

# Curator System

## Student Guide

Virginia Polytechnic Institute and State University  
Department of Computer Science  
©Copyright January 2001

### Table of Contents

1.	<a href="#">Introduction</a>	2
2.	<a href="#">Submitting an Assignment from Your Computer</a>	3
3.	<a href="#">How the Curator Scores Your Submission</a>	9
4.	<a href="#">Multiple Submissions</a>	13
5.	<a href="#">The Curator and the Honor Code</a>	14
Appendices:		
I.	Verifying File Sizes under Microsoft Windows	15
II.	Scoring Examples from the Curator	16
III.	Some Testing Issues	19
IV.	Deadly Sins	20
V.	Submitting from the Virginia Tech CS Lab	21
VI.	Known Bugs and Alarming Behaviors	21
VII.	Getting Help	21

**Disclaimer:** Every effort has been made to ensure that the contents of this document are accurate and complete. However, the Enhanced Automated Grading System Project is ongoing. It is always possible improvements and bug fixes have been implemented since the last update of this document. The current version of this document will be available from the Curator Homepage:

<http://ei.cs.vt.edu/~eags/Curator.html>

# 1. Introduction

The Curator System (also known as Curator) allows your Instructor to:

- collect programming assignments over the Internet, and to grade those programming assignments automatically.
- collect any type of assignment over the Internet, and archive that assignment for later grading.

You will submit your assignments via your network connection to the Internet, using the Curator Client software. This document describes how to use the Curator Client to submit your work.

When you submit an assignment, the client software will establish a network connection to the server software (usually called the Curator Server, or simply the Grader), and transfer your submission to the computer on which the Curator Server is running. The Curator Server will send you an e-mail message either confirming receipt of your submission, or containing your score for that submission and additional information. A general description of how the Curator scores submissions is given in Section 3, and a detailed discussion is given in Appendix II.

The Curator is a client-server application written in the Java programming language, developed by the Virginia Tech Department of Computer Science, with support from Virginia Tech and Microsoft Research Corporation. The Curator is the latest in a sequence of automated grading systems that have been used by the Virginia Tech Computer Science department over a period of more than 25 years. The immediate predecessor, the New Automated Grading system, was used for over two years, processing more than 9000 submissions from almost 2000 students.

## 2. Submitting an Assignment from Your Computer

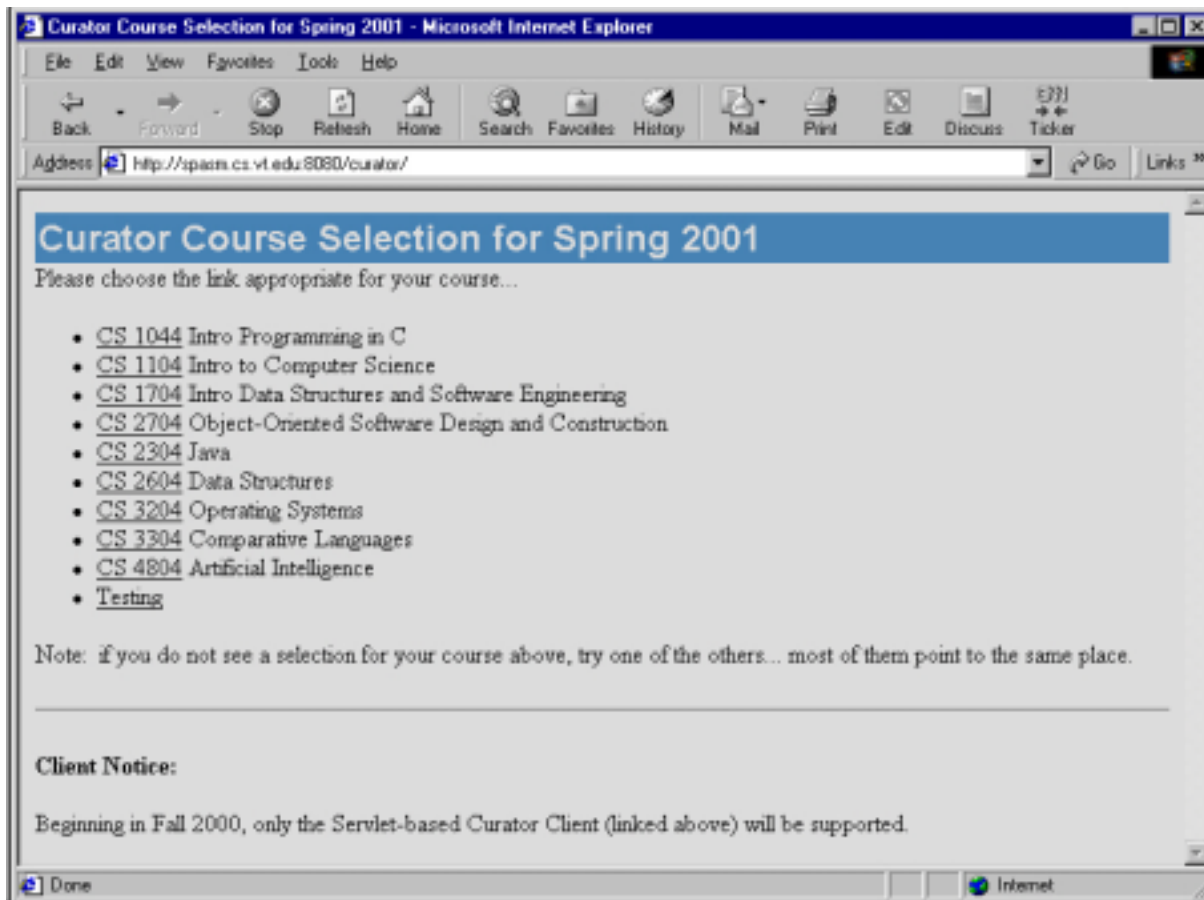
This section describes how to use the Curator Client to submit a file to the Curator Server. No special setup is required before you can submit to the Curator Server.

### Using the servlet Curator Client

There are several versions of the Curator Client software. This manual describes only the servlet version, which requires minimal work for the student. The Curator Client is a Java servlet, which means it is accessed via your Web browser.

