

Spring 1-9-2022

## Automated Grading Tools for Compter Programming

Ed Jorgensen

University of Nevada, Las Vegas, ed.jorgensen@unlv.edu

Follow this and additional works at: [https://digitalscholarship.unlv.edu/btp\\_expo](https://digitalscholarship.unlv.edu/btp_expo)



Part of the [Scholarship of Teaching and Learning Commons](#)

---

### Recommended Citation

Jorgensen, Ed, "Automated Grading Tools for Compter Programming" (2022). *UNLV Best Teaching Practices Expo*. 161.

[https://digitalscholarship.unlv.edu/btp\\_expo/161](https://digitalscholarship.unlv.edu/btp_expo/161)

This Poster is protected by copyright and/or related rights. It has been brought to you by Digital Scholarship@UNLV with permission from the rights-holder(s). You are free to use this Poster in any way that is permitted by the copyright and related rights legislation that applies to your use. For other uses you need to obtain permission from the rights-holder(s) directly, unless additional rights are indicated by a Creative Commons license in the record and/or on the work itself.

This Poster has been accepted for inclusion in UNLV Best Teaching Practices Expo by an authorized administrator of Digital Scholarship@UNLV. For more information, please contact [digitalscholarship@unlv.edu](mailto:digitalscholarship@unlv.edu).

# Automated Grading Tools for Computer Programming

Dr. Ed Jorgensen, Department of Computer Science

## Teaching Practice & Need it Addresses

This teaching practice addresses **automated grading tools for computer programming** assignments. Similar tools are in use at UNLV for other disciplines (e.g., TurnItIn). This is the first use of automation specifically for grading computer programs. This tool was initially applied to first and second programming courses (CS 135, 10 sections and CS 202, 8 sections) in Fall 2021.

## Evidence it Benefits Students

Benefits to students are both direct and indirect.

### Direct Benefits:

- A direct benefit includes very **fast scoring feedback** scoring (~5-10 minutes).
- If the student does not receive the expected score, the student can **correct and re-submit** until the final due date.
- If the cause of an issue is not clear, the student can ask for assistance based on the feedback which **allows the instructor or TA to provide more specific assistance**. Again, the student can correct and re-submit.

### Indirect Benefits:

- An indirect benefit is that the instructor or TA can **spend more time helping students** and less time scoring assignments.
- Another key benefit includes **scoring consistency** across different instructors/TAs and different sections. Based on a CS Masters Thesis (SP22) using a ~300 student test, the Levene's test for homogeneity of variance showed a large statistical difference in variance for manual-based scoring.

## Initial Learning Curve

To be most effective, programming projects must be developed so they can be tested in stages. Using the tool requires a fully developed solution prior to issuance of the assignment. This requires a more up-front effort for the instructor.

## Resources & Where to Find Them

The **codeGrade** tool is integrated into the UNLV Canvas system (listed under External Tool). However, it can only be used to score computer programs (any programming language).

Assignment #10  
Due at 2022-01-04 23:59

Course feedback

Latest submission Upload files

### Assignment Submission Page (step 1)

Assignment #10  
Due at 2022-01-04 23:59

Course feedback

linkedQueue.h	5.7 KB	✕
linkedStack.h	4.6 KB	✕
makefile	0.3 KB	✕
reverse.cpp	3.3 KB	✕

[Click here or drop file\(s\) to upload.](#)

The assignment is due in 2 days. Submit

### Assignment Submission Upload (step 2)

## References/Acknowledgments

Little formal research has been done regarding the specific benefits and impacts of automated program scoring. As such, work is being initiated locally to more fully evaluate.

Some of this effort was initiated under CRRSA grant GR12558/AC01360. Implementation was supported by the College of Engineering and the Office of Information Technology.

Test Student at 2022-01-02 14:15

Code Feedback Overview AutoTest

Functionality - Basic Linked Queue Operations Options · 100 %

No	Summary	Score	Pass
> 1	Compile Ordered Main Run <code>g++ -Wall -pedantic -g -std=c++11 -o testQueue testQueue.cpp</code> and check for successful completion.	0.1 / 0.1	✓
> 2	Must Compile Stop when you achieve less than 1% of the points possible.		✓
3	Ordered Linked List Test	2 / 2	
> 3.1	Test Ordered Linked Lists Run <code>./testQueue</code> and match its output to an expected value.	2 / 2	✓
> 4	Memory Leak Test Check the code quality using <code>valgrind --error-exitcode=1 ./testQueue</code> . Points will be deducted for each comment. Per <b>fatal</b> comment you will lose 100% of the points. Per <b>error</b> comment you will lose 20% of the points. Per <b>warning</b> comment you will lose 10% of the points. Per <b>info</b> comment you will lose 5% of the points.	1 / 1	✓

Functionality - Reverse Audio Samples Options · 100 %

No	Summary	Score	Pass
> 1	Compile Run <code>make</code> and check for successful completion.	0.1 / 0.1	✓
> 2	Verify Compilation Stop when you achieve less than 0% of the points possible.		✓
3	Audio Clip #1	0.5 / 0.5	
> 3.1	Reverse Audio Clip #1 Run <code>./reverse mstr1.dat out1.dat</code> and match its output to an expected value.	0.5 / 0.5	✓
> 4	Audio Clip #1 Data Output File Run <code>diff -w out1.dat mstrOut1.dat</code> and check for successful completion.	1 / 1	✓
5	Audio Clip #2	0.5 / 0.5	
> 5.1	Reverse Audio Clip #2 Run <code>./reverse mstr2.dat out2.dat</code> and match its output to an expected value.	0.5 / 0.5	✓
> 6	Audio Clip #2 Data Output File Run <code>diff -w out2.dat mstrOut2.dat</code> and check for successful completion.	1 / 1	✓
7	Audio Clip #3	0.5 / 0.5	
> 7.1	Reverse Audio Clip #3 Run <code>./reverse mstr3.dat out3.dat</code> and match its output to an expected value.	0.5 / 0.5	✓
> 8	Audio Clip #2 Data Output File Run <code>diff -w out3.dat mstrOut3.dat</code> and check for successful completion.	1 / 1	✓
> 9	Memory Leak Test Check the code quality using <code>valgrind ./reverse mstr1.dat out1.dat</code> . Points will be deducted for each comment. Per <b>fatal</b> comment you will lose 100% of the points. Per <b>error</b> comment you will lose 20% of the points. Per <b>warning</b> comment you will lose 10% of the points. Per <b>info</b> comment you will lose 5% of the points.	1 / 1	✓

### Assignment Scoring Example (step 3)

**Best Teaching Practices Expo 2022**