```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This function receives a table of grades for different assignments with %
% the student IDs and their names and returns the same table, arranged    %
% alphabetically, with an extra column with the calculated final grades,  %
% called "Final"                                                          %
%                                                                         %
% Input:  An N × M matrix containing grades on the 7-step-scale given to  %
%         N students on M different assignments.                          %
% Output: A vector of length n containing the final grade for each of     %
%         the N students.                                                 %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function gradesFinal = computeFinalGrades(grades)

% Determine the number of rows and columns in the matrix
[N,M] = size(grades);
averageGrades = 1:N;

% If there is only one assignment the grade equals that assignment's grade
if M < 2
    averageGrades = reshape(grades , [1 , N]);
    gradesFinal = roundGrade(averageGrades);

% Otherwise we make a loop to calculate the average grades
else
    for i = 1:N
        % Assign a value a equal to the sorted row of grades
        a = sort(grades(i , :));

        % If all grades are larger than -3 the average is calculated the
        % usual way:
        if min(grades(i , :)) > -3

        % The average is the mean of the values in 'a' from 2 to last
        averageGrades(i) = mean(a( 2:length(a) ));

        % If one grade is -3 the average is -3
        elseif min(grades(i , :)) == -3
            averageGrades(i) = -3;
        end
    end

% Display the rounded grades with the use of the function 'roundGrade'
gradesFinal = roundGrade(averageGrades);
end
```