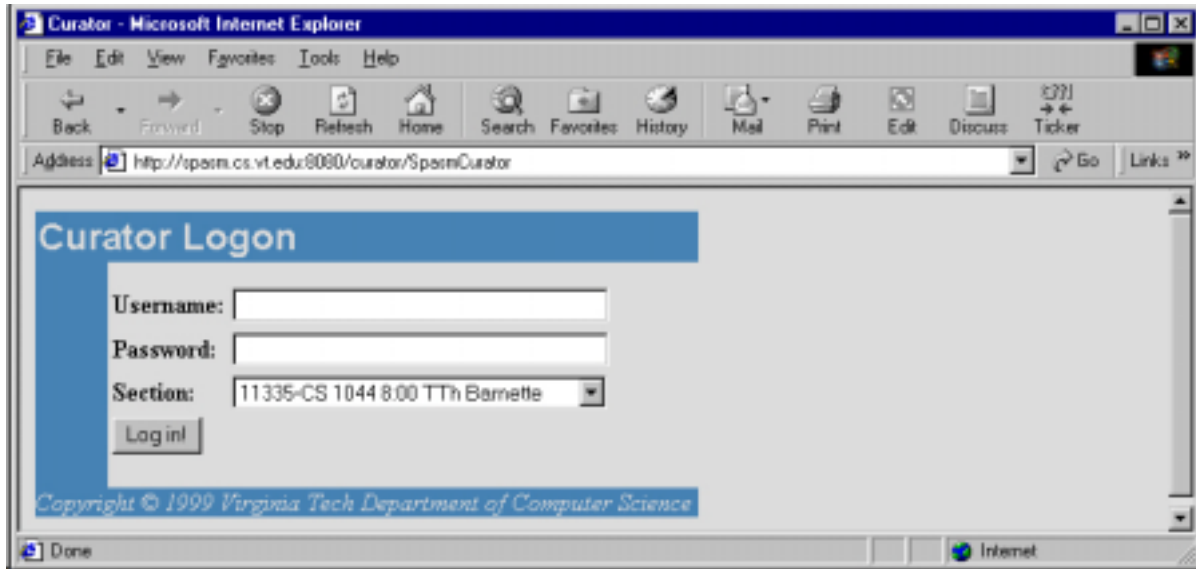
1. **Connect to the Internet:** The Curator Client is designed to work over the Internet. Before you can use it, you must run your communications software and connect to the network.
2. **Connecting to the servlet Client on the Web:** To run the Curator Client from Windows NT/2000/95/98, start your WWW browser and enter the URL given by your course instructor or on the website for your course.

The page requires a few seconds (perhaps a few minutes with a slow network connection) to load. Be patient. After a short time you should see a screen similar to this:



If you see an error dialog box when you attempt to connect to the Curator Client, wait a few minutes and try again. If you consistently receive messages indicating the server is down or unreachable, there is probably a technical problem (hardware or software) that will be corrected shortly. Be patient.

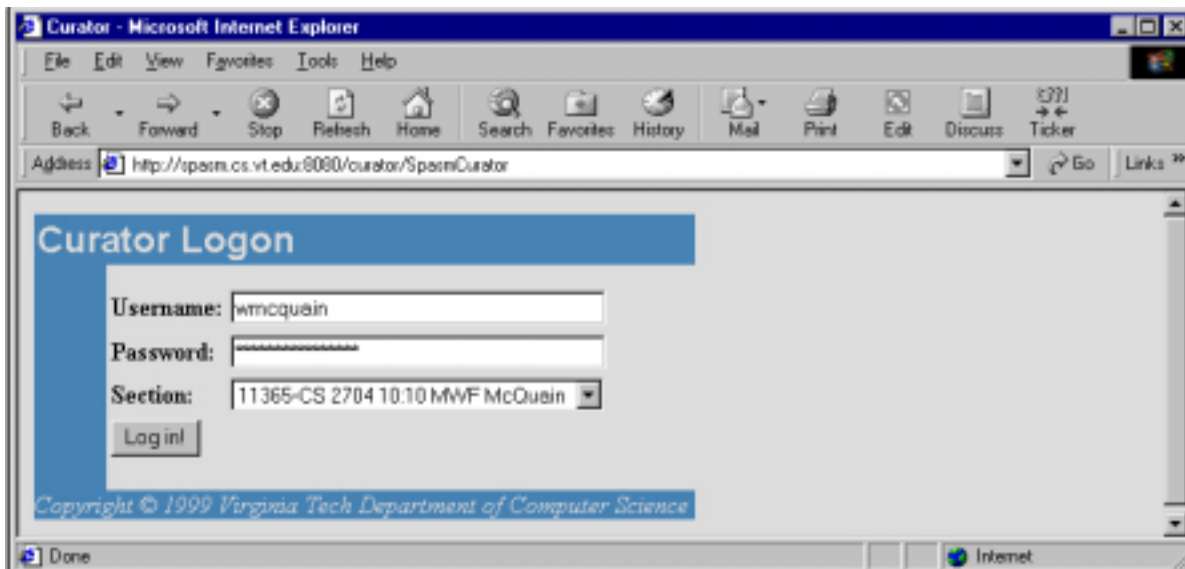
When the page has loaded, there will be a link for your course. Click on that link to access the Curator Client for your course:



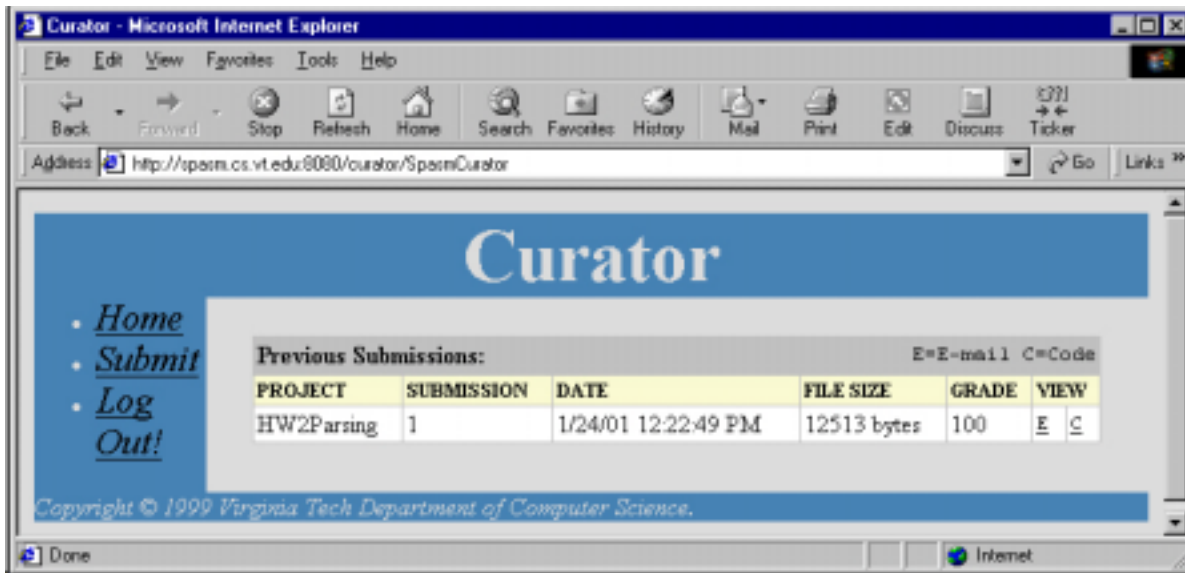
3. **Filling Out the Submission Form:** The first Curator Client window depicts a form that needs to be filled out in order for you to log onto the Curator System and make a submission or check the status of your old submissions. You must fill out all of the information correctly in order to log onto the system. Use the arrow keys, tab key, or the mouse to move between fields. Each of the fields in the form is discussed in detail below:

- **PID:** Enter your University PID (this is the same name you use when you connect to the university network or check your e-mail). This must be your original PID, not an e-mail alias. (Note: don't include @vt.edu)
- **Password:** Type in your password. This would normally be the same password you use to connect to the university network or check your e-mail, HOWEVER a new password system may be tested this term. In that case, you will be informed in class about any changes regarding the assignment of a Curator password. As you type, each letter of your password will appear as an asterisk (\*). This is done to hide your password from the sight of anyone who might be watching over your shoulder. Be careful that you type it correctly.
- **Section:** Click your mouse on the little arrow to get a list of class sections and index numbers. Make sure you select the correct time, instructor, and index number for the class you are enrolled in, or your submission may be misfiled or rejected.

