*Intelligent Information Management*, **2010, 2, 569-585**
doi:10.4236/iim.2010.210066 Published Online October 2010 (http://www.SciRP.org/journal/iim)

# Adaptive System for Assigning Reliable Students' Letter Grades—A Computer Code

**Saeid A. Alghamdi**
*Department of Civil Engineering, College of Engineering Sciences*
*King Fahd University of Petroleum & Minerals, Dhahran, Kingdom of Saudi Arabia*
*E-mail*: *saghamdi@kfupm.edu.sa*
*Received March* 19, 2010; *revised June* 10, 2010; *accepted August* 12, 2010

## Abstract

The availability of automated evaluation methodologies that may reliably be used for determining students' scholastic performance through assigning letter grades are of *utmost* practical *importance* to educators, students, and do *invariably* have pivotal values to all stakeholders of the academic process. In particular, *educators* use letter grades as *quantification metrics* to monitor students' intellectual progress within a framework of clearly specified learning objectives of a course. *To students* grades may be used as predictive measures and motivating drives for success in a study field. However due to numerous objective and subjective variables that may by be accounted for in a methodological process of assigning students' grades, and since such a process is often tainted with personal philosophy and human psychology factors, it is essential that educators exercise extra care in maximizing positive account of all objective factors and minimizing negative ramifications of subjectively fuzzy factors. To this end, and in an attempt to make assigning students' grades more reliable for assessing true-level of mastering specified learning outcomes, this paper will: i) provide a literature *review on previous works* on the most common methods that have traditionally been in use for assigning students' grades, and a short account of the virtues and/or vices of such methods, and ii) *present a user-friendly computer code* that may be easily adapted for the purpose of assigning students' grades. This would relieve educators from the overwhelming concerns associated with mechanistic aspects of determining educational metrics, and it would allow them to have more time and focus to obtain *reliable assessments* of true-level of students' mastery of learning outcomes by accounting for all possible evaluation components.

**Keywords:** Reliable Students' Grades; Computer Code; Assigning Reliable Letter Grades

## 1. Introduction

Educators are entrusted to provide their best judgments on students' intellectual progress and achievements within a specified construct of learning objectives and outcomes for a particular course. Such judgments are however more often than not tainted with personal philosophy and human psychology factors, and despite the numerous educational instruments (including: homework assignments, quizzes, examinations, projects, etc.) that influence such judgments, they are ultimately reduced to assigning a letter grade that should have high degree of reliabilities and must be always defensible under circumstances of possible filed grievances [1-3].

The use of grades as quantification metrics [4-6] to monitor the students' intellectual progress have traditionally been through utilizing one of three forms: 1) criterion grading, or 2) normative grading, or 3) rubric grading. The most common methods and some of the less common methods of assigning students grades have been previously presented and discussed in the literature [7,8], and the various goals assigning grades are expected to achieve and the *reliability of these assignments* to achieve designated goals (as means for reward or penalty, or for communication to others, or for prediction of future performance) have also been discussed by numerous other researchers [9-11].

The *criterion-referenced grading process* [12] is based on a preset-grade-range criterion (as percentage of the total possible points) assigned for each letter grade (e.g. a

 

percentage score range of 76% to 79 % may be assigned for a letter grade of $C^+$). The method compares the performance of a student against preset criteria, it is highly dependent on designated learning outcomes for a course, and it is considered a more precise diagnostic-tool for the faculty (educator) to pinpoint to students particular strengths and weaknesses. This method may *however* lead to: 1) grade-assignment *biases* towards the upper end or the lower end of a grade-scale, and 2) improved collaboration amongst students as grades assigned for a given course would not be influenced by the individual performance of others in a students' group. The method is sometimes referred to as a *domain or mastery referencing procedure* to assign grades to students.

The *norm-referenced grading process* [13] is based on the premise that students' performances in a course represent a bell-shaped curve distribution that emulates a similar distribution of the learning outcomes designated for the course, and descriptive statistics are associated with this norm-based grading procedure. The curve of grades statistical distribution is centered on the mean score and the standard deviation is used as an index of scores' dispersion around the mean score [14]. The method compares the performance of a student against the performance of other classmates, and it is highly dependent on the course content, but it is considered less precise diagnostic-indicator for a student (learner) to pinpoint to him particular weaknesses on which he must concentrate to improve his standing in a course. This method may further lead to: 1) more wide-spreading of grade-assignment with less *biases* towards the upper end or the lower end of a grade-scale, and 2) increased level of competition amongst students as grades assigned to one student in a course would be influenced by the individual performance of others in a students' group. The method is sometimes referred to as a *norm* or a *comparative referencing procedure* to assign grades to students.

The *rubric-referenced* (generic) *grading process* [15] is used less in assigning students' grades and is based on values assigned to descriptive-scale indicators. The descriptive indicators for the letter grades (namely: the passing letter grades $A^+$, A, $B^+$, B, $C^+$, C, $D^+$, D, or the *unfavorable* failing letter grade F) are clearly spilled-out with values attached to them to be most suitable for the level of performance-achievements demonstrated by a student for specified major and minor goals of a course. For instance a 75-poin ts achievement by a student-performance out of the total possible 100 points may be assigned a letter grade C indicating that "*most major goals and minor goals of the course stated learning outcomes have been truly achieved*", while a 15-points achievement by a student-performance out of the total possible 100 points may be assigned a letter grade F indicating that "*few goals of the course stated learning outcomes have been barely achieved* ", *etc*.

## 2. Grading Systems Attributes, Anchoring, and Automation

For educators it is a unanimously agreed upon fact that there is no single method of evaluation (*i.e.* assigning students' grades) that would invariably prove effective in all formats of courses, and a method of evaluating a student's performance should be adapted to fit the course learning objectives and expected outcomes [16,17]. It is however expected that once a method of assigning students' grades is selected and is judged to be most suitable for the purpose it should be characterized by key functional attributes including:1) face and content validities, and 2) reliability, and realistic expectations. To these ends, *face validity*, on one hand, of a grading procedure must have clear and suitable metrics to measure the degree of relevance of the evaluation process to the course objectives, and such relevance must be transparent to the students as well. On the other hand, *content validity*, should provide suitable analysis (or design) case-studies so that the method of evaluation conform to the course objectives. An evaluation method (for assigning students' grades) will further be termed *reliable* if it would *invariably* produce, with little variations, the same results (students' grades) for the same students. But since it is well-known by educators (and students for this matter!) that under general circumstances a grade assigned to a student is *not an absolute measure of his performance* and *true* level of achievement in a course, and is often an artifact of the educator and/or the competencies (e.g. for the criterion-referenced grading) of the classmates enrolled in a course, every efforts should be exercised to account for all evaluation factors in an objective format that would endeavor to exclude subjective factors so that grades assigned would be true measure of students' achievements. The reliability of an evaluation procedure (to assign students' grades) is also constrained *not only* by the *realistic expectations* of the tools used for the evaluation *but also by* the individual being evaluated. The realistic expectations would define the number and type of evaluation parameters that may be developed and administered by the educator within a most-suitable time-frame for a course, and should provide suitable considerations to the fact that a student is also enrolled in other courses.

Analysis of the two most commonly used grading systems (namely: the normative and criterion procedures) further indicates that while the norm-referencing may be the most suitable procedure for assigning students' grades, it requires using the unsatisfactory method of

 

grading on a class-curve [15,18,19]. Therefore and to overcome the vices of either one of the two grading assignment systems it has been recommended [20] to use various anchor measures that would enable utilizing the virtues of *norm-based grading* without introducing the vices of *class-curve-based grading*.

