Professional Teaching Portfolio Of

Curtis D'Alves

Ph.D Candidate and Sessional Instructor

Department of Computing and Software

McMaster University

Last Edited: April 15, 2021

Contact

Email: curtis.dalves@gmail.com
Phone: +905-870-3907

Contents

Part I

Teaching Philosophy

My core personal beliefs on teaching

Teaching is a responsibility. When a person assumes the role of teacher, they take on the responsibility of guiding a learners experience. On the other hand, the learner must yield a certain degree of trust with the teacher. This is not to say a students learning experience solely relies on a teacher, but it is necessary to recognize this dynamic to effectively understand the role. It is a teachers responsibility to never betray the trust of a student, this responsibility is multi-faceted and vast but put broadly the teacher must always put the students well being first and foremost. This means results achieved by conventional means of assessment cannot be the teachers only priority.

Good teaching challenges and inspires students. A teacher should offer the students more than what can be offered through a textbook or pre-recorded lectures. A teacher should present students with challenges to overcome, and inspire them to want to do so of their own volition. I entered university with a singular goal, acquire a degree. However during my first year of university, I met a professor that I'd come to do a student research project with during the summer that would change my outlook on education. He presented me with an opportunity to do real research, and although I was highly under-qualified at the time he set up series of challenges/stepping stones that allowed me to work up to the level of competence necessary to be useful. I wish to inspire other students the way he inspired me.

Teaching should be accessible. Everyone should have access to education, and teachers should be aware and accommodating of different accessibility issues that their students may face. Teachers are often to quick to assume their methods of assessment or presentation are the only reasonably effective way to teach, and that it is the responsibility of the student to adapt to them no matter their circumstances. This loses sight of the true goals of teaching, to give students the skills and knowledge they need to succeed.

My teaching strategies

I employ Active Learning methods in my lectures and teaching material. Active learning makes use of various activities to break up the otherwise passive experience of learning in traditional lectures. It puts a higher degree of responsibility on the student, challenging them during the learning process. And there is possibly no more appropriate field to employ active learning then computing and software. Interfacing with a computer allows a level of live assessment not achievable otherwise. By interspersing lecture content with live coding activities and discussion sessions students have a much deeper level of engagement with the material, and receive a much more adequate level of feedback than conventional assessment methods. Regular lecture sessions cease to be a passive experience where students show up simply to listen to a professor present a topic and become a challenge for learners to overcome.

I provide real world applications of concepts being taught wherever possible. Most students in STEM come to university with the dream of one day acquiring the skills to build something truly useful. For these students, the traditional university experience can be extremely disheartening. Whole subjects (calculus for example) can seem like nothing more than a form of academic hazing, meant to give students a hard time rather than provide them with useful skills. This is not to say STEM students shouldn't be taught calculus, but that if done in a way that doesn't reflect it's applicability it can be a hindrance to their academic development.

I design my course content following Universal Design for Learning principles. When appropriate, I provide course content in as many different means of representation as reasonable. This means implementing redundancies for visual and audio representation in lectures and video presentations, which can be as simple as making sure to fully explain slide content verbally and providing all verbally explained content in slides and text documents. I also encourage note sharing (either for bonus marks or a means of assessment with exemptions) to help students who have difficulty taking notes. Another principle of Universal Design for Learning I follow is providing students with multiple means of engaging with me. Although I encourage students to engage with me directly in class or during office hours, I understand in person experience can be anxiety inducing. I find online interfaces (like forums and instant messaging / video conferencing) can be very effective alternatives for these students.

My goals as a teacher

Although my teaching experience is limited, I already greatly value the positive impacts I've had on my students. Of all the feedback I've received, the most dear to me are from students from a different program who have stated they were inspired by me to transfer into a computing program. The most obvious goal any institutional teacher should have is for their students to finish with a sufficient understanding of the course material. This is a very valid goal and one I hold, however I don't limit myself there. My more ambitious goal is to inspire students to want to continue learning. But this isn't the only ambitious goal I have. I wish to not just acknowledge my positive feedback but critically engage with and improve from my negative feedback. The most common theme in my negative feedback (particularly from the first few sessional positions I taught) revolve around having too high expectations on students. I believe this stems partly from my desire to challenge students, but also a personality flaw of lack of patience. One of the greatest things about being a teacher is it challenges you to grow as a person. I have seen myself become a more patient and understanding person each year I teach not just through my interactions with students and course feedback, but also in my personal life. It is my greatest goal that as I continue to improve as a teacher, I continue to improve as a person.