At this point the Curator Client window should look something like:

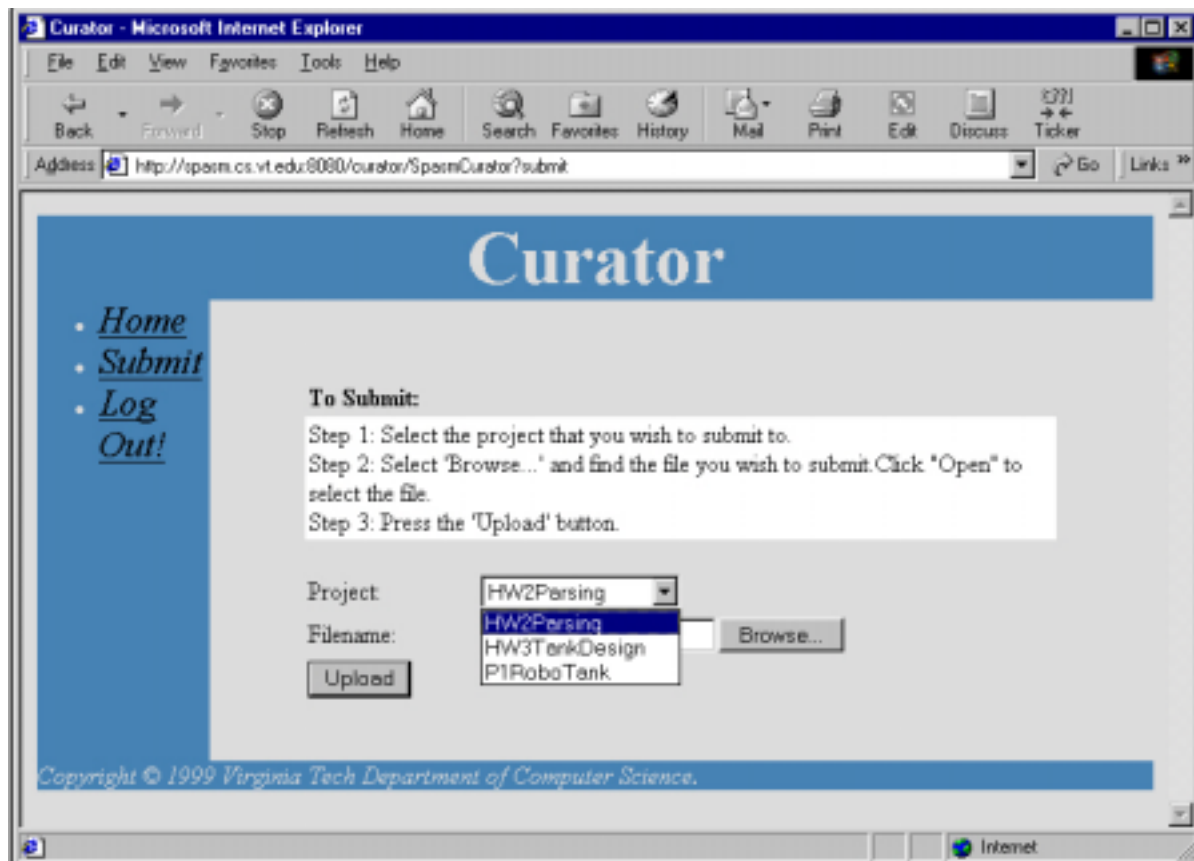


Click the Log in! button to log on to the Curator System. If you've entered all the information correctly you will see your Curator Home page:

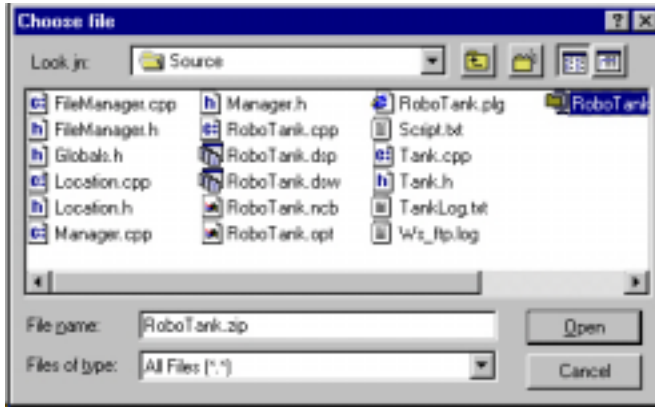


This page lists all your submissions, if you've made any, with timestamp, file size and grade (if applicable). You can view the Curator e-mail message or the source file you submitted by clicking on the appropriate link.

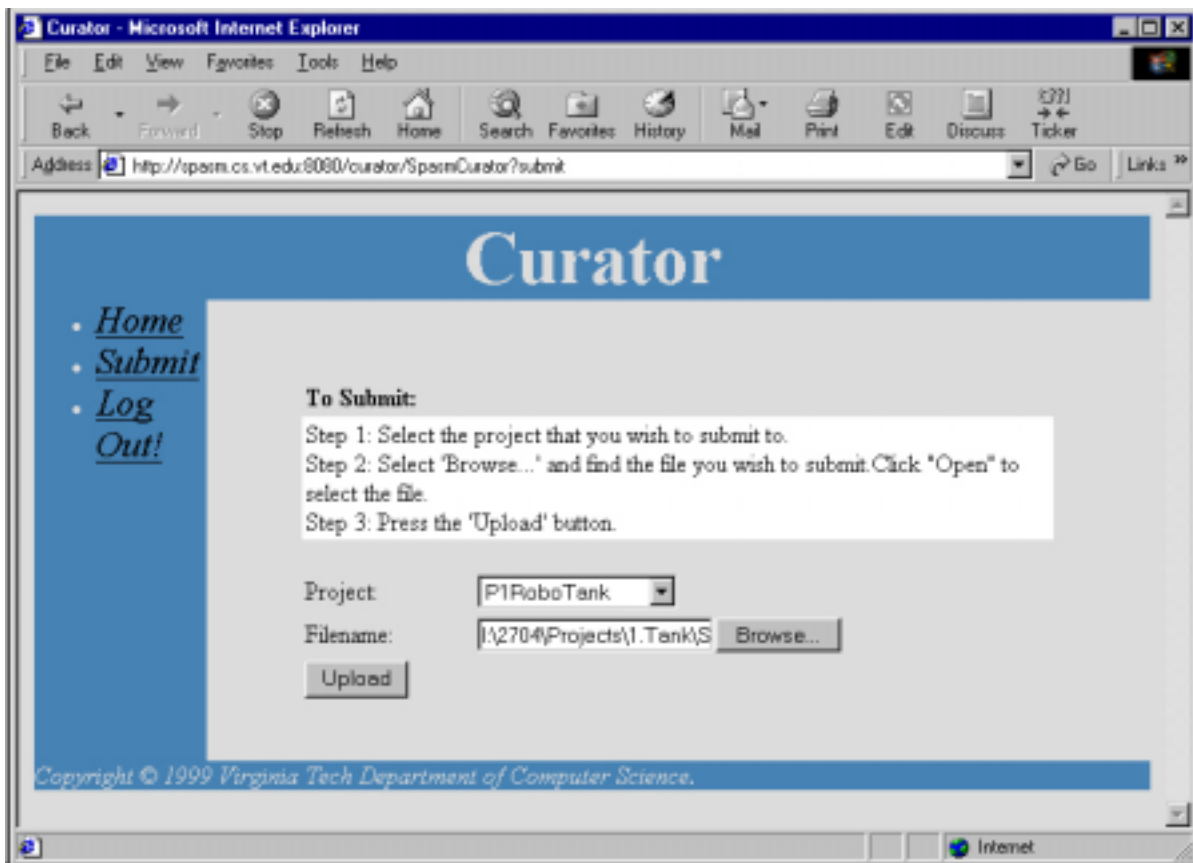
To submit an assignment, click on the Submit link, which will take you to your Curator submission page:



Use the Project drop-list to select the appropriate assignment, and then click Browse. Using the Choose file box (below), select the file you wish to submit and click on Open:



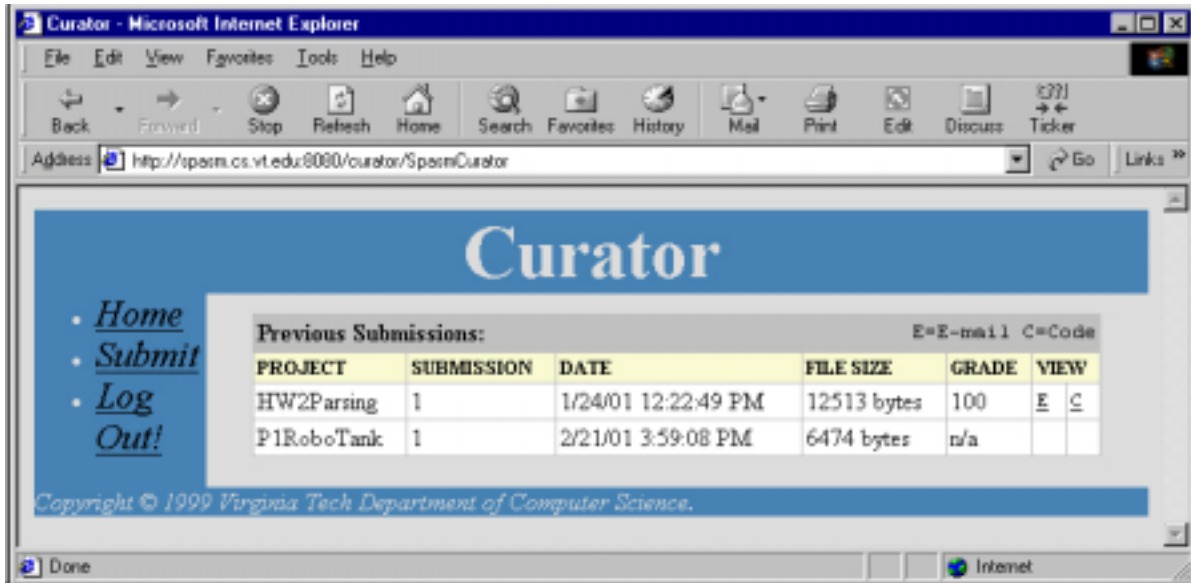
Your submit page should now look like this:



Click the Upload button to submit your file.

Choosing the wrong assignment may result in the rejection of your submission, or in your getting a very low score because Curator files your submission for the wrong assignment.

Return to your Curator Home page (take the Home link) to confirm your submission has been archived and logged. For assignments that are automatically graded, it may take a few minutes for your score information to appear. Just toggle the "refresh" button in your web browser a few times. The submission you just made should be added to the table on your Curator home page:



4. **Getting Results:** The type of response you receive from the Curator Server depends on whether your assignment is to be graded automatically or archived for later grading.

**Archive Response:** If you are submitting an assignment, such as a design outline, which will be graded by hand later, the Curator Server will send you an e-mail message immediately confirming that the file was received, the size of the file, and the time at which the file was archived:

```
Date: Tue, 6 Jul 1999 23:22:52 -0400 (EDT)
From: grader@vt.edu
Message-Id: <199907070322.XAA21957@sable.cc.vt.edu>
Subject: Curator Notification
```

Please do not reply to this email address

Your submission has been accepted and archived:

```
Submit Number: 7
Length: 2958 bytes
Time: Tue Jul 06 23:22:27 EDT 1999
```

You should verify that the file size is correct, since network transmission errors CAN occur. If you don't know how to discover the exact size of a file on your computer, see Appendix I. If the size of your file does not match the size in the e-mail message you received, the file was probably corrupted during transfer and you should resubmit it.

**Automatic Grading Response:** If you are submitting a program that is to be automatically graded, the Curator Server will compile and run your program, test it, and compute a score. It will then send you an e-mail message containing scoring information for your submission. See the following section, How the Curator Scores Your Submission, for further discussion.

Whether your assignment is graded immediately or simply archived, under normal conditions, the e-mail message should be sent within a few minutes. If you do not receive an e-mail notification, you may still check the results via your Curator Home page.

Clicking the link [E](#) in the View column displays your email message to a browser window:

Curator - Microsoft Internet Explorer

File Edit View Favorites Tools Help

### Email for project-HW2Parsing submission #1

```
// PLEASE DO NOT REPLY TO THIS EMAIL MESSAGE
//
// This information is for submit number 1 of 5 allowed submits.
//
//
// -----
// Point Information
// -----
//      Raw Score      : 100
//      Early Bonus    : 0
//      Late Penalty   : 0
//      Blank Line Penalty : 0      (0 mismatches)
//      -----
//      Total Score    : 100
// -----
// Input File
// -----
// : CS 2704 Parsing Homework Input
// :
// : Let's try a couple of precedes commands:
// :
// precedes      Guadalupe      Chaves
// precedes      47             86
```

Clicking the link [C](#) in the View column displays your submitted source code to a browser window:

Curator - Microsoft Internet Explorer

File Edit View Favorites Tools Help

### Submitted Code for project-HW2Parsing submission #1

```
// CS 2704 Spring 2001
// Homework 2: Parsing a delimited input script
//
// Programmer:    Bill McQuain
// Platform:      Win NT Workstation 4.0
// Compiler:      MS Visual C++ V6.0, SP4
// Last modified: January 16, 2001
//
// Description:
// The program processed tab-delimited command lines from an input
// script, named "Parsing.in". There are no data structures in this
// project.
//
// Any line of the input script beginning with a semicolon character
// ';' is treated as a comment.
//
// Any line not beginning with a semicolon is a command line. Each
// command line will begin with a command string, terminated by a
```

Because assignments that are not auto-graded may require you to submit almost any type of file, these two options are only available for auto-graded assignments.

### 3. How the Curator Scores Your Submission

If your assignment is automatically graded you should receive an e-mail message from the Curator Server within a few minutes containing your score. A sample e-mail message from the Curator Server is shown below.