It is therefore conspicuous that due to the numerous variables that may enter into a process of evaluating students through suitably selected quantification metrics (using a norm-referencing or a domain-criteria-referencing), this process may *invariably* suffer from a reliability-problem [2,21-24]. This may further result in un-intended negative ramifications on the learning-educational process and may even set the whole evaluative process in a true reduced-reliability dilemma. This process is, therefore, often categorized by academicians as truly one of the *least favorable* and *highly daunting activities* amongst the myriad of other activities undertaken by an educator. As such there is a real need for an educator to account for all possible parameters that would influence the evaluation process and properly weigh all relevant factors to increase the reliability of the procedure [14]. To this end the literature includes scores of disparate previous attempts that have been documented for automating the mechanistic aspects of the grading assignment process [25-26]. Therefore, the development and adapting the uses of an automated grading system would certainly be a relieve for educators from the prohibitive-drudgery of overwhelming numeric processing that is often typical within the grading-assignment process for medium-to-large size classes or for multi-section courses with unified grading-standards. It is believed that if the *mechanistic aspects* of the evaluative process are included within an automated and user-friendly procedure, educators' efforts would then be more meaningfully focused on ensuring more reliable evaluation of students' mastery of the designated learning outcomes in a given course. The following sections of this paper will present an adaptive [27] automated grading system designed as a FORTRAN Computer Code. The code has been developed to automate the mechanistic aspects of the evaluative process and has been further successfully tested and has been proven to be a reliable and practical tool for assigning students' letter grades.

## 3. The Computer Code Taxonomy and Attributes

Based on the premise that an evaluative process (e.g. assigning students' grades) would be highly more reliable if the mechanistic aspects of determining evaluation metrics are automated, an algorithmic process for assigning letter grades to students has bee automated through the development of a FORTRAN computer code "*classrecord.for*". The overall taxonomy of the coding process is summarized in the flowchart shown in **Figure 1**. The code has several *user-friendly features* that make it *easily adaptable* to either a normative-referencing procedure or to a domain-referencing procedure. Complete listing of the code is given in Appendix I-*a*, and further clarifying details of the code structure are also available elsewhere [28]. Specifically, the coding procedure presented herein would enable the educator to:

1) Devise several *input file categories* that may include rosters for class attendances, student homeworks, students quizzes, student scores on major examinations, and students' scores on the final examination (*namely*: *attendance.csv*; *homework.csv*; *quiz.csv*; *major.csv*; and *final.csv*, respectively).

2) Compute the *evaluative metrics* for a course by preparing weighted scores for each input file category entered for each student, compute pre-final (sub-total) weighted scores for all input categories, and determine weighted *grandtotal* scores for all input categories.

3) Use the *grandtotal* scores (course metrics) obtained based on considerations of all input categories to assign students' grades by adapting the computer code to use a normative-referencing procedure or a domain-referencing procedure.

Compared to the method of using an *ad hoc* spread-sheet calculation, which *invariably* requires frequent redesigns of the spread-sheet [20], this code has the following main features:

1) It has simple text-input prepared in free-format;

2) It requires no prior programming knowledge on the part of the user;

3) It is highly valuable and is easily adaptable for assigning students' grade for multi-sections courses and for providing detailed individual reports for each section [29];

4) The computational mechanics for determining the *cut-off lines* (normally implemented in a normative-referencing process) is based on the class weighted average and the class(s) standard deviation such that with N = number of students, and $x_i$ = a student score, the code module "grade Students" would compute the weighted course mean $\bar{x}$ and the standard deviation $\sigma$ (as a measure of dispersion of scores around the mean) such that with

$$\bar{x} = \frac{\sum_{i=1}^{N} x_i}{N} \tag{1}$$

$$\sigma = \frac{1}{N}\sqrt{N\sum_{i=1}^{N}\left(x_i^{\,2}\right) - \sum_{i=1}^{N}\left(x_i\right)^2} \tag{2}$$

 

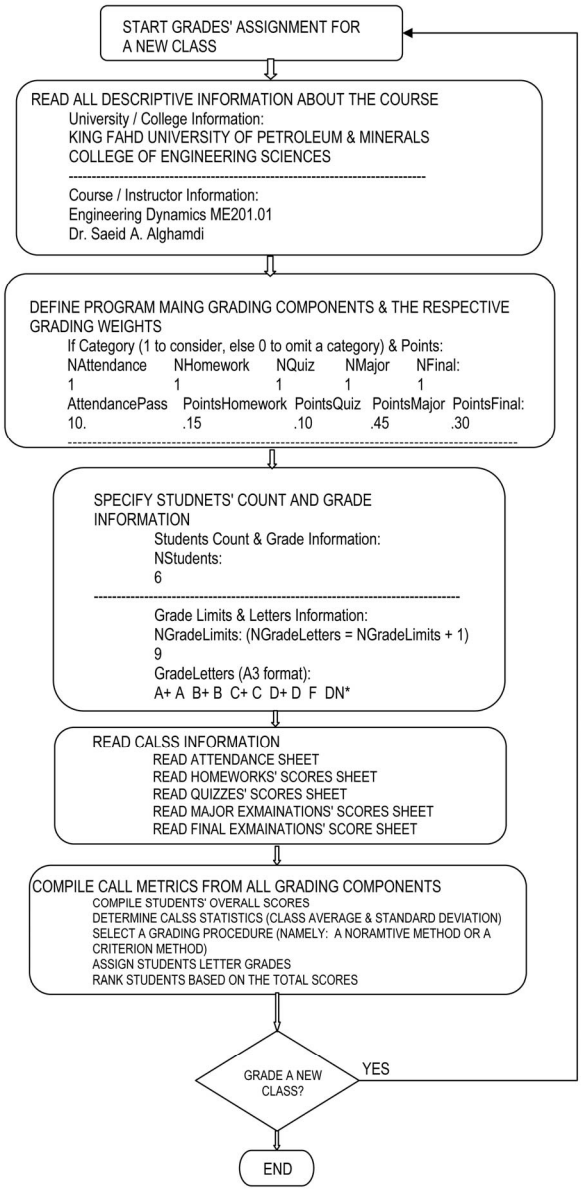


**Figure 1. Main steps of the grades' assignment computer code.**

grades' cut-off lines (defined as $c_{Li}$) for the lower bounds of the ranges for the eight passing grades (namely: for the letter grade $A^+$, A, $B^+$, B, $C^+$, C, $D^+$, D) are determined generically. For this purpose the following generic formula for determining letter grades above the mean, for mean grades, and for grades below the mean, are *respectively* given as

$$c_{Li} = \frac{1}{2}\left[2\bar{x} - 10(i-4) + \sigma\right] \quad \text{for } i = 1,\ 2,\ 3,\ 4 \qquad (3)$$

$$c_{Li} = \bar{x} \qquad \text{for } i = 5 \qquad (4)$$

$$c_{Li} = \frac{1}{2}\left[2\bar{x} - 10(i-6) - \sigma\right] \quad \text{for } i = 6,\ 7,\ 8 \qquad (5)$$

in which: the index $i$ values of 1, 5, and 8, correspond to lower bounds of the distinction grade $A^+$, the above average grade $C^+$, and the just barely passing grading D, and so on, respectively; the output of the code presents the assigned letter grades in detailed and simple formats that are easily interpretable (as shown for example in Appendix I-*a* and I-*b*).

