
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-spread 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 spelled-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-points 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 stu-
dents 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-
tended negative ramifications on the learn-
ing-educational process and may even set the whole
evaluative process in a true reduced-reliability dilemma.
This process is, therefore, often categorized by academi-
cians 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 docu-
mented for automating the mechanistic aspects of the
grading assignment process [25-26]. Therefore, the
development and adapting the uses of an automated grad-
ing 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 evalu-
ative 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 learn-
ing 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 reli-
able if the mechanistic aspects of determining evaluation
metrics are automated, an algorithmic process for as-
signing letter grades to students has been 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. Com-
plete 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 proce-
dure presented herein would enable the educator to:

1) Devise several input file categories that may in-
clude 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 pre-
paring weighted scores for each input file category en-
tered 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 fre-
fquent 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 as-
signing 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 norma-
tive-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

\[
\bar{x} = \frac{\sum_{i=1}^{N} x_i}{N}
\]

(1)

\[
\sigma = \frac{1}{N} \sqrt{N \sum_{i=1}^{N} (x_i^2) - \left( \sum_{i=1}^{N} x_i \right)^2}
\]

(2)
Figure 1. Main steps of the grades’ assignment computer code.

The 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

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

\[
c_{Li} = \bar{x} \quad \text{for } i = 5 \tag{4}
\]

in which: the index \( i \) values of 1, 5, and 8, correspond to lower bounds of the distinction grade A+, the 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 Excel 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 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


Appendix I-a: Listing of the code

```
program classrecord

*** classrecord.for ***

A user-friendly Fortran code to handle the multi-faceted process of
assessing students' academic performance and assigning reliable letter
grades for a single group or multiple groups of students.

Accepts input file categories:
* attendance.csv : roster of class attendances
* homework.csv  : roster of student homeworks
* quiz.csv      : roster of student quizzes
* major.csv     : roster of student major exams
* final.csv     : roster of student final exam

And for each student prepares:
* weighted scores for each input file category
* subtotal weighted score for all categories, except the final
* weighted grand total of all categories
* performance rank relative to the overall class
* selection of grade according to user criteria

Reference: class.record fortran program initially Developed by the

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),
major(mstudents, mweeks), averageMajor(mstudents),
.final(mstudents, mweeks), averageFinal(mstudents)
dimension
.attendanceSum(mstudents), ifAttendancePass(mstudents)
dimension mStudents(mstudents), studentsSum(mStudents),
.iStudentsRank(mStudents), iStudentsGrade(mStudents)
dimension subTotal(mStudents)

open(1, file='classrecord.txt', status='old', form='formatted')
open(2, file='classrecord.txt', status='unknown', form='formatted')
open(3, file='classrecord.txt', status='unknown', form='formatted')
open(11, file='attendance.csv', status='old', form='formatted')
open(12, file='homework.csv', status='old', form='formatted')
open(13, file='quiz.csv', status='old', form='formatted')
open(14, file='major.csv', status='old', form='formatted')
open(15, file='final.csv', status='old', form='formatted')
open(11, file='C:\Documents and Settings\Administrator\Desktop\grad'
   \ing\we2018\attendance.txt', status='old', form='formatted')
open(12, file='C:\Documents and Settings\Administrator\Desktop\grad'
   \ing\we2018\homework.txt', status='old', form='formatted')
open(13, file='C:\Documents and Settings\Administrator\Desktop\grad'
   \ing\we2018\quiz.txt', status='old', form='formatted')
open(14, file='C:\Documents and Settings\Administrator\Desktop\grad'
   \ing\we2018\major.txt', status='old', form='formatted')
open(15, file='C:\Documents and Settings\Administrator\Desktop\grad'
   \ing\we2018\final.txt', status='old', form='formatted')
```

Copyright © 2010 SciRes. IIM
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, nGradelimits
  .)