Date: Mon, 26 Jul 1999 11:30:28 -0400 (EDT)  
From: grader@vt.edu  
Subject: Curator Notification

```
// PLEASE DO NOT REPLY TO THIS EMAIL MESSAGE
//
// This information is for submit number 3 of 5 allowed submits.
//
// -----
// Point Information
// -----
//      Raw Score      : 49
//      Early Bonus    : 5
//      Late Penalty   : 0
//      Blank Line Penalty : 0      (0 mismatches)
//      -----
//      Total Score    : 54
// -----
// Input File
// -----
//      7842      49      6377.01      D      Kirk, James
//      2922      69      1423.73      D      Spock
//      5582      45      7059.99      D      Picard, Jean Luc
//      3193      54      2969.71      B      Data
//      6394      44      4362.55      B      LaForge, Geordi
// -----
// Correct Output File: 12 lines
// -----
// [ 0]Programmer:  Bill McQuain
// [ 4]StarFleet Payroll
// [ 0]
// [12]  ID      Gross Pay      FIT      SSI      Ins      Net Pay
// [ 0]=====
// [12] 7842      6377.01      1824.41      435.55      180.00      3937.05
// [12] 2922      1423.73      268.64      97.24      250.00      807.84
// [12] 5582      7059.99      2049.80      482.20      180.00      4348.00
// [12] 3193      2969.71      701.52      202.83      100.00      1965.36
// [12] 6394      4362.55      1159.64      297.96      100.00      2804.95
// [ 0]=====
// [24] Avg:      4438.60      1200.80      303.16      162.00      2772.64
// -----
// Student Output File: 12 lines
// -----
// Joe Bob Hokie
// Macro$oft Corporation Payroll
// -----
// IdNum      Gross Pay      F.I.T.      SSI      Ins.      Net Pay
// =====
// 7842      6377.01      2104.41      435.55      220.00      3617.05
// 2922      1423.73      398.64      97.24      400.00      527.84
// 5582      7059.99      2329.80      482.20      220.00      4028.00
// 3193      2969.71      831.52      202.83      160.00      1775.36
// 6394      4362.55      1439.64      297.96      160.00      2464.95
// =====
// Avg:      4438.60      1420.80      303.16      232.00      2482.64
// -----
// -----
```

Submission Info

Point Info

Test Input

Correct Output

Student Output

The e-mail notification includes the following sections:

- **Submission Information:** The message tells you which submission this scoring was for and how many submissions are allowed for this project.
- **Point Information:** Provides your score, with a breakdown indicating any early bonus or late penalty points assessed. Your Instructor will determine how bonus and/or penalty points are to be assigned.
- **Test Input File:** Shows the input file on which this submission was tested.
- **Correct Output File:** The output file produced by the Instructor's solution using the preceding input file.
- **Student Output File:** The output file your submission produced from the same input file.

Your score is determined entirely by how nearly correct your output is, and how early or late your submission is. The Curator compares your output file to the correct output file, line by line. Each line is broken into "tokens" consisting of characters separated by whitespace (blanks or tabs). The tokens from each line of your output file are compared to those from the corresponding line of the correct output file; your score for each line is determined by the number of correct tokens you produce and the number of points assigned to that line. The point value assigned by your Instructor to each line is shown in square brackets at the beginning of each line of the correct output file.

Note that the amount of horizontal space you put between tokens does not generally matter to the Curator. Of course, if you manage to run two tokens together so the Curator treats them as a single token (for example, printing "GrossPay" instead of "Gross Pay"), then you will lose points. And, if you separate what should be a single token in two (for example, printing "F I T" instead of "FIT") you will also lose points.

It is important that your output file not contain any extra lines, or omit any lines. If you do have extra lines or missing lines, then the Curator may compare the wrong lines, in which case you will probably receive a very low score.

**Extra or missing lines:** Notice that some of the lines are given a value of 0 points. For instance, the first line is supposed to contain the name of the programmer. "Joe Bob Hokie" doesn't match the line in the correct output. In fact, the label "Programmer" is missing as well. But that's OK since the point value of the line is 0. That means that you don't have to match anything on that line exactly; however, if you don't have a line there it may throw the comparison off.

If your output file has extra nonblank lines, when compared to the correct output file then you will be penalized for each of those; the default deduction for each extra nonblank line is 10 points. Compare the line counts given in the email message to see if you have too many or too few lines of output. On the other hand, if your output has fewer lines than the correct output, then you're penalized the number of points that are assigned to those lines.

How extra or missing blank lines are handled depends on where they occur. Extra blank lines at the end of your output file should be ignored by the Curator. If you insert an extra blank line in the middle of your output file, the Curator will assess a penalty for that, determined by your instructor, and then attempt to resynchronize its comparison by reading the next line of your output. If you omit a specified blank line, that is handled in a similar manner. Note that the e-mail header specifies how many mismatches there were involving blank lines and the total penalty assessed for them.

**Other messages, other problems:** The two example Curator e-mail messages discussed above are the most common sort. However, there are several other scenarios that you should be aware of. If your submission does not compile (without errors) you will receive a message similar to:

```
Your source code failed to compile.  
Please correct your code so that it compiles and then submit it again.  
Your score for this submission is ZERO.
```

```
Compiler error messages follow:  
Compiling student program ...  
c:\CuratorRoot\CompileArea\johokie\nazmul.cpp(14) : error C2065: 'cnst' : undeclared  
identifier  
c:\CuratorRoot\CompileArea\johokie\nazmul.cpp(14) : error C2144: syntax error :  
missing ';' before type 'float'
```

Of course, you should usually not submit a program that does not compile since that guarantees a score of zero. Two possible causes of this are using a different compiler (the Curator uses MS Visual C++ 6.0) or submitting the wrong file. There have even been instances where students have submitted term papers to the Curator – you shouldn't be surprised that a term paper usually doesn't compile properly as a program in C++ or any other programming language.

The Curator gives your program a fixed amount of time to finish (the default is 10 seconds). If your program has not finished within the time limit, the Curator kills your program and you'll receive the message:

```
Your source code failed to exit and took too long to execute.
Possible problems include:
- an infinite loop
- a runtime error such as an array index out of bounds or divide by zero
- your program expects input from the keyboard
- your program uses an incorrect name for the input file
Please correct your code so that it terminates properly.
Your score for this submission is ZERO.
```

When this happens, you need to do further testing and fix the problem before resubmitting. There is absolutely no point in resubmitting the same source code. This sort of problem may occur for a variety of reasons, including but not limited to the ones listed in the Curator message. The message will also include the input file and correct output, so you may use that input file to try to determine why your program misbehaved.

It is also possible that your program may be terminated abnormally (killed) by the operating system on the Curator machine before the time limit has expired. In that case, the Curator doesn't actually "know" that happened. It will look for an output file to score as usual and you will receive an e-mail message similar to the first one discussed above. However, if this happens you may notice that your output is incomplete since your program did not run to a normal termination. If that happens, you need to debug your program to eliminate the error before using another submission.

You may also find that when you run your program on your computer with the input file the Curator used, your program produces a complete output file and appears to operate correctly. If the Curator indicates your program is not producing correct results, but it does something different on your computer, the most likely explanation is that you are testing your program under Windows 95/98. The Curator operates on a machine using Windows NT, which is much less forgiving of misbehaving programs than Windows 95/98. You may see different behavior because of a variety of errors, including those listed in the e-mail message above and also failure to initialize variables before using them.

You may not be able to discover the source of the problem and fix it under Windows 95/98. In that case, you should test your program in the Computer Science Undergraduate Lab on one of the computers equipped with Windows NT. Understand: the fact that your program appears to run correctly under Windows 95/98 but not under Windows NT indicates only that Windows 95/98 tolerates bad behavior, not that your program is correct.

You may also get an e-mail message similar to:

```
Your source code failed to produce the output file: payola.dat
Possible problems include:
- you specified the wrong output file name
- you specified the wrong input file name
- your program had a runtime error and was terminated
  by the system
Please correct your code so that it produces this output file.
Your score for this submission is ZERO.
```

In this case, the Curator couldn't find your output file. The correct name for your output file will be given in the specification for your assignment. If you use another name, the Curator will not find your output. If your program is terminated by the system because of a runtime error, before it produces any output, you'll see the same message. Again, test and determine what the problem is, and fix it, before using another submission.