## 4. Code Utilization and Discussions

To demonstrate the ease this code would provide to educators, that are frequently involved in evaluating students' performance in given course for assigning letter grades, the *utilization* of the code is shown herein for a class size of only seven students. The *output obtained from this code is further compared* to the output obtained from an *ad hoc* Excel spread-sheet that has been prepared to determine the letter grades for the same group of students.

Sample input-control data and output-summary details of the code outputs obtained for a class-size of just eleven students given in Appendix I-*b*. The code has also been utilized in a comparative study to determine students' letter grades for another class of small size with only seven students and the students' letter grades assigned by the code are summarized in Appendix II-*a*, and in Appendix II-*b*, and the comparison is shown with reference to the letter grades reported by an *ad hoc* spread-sheet calculations (shown in Appendix II-*c*). The comparison of the results (*i.e.* the letter grades assigned to the students in the class) obtained from the two automated procedures *clearly show identical* results and as this has been repeatedly noted by the author on several occasions it should represent a strong *empirical evidence* of the numerous *advantages* and *robustness* of the code presented herein. *These advantages would of course be more indisputable* particularly when reporting letter grades for classes with large students' population.

This shows that while the results obtained are identical the code presented herein has the advantages of being *more general* and *user-friendly* as the user is *not* presumed to have a prior programming knowledge to use the code and all the inputs required are entered merely as separate text sheets that can be easily prepared with free format to be read by the code.

In particular, the code has clear and distinct advantages for the grading assignment process of multi-section courses as *near-fair* and *less-disparate grade distributions* are made *more likely* possible and attainable. The code would make this easily achieved as educators would be relieved from the overwhelming concerns that

 

frequently accompany the mechanistic aspects (to determine numerous evaluative metrics that may be included for a specific course) of an academic evaluation process. The utilization *of the code would certainly enable educators have more time to focus on getting rational assessment of students' achievements* within the framework of designated learning outcomes, and would certainly help minimize the number of possible grievances that may be filed by some students for different *subjective* justifications.

## 5. Closure

It is an undeniable reality among educators and academicians that a process of assigning students' grades is invariably influenced by *numerous objective and subjective variables*, and *quite often* it is tainted with the *subjective* influences of *personal philosophy* and *human psychology* factors. Based on this, and due to the vital importance of using grades' as *quantification metrics* to monitor students' intellectual progress within the framework of clearly specified learning objectives of a course, and *since most students* see grades as predictive measures and motivating drives for success in a study field, it is essential that educators exercise extra care in maximizing the positive accounts of all objective factors and minimizing the negative ramifications of subjectively fuzzy factors. This can be easily achieved only if the mechanistic aspects of determining the students' performance metrics are taken care of by an automated methodology.

For this purpose, the FORTRAN Computer Code presented in this paper that has been tested by the author on several occasions to report students' letter grades and the results obtained have provided strong empirical evidence of the *robustness of the code as an instrument for assigning students' grades* with relative ease. The code is capable of accounting for numerous parameters that may be deemed essential for the process of academic evaluation. Furthermore and compared to an *ad hoc* grading spread-sheet, the *user-friendly features of the code make it easily* adaptable for the purpose of *assigning more reliable students' grades* that reflect the true-level of students' mastery of learning outcomes when all possible evaluation components are accounted for.

## 6. Acknowledgements

Initial versions of the program described herein were developed and tested using the computational facilities made available by King Fahd University of Petroleum and Minerals (KFUPM; Dhahran, KSA). The support provided by the staff of the Academic Computing Services (ACS) of the ITC-KFUPM is acknowledged with appreciations.

## 7. References

[1] E. J. Pedhazur and L. P. Schmelkin, "Measurement, Design and Analysis: An Integrated Approach," Lawrence, Erlbaum, Hillsdale, 1991.

[2] B. Thompson and T. Vacha-Haase, "Psychometrics is Datametrics: The Test is Not Reliable," *Educational and Psychological Measurement*, Vol. 60, No. 2, 2000, pp. 174-195.

[3] L. M. Aleamoni, "Why is Grading Difficult? Note to the Faculty," University of Arizona, Office of Instructional Research and Development, Tucson, 1978.

[4] D. A. Frisbie, N. A. Diamond and J. C. Ory, "Assigning Course Grades," Office of Instructional Resources, University of Illinois, Urbana, 1979.

[5] W. Hornby, "Assessing Using Grade-Related Criteria: A Single Currency for Universities?" *Assessment & Evaluation in Higher Education*, Vol. 28, No. 4, 2003, pp. 435-454.

[6] B. Cheang, A. Kurnia, A. Lim and W.-C. Oon, "On Automated Grading of Programming Assignment in an Academic Institution," *Computers and Education*, Vol. 41, No. 2, 2003, pp. 121-131.

[7] W. J. McKeachie, "College Grades: A Rational and Mild Defense," *AAUP Bulletin*, Vol. 320, 1976.

[8] J. S. Terwilliger, "Classroom Standard Setting and Grading Practices," *Educational Measurement*: *Issues and Practices*, Vol. 8, No. 2, 1989, pp. 15-19.

[9] G. R. Johnson, "Taking Teaching Seriously: A Faculty Handbook," Texas A & M University, Center for Teaching Excellence, College Station, TX, 1988.

[10] J. E. Stice, "Grades and Test Scores: Do They Predict Adult Achievement?" *Engineering Education*, No. 390, 1979.

[11] K. E. Eble, "The Craft of Teaching," 2nd Edition, Jossey-Bass, San Francisco, 1988.

[12] W. J. McKeachie, "The A B C's of Assigning Grades-Teaching Tips: Strategies, Research, and Theory for College and University Teachers," In: W. J. McKeachie Ed., 9th Edition, D. C. Heath, Lexington, 1994.

[13] J. S. Terwilliger, "Assigning Grades to Students," IL: Scott, Forsman, Glenview, 1971.

[14] M. J. Evans and J. S. Rosenthal, "Probability and Statistics—The Science of Uncertainty," W. H. Freeman, 2004, p. 638.

[15] K. A. Smith, "Grading and Distributive Justice," *Proceedings ASEE/IEEE Frontiers in Education Conference*, *IEEE*, New York, 1986, p. 421.

[16] R. L. Ebel, "Essentials of Educational Measurement," 3rd Edition, , Prentice Hall, Englewood Cliffs, 1979.

[17] C. Adelman, "Performance and Judgment: Essay on Principles and Practice in the Assessment of College Student

 

Learning," U.S. Department of Education, Washington, D. C., 1988.

[18] R. M. W. Travers, "Appraisal of the Teaching of the College Faculty," *Journal of Higher Education*, Vol. 21, No. 41, 1950.

[19] R. L. Thornedike, "Applied Psychometrics," Houghton Mifflin, Boston, 1982.

[20] G. S. Hanna and W. E. Cashin, "Matching Instruction Objectives, Subject Matter, Tests, and Score Interpretations. (IDEA Paper 18), " Kansas State University, Center for Faculty Evaluation and Development, Manhattan, 1987.

[21] P. Elbow, "Embracing Contraries: Explorations in Learning and Teaching," Oxford University Press, New York, 1986.

