Gradiance On-Line Accelerated Learning Users' Manual

Jeffrey D. Ullman
Stanford University and Gradiance Corp.

Abstract

Gradiance On-Line Accelerated Learning (GOAL) is a system for creating and automatically grading homeworks, programming laboratories, and tests. Through the concept of a "root question," Gradiance discourages collaboration and encourages the student to work a complete problem, rather than guessing or solving only the part of a problem needed to select among choices. The instructor has the option to offer a hint or advice every time the student gets a wrong answer, and the student may be allowed to try each question until getting it right. This manual covers how one uses the Gradiance system to create root problems, and programming labs (in SQL, particularly), how one assembles questions into homeworks, and how one issues assignments to a class.

Contents

1 Root Questions

A *root question* is a multiple-choice question that has several right answers and many wrong answers. In this section, we first explain the motivation for using root questions and then go through the process of creating root questions using the Gradiance system.

Root questions automate the process of assigning and grading "analytic" questions, where, conventionally, a student in Math, Science, or Engineering is asked to solve a problem and hand in the solution to a TA, who grades it and hands it back a week or so later. With root questions, the student is asked to solve the same problem, but their knowledge of the answer is sampled by a multiple-choice question. If they get the question wrong, they are given a hint, called a *choice explanation*, and invited to try again. As a result, homework is no longer a little test, but rather an opportunity for students to learn how to solve problems in the subject matter of the course.

While the instructor has several options, we believe that the most effective use of root questions is in groups of 4–6 questions, called an *assignment*. Students should be allowed to take the assignment until they get all the questions right. By grouping several questions together, and expecting the student to work out all the questions, it is unlikely that one could get a perfect score by guessing. Yet as they solve the individual problems, and keep the answers in front of them for reference, it doesn't take long for students to pick right answers if they have to repeat an assignment to get a better grade.

Instructors and TA's are spared the effort of grading many nearly identical homeworks, while students appreciate the immediate feedback in the case of errors. In addition, because we randomize both the order of questions and the selection of right and wrong choices, there is some inhibition to cheating, as students cannot easily pass information like "the answer to question 3 is b"; there is no notion of "question 3" or a particular "choice b."

To see how root questions work, let us consider a typical long-answer question from database systems:

> If relation R has set of tuples x and relation S has set of tuples y, what is the natural join of R and S?

A root question about joins of relations would be phrased as follows. The *stem* (portion before the choices) of the question would give particular, small sets of tuples that constitute the relations R and S and then say: "compute the natural join of R and S, and then indicate which of the tuples below is in the natural join." The student would see four choices, one of which would be a randomly chosen tuple of the answer, and the other three would be chosen from a list of incorrect answers provided by the question designer. As a result, the student has to compute the join, just as in the conventional form of question, but they get their answer sampled for correctness, rather than graded in its entirety.

The question designer gives a list of possible correct answers. For this question, the list would naturally consist of all and only the tuples in the natural join of these relations. The designer also gives a supply of incorrect answers, which in this case could be any other tuples. To make the *distractors* (incorrect answers) look plausible, it would be wise to give them the right number of components and to use values that appear in R and/or S.

Finally, the designer should provide choice explanations, at least for the incorrect answers; these are normally shown to the student if they make the corresponding incorrect choice. In general, a choice explanation could be a reason why the choice is incorrect, or perhaps an outline of how to approach the problem. For instance, in the case of the question about joins, we could explain to the student which tuples would have to be in R and/or S for the incorrect choice to be correct.

That explanation not only should convince the student that their answer is wrong, but give them the reasoning needed to figure out how to work the problem if they are unsure of how to compute joins.

2 Creating Questions

We'll go through the mechanics of creating a root question momentarily, but to start, you need to think of an idea for the question. While there is no sure way to invent a good root question, start by thinking of an ordinary, long-answer question, just as you would for an ordinary homework assignment. Visualize the answer. Unless the answer is something like "6," it probably has distinct components, just as the answer to the example question about joins, above, did. Phrase your question so that it asks not for the whole answer, but an identification of a part of the answer. The correct choices may turn out to be pairs, where the first component of the pair is a description of which part of the answer is requested, and the second component is a proposed value for that part.

As an example, you will find in our question bank a question about computing PageRank — the Google technique for estimating the importance of Web pages. The stem describes a simple example of a graph representing links among three Web pages, and the student is asked to compute the PageRank of each of the pages. The correct choices are pairs consisting of one of the nodes, and the correct PageRank for that node; the incorrect choices are similar, but with the wrong PageRank value.

