Tutorial 8

In this tutorial, we will write a simple program to perform some processing for students. The student information will be stored in an array of derived data types. There will be no more than 50 students per class. The main will call a subroutine to read student information (name and score) and another subroutine to set the student grades. Finally, the main will call a function to calculate the class average. The main will display the average. Routines for displaying the students are left as an exercise.

Understand the Problem

The main is expected to define the appropriate derived data type for student, declare some variables of that type and call the subroutines. The first subroutine will read student information, including a name (up to 60 characters) and score from the user. Names and scores should continue to be read until a blank name is entered. The score value must be between 0.0 and 100.0 (inclusive). For this simple example, we will re-prompt for incorrect data entry (until correct data is provided). The routine must return the count of students entered. The second subroutine sets the grades based on the following standard scale.

A	B	C	D	F
A>=90	80 - 89	70 - 79	60 - 69	<=59

When determining the final grade, the program should round up when appropriate. The main will call a function to calculate and return the average of the scores. Additionally, the main will display the final average.

Create the Algorithm

For this example main, part for the main includes a declaration, display header, call read time subroutine, and the call the time summation subroutine. The basic steps for the main include:

! define derived data type for student

! must include → name, id, grade

! declare variables

! includes → array for up to 50 students

! display initial header

! call subroutine to read student information

! call subroutine to set grades

! use function to calculate average of scores

! display average

The basic steps for the read student information subroutine include:

! subroutine header and appropriate declarations

! loop

! prompt for student name

! read name

! if name is empty, exit loop

! loop

! prompt for student score

! read score

! check score entered

! if [score is between 0.0 and 100.0, inclusive] exit

! display error message

! end loop

! update count of students

! place values in student array

! end loop

The basic steps for the set grades subroutine include:

! subroutine header and appropriate declarations

! loop

! read score / set grade

! ≥ 90 → A; 80 - 89 → B; 70 – 79 → C; 60 - 69 → D; ≤ 59 → F

! end loop

When determining the final grade, the nearest integer intrinsic function, nint(), can be used to perform the appropriate rounding. The basic steps for the calculate average score function include:

! function header and appropriate declarations

! loop

! sum scores

! end loop

! calculate and return average

For convenience, the steps are written as program comments.