**The moral of all this is simple:**

**Follow the project specifications precisely!**

**Grading Options Set by Your Instructor:** Your Instructor has quite a bit of flexibility when setting up grading on the Curator. Here's a short list:

- The date your program is due. Programs are always due at midnight on that date.
- The date on which submissions of an assignment are no longer accepted (a.k.a. the drop dead date).
- The number of submissions you are allowed.
- The total number of points a program is worth, and how those points are distributed to lines in the output file.
- The number of bonus points you will receive for each day your program submission is early, and the maximum early bonus you can earn.
- The number of late points you will lose for each day your program submission is late, and the maximum late penalty you can suffer. (Early and late points are calculated based on a due time of midnight on the due date and fractional days are not used.)
- The number of points to be deducted each time you have a blank line that shouldn't be there or omit a blank line that should be there (called a blank line mismatch), and the maximum number of points you can lose for that type of error.
- The number of points to be deducted for each extra nonblank line you have at the end of your output file, and the maximum number of points you can lose for that type of error.

Some of these may vary from assignment to assignment, and will usually be specified in the assignment statement. Others will usually be the same for all your assignments (although not for all instructors). Any questions about these settings should be addressed to your GTAs or Instructor, not to the Curator administrator.

## 4. Multiple Submissions

Some instructors allow programming assignments to be submitted more than once. This gives students a chance to fix problems that were detected by the Curator. The number of submissions that are allowed depends on your instructor. The Curator server will automatically ignore any submissions you send beyond the maximum number allowed by your teacher. How grades are assigned to multiple submissions are also up to the instructor. Some will use the highest grade from all of your submissions. Others may use the grade from your last submission.

If multiple submissions are allowed, the Curator will typically use a different input file each time. These input files are generated by a program provided by your Instructor, and should conform to the program specification provided for your assignment.

You should send another submission only after you have received the results from the previous submission, determined what errors you had in that version of your program, and attempted to fix those errors.

### Tips on Using Multiple Submissions Effectively

Here are a few tips that can help you use multiple submission effectively (if your instructor allows multiple submissions). Following these hints should help you get the most out of your programming assignments.

**Test your program thoroughly before you send it.** Sending in a program that has not been adequately tested will very likely result in a poor grade. Your instructor may have supplied some example input and output data with the assignment specification. Testing your program with only one input sample is not enough to assure that your program is correct. In fact, it is generally impossible to ensure your program is entirely correct merely by testing it. However, the more testing you do before submitting, the higher the probability you will achieve a good score.

The Curator uses its own set of input and output data which will conform to your program specification, but may provide a more rigorous test of your program than your assignment's sample data. Try a wide range of input values, and study the output carefully to be sure that it is correct.

**Use the results from a submission to diagnose your problem.** When you get the results back from the Curator on your last submission, it will include the input data file the Curator used to test your program, the correct output, and the output from your program. If points were deducted, study the correct output to see if you can determine what you are doing wrong. When you try to fix the problem, test it with the Curator input file so you can compare it to the correct output. It is important to remember that the Curator uses different input data for every submission, so make sure that you design your program so that it works for all possible ranges of input values, not just the assignment's sample input or the input that came back from a previous submission.

The bottom line on point deductions is this: the Curator only deducts points if your program is late or if there is a difference between the output your program produced and the correct output produced by your Instructor's solution. In some cases the difference may be subtle and you may have difficulty finding it. That is all part of the programming experience.

**Be careful of deadlines.** The clock on the Curator machine is synchronized with a timeserver, so we have great faith in its accuracy. Remember that deadlines are ill-tempered beasts. Hugging one too tightly may result in a bite.

## 5. The Curator and the Honor Code

Each program you submit to the Curator is subject to the Virginia Tech Honor Code, just as if you had given the program to a human for evaluation.

Your Instructor will specify precisely what sources of help are allowed and how much, if any, collaboration with other students is allowed. Be certain you understand and follow the rules set by your Instructor. Ignorance of those rules is not an effective defense before the Virginia Tech Honor Court.

Regardless of the rules set by your Instructor, each of the following is considered a flagrant violation of the Honor Code and will result in a formal charge:

- submitting a program written, in whole or in part, by another student as your own, individual work
- submitting a program designed to alter or circumvent the operation of the Curator's scoring mechanism
- submitting a program designed to crash or otherwise damage the operation of the Curator software or the machine on which it is installed
- editing computer generated output, such as an e-mail message sent by the Curator, and presenting that altered version when raising a question relating to the scoring of your program
- attempting to access or alter files on the machine on which the Curator is installed, whether physically or via a network connection; the sole exceptions would be normal use of the Curator client and QueryTool, and if the Curator machine is also used as an FTP repository for course-related files

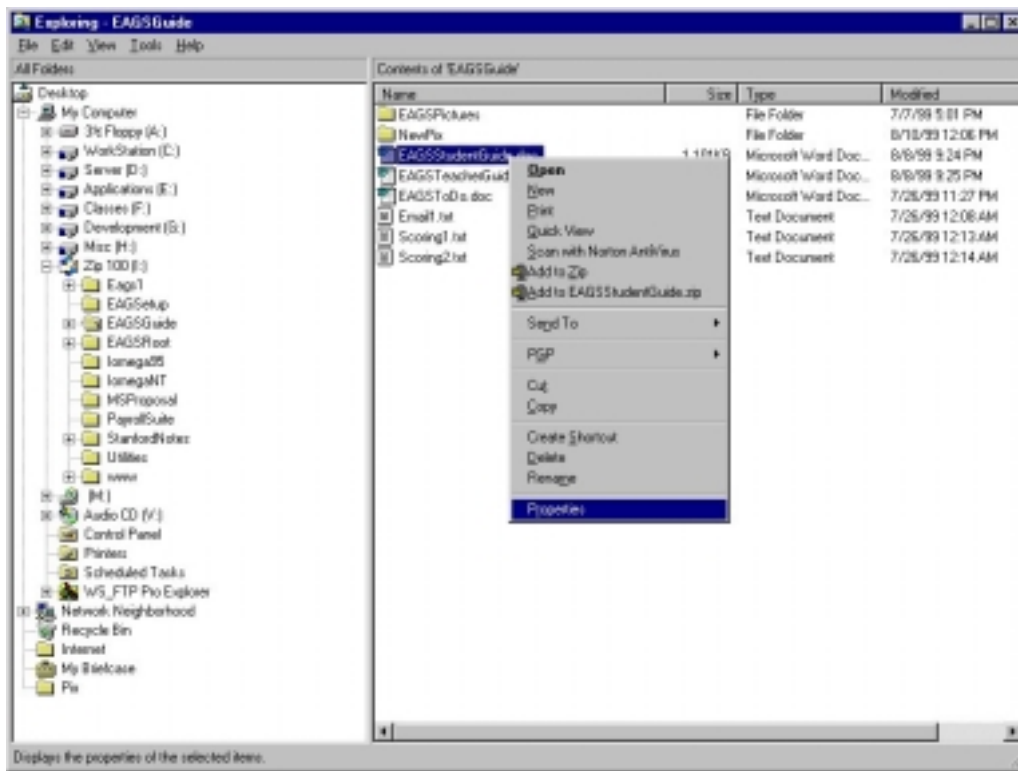
This list is not intended to be comprehensive; resolve any questions you have about these policies with the Instructor of your course.

