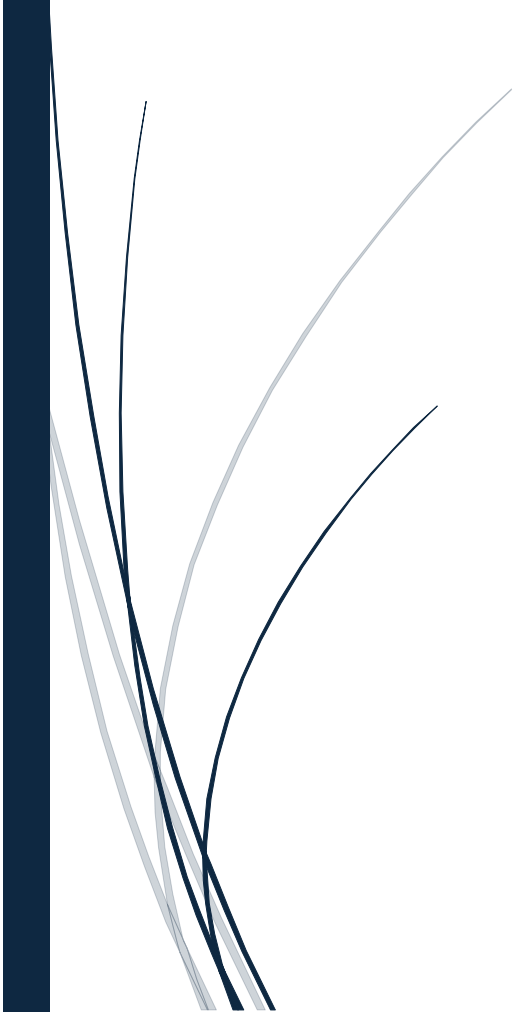
17/01/2025

Rapport de projet : Recherche et gestion des livres dans une bibliothèque



Amadis Senelet, Bekir Canan

Table des matières

1. Description du projet

L'objectif principal du projet est de se servir d' au moins 2 structures de données parmi les piles, files, listes et arbres afin de développer une application permettant :

- La recherche de livres dans une bibliothèque numérique.
- La gestion d'un historique des recherches.
- Un affinage des résultats de recherche à travers des critères multiples tels que le titre, l'auteur, la langue, le style ou le genre.

L'interface utilisateur est construite en Python à l'aide de **Tkinter** pour permettre des interactions dynamiques. Elle offre des fonctionnalités telles que :

- La navigation dans l'historique des recherches (avant et arrière).
- La visualisation des livres consultés.
- Le tri dynamique des résultats selon plusieurs colonnes.

La recherche d'optimisation est également au centre du projet, notamment au niveau du choix du tri. Au total quatre tris seront disponibles, et a chaque appel, le meilleur sera sélectionné et donc appliqué.

2. Justification des structures de données choisies

Liste chaînée

- Gestion des livres : Chaque livre est représenté comme un maillon dans une liste chaînée. Les livres sont représentés par des dictionnaires dont les clés sont les paramètres de recherche.
- Gestion des piles et files : Ces structures sont implémentées à l'aide de listes chaînées pour gérer l'historique des recherches et des consultations.

Pile

- Utilisée pour gérer l'historique des recherches, permettant un accès en mode LIFO (Last In, First Out).

File

- Utilisée pour gérer des structures nécessitant un accès FIFO (First In, First Out), bien que peu exploitée dans ce projet spécifique.

3. Évaluation de la complexité des algorithmes

Recherche

- **Algorithme :** Parcours séquentiel de la liste chaînée.
- **Complexité :** $O(n \cdot m)$, où est le nombre total de livres et est le nombre de critères de recherche.

Tri

- Dans un souci d'optimisation, plusieurs tris on été implémentés, les voici accompagnés de leur complexité :
    - **Tri à bulles :** $O(n^2)$
    - **Tri par insertion :** $O(n^2)$ (meilleur cas $O(n)$ si la liste est presque triée).
    - **Tri par sélection :** $O(n^2)$
    - **Tri par fusion :** $O(n\ log\ n)$, le plus efficace pour des grands ensembles de données.
- **Stratégie adoptée :** Le tri le plus optimisé est déterminé dynamiquement en fonction de la complexité de chaque algorithme sur les données actuelles.

Gestion des piles et files

- **Empilement/Dépilement :** $O(1)$, car il s'agit d'ajouter ou de retirer un maillon au début de la liste chaînée.
- **Complexité mémoire :** Faible, grâce à l'utilisation de listes chaînées.

4. Fonctionnalités principales

1. **Affinage des recherches** : L'historique est parcouru pour affiner les résultats au fur et à mesure de l'ajout de paramètres.
2. **Tri dynamique** : Les colonnes des résultats peuvent être triées par un simple clic.
3. **Navigation historique** : L'utilisateur peut naviguer dans les recherches précédentes ou futures (si elles ont déjà été effectuées) .
4. **Affichage des livres consultés** : Les livres consultés sont affichés séparément pour un suivi.