[22] J. D. Krumboltz and C. J. Yeh, "Competitive Grading Sabotage Good Teaching," *Phi Delta Kappan*, Vol. 78, No. 4, 1996.

[23] T. R. Guskey, "Grading Policies That Work Against Standards and How to Fix Them," *NASSP Bulletin*, Vol. 84, No. 620, 2000, pp. 20-29.

[24] J. M. Graham, "Congeneric and (Essentially) Tau-Equivalent Estimates of Score Reliability—What They are and How to Use Them," *Educational and Psychological Measurement*, Vol. 66, No. 6, 2006, pp. 930-944.

[25] I. Krunger, "A Computer Code to Assign Student Grades," *Educational and Psychological Measurement*, No. 34, 1974, pp. 179-180.

[26] Microsoft Corporation, "Managing grade with Excel 2002," 2002. http://www.microsoft.com/education/ ManagingGrades.mspx

[27] E. Wilson, C. L. Karr and L. M. Freeman, "Flexible, Adaptive, Automatic Fuzzy-Based Grade Assigning System," *Proceedings Fuzzy Information Processing Society—NAFIPS, Conference of the North American*, Vol. 60, 1998, pp. 174-195.

[28] S. A. Alghamdi, "Towards Automated and Reliable Evaluation of Students' Scholastic Performance: A FORTARN Computer Code," Unpublished report, CE-Department, King Fahd University of Petroleum & Minerals, Dhahran, KSA, 2008.

[29] S. R. Cheshier, "Assigning Grades More Fairly," *Engineering Education*, Vol. 343, 1975, pp. 4-8.

 

## Appendix I-*a*: Listing of the code

```
                The Code (Assigning Reliable Students' Letter Grades)
      program classrecord
c
c.....................................................................
c                  ***  classrecord.for  ***                          .
c A user-friendly Fortran Code to handle the multi-faceted process of   .
c assessing students' academic performance and assigning reliable letter.
c grades for a single group or multiple groups of students.            .
c.....................................................................
c                                                                      .
c   Accepts input file categories:                                     .
c    * attendance.csv : roster of class attendances                    .
c    * homework.csv   : roster of student homeworks                    .
c    * quiz.csv       : roster of student quizes                       .
c    * major.csv      : roster of student major exams                  .
c    * final.csv      : roster of student final exam                   .
c                                                                      .
c  And for each student prepares:                                      .
c    * weighted scores for each input file category                    .
c    * subtotal weighted score for all  categories, except the final   .
c    * weighted grandTotal of all categories                           .
c    * performance rank relative to the overall class                  .
c    * selection of grade according to user criteria                   .
c.....................................................................
c
c                                                                      .
c    Reference:class.record fortran program  initially Developed by the .
c    the author in June 1989.                                          .
c.....................................................................
c
        parameter(mstudents=100,mweeks=100,mGradeLimits=20)

      implicit real*8 (a-h,o-z)
      dimension gradeLimits(mGradeLimits)
        character*3 gradeLetters(mGradeLimits+1)
        character*80 blank,university,college,course,instructor

        dimension
        .attendance(mstudents,mweeks),averageAttendance(mstudents),
        .homework(mstudents,mweeks),averageHomework(mstudents),
        .quiz(mstudents,mweeks),averageQuiz(mstudents),
        .xmajor(mstudents,mweeks),averageMajor(mstudents),
        .final(mstudents,mweeks),averageFinal(mstudents)


        dimension
        .attendanceSum(mstudents),ifAttendancePass(mstudents)

        dimension IDStudents(mStudents),studentsSum(mStudents),
        .iStudentsRank(mStudents),iStudentsGrade(mStudents)
      dimension subTotal(mStudents)


      open(1,file='classrecordd.txt',status='old',form='formatted')
      open(2,file='classrecorde.txt',status='unknown',form='formatted')
      open(3,file='classrecordo.txt',status='unknown',form='formatted')

c      open(11,file='attendance.csv',status='old',form='formatted')
c      open(12,file='homework.csv',status='old',form='formatted')
c      open(13,file='quiz.csv',status='old',form='formatted')
c      open(14,file='major.csv',status='old',form='formatted')
c      open(15,file='final.csv',status='old',form='formatted')

c      open(11,file='C:\Documents and Settings\Administrator\Desktop\grad
c        .ing\me20189\attendance.txt',status='old',form='formatted')
c      open(12,file='C:\Documents and Settings\Administrator\Desktop\grad
c        .ing\me20189\homework.txt',status='old',form='formatted')
c      open(13,file='C:\Documents and Settings\Administrator\Desktop\grad
c        .ing\me20189\quiz.txt',status='old',form='formatted')
c      open(14,file='C:\Documents and Settings\Administrator\Desktop\grad
c        .ing\me20189\major.txt',status='old',form='formatted')
c      open(15,file='C:\Documents and Settings\Administrator\Desktop\grad
c        .ing\me20189\final.txt',status='old',form='formatted')
      open(11,file='attendance.txt',status='old',form='formatted')
      open(12,file='homework.txt',status='old',form='formatted')
      open(13,file='quiz.txt',status='old',form='formatted')
      open(14,file='major.txt',status='old',form='formatted')
      open(15,file='final.txt',status='old',form='formatted')
```

 

The Code (Assigning Reliable Students' Letter Grades)

```
        blank=''

        call getInformation(university,college,course,instructor,
        .nAttendance,nHomework,nQuiz,nMajor,nFinal,attendancePass,
      .pointsAttendance,pointsHomework,pointsQuiz,pointsMajor,
      .pointsFinal,nStudents,nGradeLimits,gradeLetters,mGradeLimits
      .)

c     retrieve and process categories
c       first initialize student grand total
      do is=1,mstudents
          studentsSum(is)=0.
        enddo

c ** deal with attendance:
      if(nAttendance.eq.1) then
         itape=11
         call getCategory(itape,attendance,nstudents,nAttendance,
      .mstudents,mweeks,IDStudents)
         call passCategory(attendance,nstudents,nAttendance,
        .attendanceSum,ifAttendancePass,attendancePass,mstudents,mweeks,
      .nAttendancePass)

        do is=1,nstudents
          sum=0.0
          do jc=1,nAttendance
            sum=sum+attendance(is,jc)
          enddo
          averageAttendance(is)=PointsAttendance*(sum-attendancePass)/
      .  (nAttendance-attendancePass)*100
       write(2,'(i15,f15.3)') is,averageAttendance(is)
        enddo
          call addCategory(studentsSum,nstudents,averageAttendance,
      . mstudents)

        endif


c ** deal with homeworks:
      if(nHomework.eq.1) then
         itape=12
         call getCategory(itape,homework,nstudents,nHomework,
      .mstudents,mweeks,IDStudents)
         call averageCategory(homework,nstudents,nHomework,
        .averageHomework,pointsHomework,mstudents,mweeks)
         call addCategory(studentsSum,nstudents,averageHomework,
        .mstudents)
        endif

c ** deal with quizes:
      if(nQuiz.eq.1) then
         itape=13
         call getCategory(itape,quiz,nstudents,nQuiz,
      .mstudents,mweeks,IDStudents)
         call averageCategory(quiz,nstudents,nQuiz,
        .averageQuiz,pointsQuiz,mstudents,mweeks)
         call addCategory(studentsSum,nstudents,averageQuiz,
        .mstudents)
        endif

c ** deal with majors:
      if(nMajor.eq.1) then
         itape=14
         call getCategory(itape,xmajor,nstudents,nMajor,
        .mstudents,mweeks,IDStudents)
         call averageCategory(xmajor,nstudents,nMajor,
        .averageMajor,pointsMajor,mstudents,mweeks)
         call addCategory(studentsSum,nstudents,averageMajor,
        .mstudents)
        endif

c ** computed the subTotal prior to the final
      do is=1,nStudents
         subTotal(is)=studentsSum(is)
        enddo

c ** deal with final(s):
```