c retrieve and process categories

c first initialize student grand total

do is=1, mstudents
  studentsSum(is)=0.
endo
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,'(15, 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 quizzes:
  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)

```c
if(nFinal.eq.1) then
    i=5
    call getCategory(i,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 assignment
        averageHomework(is) = -1
        averageQuiz(is) = -1
        averageMajor(is) = -1
        subtotal(is) = -1
        averageFinal(is) = -1
        studentsSum(is) = -1
    endif
endo

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

c ** synchronize student rank with failed attendance:
do is = 1, nstudents
    ifPass=ifAttendancePass(is)
    if(ifPass.eq.0) then
        flag = -1 for grade DN assignment
        studentsRank(is) = -1
    endif
endo

c ** grade code students according to grade limits selections
    call gradeCodeStudents(studentsSum,nstudents,gradeLimits,
                        .ngradeLimits,studentsGrade)

c echo some results
    write(2,'((/7a15))','student','IDStudents','studentsSum',
           'studentsRank','studentsGrade','gradeLetter'
            do is=1,nstudents
               write(2,'((/7i5,F15.2,7i15,a15))',is,IDStudents(is),
                   .studentsSum(is),studentsRank(is),studentsGrade(is),
                       .gradeLetters(is))
            endo

c output results
    write(3,'((a))'university
    write(3,'((a))'college
    write(3,'((a))'course
    write(3,'((a))'instructor
    write(3,'((a))'year
    write(3,'((a))'semester
    write(3,'((a))'credit
    write(3,'((a))'GPA
    write(3,'((a))'total
    write(3,'((a))'rank
    write(3,'((a))'grade
    write(3,'((a))'gradeLetter
    do is=1,nstudents
       write(3,'((a5,10a12))'\'SN\',\'STUDENT\',\'ATTEND\','\'HWork\',\'QUIZES\',
               \'MAJORS\',\'SUBTOTAL\','\'FINAL\',\'TOTAL\',\'RANK\',\'GRADE\'
        write(3,'((a5,10a12))')
       if(mod(is,5).eq.0) write(3,'((a5,10a12))'\'125(1h-))'
       endif
    endo
    write(3,'((a))')
stop
```

C--
The Code (Assigning Reliable Students' Letter Grades)

```fortran
subroutine gradeStudents(studentsSum,nStudents,gradeLimits,
   .nGradeLimits,1StudentsGrade)

  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)
  do il=1,nGradeLimits-1
    gradeLimits(il) =average+(2-(il-1)*0.5)*std
  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, 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
```