2.1 Entering a Question into the Question Bank

After logging on to the Gradiance system as an instructor, go to the menu on the left and select *Question Bank*. You should see three options appear below:

1. *Find Questions*: Used for searching for questions. We'll need this choice if we ever need to edit our question, or just for browsing to see what other instructors have created.
2. *Create Question*: Used to enter new questions into the question bank.
3. *Upload File*: Needed to embed diagrams in our questions.

Click on *Create Question*. You should see a form to fill, with four fields:

1. *Question Type*: Normally leave this field at *Root Multiple-Choice Question*. The alternative is a conventional multiple-choice question with one correct answer, which is useful for tests, e.g., where you want students to take the question only once, and you want all students to get the same choices, for fairness.
2. *Question Category*: Select the closest category.
3. *Difficulty Level*: The choices are 1–5, with 1 the easiest. You can leave this value at the default 3, but if your question is intended to be noticeably hard or easy, it would be a good idea to warn other instructors, should you choose to make your question public.
4. *Search Words*: These can be any list of words, separated by blanks. The purpose of these words is to make it easier to find questions on a topic later. They are especially useful if your question does not fit the predefined categories for your subject.

Having completed the form, click *Create Question*, and you will see another screen into which the question is typed. There are two large boxes, and smaller lines for entering correct and incorrect answer choices. The big boxes are:

1. The question text, i.e., the stem of the question, which the student sees.
2. A question explanation, which can, at the option of the instructor, be shown to the student after they have answered the question, or after the deadline for their homework.

The language used in this form is HTML, so you can format the question as you like, e.g.,

```
Let <I>R</I> be the relation with tuples (1,2),...
```

Below the question stem are four boxes for entering correct answers and twelve boxes for incorrect answers. Again HTML is used in these boxes. You need not fill in all the boxes, and if you want more choices, there will be a way to add more on the next screen. After entering choices, click *Create Question*.

The next screen shows what you entered on the previous screen, and has place for you to add choice explanations for any of the choices. There are also four options on the bottom:

1. *Add New Choices* in case you want to use more than four correct and/or twelve incorrect answers.
2. *Delete Choices* to eliminate some of the choices you already have entered.
3. *Edit Question* to change the stem or question expanation (not choice explanations).
4. *Delete Question*, with the obvious effect. But note that the deletion needs to be confirmed.

To include choice explanations, click *Add Explanation* next to the choice you wish to explain. We advise that a choice explanation be given for each incorrect answer. For example, you may wish to explain why a particular choice is wrong, or give some general hints about how to solve the problem. However, we would not use the entire question-explanation as the explanation for an individual answer; the latter is best shown only after the homework's deadline has passed. If you want to say "congratulations" for a correct answer, that is OK too, but we generally leave the choice explanations for correct answers blank. *Hint*: Since choice explanations are frequently similar, be sure to use copy-and-paste.

2.2 Using Diagrams in Questions

If you need a figure as part of your question, select *Upload File* from the menu on the left. You are given a screen in which you can name, or browse for, the file that holds the figure you need for your question; its operation should be obvious.

You may address the uploaded file f by the path `../pictures/`f . For instance, a question using the uploaded file `diagram1.GIF` might start out:

```
Consider the following E/R diagram:
<P>
<IMG SRC = "../pictures/diagram1.GIF">
    ...
```

2.3 Editing Questions

You may discover later that you need to change a question. The wording may be poor, or it may even be that some answers were misclassified as correct or incorrect. If so, do the following:

1. From the left menu, select *Find Questions*, which is under *Question Bank* and only appears after you click *Question Bank*.
2. Fill out the search form that is presented to you. As a default, you search for only your own questions, which are the only ones you can edit.
3. Click *Find*. You should be given a list of the questions that match your search.
4. Locate the question you want to edit. Click on *Details* at the right. The question, with all its answer choices should appear.
5. Click on *Edit* at the bottom. Now, the original form on which the question was created should appear.
6. Make whatever changes you wish; your options are the same as for the screen you get when you created the question.
7. Click *Submit Changes*, and the edited question replaces its old version. Note that if this question is currently being used in a homework, then the new version appears in the homework, and every student who later opens the homework gets an instance of the new version.

3 Creating Homeworks

There are two different approaches to creating a homework for a class:

1. Locate an existing homework. You may use it intact, or you may modify it by adding or deleting questions. You may also adjust certain parameters, such as the time allowed for completion of the homework.
2. Create a new homework from questions. You may use either questions you have created or others that appear in the question bank.