 

The Code (Assigning Reliable Students' Letter Grades)

```
      if(nFinal.eq.1) then
         itape=15
         call getCategory(itape,final,nstudents,nFinal,
        .mstudents,mweeks,IDStudents)
         call averageCategory(final,nstudents,nFinal,
        .averageFinal,pointsFinal,mstudents,mweeks)
         call addCategory(studentsSum,nstudents,averageFinal,
        .mstudents)
        endif

c ** deal with failed attendance:
      do is = 1, nstudents
         ifPass=ifAttendancePass(is)
         if(ifPass.eq.0) then  !flag -1 for grade DN asignement
           averageHomework(is)= -1
           averageQuiz(is)    = -1
           averageMajor(is)   = -1
           subTotal(is)       = -1
           averageFinal(is)   = -1
           studentsSum(is)    = -1
         endif
      enddo

c ** rank students, 1 to nStudents in descending order of studentsSum
      call rankStudents(studentsSum,nStudents,iStudentsRank,mStudents)

c ** synchronize studentRank with failed attendance:
      do is = 1, nstudents
         ifPass=ifAttendancePass(is)
         if(ifPass.eq.0) then  !flag -1 for grade DN asignement
           iStudentsRank(is) = -1
         endif
      enddo

c ** gradeCode students according to gradeLimits selections
        call gradeStudents(studentsSum,nStudents,gradeLimits,
        .nGradeLimits,iStudentsGrade)

c     ------------------------------------------------------------
c     echo some results
        write(2,'(/7a15)')'iStudent','IDStudents','studentsSum',
        .'iStudentsRank','iStudentsGrade','gradeLetter'
        do is=1,nstudents
          write(2,'(2i15,f15.2,2i15,a15)')is,IDStudents(is),
        .studentsSum(is),iStudentsRank(is),iStudentsGrade(is),
     .gradeLetters(iStudentsGrade(is))
        enddo

c     ------------------------------------------------------------
c     ouput results
        write(3,'(a)')university
        write(3,'(a)')college
        write(3,'(a)')' '
        write(3,'(a)')course
        write(3,'(a)')instructor
        write(3,'(a)')' '
        write(3,'(125(1h-))')

        write(3,'(a5,10a12)') 'SN','STUDENT','ATTEND.','HWORK','QUIZES',
        .'MAJORS','SUBTOTAL','FINAL','TOTAL','RANK','GRADE'
        write(3,'(125(1h-))')

        do is=1,nStudents
          write(3,'(i5,i12,7f12.2,i12,a12)') is,
        .IDStudents(is),averageAttendance(is),averageHomework(is),
        .averageQuiz(is),averageMajor(is),subTotal(is),
        .averageFinal(is),studentsSum(is),iStudentsRank(is),
     .gradeLetters(iStudentsGrade(is))

        if(mod(is,5).eq.0) write(3,'(125(1h-))')
        enddo
      if(mod(nStudents,5).ne.0) write(3,'(125(1h-))')

        write(3,'(a)')

      stop
        end
c---------------------------------------------------------------------
```

 

```
                    The Code (Assigning Reliable Students' Letter Grades)
      subroutine gradeStudents(studentsSum,nStudents,gradeLimits,
      .nGradeLimits,iStudentsGrade)

      implicit real*8 (a-h,o-z)
        dimension studentsSum(nStudents),gradeLimits(nGradeLimits),
        .iStudentsGrade(nStudents)
        average=0
        icount=0
      do is=1,nStudents
        if(studentsSum(is) > 0) then
         average=average+ studentsSum(is)
         icount=icount+1
        endif
      enddo
        average=average/icount
        var=0
        do is=1,nStudents
        if(studentsSum(is) > 0)  then
        var=var+(studentsSum(is)-average)*(studentsSum(is)-average)
        endif
        enddo
        var=var/icount
        std=dsqrt(var)
c       do il=1,nGradeLimits-1
c       gradeLimits(il) =average+(2-(il-1)*0.5)*std
c      enddo
        gradeLimits(1) =average+0.5*std+15
        gradeLimits(2) =average+0.5*std+10
        gradeLimits(3) =average+0.5*std+5
        gradeLimits(4) =average+0.5*std
        gradeLimits(5) =average
        gradeLimits(6) =average-0.5*std
        gradeLimits(7) =average-0.5*std-5
        gradeLimits(8) =average-0.5*std-10
        gradeLimits(9)=0
        write(2,*)'-------------------------------------'
        write(2,*)'average= ',average
        write(2,*)'standard deviation= ', std
        write(2,*)'-------------------------------------'
      do is=1,nStudents
          score=studentsSum(is)
          igrade=0
          do ig=1,nGradeLimits
            grade=gradeLimits(ig)
            if(score.ge.grade .and. igrade.eq.0) igrade=ig
          enddo
          if(igrade.eq.0) igrade=nGradeLimits+1
          iStudentsGrade(is)=igrade
        enddo
      return
        end

c-----------------------------------------------------------------------
      subroutine rankStudents(glist,nlist,lrank,mStudents)

      implicit real*8 (a-h,o-z)
      dimension glist(nlist),lrank(mStudents)
      dimension wlist(mStudents),llist(mStudents)


c     create a working array and use -ve glist() to
c     achieve descending order ranking of glist()
      do ilist = 1,nlist
        wlist(ilist) = -glist(ilist)
        llist(ilist)=ilist
      enddo

c     sort llist() in ascending order of wlist()
      do ilist = 1,nlist
        si = wlist(ilist)
        imin = llist(ilist)
        do jlist = ilist,nlist
          sj = wlist(jlist)
          jmin = llist(jlist)
          if (sj.lt.si) then
            wlist(ilist) = sj
            wlist(jlist) = si
            si = sj
```

 

```
                    The Code (Assigning Reliable Students' Letter Grades)
              llist(ilist) = jmin
                llist(jlist) = imin
              imin = jmin
            endif
          enddo
        enddo

c       retrieve inorder ranking of in llist() entries
        do irank=1,nlist
            ilist=llist(irank)
            lrank(ilist)=irank
          enddo

        return
          end

c-------------------------------------------------------------------------
          subroutine addCategory(studentsSum,nstudents,categoryAverage,
          .mstudents)

        implicit real*8 (a-h,o-z)
          dimension studentsSum(mstudents),categoryAverage(mstudents)

          do is=1,nstudents
           studentsSum(is)=studentsSum(is) + categoryAverage(is)
          enddo

          return
          end
c-------------------------------------------------------------------------
          subroutine passCategory(category,nstudents,nCategory,
          .categorySum,ifCategoryPass,categoryPass,mstudents,mweeks,
       .nCategoryPass)

        implicit real*8 (a-h,o-z)
          dimension category(mstudents,mweeks),categorySum(mstudents)
          dimension ifCategoryPass(mstudents)

          write(2,'(/4a15)')'istudent','nCategory','categoryPass',
          .'catSum','ifCatPass'

          nCategoryPass=0
          do is=1,nstudents
            sum=0.0
            do jc=1,nCategory
              sum=sum+category(is,jc)
          enddo
            categorySum(is)=sum
            if(sum .ge. categoryPass) then
              ifCategoryPass(is)=1
            nCategoryPass=nCategoryPass+1
            else
              ifcategoryPass(is)=0
            endif
            write(2,'(2i15,2f15.3,i15)') is,nCategory,categoryPass,
       .    sum,ifCategoryPass(is)
          enddo

          return
          end

c-------------------------------------------------------------------------
          subroutine averageCategory(category,nstudents,nCategory,
          .categoryAverage,pointsCategory,mstudents,mweeks)

        implicit real*8 (a-h,o-z)
          dimension category(mstudents,mweeks),categoryAverage(mstudents)

          write(2,'(/4a15)')'istudent','nCategory','sum','catAverage'

          do is=1,nstudents
            sum=0.0
            do jc=1,nCategory
              sum=sum+category(is,jc)
            enddo
            categoryAverage(is)=sum*pointsCategory/nCategory
            write(2,'(2i15,2f15.3)') is,nCategory,sum,categoryAverage(is)
          enddo
```