---

```
subroutine rankStudents(glist,nlist,1rank,mStudents)

  implicit real*8 (a-h,o-z)
  dimension glist(nlist),1rank(mStudents)
  dimension wlist(mStudents),1list(mStudents)

  do ilist = 1,nlist
    wlist(ilist) = -glist(ilist)
    1list(ilist) = ilist
  enddo

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

---

Copyright © 2010 SciRes.
The Code (Assigning Reliable Students' Letter Grades)
llist(illist) = jmin
llist(jlist) = lmin
jmin = jmin
endf
enddo
enddo

retrieve inorder ranking of in llist() entries
do i.rank=1, nlist
llist = llist(i.rank)
trank(i.lrank) = i.rank
enddo
return
end

subroutine addCategory(studentsSum, nstudents, categoryAverage, mstudents)
implicit real*8 (a-h.o-z)
dimension studentsSum(mstudents), categoryAverage(mstudents)
do i.is=1, nstudents
studentsSum(is) = studentsSum(is) + categoryAverage(is)
enddo
return
end

subroutine passCategory(category, nstudenst, n_Category, 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)'), 'isstudent', 'nCategory', 'categoryPass', 'catSum', 'ifCatPass'
nCategoryPass = 0
do i.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

subroutine averageCategory(category, nstudents, n_Category, categoryAverage, pointsCategory, mstudents, mweeks)
implicit real*8 (a-h.o-z)
dimension category(mstudents, mweeks), categoryAverage(mstudents)
write(2, '(/4a15)'), 'isstudent', 'nCategory', 'sum', 'catAverage'
do i.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,
  nstudents,nweeks,IDStudents)
  implicit real*8 (a-h,o-z)
  dimension category(nstudents,nweeks),IDStudents(nstudents)
  character*80 blank
read(itape,'*') kStudents,ncategory
write(2,'(3x10)') kStudents,ncategory
if (kStudents.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,'(2i10,5f10.2)') jsnid, IDStudents[jsnid],
  category(is,jc),jc=1,ncategory
endo
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 *.csv 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 *.csv 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,nGradeLimits + 1)
  implicit real*8 (a-h,o-z)
  dimension gradeLimits(nGradeLimits)
  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 snippet above includes functions for handling student data, such as reading from files, processing student categories, and assigning reliable letter grades. The `getCategory` subroutine reads student data from a file, checks for errors in the count of students, and calls `stopCategory` if necessary. The `stopCategory` subroutine provides error messages and stops the program execution. The `getInformation` subroutine reads various pieces of student information, including attendance, homework, quizzes, majors, and final grades, among other data.
The Code (Assigning Reliable Students' Letter Grades)

write(2,'(a)') course
read(2,'(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(2,'(a)') nAttendance, nHomework, nQuiz, nMajor, nFinal
write(2,'(a)') nAttendance, nHomework, nQuiz, nMajor, nFinal

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

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

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

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

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

### classrecord_control data

<table>
<thead>
<tr>
<th>Student ID</th>
<th>Name</th>
<th>Department</th>
<th>Course Info</th>
<th>Instructor Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>123456</td>
<td>John Smith</td>
<td>CS</td>
<td>Calculus</td>
<td>Prof. Johnson</td>
</tr>
</tbody>
</table>

### attendance_records

<table>
<thead>
<tr>
<th>Date</th>
<th>Name</th>
<th>Class</th>
<th>Attendance</th>
</tr>
</thead>
<tbody>
<tr>
<td>01/01/2023</td>
<td>John Smith</td>
<td>CS</td>
<td>90%</td>
</tr>
<tr>
<td>01/15/2023</td>
<td>John Smith</td>
<td>Math</td>
<td>80%</td>
</tr>
</tbody>
</table>

### major exams_records

<table>
<thead>
<tr>
<th>Date</th>
<th>Name</th>
<th>Subject</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>03/01/2023</td>
<td>John Smith</td>
<td>CS</td>
<td>95%</td>
</tr>
<tr>
<td>03/15/2023</td>
<td>John Smith</td>
<td>Math</td>
<td>85%</td>
</tr>
</tbody>
</table>

### final exam_records

<table>
<thead>
<tr>
<th>Date</th>
<th>Name</th>
<th>Subject</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>04/01/2023</td>
<td>John Smith</td>
<td>CS</td>
<td>90%</td>
</tr>
<tr>
<td>04/15/2023</td>
<td>John Smith</td>
<td>Math</td>
<td>80%</td>
</tr>
</tbody>
</table>

### classrecord_output_summary

<table>
<thead>
<tr>
<th>SN</th>
<th>Student ID</th>
<th>Name</th>
<th>Attended</th>
<th>Homework</th>
<th>Quizzes</th>
<th>Majors</th>
<th>Subtotal</th>
<th>Final</th>
<th>Total</th>
<th>Rank</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>234567</td>
<td>Jane Doe</td>
<td>90%</td>
<td>85%</td>
<td>90%</td>
<td>95%</td>
<td>95%</td>
<td>95%</td>
<td>95%</td>
<td>2</td>
<td>B</td>
</tr>
<tr>
<td>2</td>
<td>789012</td>
<td>Bob Smith</td>
<td>85%</td>
<td>90%</td>
<td>80%</td>
<td>90%</td>
<td>90%</td>
<td>90%</td>
<td>90%</td>
<td>3</td>
<td>C</td>
</tr>
<tr>
<td>3</td>
<td>123456</td>
<td>Emily White</td>
<td>95%</td>
<td>90%</td>
<td>95%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
<td>1</td>
<td>A</td>
</tr>
<tr>
<td>4</td>
<td>654321</td>
<td>Michael Lee</td>
<td>80%</td>
<td>85%</td>
<td>80%</td>
<td>85%</td>
<td>85%</td>
<td>85%</td>
<td>85%</td>
<td>4</td>
<td>D</td>
</tr>
</tbody>
</table>

*Note: DN = Did Not Attend*
Appendix II-a: A brief sample summary output of the code

```plaintext
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:
   NTest / 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
GradLimits:
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*