All submissions to the Curator are archived. The programs submitted to the Curator are automatically analyzed for suspicious similarities. When such similarities are found, the programs involved are compared (by humans) and charges are filed with the Honor Count if the similarities warrant action.

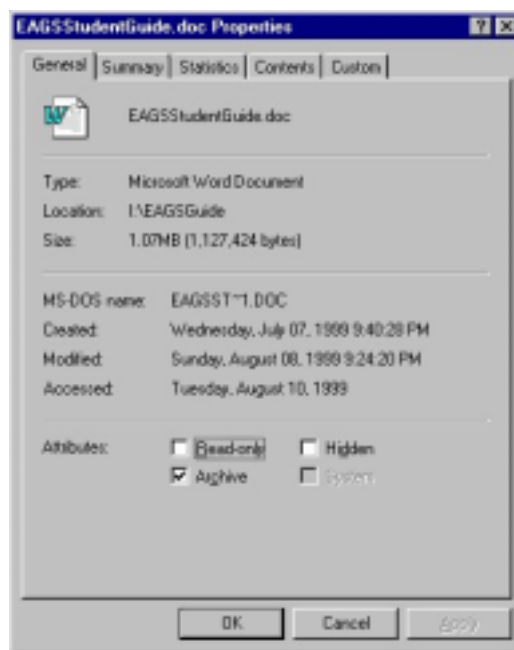
The Honor Code will be strictly enforced by the Instructors and GTAs who use and administer the Curator. All assignments submitted shall be considered pledged graded work, unless otherwise noted. All aspects of your work will be covered by the Honor System. Honesty in your academic work will develop into professional integrity. The faculty and students of Virginia Tech will not tolerate any form of academic dishonesty.

## Appendix I Verifying File Sizes under Microsoft Windows

Here's how to find the exact size of a file on your Windows system. Using Windows Explorer or going through My Computer, find and open the folder containing your file. Point to the file with your mouse and click the **right** mouse button to bring up a pop-up menu:



Now click on "Properties" at the bottom of the pop-up menu. That will bring up a dialog box that shows basic information about the file. The exact file size, measured in bytes, is shown near the middle of the dialog:



In this case, the file CuratorStudentGuide.doc is 1,127,424 bytes in size.

## Appendix II Scoring Examples from the Curator

This section provides a detailed discussion of how the Curator scores student assignments. Consider the correct output and student output sections shown in the e-mail message shown on the following page. Basically, the Curator compares the two output files, line by line from the beginning.

The following table shows the comparisons that are made, and highlights the student tokens that do not match the Instructor's output. The Curator stores line deductions internally as doubles (rounded here to two digit precision). The final score is rounded to the nearest integer for recording.

Line	Value	Tokens compared	Deduction
1	0	Inst: None Stud: None	0.00
2	4	Inst: "Newton's" "Law" "of" "Cooling" Stud: "Newton's" "Law" "of" "Cooling"	0.00
3	0	Inst: None Stud: None	0.00
4	10	Inst: "Initial" "temp" "of" "object:" "748.64" Stud: "Initial" "temp" "of" "object:" "800.00"	2.00
5	10	Inst: "Temperature" "of" "medium:" "150.17" Stud: "Temperature" "of" "medium:" "100.00"	2.50
6	0	Inst: None Stud: None	0.00
7	6	Inst: "Time" "Temp" "Rate" Stud: "Time" "Temp" "Rate"	0.00
8	0	Inst: None Stud: None	0.00
9	14	Inst: "1.0" "719.45" "28.46" Stud: "1.0" "765.86" "33.29"	9.33
10	14	Inst: "3.0" "665.28" "25.76" Stud: "30" "702.50" "30.12"	9.33
11	14	Inst: "50" "616.26" "23.30" Stud: "50" "645.16" "27.26"	9.33
12	14	Inst: "7.0" "571.90" "21.09" Stud: "7.0" "593.28" "24.66"	9.33
13	14	Inst: "9.0" "531.77" "21.09" Stud: "9.0" "546.34" "22.32"	9.33
14	0	Inst: None Stud: None (no comparison since Instructor line is worth 0 points)	0.00
15	14	Inst: No more data Stud: Extra line and it's a line of delimiters, not blank.	10.00

So the comparison of tokens results in a total deduction of approximately 61.17 points, including the deduction for the extra nonblank line. That yields a raw score of 38.83 which is rounded to 39 for recording. The student submitted the program early and a bonus of 2 points was assessed, producing the total score of 41.

**Post Mortem Analysis:** Comparing the Student and Instructor's output, it is evident that both the Temperature and Rate calculations are incorrect. In addition, the student has printed incorrect (and irrelevant) values in the table header, and has printed the last line of the output table twice. These observations suggest where the student should look for errors.

E-mail from a corrected submission is shown on page 23. In this case, the student has fixed all the original mistakes except the one that causes the last line of the output table to be duplicated. The student has also introduced an unspecified blank line at the beginning of the table, resulting in a deduction.

Date: Mon, 26 Jul 1999 11:30:28 -0400 (EDT)  
From: grader@vt.edu  
Subject: Curator Notification

PLEASE DO NOT REPLY TO THIS EMAIL MESSAGE

This information is for submit number 1 of 4 allowed submits.

---

Point Information

---

```
Raw Score      : 39
Early Bonus    : 2
Late Penalty   : 0
Blank Line Penalty : 0 (0 mismatches)
-----
Total Score    : 41
```

---

The Raw Score reflects all deductions for mismatched output tokens, but no adjustments for early or late submissions, or deductions for mismatches involving blank lines.

---

Input File

---

```
Object      Medium
748.64      150.17
Time
1.0
3.0
5.0
7.0
9.0
```

---

---

Correct Output File: 14 lines

---

```
[ 0]Programmer:  Bill McQuain
[ 4]Newton's Law of Cooling
[ 0]
[10]Initial temp of object:  748.64
[10]Temperature of medium:  150.17
[ 0]
[ 6]  Time      Temp      Rate
[ 0]-----
[14]  1.0      719.45    28.46
[14]  3.0      665.28    25.76
[14]  5.0      616.26    23.30
[14]  7.0      571.90    21.09
[14]  9.0      531.77    19.08
[ 0]-----
```

---

Both temperatures are incorrect. CURATOR deducts 2.0 for the first and 2.5 for the second.

Each of the temperature and rate values is incorrect. CURATOR deducts about 4.667 points for each. Since there are 10 incorrect values, that results in a total deduction of 46.67 points for this error.

---

Student Output File: 15 lines

---

```
Programmer:  Joe Bob Hokie
Newton's Law of Cooling

Initial temp of object: 800.00
Temperature of medium:  100.00

      Time  Temp  Rate
-----
1.0    765.86  33.29
3.0    702.50  30.12
5.0    645.16  27.26
7.0    593.28  24.66
9.0    546.34  22.32
9.0    546.34  22.32
```

---

This line is compared to the delimiter line above, resulting in no deduction.

This is an extra nonblank line, resulting in a deduction of 10.0 points.

Date: Mon, 26 Jul 1999 11:52:54 -0400 (EDT)  
From: grader@vt.edu  
Subject: Curator Notification

PLEASE DO NOT REPLY TO THIS EMAIL MESSAGE

This information is for submit number 2 of 4 allowed submits.