 

The Code (Assigning Reliable Students' Letter Grades)

```
        return
        end

c-------------------------------------------------------------------------
        subroutine getCategory(itape,category,nStudents,nCategory,
        .mstudents,mweeks,IDStudents)

      implicit real*8 (a-h,o-z)
        dimension category(mstudents,mweeks),IDStudents(mstudents)
        character*80 blank

        read(itape,*) kStudents,nCategory
        write(2,'(/2i10)') kStudents, nCategory
c       write(2,'(3a10)')'jsnid','stuID','catValues...'

c        stop if error in count of students across input files
        if(nStudents.ne.kStudents) then
          call stopCategory(nStudents,kStudents)
        endif

        read(itape,'(a)') blank
        write(2,'(a)') blank

        do is=1,nStudents
          read(itape,*) jsnid,IDStudents(jsnid),
        .  (category(is,jc),jc=1,nCategory)
          write(2,'(2i10,50f10.2)') jsnid, IDStudents(jsnid),
        .  (category(is,jc),jc=1,nCategory)
      enddo

        return
        end

c-------------------------------------------------------------------------
        subroutine stopCategory(nStudents,kStudents)
      implicit real*8 (a-h,o-z)

         write(*,'(a)')    ' ** Error students count'
         write(*,'(a,i5)') ' Student count in *.dat file  = ',nStudents
         write(*,'(a,i5)') ' Student count in *.cvs file  = ',kStudents
         write(*,'(a)')    ' program execution stops ...'

         write(2,'(a)')    ' ** Error students count'
         write(2,'(a,i5)') ' Student count in *.dat file  = ',nStudents
         write(2,'(a,i5)') ' Student count in *.cvs file  = ',kStudents
         write(2,'(a)')    ' program execution stops ...'

         stop
        return
        end


c-------------------------------------------------------------------------
        subroutine getInformation(university,college,course,instructor,
        .nAttendance,nHomework,nQuiz,nMajor,nFinal,attendancePass,
     .pointsAttendance,pointsHomework,pointsQuiz,pointsMajor,
     .pointsFinal,nStudents,nGradeLimits,gradeLetters,mGradeLimits
     .)

      implicit real*8 (a-h,o-z)
c      dimension gradeLimits(mGradeLimits)
        character*3 gradeLetters(mGradeLimits+1)
        character*80 blank,university,college,course,instructor

        read(1,'(a)') blank
        write(2,'(a)') blank
        read(1,'(a)') university
        write(2,'(a)') university
        read(1,'(a)') college
        write(2,'(a)') college

        read(1,'(a)') blank
        write(2,'(a)') blank

        read(1,'(a)') blank
        write(2,'(a)') blank
        read(1,'(a)') course
```

 

The Code (Assigning Reliable Students' Letter Grades)

```
      write(2,'(a)') course
      read(1,'(a)') instructor
      write(2,'(a)') instructor

      read(1,'(a)') blank
      write(2,'(a)') blank
      read(1,'(a)') blank
      write(2,'(a)') blank

      read(1,'(a)') blank
      write(2,'(a)') blank
      read(1,*) nAttendance,nHomework,nQuiz,nMajor,nFinal
      write(2,'(5i5)') nAttendance,nHomework,nQuiz,nMajor,nFinal

      read(1,'(a)') blank
      write(2,'(a)') blank
      read(1,*) attendancePass,pointsAttendance,pointsHomework,
     .pointsQuiz,pointsMajor,pointsFinal
      write(2,'(6f7.3)') attendancePass,pointsAttendance,pointsHomework,
     .pointsQuiz,pointsMajor,pointsFinal

      read(1,'(a)') blank
      write(2,'(a)') blank
      read(1,'(a)') blank
      write(2,'(a)') blank

      read(1,'(a)') blank
      write(2,'(a)') blank
      read(1,*) nStudents
      write(2,'(i5)') nStudents

      read(1,'(a)') blank
      write(2,'(a)') blank
      read(1,'(a)') blank
      write(2,'(a)') blank

      read(1,'(a)') blank
      write(2,'(a)') blank
      read(1,*) nGradeLimits
      write(2,'(i5)') nGradeLimits
c
c     read(1,'(a)') blank
c     write(2,'(a)') blank
c     read(1,*) (gradeLimits(i),i=1,nGradeLimits)
c     write(2,'(20f7.3)') (gradeLimits(i),i=1,nGradeLimits)
c
      read(1,'(a)') blank
      write(2,'(a)') blank
      read(1,'(20a3)') (gradeLetters(i),i=1,nGradeLimits+1)
      write(2,'(20a3)') (gradeLetters(i),i=1,nGradeLimits+1)
      return
      end
```

 

## Appendix I-*b*: Sample input-control data and output-summary details of the code

classrecord_Control data

```
University / College Information:
SCHOOL NAME
COLLEGE/DEPARTMENT NAME
-----------------------------------------------------------------
Course / Instructor Information:
COURSE NAME: COURSE & SECTION IDENTIFICATION
INSTRUCTOR's NAME
-----------------------------------------------------------------
If Category (1 to consider, else 0 to omit a category) & Points:
NAttendance       NHomework       NQuiz        NMajor       NFinal:
1                 1               1            1            1
AttendancePass    PointsHomework  PointsQuiz   PointsMajor  PointsFinal:
10.               .10             .05          .45          .35
-----------------------------------------------------------------
Students Count & Grade Information:
NStudents:
11
-----------------------------------------------------------------
Grade Limits & Letters Information:
NGradeLimits: (NGradeLetters = NGradeLimits + 1)
9
GradeLetters (A3 format):
A+ A  B+ B  C+ C  D+ D  F  DN*
-----------------------EOF-------------------------------------
```

homeworks

```
11,12
-----------------
1,223512  ,62,77,95  ,72,92 ,64,83,90,88,87,95,94
2,224012  ,70,69,80  ,58,84 ,61,71,95,60,97,93,92
3,233767  ,14,62,45  ,39,52 ,59,32,43,55,93,0,0
4,234243  ,42,90,70  ,62,56 ,56,66,0,0,93,0,0
5,235141  ,0,0,80,42 ,66,54 ,68,45,0,0,0,0
6,244072  ,42,39,33  ,0,86  ,54,32,83,78,80,88,0
7,244262  ,0,0,0,62  ,0,79  ,0,90,0,0,0,0
8,244736  ,65,83,80  ,62,86 ,74,0,0,83,100,93,0
9,255649  ,47,75,90  ,60,68 ,59,0,80,85,90,93,82
10,259495 ,1,30,30   ,0,76  ,60,0,60,0,0,0,0
11,259497 ,26,83,53  ,20,82 ,64,59,85,73,80,93,57
```

