ère partie

## 1. Objectif

La première partie consiste à concevoir le cœur du programme, celle qui utilisera l'algorithme des kPPV en s'appuyant sur les élèves références (Caracteristiques_des_persos.csv) et sur un profil d'élève cible prédéfini.

Le programme devra renvoyer les k plus proches voisins de plusieurs profils d'élèves donnés ci-dessous, puis leurs maisons attitrées.

## 2. Cahier des charges

Votre programme doit être conçu pour faciliter son test et son évaluation. Il doit donc être complet, exécutable immédiatement avec un interpréteur Python.

L'utilisation du tri built-in de Python sort() est autorisée.

Il est à nouveau conseillé de construire une fonction par objectif et de structurer au mieux votre programme (Encodage, commentaire introductif, variables globales, définitions des fonctions, appel des fonctions dans un corps de programme structuré).

Vos fonctions doivent être parfaitement documentées dans leur docstring.

Le choixpeau magique devra se prononcer sur les profils suivants :

- Courage : 9 Ambition : 2 Intelligence : 8 Good : 9
- Courage : 9 Ambition : 4 Intelligence : 8 Good : 9
- Courage : 3 Ambition : 8 Intelligence : 6 Good : 3
- Courage : 2 Ambition : 3 Intelligence : 7 Good : 8

Les données doivent résulter d'une fusion entre les fichiers Characters.csv et Caracteristiques_des_persos.csv.

Il n'est pas autorisé de modifier les caractéristiques des personnages.

Les données étant en anglais, il est préférable d'utiliser des noms de fonctions et de variables en langue anglaise.

La distance par défaut sera la distance euclidienne.

La valeur de k sera fixée à 5 par défaut.

Votre IHM doit nécessairement automatiser les passages de chaque profil devant le choixpeau magique, avec une visualisation claire des résultats obtenus (la maison choisie + les plus proches voisins et leurs maisons respectives).

L'IHM ne doit pas nécessairement passer par une interface graphique de type Tkinter, Pygame ou page web, même si cette option est envisageable (bonus). Considérant que ce sera l'objet du cahier des charges obligatoire de la seconde partie, il paraît inutile d'y dépenser trop d'énergie dans un premier temps.

Un algorithme doit être rédigé en pseudo-code et ne doit décrire que le cœur de la problématique (l'algorithme kPPV), pas les créations de structures de données ni l'IHM.

Au besoin, votre algorithme pourra utiliser des appels de fonctions…

- dont il est inutile de détailler le pseudo-code.
- avec un nom suffisamment explicite pour comprendre leur utilité.
- avec un descriptif sous le mot clé FONCTIONS au dessus du détail des VARIABLES.
- avec un équivalent de leur docstring dans ce descriptif

Enregistrer le programme sous le format : kPPV_1_NOM1_NOM2_NOM3.py

Si vous avez un document PDF (ex : algorithme, lien vers vos outils collaboratifs, aide à l'utilisation de votre programme,...) : kPPV_1_NOM1_NOM2_NOM3.pdf

Créer un fichier compressé (kPPV_1_NOM1_NOM2_NOM3.zip) pour organiser vos fichiers. Après décompression, la structure de dossiers / fichiers doit permettre une exécution facile du programme (pas de changement de nom de fichier ou de dossier à effectuer).

Votre projet doit être déposé sur l'espace de travail NSI d'e-lyco (« Vue d'ensemble », « tâches »), en temps voulu. Vous devrez y créer votre groupe avant de déposer vos fichiers.

**Remarque :** les notebooks 4_11, 4_12, 4_13, P1, P2 et P4 pourront vous servir.

## 3. Bonus possibles

- La saisie manuelle d'un profil d'élève à tester par le choixpeau peut être ajoutée.
- La valeur de k doit pouvoir être modifiée dans l'IHM.
- Une validation croisée est effectuée, avec affichage clair des résultats en pourcentage.
- Une optimisation des résultats est recherchée (à l'aide de la validation croisée, avec un travail sur les données, sur la valeur de k, …).