---

Point Information

---

Raw Score : 90  
Early Bonus : 2  
Late Penalty : 0  
Blank Line Penalty : -2 (1 mismatches)  
-----  
Total Score : 90

This indicates how many times there was a blank line in the correct output, but not in the student's output, or vice versa. The total deduction for such mismatches is also shown.

---

Input File

---

Object Medium  
640.79 86.33  
Time  
1.0  
6.0  
11.0  
16.0  
21.0

---

Correct Output File: 14 lines

---

```
[ 0]Programmer: Bill McQuain
[ 4]Newton's Law of Cooling
[ 0]
[10]Initial temp of object: 640.79
[10]Temperature of medium: 86.33
[ 0]
[ 6] Time      Temp      Rate
[ 0]-----
[14]  1.0      613.75    26.37
[14]  6.0      497.08    20.54
[14] 11.0      406.23    15.99
[14] 16.0      335.46    12.46
[14] 21.0      280.36     9.70
[ 0]-----
```

---

Student Output File: 16 lines

---

Programmer: Joe Bob Hokie  
Newton's Law of Cooling  
  
Initial temp of object: 640.79  
Temperature of medium: 86.33

The student has introduced an extra blank line at the beginning of the table body, accounting for the blank line mismatch reported above. This results in a deduction of 2.0 points.

---

Time	Temp	Rate
1.0	613.75	26.37
6.0	497.08	20.54
11.0	406.23	15.99
16.0	335.46	12.46
21.0	280.36	9.70
21.0	280.36	9.70

---

The student is still repeating the last line of the output table. So, the last line is still counted as an extra nonblank line, just as before, resulting in the same deduction of 10.0 points.

## Appendix III Some Testing Issues

The Curator Server will be run under Windows NT Server 4.0.

Ideally, students will develop and test their programming assignments under Windows NT as well. If that is not the case, students may discover that their programs behave differently during Curator testing than on their systems. It is probably impossible to give a comprehensive list of causes, but here are some that we have seen at Virginia Tech, where most of our introductory programming students are developing programs under Windows 95 or Windows 98.

**File Names:** One of the most common errors we have seen is the use of an incorrect name for the input and/or output files. These names must be specified prominently in the assignment statement. Even so, many students will fail to follow instructions. Several scenarios may play out if a student uses the wrong name for the input file. The most common is that his/her program will produce a trivial output file, terminating normally. It is also possible that a design flaw may lead to an infinite loop or runtime error in this case. If a student uses the wrong name for the output file, the Curator will not find any output, assign a score of zero, and notify the student that no output was produced. This typically perplexes students because they are using the same incorrect file names when testing their program, and hence do not reproduce the error.

**Array Indices:** Failure to restrict array indices to valid ranges can lead to a plethora of unfortunate behaviors. Some of these will be exhibited regardless of the operating system being used, others only with systems that provide adequate memory protection for user processes.

It is possible for an out-of-bounds array index to cause no logical errors. In that case, the program may appear to be correct when tested under Windows 9x. Executed under Windows NT, the same program may exhibit a runtime access violation, or simply produce different results.

One nasty truth is that the behavior of a program containing this type of error may depend upon the contents of system memory and hence be nondeterministic. Another nasty truth is that memory protection faults are not reliably detected by the Windows 9x kernel. It is important to emphasize to students that the fact their program produces a correct result under Windows 9x does not mean their program is correct, merely that it caused no error Windows 9x is capable of detecting. Requiring students to make proper use of asserts or exceptions can alleviate this problem.

## Appendix IV Deadly Sins

Here are some simple things you can do to increase the probability you will receive low scores on your programs:

**1. Submit an untested program.**

Testing is the programmer's responsibility. Submissions to the Curator are precious; using the Curator to test your programs is a good way to ensure you will never receive a good score.

**2. Submit an inadequately tested program.**

Test data files are provided for each assignment. Usually several sets of input and correct output are provided. Just because your program works correctly with one test case doesn't imply it will work with others. In fact, it's usually impossible to test any program thoroughly enough to prove it's correct. That said, the more testing you do, the more likely you are to discover your errors yourself.

**3. Use incorrect names for the input and/or output files.**

Do this and you'll never find the data you're supposed to be processing... or you'll write your output somewhere the Curator will never look. Either way, you'll receive a very low score.

**4. Fail to initialize all of your variables.**

Do this and your program may behave differently on different computers. In fact, your program may behave differently every time you run it. Or maybe not... This is especially important with counters, running totals, and any array variable.

**5. Fail to properly control array index values.**

This is discussed in the course notes and in Appendix V. All sorts of puzzling and nasty behavior can result.

**6. Fail to check results and to take those results seriously.**

Count on this: if the Curator deducts points, there is a difference between your output and the correct output.

**7. Submit the wrong file.**

Careless error, and costly.

## Appendix V Submitting From the CS Lab

If you do not have Internet access at home, you can make a submission from the Computer Science undergraduate laboratory, provided you have requested an account on the lab machines. Bring the assignment you want to submit on a floppy disk to McBryde Hall room 116-118. Ask for a system running Windows NT. Log on using your PID and lab account password, and then follow the instructions given in section 2 for submitting your work.

## Appendix VI Known Bugs and Alarming Behaviors

**Curator Server Problems:** In most cases, if a student submits a program that commits a runtime error (such as a divide by zero or an infinite loop), the Curator will simply kill the program, assign a score of zero, and proceed with the next submission. However, it is possible that some programs may misbehave in such a way that Windows NT prevents the Curator from killing the program. In such cases, the Curator may fail to score, or incorrectly score, any subsequent submissions from **that** student correctly until a human operator kills the offending program.

No submissions will be lost in this situation, and the Curator will rescore any pending project submissions, with the correct timestamp, once the misbehaving program is killed. You may, however receive multiple e-mail messages in the interim, each indicating a problem with your program and a score of zero. Don't be alarmed by this if you're sure your program does run properly, but notify your GTA or Instructor, or the Curator Administrator ([grader@cs.vt.edu](mailto:grader@cs.vt.edu)) to be sure the problem is fixed as quickly as possible.

**Enrollment Issues:** The Curator will indicate you are not enrolled in the class if your PID is not correctly recorded in the roll file the Curator uses. This may be because you selected the wrong section, or because there is an error in the roll file. If the Curator insists you are not enrolled in your section, contact your GTA or Instructor as soon as possible to resolve the problem. Note that the Curator uses an internal roll file; it does not access University records to confirm your enrollment.

**E-mail Notification Problems:** The rate at which the Curator is receiving submissions will determine how long it takes for an e-mail message to be sent to you; normally you should receive a message within 5 minutes. In some cases, the Curator may fail to get a connection with the campus e-mail server when it attempts to send your e-mail scoring notification. In that case, you will not receive an e-mail message, although the scoring and recording of your grade will be completed. If you do not receive an e-mail notification, you may use the Curator Client to check your submission.

## Appendix VII Getting Help and Reporting Problems

**Sources of Help:** If you need additional help using the Curator Client, see your Instructor or your GTA, not necessarily in that order. If you have questions about the score your program received, always bring or e-mail a copy of the e-mail message sent to you by the Curator.

**Bug Reports:** Problems and possible bugs can be reported to the Curator Administrator by e-mail:

[eags@ei.cs.vt.edu](mailto:eags@ei.cs.vt.edu).

Do not expect an immediate reply to email sent to this address. The Curator project staff is at low ebb.

**Do not** send an e-mail reply to the messages from the Curator — no one will either read or respond to such messages.