quizzes

```
11,8
------------
1,223512,50,95,85,80,90,70,70,80
2,224012,35,80,95,50,70,60,0,70
3,233767,85,75,95,50,70,35,65,87
4,234243,35,80,70,30,35,30,20,85
5,235141,68,20,70,0,0,70,70,70
6,244072,55,35,80,20,15,10,60,70
7,244262,45,65,95,0,0,0,30,75
8,244736,65,75,75,0,0,60,30,85
9,255649,50,0,100,70,100,95,0,80
10,259495,65,95,65,0,0,0,0,0
11,259497,30,100,95,65,60,30,75,85
```

attendance_records

```
11,15
---------------------------------------
1,223512,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
2,224012,1,1,1,0,0,0,1,1,1,1,1,1,1,1,1
3,233767,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
4,234243,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
5,235141,1,0,0,0,0,0,1,1,1,1,1,1,1,1,1
6,244072,1,0,0,0,0,0,1,1,1,1,1,1,1,1,1
7,244262,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
8,244736,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
9,255649,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
10,259495,1,0,0,0,0,0,1,1,1,1,1,1,1,1,1
11,259497,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
```

major exams_records

```
11,2
------------
1,223512,80,78
2,224012,83,61
3,233767,85,65
4,234243,67,65
5,235141,66,69
6,244072,56,62
7,244262,70,52
8,244736,45,60
9,255649,80,71
10,259495,61,60
11,259497,67,61
```

final exam_records

```
11,1
---------
1,223512,66.5
2,224012,85.1
3,233767,78.8
4,234243,63.7
5,235141,38.4
6,244072,39.3
7,244262,67.1
8,244736,47.5
9,255649,88.8
10,259495,57.3
11,259497,79.2
```

## Appendix I-*b* (cont'd)

classrecord_Output_summary

SCHOOL NAME
COLLEGE/DEPARTMENT NAME

COURSE NAME: COURSE & SECTION IDENTIFICATION
INSTRUCTOR's NAME

| SN | STUDENT | ATTEND. | HWORK | QUIZZES | MAJORS | SUBTOTAL | FINAL | TOTAL | RANK | GRADE |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 223512 | 5.00 | 8.33 | 3.88 | 35.55 | 52.75 | 23.27 | 76.03 | 2 | B |
| 2 | 224012 | 1.00 | 7.75 | 2.88 | 32.40 | 44.02 | 29.78 | 73.81 | 4 | B |
| 3 | 233767 | 5.00 | 4.12 | 3.51 | 33.75 | 46.38 | 27.58 | 73.96 | 3 | B |
| 4 | 234243 | 5.00 | 4.46 | 2.41 | 29.70 | 41.56 | 22.29 | 63.86 | 6 | C |
| 5 | 235141 | -1.00 | -1.00 | -1.00 | -1.00 | -1.00 | -1.00 | -1.00 | -1 | DN* |
| 6 | 244072 | -1.00 | -1.00 | -1.00 | -1.00 | -1.00 | -1.00 | -1.00 | -1 | DN* |
| 7 | 244262 | 5.00 | 1.93 | 1.94 | 27.45 | 36.31 | 23.48 | 59.80 | 7 | D+ |
| 8 | 244736 | 5.00 | 6.05 | 2.44 | 23.63 | 37.11 | 16.63 | 53.74 | 8 | D |
| 9 | 255649 | 5.00 | 6.91 | 3.09 | 33.98 | 48.98 | 31.08 | 80.06 | 1 | B+ |
| 10 | 259495 | 1.00 | 2.14 | 1.41 | 27.23 | 31.77 | 20.05 | 51.83 | 9 | F |
| 11 | 259497 | 5.00 | 6.46 | 3.38 | 28.80 | 43.63 | 27.72 | 71.35 | 5 | C+ |

 

## Appendix II-*a*: A brief sample summary output of the code

```
                                                classrecorde_Pre_summary Echo_print
University / College Information:
KING FAHD UNIVERSITY OF PETROLEUM & MINERALS
COLLEGE OF ENGINEERING SCIENCES
-----------------------------------------------------------------
Course / Instructor Information:
Engineering Dynamics ME201.04
Dr. Saeid A. Alghamdi
-----------------------------------------------------------------
If Category (1 to consider, else 0 to omit a category) & Points:
NAttendance       NHomework       NQuiz        NMajor       NFinal:
    1    1    1    1    1
AttendancePass    PointsHomework  PointsQuiz   PointsMajor  PointsFinal:
 10.000   .150   .100   .450   .300
-----------------------------------------------------------------
Students Count & Grade Information:
NStudents:
    7
-----------------------------------------------------------------
Grade Limits & Letters Information:
NGradeLimits: (NGradeLetters = NGradeLimits + 1)
   10
GradeLimits:
 85.000 80.000 75.000 70.000 65.000 60.000 55.000 50.000 45.000   .000
GradeLetters (A3 format):
A+ A  B+ B  C+ C  D+ D  E  F  DN*

         7        15
-------------------------------------
         1      1001      1.00      1.00      1.00      1.00       .00       .00      1.00      1.00      1.00      1.00
 .00      1.00      1.00      1.00       .00
         2      1002      1.00      1.00       .00      1.00      1.00       .00      1.00      1.00       .00
 .00       .00      1.00      1.00      1.00
         3      1003      1.00      1.00      1.00      1.00      1.00      1.00      1.00      1.00      1.00      1.00
1.00      1.00      1.00      1.00      1.00
         4      1004      1.00       .00       .00       .00       .00      1.00      1.00      1.00       .00       .00
1.00      1.00      1.00      1.00      1.00
         5      1005      1.00      1.00      1.00      1.00      1.00       .00       .00       .00       .00      1.00
1.00      1.00      1.00      1.00      1.00
         6      1006      1.00      1.00      1.00       .00       .00      1.00      1.00      1.00      1.00      1.00
1.00      1.00      1.00      1.00      1.00
         7      1007      1.00      1.00      1.00      1.00      1.00      1.00      1.00      1.00      1.00      1.00
1.00      1.00      1.00      1.00      1.00

       istudent      nCategory   categoryPass         catSum

      ifCatPass
             1             15         10.000         11.000             1
             2             15         10.000         10.000             1
             3             15         10.000         15.000             1
             4             15         10.000          9.000             0
             5             15         10.000         11.000             1
             6             15         10.000         13.000             1
             7             15         10.000         15.000             1
```

 