## 4. Évaluation

Critères d'évaluation (note sur 10 points) :

- Respect du cahier des charges (5 points)
- Algorithme (2 points)
- Bonnes pratiques PEP-8 (dont commentaires et docstrings) (1 point)
- Travail d'équipe / Utilisation d'outils collaboratifs numériques (1 point)
- Point « subjectif » (1 point)
- Bonus (+ 2 points envisageables en fonction de l'ampleur du bonus)

# IV. Le choixpeau magique : 2ème partie

## 1. Objectif

L'objectif est double :

- Construire un questionnaire permettant de créer le profil de l'utilisateur qui y répondra .
- Créer une IHM structurée sur une page web

## 2. Cahier des charges

### a. Le questionnaire

Le QCM doit comporter entre 10 et 20 questions.

Chaque question doit proposer un choix d'au moins 3 réponses.

Les questions doivent être, si possible, basée sur une mise en situation permettant d'établir une ébauche de profil psychologique. Il faut donc, dans la mesure du possible, éviter les questions qui suggèrent trop grossièrement un rapport avec les 4 maisons de Poudlard.

Chacune des réponses sera liée à un tuple de valeurs numériques, correspondant aux 4 caractéristiques : (Courage, Ambition, Intelligence, Bonté).

Un exemple simpliste :

- Question : As-tu peur du noir ?
  - Oui, j'en suis terrifié (0, 5, 5, 5)
  - Oui, ça ne me met pas à l'aise (3, 5, 5, 5)
  - Non, ça ne me fait ni chaud ni froid (5, 5, 5, 5)
  - Non, au contraire, je m'y sens bien (8, 5, 5, 5)
- On voit que cette question ne teste que la caractéristique de courage, mais rien n'empêche de construire des questions plus complexes, mettant en jeu plusieurs des caractéristiques à évaluer.

Le questionnaire permet d'obtenir un profil numérique de l'utilisateur, sous la forme d'un tuple de 4 entiers.

Un ajustement sera probablement utile pour obtenir des valeurs proches de la moyenne de celles des élèves de Poudlard, sinon les résultats risquent d'être déséquilibrés.

### b. La page web

La page web doit être d'un design simple et efficace. Par exemple, le clic sur les réponses proposées paraît la solution la plus pertinente pour répondre à un QCM.

A l'issu de ce questionnaire, la page doit afficher :

- La maison attribuée à l'utilisateur.
- Les noms et les maisons des k plus proches voisins.
- La possibilité de rejouer.

Elle doit se baser sur les langages HTML / CSS. Le CCS doit être réuni dans un fichier séparé.

Le programme, permettant de déterminer la meilleure maison, peut être conservé en langage Python, en utilisant le langage Brython pour intégrer Python à la page web.

Dans la mesure du possible…

- séparer le code Brython dans un fichier séparé.
- minimiser l'utilisation du HTML pour privilégier Brython.

## 3. Bonus possibles

- Ajouter un sélecteur pour changer la valeur de k
- Ajouter 4 sélecteurs pour tester des profils types (ex : profil (7, 6, 3, 8)), sans passer par le questionnaire.
- Gérer les situations d'ex æquo, en augmentant ou diminuant ponctuellement la valeur de k par exemple.
- Mettre votre page web en ligne, à une adresse discrète, si possible non référencée.
- Pour les très courageux : recoder tout le cœur de votre programme Python en Javascript pour vous passer de Brython. Dans ce cas, il faudra faire preuve d'une grande autonomie...

## 4. Évaluation

Critères d'évaluation (note sur 10 points) :

- Respect du cahier des charges (7 points)
- Bonnes pratiques PEP-8 (1 point)
- Travail d'équipe / Utilisation d'outils collaboratifs numériques (1 point)
- Point « subjectif » (1 point)
- Bonus (+ 2 points envisageables en fonction de l'ampleur du bonus)

**Remarque :** il ne vous est pas demandé d'algorithme en pseudo-code pour cette 2ème partie.