Creating a homework is different from assigning to your class. Whichever approach you follow, you will have to assign to the class, as described in Section 3.2.

3.1 Creating a New Homework

Go to *Homeworks* in the menu on the left, and click it to open two choices: finding and creating homeworks. Choose *Create Homework*, which opens up a screen in which you place the following information:

1. Give your homework a *Title*, for example, `CS145 functional-dependency homework`. The words in this title field may be used later to search for homeworks, so it is a good idea to make this title descriptive.

2. Pick a *Duration* for the homework. If you are creating a homework of 4–5 root questions, the default 60 minutes is adequate. If the homework is really an exam, use whatever time is appropriate.

3. Select *Positive* and *Negative Points Per Question*. We recommend using 3 positive points and one negative point, since then guessing nets nothing in a 4-way multiple choice question. However, if you are hoping for an eventual perfect score, it is fine to use one positive point and zero negative points. Note that some students get confused by negative points and will be convinced the system is malfunctioning.

4. The *Difficulty Level* is in the range 1–5; set it to something other than the default 3 if you feel another level is warranted. The level can be used when searching for homeworks, incidentally.

5. If you wish, choose another *Randomization Seed*. Usually, the default 1234 is fine. This number is used to randomize the display of the homework and the selection and order of choices for root questions, if desired.

You will find a box for a *Description*, which must be filled out and should hold some comment about the area covered by the homework. Now, we are ready to add questions to the homework, using the question bank. Click on *Add Questions*. You are given a screen with several fields that let you describe the kind of question you want.

1. The *Question Type* is either multiple-choice or true/false; typically we want the former.

2. Tell whether you want *Root Questions* or not. We recommend that root questions be used for homework but not for exams. The reason is that some variations of a root question can be harder than others, which is OK for a homework that people will take until they get a perfect score, but not for an exam.

3. Set the *Difficulty Level* if you wish; generally we leave it at *Any*.

4. The best ways to identify questions are to use either the *Question Category* or *Search Words* fields, or both. Remember that when you create a question, you are asked to assign it to a category, and you must give it at least one search word.

5. You may ask for homeworks created by anyone, or by one specific instructor — presumably you.

Click *Search*, and you will get a page with all the matching questions. You may examine a question, including its answer choices, by clicking *Details* next to that question. Use the check boxes in the column *Select Questions* to pick zero or more questions to add to your homework. Click *Add Questions*.

You are taken back to the screen that lets you add more questions. Repeat the process of adding questions until you have a complete homework. At this screen, you also have the option of selecting certain previously added questions and deleting them from the homework. The questions remain in the question bank. Finally, when you have exactly the questions you want in that homework, click *Submit Homework*. You now have a new homework in the bank of assignments, and you are ready to assign it to a class, either now or later.

3.2 Giving an Assignment to Your Class

You can assign students any assignment (homework or lab), either a public assignment that someone else has created or an assignment that you yourself have created. The steps are as follows:

1. In the menu at the left, choose *Administer Groups*.
2. You will be given a screen with a line for each group that you own, typically only the one group for your class. Select *Assign Homeworks*.
3. You now have a screen with three boxes, which allow you to search for homeworks using the *Difficulty Level* and search words from either the title or description. That is, you may specify one or more words that you remember from the title or description of the desired homework.
4. Click *Search*. You are given a page with information about all the matching homeworks, including those already assigned to the group and those available in the assignment bank but not yet assigned.
5. Clicking *Preview* next to a homework lets you see the homework as the student sees it. If you submit the homework, you get to see the Gradiance response to the student. For example, by answering incorrectly, you get to see a typical choice explanation, as well as the question explanation. Note that normally, the question explanation does not appear until after the homework due date.
6. When you have found the homework you wish to select, click *Assign* for that homework.
7. Then, fill out the conditions of the homework, as follows:
   (a) Change the homework title and/or the number of positive or negative points per question, if you like.
   (b) Fill out the *Open Date* and *Close Date*. For a homework, you might choose a week between opening and closing, but for a one-hour exam, you might choose 70 minutes, with 5 minutes leeway on either side of the intended exam time. Note the format of date/times, which follows the pattern in the defaults on the form. The day precedes the month, which is always a 3-letter abbreviation, followed by the year and the time on a 24-hour clock.

