Rapport du Code

Gestion des Fichiers et des Blocs Mémoire

Section B

ATTIGUI Fatima Zohra 222231639114 / G4

BOUSSADA Abderraouf 212231447313 / G1

CHERIFI Kaouther 232331717610 / G3

HAMIDI Assala 232331640907 / G3

TABTI Sabrina 232331546303 / G2

10.01.2025

- Introduction

Ce rapport présente le code développé pour la gestion des fichiers et des blocs mémoire. L'objectif principal est de manipuler des fichiers structurés sous forme de blocs, permettant des opérations comme l'insertion, la modification, la recherche, la suppression, et bien d'autres. Ce système inclut également des fonctionnalités avancées telles que la défragmentation et la sauvegarde/restauration des fichiers.

- Description des structures de données et algorithmes utilisés :

2.1 Structures de données principals :

Record (Enregistrement)

- Structure représentant un enregistrement.
- Champs :
- record_ID : Identifiant unique de l'enregistrement (type entier).
- content : Contenu de l'enregistrement (tableau de caractères de taille fixe).

Block (Bloc)

- Représente un bloc mémoire contenant un enregistrement.
- Champs :
- is_empty : Indique si le bloc est vide ou occupé.
- file_name : Nom du fichier auquel le bloc est associé.
- record_data : Contenu de l'enregistrement stocké dans ce bloc.

File (Fichier)

- Représente un fichier composé de plusieurs blocs.
- Champs :
- name : Nom du fichier.
- total_records : Nombre total d'enregistrements alloués au fichier.
- blocks : Tableau de pointeurs vers des blocs de mémoire.

MetaInfo (Métadonnées)

- Fournit des informations sur les métadonnées d'un fichier.
- Champs :
- meta_file_name : Nom du fichier de métadonnées.
- block_count : Nombre total de blocs utilisés.
- record_count : Nombre total d'enregistrements dans le fichier.
- first_block_addr : Adresse du premier bloc utilisé dans le fichier.

2.2 Algorithmes principaux :

- Insertion d'un enregistrement :

• Parcours linéaire des blocs pour trouver le premier bloc vide.
• Mise à jour des métadonnées du bloc avec les informations de l'enregistrement.

2.Recherche d'un enregistrement :

• Recherche séquentielle dans les blocs associés à un fichier pour localiser l'enregistrement par son record_ID.

3.Modification d'un enregistrement :

• Recherche de l'enregistrement cible par son record_ID.
• Mise à jour du champ content de l'enregistrement correspondant.

4.Suppression d'un enregistrement:

• Recherche de l'enregistrement par son record_ID.
• Marquage du bloc comme vide en mettant à jour son champ is_empty.

5.Tri des enregistrements :

• Algorithme de tri par échange (bubble sort).
• Comparaison des record_ID des enregistrements pour les trier dans l'ordre croissant.

6.Défragmentation :

• Réorganisation des blocs associés à un fichier pour regrouper les blocs utilisés en début de tableau

7.Compaction de la mémoire :

• Réorganisation de l'ensemble des blocs mémoire pour éliminer les zones vides et regrouper les blocs utilisés..

3. Résultats des tests

3.1 Test 1 : Création d'un fichier

- Description : Création d'un fichier avec un nombre défini d'enregistrements.
- Résultat attendu : Le fichier est correctement initialisé avec des blocs vides.
- Capture d'écran :

3.2 Test 2 : Insertion d’un enregistrement

- Description : Insertion d’un enregistrement dans un fichier.
- Résultat attendu : L’enregistrement est inséré dans le premier bloc vide disponible.
- Capture d’écran :

3.3 Test 3 : Recherche d’un enregistrement

- Description : Recherche d’un enregistrement par son record_ID.
- Résultat attendu : L’index du bloc contenant l’enregistrement est retourné.
- Capture d’écran