## Appendix II-*a* (cont'd)

```
                                                    classrecorde_Pre_summary Echo_print
        7        3
---------------
        1      1001     59.00     82.00    100.00
        2      1002     70.00     55.00     99.00
        3      1003     50.00     32.00     25.00
        4      1004     66.00     77.00     88.00
        5      1005     23.00     88.00     74.00
        6      1006     77.00     68.00     95.00
        7      1007     77.00     68.00     10.00

      istudent      nCategory           sum     catAverage
             1              3       241.000         12.050
             2              3       224.000         11.200
             3              3       107.000          5.350
             4              3       231.000         11.550
             5              3       185.000          9.250
             6              3       240.000         12.000
             7              3       155.000          7.750

        7        2
------------
        1      1001     66.00     99.00
        2      1002     35.00     69.00
        3      1003     55.00     82.00
        4      1004     88.00     96.00
        5      1005     75.00     96.00
        6      1006     55.00     23.00
        7      1007       .00       .00

      istudent      nCategory           sum     catAverage
             1              2       165.000          8.250
             2              2       104.000          5.200
             3              2       137.000          6.850
             4              2       184.000          9.200
             5              2       171.000          8.550
             6              2        78.000          3.900
             7              2          .000           .000

        7        2
------------
        1      1001     88.00     95.00
        2      1002     23.00     12.00
        3      1003     88.00     75.00
        4      1004     75.00     62.00
        5      1005     90.00     88.00
        6      1006     78.00    100.00
        7      1007     78.00       .00

      istudent      nCategory           sum     catAverage
             1              2       183.000         41.175
             2              2        35.000          7.875
             3              2       163.000         36.675
             4              2       137.000         30.825
             5              2       178.000         40.050
```

```
                                    classrecorde_Pre_summary Echo_print
             6              2       178.000         40.050
             7              2        78.000         17.550

        7        1
---------
        1      1001     90.00
        2      1002     85.00
        3      1003     55.00
        4      1004     23.00
        5      1005     66.00
        6      1006     80.00
        7      1007    100.00

      istudent      nCategory           sum     catAverage
             1              1        90.000         27.000
             2              1        85.000         25.500
             3              1        55.000         16.500
             4              1        23.000          6.900
             5              1        66.000         19.800
             6              1        80.000         24.000
             7              1       100.000         30.000

      iStudent     IDStudents    studentsSum  iStudentsRank iStudentsGrade    gradeLetter
             1           1001          88.47              1              1             A+
             2           1002          49.78              6              9              E
             3           1003          65.38              4              5             C+
             4           1004          -1.00             -1             11            DN*
             5           1005          77.65              3              3             B+
             6           1006          79.95              2              3             B+
             7           1007          55.30              5              7             D+
```

 

## Appendix II-*b*: A brief sample summary output of the code

classrecordo

SCHOOL INFORMATION
COLLEGE OF ENGINEERING SCIENCES

Course Title & Indentifications
Instructor's Name

| SN | STUDENT | ATTEND. | HWORK | QUIZES | MAJORS | SUBTOTAL | FINAL | TOTAL | RANK | GRADE |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1001 | 11.00 | 12.05 | 8.25 | 41.18 | 61.48 | 27.00 | 88.47 | 1 | A+ |
| 2 | 1002 | 10.00 | 11.20 | 5.20 | 7.88 | 24.28 | 25.50 | 49.78 | 6 | E |
| 3 | 1003 | 15.00 | 5.35 | 6.85 | 36.68 | 48.88 | 16.50 | 65.38 | 4 | C+ |
| 4 | 1004 | 9.00 | -1.00 | -1.00 | -1.00 | -1.00 | -1.00 | -1.00 | -1 | DN* |
| 5 | 1005 | 11.00 | 9.25 | 8.55 | 40.05 | 57.85 | 19.80 | 77.65 | 3 | B+ |
| 6 | 1006 | 13.00 | 12.00 | 3.90 | 40.05 | 55.95 | 24.00 | 79.95 | 2 | B+ |
| 7 | 1007 | 15.00 | 7.75 | .00 | 17.55 | 25.30 | 30.00 | 55.30 | 5 | D+ |

## Appendix II-*c*: A sample *ad hoc* Excel spread sheet grades' assignment

| Serial No. | Sta_id & Name | total score | Attend | Quizzes scores | | | | | | | Quiz_score | Major Exams score | | | Pre-final score | Final Exam score | Score-I | Bonus | Total Score | Course Letter Grade | Grade Quality Points |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 15% | 5% | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 5% | I | II | Σ (45%) | 70% | 30% | 100% | ≤ 3% | 100% | A+, A, B, ..., F | 3.0; 2.5; 2.0; 1.5; 1.0; 0. |
| 1 | 10001 | 81.0 | 3.7 | 66 | 99 | 97 | 50 | 25 | 80 | 50 | 8.3 | 88.0 | 95.0 | 41.4 | 65.5 | 90.0 | 92.5 | 3.0 | 96 | A+ | 4 |
| 2 | 10002 | 75.0 | 3.3 | 35 | 69 | 91 | 60 | 10 | 70 | 0 | 5.2 | 23.0 | 12.0 | 7.6 | 27.4 | 85.0 | 52.9 | 0.0 | 53 | F | 0 |
| 3 | 10003 | 36.0 | 5.0 | 55 | 82 | 82 | 0 | 20 | 60 | 0 | 6.9 | 88.0 | 75.0 | 36.4 | 53.6 | 55.0 | 70.1 | 3.0 | 73 | C+ | 2 |
| 4 | 10004 | 77.0 | 3.0 | 88 | 96 | 87 | 25 | 0 | 90 | 0 | 9.2 | 75.0 | 62.0 | 30.5 | 54.3 | 23.0 | 61.2 | 0.6 | 62 | D+ | 1 |
| 5 | 10005 | 62.0 | 3.7 | 75 | 96 | 90 | 0 | 10 | 70 | 0 | 8.6 | 90.0 | 88.0 | 40.0 | 61.6 | 66.0 | 81.4 | 2.0 | 83 | B+ | 3 |
| 6 | 10006 | 80.0 | 4.4 | 55 | 23 | 0 | 0 | 35 | 0 | 0 | 3.9 | 78.0 | 100.0 | 40.6 | 60.9 | 80.0 | 84.9 | 2.5 | 87 | B+ | 3.5 |
| 7 | 10007 | 52.0 | 5.0 | 0 | 0 | 88 | 25 | 0 | 40 | 40 | 0.0 | 78.0 | 0.0 | 15.6 | 28.4 | 100.0 | 58.4 | 3.0 | 61 | D+ | 1.5 |
| | | | | | | | | | | | | | | | | | | | | Sum of QPs | 15 |
| Statistics | Average | 66.1 | 4.0 | 53.4 | 66.4 | 76 | 23 | 14.3 | 58.6 | 12.9 | 6.0 | 74.3 | 61.7 | 30.3 | 50.2 | 71.3 | 70.6 | 2.0 | 74 | No. Studnets | 7 |
| | Standard deviation | 17.0 | 0.8 | 28.9 | 39.5 | 34 | 25 | 13 | 30.2 | 22.1 | 3.3 | 23.4 | 40.3 | 13.5 | 15.8 | 26.0 | 15.0 | 1.2 | 16 | | |
| | | | | | | | | | | | | | | | | | | | | CLASS GPA | 2.14 |

Basis for grade calculations: using the total score average x and its standard deviation s:

| Scores subsets | | Letter grade | Grade cut-off lines |
|---|---|---|---|
| Mean score x: | 74 | A+ | 93.475 |
| Standard deviation s: | 16 | A | 91.475 |
| Mean score x + 0.5 σ | 81 | B+ | 82.475 |
| Mean score x - 0.5 σ | 66 | B | 80.475 |
| | | C+ | 73 |
| | | C | 65.754 |
| | | D+ | 60.754 |
| | | D | 55.754 |

 