There are a large number of options available to the instructor regarding how the homework is managed. The choices that we believe should be used are listed on the screen, and if they are satisfactory, you may simply click *Assign Homework*. Students are allowed to take the homework as many times as they like, they get choice explanations each time they submit a wrong answer, and they get to see the full question explanation only after the due date. Root questions are given to them with new choices every time they try the homework.

The instructor who wants to control the choices can instead click *Advanced Assignment Screen* and choose from among all the options. We shall describe the options on the advanced assignment screen, which will also explain what the default choices on the main screen do. In addition to the options (open and close dates, etc.) available on the main screen, you have the following choices to make on the advanced screen:

1. *Question Randomization Type*: The choices are:

(a) *No Randomization*: Every student gets an identical homework. If root questions are used, the selection of right and wrong answers occurs once,[1] and the order of the four choices is fixed.

(b) *Display Randomization*: The homework is fixed, but different students get the set of questions in different orders, and the same four choices will appear in different orders for different students.

(c) *Choice Randomization*: The questions appear in fixed order. However, the order of choices is random, and if the question is a root question, the choices themselves are chosen at random.

(d) *Full Randomization*: We recommend this choice for taking advantage of root questions in homeworks. Here, each time a student accesses the homework, root questions are presented with one random correct answer and three random distractors. The questions and their choices of answer are presented in random order.

2. *Randomization Seed*: Pick an integer to start whatever randomization process you have chosen. 1234 is usually fine.

3. *Number of Attempts Allowed*: Choose how many times a student may try the homework. The choices are *One*, *Unlimited*, or *Others*. If you choose *Others*, then fill in the number of attempts in the next box: *Specify Limit*. Otherwise, the system ignores *Specify Limit*, and you should too. We recommend *Unlimited* for homework and *One* for exams.

4. *Attempt Interval*: One of the first things we noticed was that some students treated the system as a challenge to get the correct answers without understanding the questions. If you think about it, a statistical analysis will almost surely allow you to identify correct answers, if you see random presentations of the homework enough times. To discourage such behavior, we allow them to open a homework only once per 10 minutes. You have the option to block the reopening of a homework for any number of minutes that you choose.

5. *Student Score Records*: You have the option to let students see either all their scores, or just their most recent score for this homework. It is probably adequate to let them see only their most recent.

6. *After Submission*: There are six options, which determines what the student sees when they submit a homework. The first two *No Feedback* and *Total Score Obtained* should be self-explanatory. *Individual Question Grades* lets them know which questions they got right and wrong, but not the correct answer for those they got wrong. *Choice Explanation*, the option we recommend, lets them see, for each incorrect answer, the choice explanation associated with their answer, if there is one. Otherwise, they are just told their answer is incorrect. *Question Explanation* gives them, in addition, the solution to the problem. Finally, *Answer Key* gives them not only the choice and question explanations, but the correct answer.

7. *After Due Date*: You have the same six options as for *After Submission*. This selection determines what students will see after the due date for the homework. We recommend going all the way with *Answer Key*.

[1] And the selection of the four choices is random, so there really is some randomization going on, even though the specification is "*No Randomization*.

8. *Instructor Score Records*: When you examine grades for this homework, you can arrange to see only the most recent, the average, the maximum or all scores obtained by a student. These choices are relevant only if students may take the homework more than once.

After making your selections in all the above categories, click *Assign Homework*, and you are done.

3.3 Repairing Homeworks

Possibly, as a homework is being done by your class, a bug will be discovered. If the problem is that a question needs to be reworded or its answer choices changed, use the question-editing procedure of Section 2.3. Note that the change will appear in your homework immediately; you do not have to manage the homework in any way.

However, if the homework itself needs to be changed, say by adding or deleting entire questions, a different procedure, outlined below, must be followed.

1. In the menu at the left, select *Homeworks* → *Find Homeworks*.
2. Fill out the search form as best you can, to receive a list of homework titles, including the homework you need to change. Next to this homework, click *Details*.
3. You will see a page summarizing this homework. At the bottom, click *Edit*.
4. Now, you have a page with editable boxes that specify all the important information about the homework. Edit these if you need to. To change the questions, you may:
   (a) Delete questions. Select the questions to be deleted, using the check boxes at the beginning of each question. Then click *Delete Selected Questions* at the bottom.
   (b) Add questions. Simply select *Add Questions* at the bottom, and then follow the procedure for adding questions outlined in Section 3.1.
5. Finally, click *Submit Homework* at the bottom of the form, and the new version of the homework will now be seen by your class.