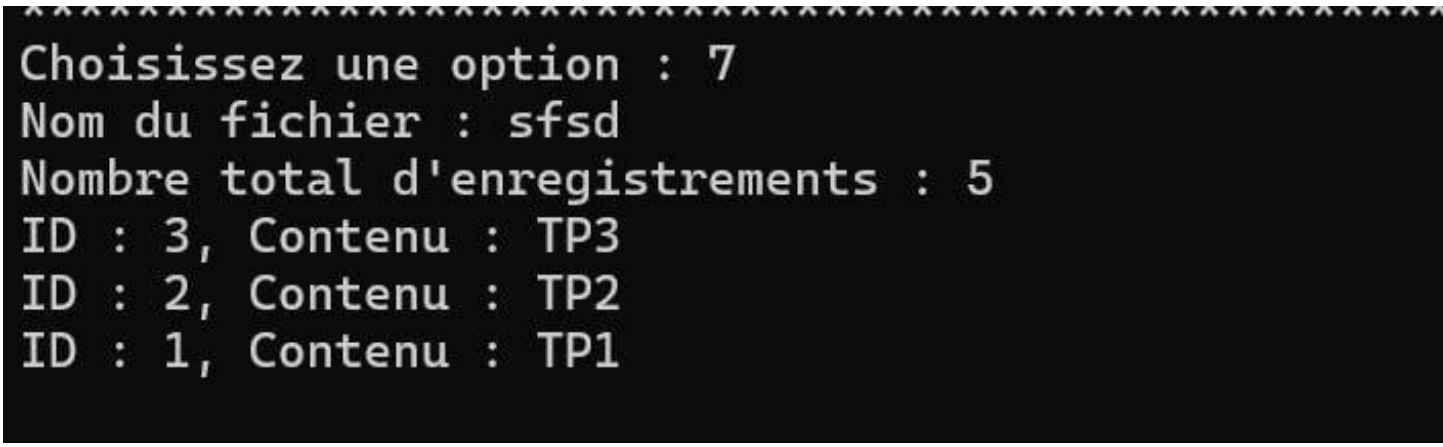
3.4 Test 4 : Modification d’un enregistrement

- Description : Modification du contenu d’un enregistrement existant.
- Résultat attendu : Le champ content de l’enregistrement est mis à jour avec succès.
- Capture d’écran :

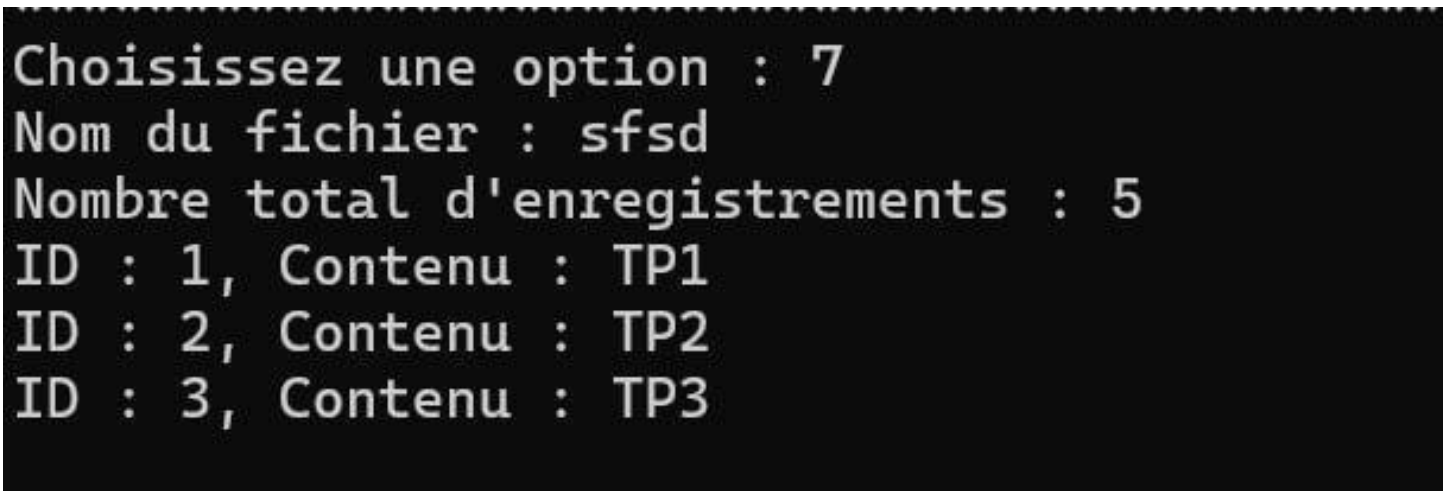
3.5 Test 5 : Tri des enregistrements

- Description : Tri des enregistrements d’un fichier par ordre croissant de leurs record_ID.
- Résultat attendu : Les enregistrements sont triés dans l’ordre correct.
- Capture d’écran :

Avant :



Apres :



3.6 Test 6 : Défragmentation d'un fichier

- Description : Regroupement des blocs utilisés en début de fichier.
- Résultat attendu : Les blocs vides sont déplacés à la fin.
- Capture d'écran :

3.7 Test 7 : Ajouter un fichier :

- Description : Sauvegarde le deuxième fichier après le premier
- Résultat attendu : Le fichier est sauvegarder après le premier est les blocs vides sont déplacés à la fin.
- Capture d'écran :

4. Conclusion

Le programme implémente efficacement la gestion des fichiers et des blocs mémoire en simulant des opérations complexes comme l'insertion, la recherche, et la défragmentation. Les tests confirment le bon fonctionnement des fonctionnalités. Cependant, certaines optimisations, notamment pour le tri et la recherche, pourraient être envisagées pour améliorer les performances.