Du sollst ein Konsolenprogramm entwickeln, das als einfacher Task-Manager fungiert. Der Task-Manager ermöglicht es Benutzern, Aufgaben zu verwalten, Prioritäten zuzuweisen, Aufgaben zu sortieren und verschiedene Aktionen auszuführen. Dies soll ohne die Verwendung von Streams implementiert werden, um grundlegende Java-Konzepte zu üben und ein tieferes Verständnis von Schleifen, Datenstrukturen und Objektorientierung zu erlangen.

Ziele:

Der Task-Manager muss folgende Anforderungen erfüllen:

1. **Aufgaben hinzufügen**
    - Der Benutzer soll eine neue Aufgabe mit Name, Beschreibung und Priorität (1 = höchste, 5 = niedrigste) erstellen können.
    - Jede Aufgabe hat eine eindeutige ID, die automatisch generiert wird.
2. **Aufgaben anzeigen**
    - Der Benutzer soll offene Aufgaben oder abgeschlossene Aufgaben separat anzeigen lassen können.
    - Die Liste der Aufgaben soll sortiert nach Priorität ausgegeben werden (1 hat die höchste Priorität).
3. **Aufgabe als erledigt markieren**
    - Der Benutzer soll eine Aufgabe anhand der ID als erledigt markieren können.
4. **Aufgaben löschen**
    - Der Benutzer soll eine Aufgabe anhand der ID dauerhaft löschen können.
5. **Aufgabe priorisieren**
    - Der Benutzer soll die Priorität einer bestehenden Aufgabe ändern können.
6. **Aufgaben nach Priorisierung ausgeben (optional auch nach anderen Kriterien)**
    - Eine Funktion soll die Aufgabenliste sortieren
7. **Aufgaben suchen (nach Namen oder auch nach Beschreibung,...)**
    - Eine Suchfunktion ermöglicht es dem Benutzer, nach Aufgaben anhand eines Namens oder eines Teilstrings zu suchen.
8. **Statistiken (optional)**
    - Zeige die Gesamtanzahl der Aufgaben sowie die Anzahl offener und abgeschlossener Aufgaben.
9. **Einfache Navigation**
    - Implementiere ein Menüsystem, das dem Benutzer die Navigation zwischen den oben genannten Funktionen ermöglicht.

Technische Anforderungen:

1. **Klasse Task:**
    - Diese Klasse soll Attribute wie id, name, description, priority und isCompleted enthalten.
    - Sie enthält Methoden wie markAsCompleted() und setPriority().
2. **Menüsystem:**
    - Implementiere ein textbasiertes Menü, das den Benutzer durch die verschiedenen Funktionen führt (z. B. 1. Aufgabe hinzufügen, 2. Aufgaben anzeigen, etc.).

Optional: Versuche das in einer Datei zu speichern. Dazu gibt es weiter unten Hinweise!

Viel Erfolg 😊

BufferedWriter/Reader

- **Vorgaben:**
    - Die Datei soll **"tasks.txt"** heißen.
    - Jede Zeile in der Datei soll eine Aufgabe enthalten, getrennt durch Kommas (ID,Name,Beschreibung,Priorität,Erledigt).
    - Verwende BufferedWriter, um die Datei zu speichern.
    - Falls ein Fehler auftritt, soll eine Fehlermeldung ausgegeben werden.

```
 class TaskFileHandler {
    private static final String FILE_NAME = "tasks.txt";

    public static void saveTasksToFile(List<Task> tasks) {
        try (BufferedWriter writer = new BufferedWriter(new
FileWriter(FILE_NAME))) {
            //TODO: ForSchleife über die Liste. Aufrufen der
writer.write(String s) -Methode. Als s einsetzen, wie
            // Der Task in der File gespeichert werden soll. Einzelne
Elemente sollen mittels Sonderzeichen getrennt werden. Mittels
writer.newLine() eine neue Zeile einfügen.
            System.out.println("Aufgaben erfolgreich gespeichert!");
        } catch (IOException e) {
            System.out.println("Fehler beim Speichern der Aufgaben: " +
e.getMessage());
        }
    }
}
```

```java
public static List<Task> loadTasksFromFile() {
    List<Task> tasks = new ArrayList<>();
    try (BufferedReader reader = new BufferedReader(new FileReader(FILE_NAME))) {
        String line;
        while ((line = reader.readLine()) != null) {
            //TODO: Für jede Zeile in der Datei soll nach dem Sonderzeichen
getrennt werden. Nutze die split(String s)-Methode.
            // Dein String Array beinhaltet einzelne Elemente eines Tasks. Füge
diese einem Konstruktor hinzu.
            // Speichere den neuen Task in einer Liste
        }
        System.out.println("Aufgaben erfolgreich geladen!");
    } catch (IOException e) {
        System.out.println("Fehler beim Laden der Aufgaben: " + e.getMessage());
    }
    return tasks;
}
```