4 SQL Laboratories

SQL labs give the student a database schema, against which some SQL queries must be written. In a properly designed lab, when the student makes a mistake that is semantic (rather than a syntax error), they are given an example database and shown both what their query did, and what it should have done. In unusual cases, the sample database will fail to exhibit their error, but if the lab designer is careful, that situation will occur rarely. Rather, in the normal situation, the student gets valuable help from the Gradiance system when they make a semantic error.

4.1 Creating an SQL Lab

Start by choosing *Lab Projects* → *Create Lab Project* in the left menu. In the next screen, select *SQL Lab* as the assignment type. Give the lab a title and change the difficulty level if you like. When you click *Continue*, you will get a screen where you can specify the problem and the database on which students' work will be tested. That is, an SQL laboratory gives students an informal description

of the schema and the queries they are to write in SQL. Behind the scenes, the system needs to know, in SQL terms, what the schema is, and it needs to have a database on which to test queries. It also needs a *reference query* for each query the students are to write, so the system can know what the correct answer to the query on the test data is. There are five pieces of information you need to provide in this screen:

1. In the third of four small boxes, at the top, enter the number of queries for this laboratory. You can have as many queries as you like in one lab, but they all refer to one schema.

2. In the first large box, you write the *Assignment Description*, that is, the stem in which you describe the database schema and give English statements of the queries that the students are to write in SQL.

3. The second large box is the *Create Script*. Here, you enter `CREATE TABLE` commands or other SQL that you need to create the schema. You should begin the create script by dropping the tables, in case tables of the same name are used by some other lab.

4. The third large box is the *Load Script*, which will be used to populate the schema with tuples to form the *evaluation database*. You typically need to write one `INSERT` statement for each tuple in any of the relations of the schema, but more creative ways of populating the database are possible. This data is never seen by the student, but is used by Gradiance to test whether the student's query is (likely to be) correct.

5. The last large box is another *Load Script*, this time for the *example database*. It is this database that the student sees when they make a semantic error.

When you click *Create SQL Lab* for this screen, you are presented with another screen where reference queries are to be entered. There will be one box for each query, according to the number you entered in (1) above. For each, in the order you expect the students to write their answers, you give a correct SQL query. This "reference" query will be used to determine whether or not a student's answer is correct. There is a certain leeway in how the comparison between the student's result and the reference query's result is made. For example, neither order of tuples nor order of columns matters.

4.2 Advice on Designing Example and Evaluation Databases

A pitfall of lab design is that the evaluation database may exhibit an error, i.e., the students query gives a different result from the reference query, yet the example database gives the same result for both queries. Obviously, the evaluation database should have enough tuples, and varied-enough tuples, that the typical incorrect query will do something wrong on that database. However, it is also necessary that errors detected by the evaluation DB be shown as well in the example DB. Yet if those two DB's were the same, the student would immediately know the evaluation DB and could just write a query that generated the proper result for that DB and no other DB.

Thus, we suggest that you use the "real" data as the example DB. For example, when we developed a lab called *Kings* about the kings and queens of England, the data was initially typed into the example database. We found it easiest to write the data as tuples, and then surround each by the necessary `INSERT` statement using a text editor.

Next, make a copy of the `INSERT` statements from the example DB, and perform certain edits on the values in those statements. One constraint is that a constant appearing in any of the queries

must remain unchanged. Another constraint is that at least some of the constants appearing in any answer must be changed. For example, if we use a query "Who is the parent of Elizabeth II?" then we must not change `'Elizabeth II'` anywhere. However, if we don't change the name of her father, `'George VI'`, in the evaluation DB, then students will simply write the query

```
SELECT 'George VI' AS parent FROM Parents;
```

Thus, we globally replaced `George` by `ggg` in the evaluation database. (That's a lie; we don't want anyone to know the actual replacement string.)

Another tricky issue is a query that asks for a count. For example, consider "How many kings were named Edward?" We can't replace `Edward` by `eee` or anything else, because a pattern `'Edward%'` will appear in the query. However, if we don't make any changes involving the Edwards, the number the student sees in the example DB will work in the evaluation DB as well. The solution is to either delete some of the Edward tuples in the evaluation DB, or add some imaginary Edwards.

4.3 Assigning Labs

Remember that an SQL lab is an assignment, not a question. Thus, we assign it to a class as we did homeworks in Section 3.2. Begin with *Administer Groups* → *Assign Lab Projects*, which currently only allows you to assign SQL labs. Fill out the search form as for a homework, and click *Search*. You are presented with the available SQL labs, and you may click *Assign* for the one you want.