<table>
<thead>
<tr>
<th>7</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td>.00 1.00</td>
<td>1.00 1.00</td>
</tr>
<tr>
<td>.00 1.00</td>
<td>1.00 1.00</td>
</tr>
<tr>
<td>3.00 .00</td>
<td>1.00 1.00</td>
</tr>
<tr>
<td>1.00 1.00</td>
<td>1.00 1.00</td>
</tr>
<tr>
<td>1.00 1.00</td>
<td>1.00 1.00</td>
</tr>
<tr>
<td>1.00 1.00</td>
<td>1.00 1.00</td>
</tr>
<tr>
<td>1.00 1.00</td>
<td>1.00 1.00</td>
</tr>
<tr>
<td>1.00 1.00</td>
<td>1.00 1.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>istudent</th>
<th>nCategory</th>
<th>categoryPass</th>
<th>catSum</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>15</td>
<td>11.000</td>
<td>11.000</td>
</tr>
<tr>
<td>2</td>
<td>15</td>
<td>10.000</td>
<td>10.000</td>
</tr>
<tr>
<td>3</td>
<td>15</td>
<td>10.000</td>
<td>10.000</td>
</tr>
<tr>
<td>4</td>
<td>15</td>
<td>9.000</td>
<td>9.000</td>
</tr>
<tr>
<td>5</td>
<td>15</td>
<td>11.000</td>
<td>11.000</td>
</tr>
<tr>
<td>6</td>
<td>15</td>
<td>10.000</td>
<td>10.000</td>
</tr>
<tr>
<td>7</td>
<td>15</td>
<td>10.000</td>
<td>10.000</td>
</tr>
</tbody>
</table>
```

Copyright © 2010 SciRes.
Appendix II-a (cont'd)

<table>
<thead>
<tr>
<th>iStudent</th>
<th>nCategory</th>
<th>sum</th>
<th>catAverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>241</td>
<td>12.050</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>224</td>
<td>11.200</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>107</td>
<td>5.350</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>231</td>
<td>11.350</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>240</td>
<td>12.000</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>135</td>
<td>7.250</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>iStudent</th>
<th>nCategory</th>
<th>sum</th>
<th>catAverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>165</td>
<td>8.250</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>104</td>
<td>5.200</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>117</td>
<td>6.850</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>184</td>
<td>9.200</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>171</td>
<td>8.550</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>78</td>
<td>3.900</td>
</tr>
<tr>
<td>7</td>
<td>2</td>
<td>0</td>
<td>0.000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>iStudent</th>
<th>nCategory</th>
<th>sum</th>
<th>catAverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>183</td>
<td>41.175</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>161</td>
<td>36.675</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>137</td>
<td>30.825</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>178</td>
<td>40.050</td>
</tr>
<tr>
<td></td>
<td></td>
<td>178</td>
<td>40.050</td>
</tr>
<tr>
<td></td>
<td></td>
<td>78</td>
<td>27.550</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>iStudent</th>
<th>IDStudents</th>
<th>studentsSum</th>
<th>iStudentsRank</th>
<th>iStudentsGrad</th>
<th>gradeLetter</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1001</td>
<td>89.47</td>
<td>1</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1002</td>
<td>49.78</td>
<td>6</td>
<td>E</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1003</td>
<td>65.38</td>
<td>4</td>
<td>C</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1004</td>
<td>1.00</td>
<td>-1</td>
<td>D+</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>1005</td>
<td>77.65</td>
<td>3</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>1006</td>
<td>79.95</td>
<td>2</td>
<td>B+</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>1007</td>
<td>55.30</td>
<td>5</td>
<td>D</td>
<td></td>
</tr>
</tbody>
</table>
### Appendix II-b: A brief sample summary output of the code

classrecordo

<table>
<thead>
<tr>
<th>SN</th>
<th>STUDENT</th>
<th>ATTEND.</th>
<th>WORK</th>
<th>QUIZES</th>
<th>MAJORS</th>
