Task: Exam Session Planning

Description

Develop a program in Java (or in C, depending on the chosen language) that allows university students to organize their exam session. The program must support full management of exams, allowing the user to add, view, modify, and mark as completed the exams previously registered.

Interaction will take place exclusively via the console; therefore, no graphical interface is required.

1. Exam Attributes

Each exam will be characterized by the following attributes:

- **ID**: unique numeric identifier, automatically generated by the program.
- **Exam name**: mandatory field.
- **Exam date**: mandatory field.
- **Difficulty**: perceived difficulty level, possible values: High, Medium, Low.
- **Status**: indicates the preparation progress with three possible values: Scheduled, In preparation, Completed.
- **Study progress**: percentage between 0% and 100%, representing the preparation level for the exam.
- **Grade**: score obtained, assigned only if the exam status is completed.

2. Program Features

The program must implement the following features:

- **Adding a new exam:** The user must enter the exam name, exam date, and perceived difficulty level. The program must validate the data entered by verifying that:
  - Mandatory fields are filled in.
  - The exam date is later than today's date.

  If the data is valid, the exam will be registered with the status "Scheduled" and an initial study progress of 0%.

- **Viewing the list of exams:** The program must show the list of exams to be taken, including for each one:
  - Exam name.
  - Exam date.
  - Difficulty level.
  - Exam status.
  - Study progress.
  - Number of days remaining until the exam date.

  Exams must be sorted by ascending exam date. If multiple exams are on the same date, the second sorting criterion will be difficulty level (High, Medium, Low).

- **Editing an exam:** The user will be able to update the information of an already registered exam. The program must ensure data validation also during updates, verifying that:
  - The exam date remains in the future.
  - The exam name is not empty.

- **Monitoring exam preparation:** The program must calculate the preparation status of each exam based on:
  - Exam date.
  - Today's date.
  - Study progress.

  The preparation status will be classified according to the following rules:
  - Let n be the number of days remaining until the exam.
  - Let PR be the remaining percentage of study needed.

  Calculate the value PR/n and assign the corresponding label to the exam:
  - if $PR/n < 8$, the status is "Ahead of schedule".
  - if $8 \leq PR/n \leq 12$, the status is "On track".
  - if $PR/n > 12$, the status is "Behind schedule".
  - if less than 4 days remain and progress is below 70%, the status will be "Catch up".

- **Completing an exam:** Once an exam is completed, the user must update its status to "Completed" and enter the grade obtained. The program must validate the grade, accepting only values between 0 and 30, with the possibility of assigning 31 for lode. After completion, the exam information can no longer be modified, but the exam will remain visible in the list as "Completed Exam".

Checklist for Evaluation

□ Adding a new exam with name, date, difficulty level, status, study progress. (15%)

□ Input validation (e.g., non-empty name and data). (5%)

□ Calculation of remaining days. (20%)

□ Displaying exams. (10%)

□ Editing exam data. (10%)

□ Evaluation of preparation status. (15%)

□ Completing an exam. (15%)

□ Input validation and non-modifiability of the exam. (10%)