However, most of the parameters of the assignment are irrelevant to labs. The only things you need to concern yourself with are:

1. The title, if you wish the students to see another title.
2. The open and close dates for the assignment.
3. Number of points per question.
4. Whether the instructor gets to see all scores or just the most recent score.

Note that students always get to take a lab as many times as they wish.

5 Managing Groups

You need to create a group for your class before you can assign homework. There are also a number of things you'll need to know about reading student grades, and handling the "I forgot my password" problem.

5.1 Creating a Group for Your Class

Before you can use Gradiance for your class, you have to create a group, which consists of an account for each student. Since Gradiance does not know the names of the students, it gives dummy names to the accounts, and gives each account a 10-character password, which it calls a "token." You should create a group with at least twice as many accounts as there are students, since there may be others wishing to use the system, students who need an account in an emergency, or other reasons we cannot predict.

One of the first things you need to do when the class begins is get students to sign up for accounts, enter their names into the account, and pick a secure password. Here is the recommended procedure:

1. From the left menu, choose *Add A Group*.

2. The screen you next see asks only for a group name and a number of members. Remember to pick a sufficiently large number of members, because it is not easy to increase it later, and costs little to have extras.

3. Click *Create Group*.

4. You will now see a page with a table of "user ID's" (initial account names of the form `<group name>_user`*n*) and "tokens," which are the initial passwords for the accounts. **Warning**: This page is the only place you can conveniently determine the token associated with each user ID. Be sure to save it from your browser, either in a file or as a printed copy, or both.

5. Using a printed copy of this page, and ask students, perhaps at the first class, to:

   (a) Sign their names next to one account line.

   (b) Remember the user ID and token they have chosen.

   (c) As soon as possible, login to Gradiance and, using the procedure outlined in Section 5.2, enter their name and a new password.

6. **Warning**: It is important that you save the sign-up sheet, because if a student forgets his/her password, the only way you can fix the problem is to reset the password, using the student's original user ID. See Section 5.3.

5.2 How Students Initialize Their Accounts

Here is what to tell students about how to change their account name and password. They must direct their browser to the URL of the Gradiance installation they are using. They will see the login screen, just as the instructor does. On entering the user ID and password (token) they have been given, they will be admitted to the system, where they see a menu on the left that is rather different from what the instructor sees. Students have only five choices. The first three — *Homeworks, Lab Projects*, and *Tests* — are where they find work of these three types when they are assigned. The last — *Logout* — has the obvious effect.

It is the fourth choice — *Update Account* — that they must now select. They are given a series of boxes to fill out, including two for the password and its confirmation. The only point that may not be obvious is that the *Nickname* becomes the user ID for the account. They should replace the user ID in the nickname field by their preferred account name. Gradiance will not allow two students using the same site to choose the same nickname.

5.3 Fixing Student Accounts

Another aspect of group administration is dealing with the lost password, forgotten login name, or problem student who needs to be taken off the system. When you click *Administer Groups* in the left menu, you find four items. One option, *Assign*, was discussed in Section 3.2. The others are *Group Students*, *Check Grades*, and *Verify Certificate*.

Choose *Group Students*. You are given a menu with all the accounts in the group and three things that you can do after you have selected one of the accounts:

1. *Reset Password*. This choice sets the password to the initial ID (token) with which the account was created; see Section 5.1). Again, please remember that it is important the initial list of accounts and tokens be preserved.

2. *Inactivate* freezes the account so it no longer can be used.

3. *Activate* restores an inactive account.

5.4 Checking Grades

Another choice under *Administer Groups* is *Check Grades*. You are given a table with all the assignments for which grades exist. Select the assignment whose grades you want to see and click *Group Grade List* next to it. A table of grades appears. It will include only those students who have taken the assignment one or more times.

If you want to extract grades from the Gradiance system, say for entering into a spreadsheet, mail them to yourself or an assistant. You may place an email address at the bottom of the grade list and click *Email Grade Report*.

5.5 Verifying Certificates

The fourth thing you can do under *Administer Groups* is *Verify Certificate*. Every time a student submits an assignment, they are given an alphanumeric certificate. This certificate lets you deal with a situation in which the student claims they submitted something other than what Gradiance records for them. You ask them for the certificate, the submission number and timestamp for the submitted work; all three items appear on the Gradiance response to the student when they submit the work. You then enter the list of answers that they claim, and Gradiance will tell you whether or not their claim is valid.

Incidentally, you can also view their entire submission, given the submission number (not certificate). Just choose *View Submission* from the left menu and enter the number.