Midterm Prep: Account Management System

Requirements:

1. Define the `Account` Class:
   - The `Account` class should include the following fields:
     - `String firstName`
     - `String lastName`
     - `int id`
     - `double balance`
   - Implement a constructor, getters, setters, and a `toString()` method for easy display.

2. Write Accounts to a Binary File:
   - Implement a method:
     public static void writeAccountsToFile(ArrayList<Account> accounts, String filename) throws IOException
   - Requirements:
     - Use `ObjectOutputStream` to serialize and save `Account` objects to a binary file.
     - Handle file operations using a try-with-resources block.

3. Read Accounts from a Binary File:
   - Implement a method:
     public static ArrayList<Account> readAccountsFromFile(String filename) throws IOException, ClassNotFoundException
   - Requirements:
     - Use `ObjectInputStream` to deserialize `Account` objects from a binary file.
     - Ensure proper handling of `EOFException` to avoid errors when reading the file.

4. Group Accounts by Balance Range:
   - Implement a method:
     public static Map<String, List<Account>> groupAccountsByBalance(ArrayList<Account> accounts)
   - Requirements:
     - Define three balance categories:
       - `"High"`: Balance >= 10,000
       - `"Medium"`: Balance between 5,000 and 9,999
       - `"Low"`: Balance < 5,000
     - Group `Account` objects into these categories and store them in a `HashMap`.

5. Main Method to Test All Features:
   - Implement a `main` method to:
     1. Create sample `Account` objects and store them in an `ArrayList`.
     2. Write the accounts to a binary file using the `writeAccountsToFile` method.
     3. Read the accounts back from the binary file using the `readAccountsFromFile` method.

4. Group the accounts by balance range using the `groupAccountsByBalance` method.
5. Print the grouped accounts to verify the output.

6. Handle Exceptions Gracefully:
   - Ensure all methods handle relevant exceptions, such as:
     - `IOException` for file operations.
     - `ClassNotFoundException` for deserialization.
   - Print appropriate error messages if exceptions occur.

7. FileInputStream - unrelated to the Account system

- Write a method public static void readFile(String filename) that:

    1. Opens the specified file using FileInputStream.
    2. Reads the file byte by byte and prints each character to the console.
    3. Closes the file stream after reading.

Optional Extensions:
1. Add functionality to Sort Accounts by Balance:
   - Create a method that sorts accounts in ascending or descending order of balance.
2. Filter Accounts by Criteria:
   - Implement a method to filter accounts based on custom criteria, such as accounts with a balance above a specific threshold.