<th>SUBTOTAL</th>
<th>FINAL</th>
<th>TOTAL</th>
<th>RANK</th>
<th>GRADE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1001</td>
<td>13.00</td>
<td>12.05</td>
<td>8.25</td>
<td>41.38</td>
<td>61.48</td>
<td>27.00</td>
<td>88.47</td>
<td>1</td>
<td>A</td>
</tr>
<tr>
<td>2</td>
<td>1002</td>
<td>10.00</td>
<td>11.20</td>
<td>5.20</td>
<td>24.28</td>
<td>25.50</td>
<td>49.78</td>
<td>6</td>
<td>E</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>1003</td>
<td>12.00</td>
<td>5.35</td>
<td>6.85</td>
<td>36.68</td>
<td>48.88</td>
<td>16.10</td>
<td>65.38</td>
<td>4</td>
<td>C+</td>
</tr>
<tr>
<td>4</td>
<td>1004</td>
<td>9.00</td>
<td>-1.00</td>
<td>-1.00</td>
<td>-1.00</td>
<td>-1.00</td>
<td>-1.00</td>
<td>-1.00</td>
<td>-1</td>
<td>D</td>
</tr>
<tr>
<td>5</td>
<td>1005</td>
<td>12.00</td>
<td>9.25</td>
<td>8.55</td>
<td>40.05</td>
<td>57.85</td>
<td>19.80</td>
<td>77.65</td>
<td>3</td>
<td>B+</td>
</tr>
<tr>
<td>6</td>
<td>1006</td>
<td>13.00</td>
<td>12.00</td>
<td>3.00</td>
<td>40.05</td>
<td>55.85</td>
<td>24.00</td>
<td>79.35</td>
<td>2</td>
<td>B+</td>
</tr>
<tr>
<td>7</td>
<td>1007</td>
<td>15.00</td>
<td>7.75</td>
<td>0.00</td>
<td>17.55</td>
<td>25.30</td>
<td>30.00</td>
<td>55.30</td>
<td>5</td>
<td>D+</td>
</tr>
</tbody>
</table>

### Appendix II-c: A Sample ad hoc Excel spreadsheet grades' assignment

<table>
<thead>
<tr>
<th>Serial No.</th>
<th>Raw score</th>
<th>Grade</th>
<th>Final score</th>
<th>Final letter</th>
<th>Course letter</th>
<th>Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>81.0</td>
<td>A+</td>
<td>88.0</td>
<td>90.0</td>
<td>A</td>
<td>5.0</td>
</tr>
<tr>
<td>2</td>
<td>75.0</td>
<td>F</td>
<td>80.0</td>
<td>90.0</td>
<td>F</td>
<td>0.0</td>
</tr>
<tr>
<td>3</td>
<td>50.0</td>
<td>C+</td>
<td>65.0</td>
<td>70.0</td>
<td>C</td>
<td>3.0</td>
</tr>
<tr>
<td>4</td>
<td>75.0</td>
<td>B+</td>
<td>90.0</td>
<td>90.0</td>
<td>B+</td>
<td>3.0</td>
</tr>
<tr>
<td>5</td>
<td>90.0</td>
<td>A+</td>
<td>100.0</td>
<td>100.0</td>
<td>A+</td>
<td>5.0</td>
</tr>
<tr>
<td>6</td>
<td>50.0</td>
<td>B+</td>
<td>60.0</td>
<td>60.0</td>
<td>B+</td>
<td>3.0</td>
</tr>
<tr>
<td>7</td>
<td>70.0</td>
<td>C+</td>
<td>80.0</td>
<td>80.0</td>
<td>C+</td>
<td>3.0</td>
</tr>
</tbody>
</table>

### Statistics
- Average: 80.4
- Standard deviation: 10.3
- Total Students: 7
- Class GPA: 2.14

### Class GPA

<table>
<thead>
<tr>
<th>Grade</th>
<th>GPA</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>4.0</td>
</tr>
<tr>
<td>A-</td>
<td>3.7</td>
</tr>
<tr>
<td>A-</td>
<td>3.3</td>
</tr>
<tr>
<td>B+</td>
<td>3.0</td>
</tr>
<tr>
<td>B</td>
<td>2.7</td>
</tr>
<tr>
<td>B-</td>
<td>2.3</td>
</tr>
<tr>
<td>C+</td>
<td>2.0</td>
</tr>
<tr>
<td>C</td>
<td>1.7</td>
</tr>
</tbody>
</table>

### Class GPA Calculations

<table>
<thead>
<tr>
<th>Grade</th>
<th>GPA</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>4.0</td>
</tr>
<tr>
<td>A-</td>
<td>3.7</td>
</tr>
<tr>
<td>A-</td>
<td>3.3</td>
</tr>
<tr>
<td>B+</td>
<td>3.0</td>
</tr>
<tr>
<td>B</td>
<td>2.7</td>
</tr>
<tr>
<td>B-</td>
<td>2.3</td>
</tr>
<tr>
<td>C+</td>
<td>2.0</td>
</tr>
<tr>
<td>C</td>
<td>1.7</td>
</tr>
</tbody>
</table>

### Data

<table>
<thead>
<tr>
<th>Score</th>
<th>A+</th>
<th>A-</th>
<th>A-</th>
<th>B+</th>
<th>B</th>
<th>B-</th>
<th>C+</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>93.475</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>91.475</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>89.475</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>87.475</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>85.475</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>83.475</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>81.475</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>79.475</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Copyright © 2010 SciRes.