Part II

Teaching Practice

To practically apply my teaching philosophy, I continually strive to better integrate my teaching strategies (using real-world applications, active learning and Universal Design for Learning) in all of my course content. This includes how I develop/present my lectures, how I develop my assessments and what types of assessments and course materials I focus on. When I taught my first sessional position, an experiential learning based course in the second semester of the first year in a CS program (known as CS 1XA3), I laid out a clear road map with an end goal of developing a Django stack (Javascript - Python - Django - MySQL) web app managed under a GitHub repository that they could showcase as part of their personal portfolio. I found that as long as I could relate the concepts I was teaching as part of this end goal students were far more inspired to engage with the content.

When I was a teaching assistant for another intro CS course (known as CS 1JC3), I assisted a professor in a series of lecture sessions he referred to as "Discussion Sessions". In one of the three lecture sessions a week, we would hold a lecture session that focused on calling on students and asking them questions about the topics discussed in the previous lecture sessions that week. The questions were designed to be high-level and divergent so they engaged the students far more than questions with simple right or wrong answers. When I eventually "inherited" the course as a sessional instructor, I continued this practice as the students always seemed to remark on their value in the year end evaluations. In the proceeding year when I taught the course online during the 2020 pandemic, I changed things up a bit by making every lecture section a "Discussion Session", interleaving presenting lecture content and questions. Although this cut down the amount of content I could cover this proved to be a worthy trade-off, actively engaging students in content that they would demonstrate they better retained come exam time.

In both courses I've taught as a sessional instructor, I've come to find a focus on project based assessment to be more effective than exams. This is not to say I didn't administer an exam, but I was open to allowing students to shift more weight of their final mark onto projects that could be better tailored to suit their learning needs. Although exams have their place they are often more limited in their ability to fully assess a students capabilities/knowledge, and often disadvantage students who have accessibility issues making it difficult to complete assessments with short time constraints. In general I try to be as accomadating as reasonable when finding the correct method of assessment for an individual student. Another example of this is the Discussion Sessions I mentioned in CS 1JC3, although I found these sessions to be very valuable for the majority of the class, some students with anxiety issues found answering questions on the spot in front of a class an in-due burden. For these students I offered alternatives such as writing up and sharing notes taken during the lecture. When teaching you encounter a large variety of students with different needs, and it's important to be flexible and consult students who struggle on a individual basis to find the correct methods to accommodate them.

Part III

Teaching Experience

Teaching Assistantships

Teaching Assistant CS 1MA3

- **Sept-Dec 2011/2012**
- **McMaster University**
- Instructed course tutorials and assisted in developing tutorial content and slides. Marked assignments, midterms and exams.

Teaching Assistant CS 1JC3

- **Sept-Dec 2013/2014/2017/2018**
- **McMaster University**
- Developed and lead tutorial content for introductory Haskell and Elm and assisted marking assignments and midterm tests.

Teaching Assistant CS/SE 4F03

- **Feb-May 2015**
- **McMaster University**
- Lead tutorials and taught content pertaining to parallel programming with MPI and OpenMP in C/C++ and marked assignments.

Teaching Assistant CS 3EA3

- **Feb-May 2017**
- **McMaster University**
- Lead tutorials and taught content pertaining to formal methods and automated verification of C programs in Frama-C

Sessional Instructor Positions

Sessional Position CS 1XA3

- **Feb-May 2018/2019/2020**
- **McMaster University**
- Developed entirely original course content for lectures, labs, projects, exams and taught all lectures and labs. The course focused around 3 projects and taught version control with Git, Bash scripting, Web App development, and developing a parser and simple interpreter.

Sessional Position CS 1JC3 C02

- **Sep-Dec 2019/2020**
- **McMaster University**
- Taught to Non-Computer Science students. Lead lectures and developed new assignment content and midterms.

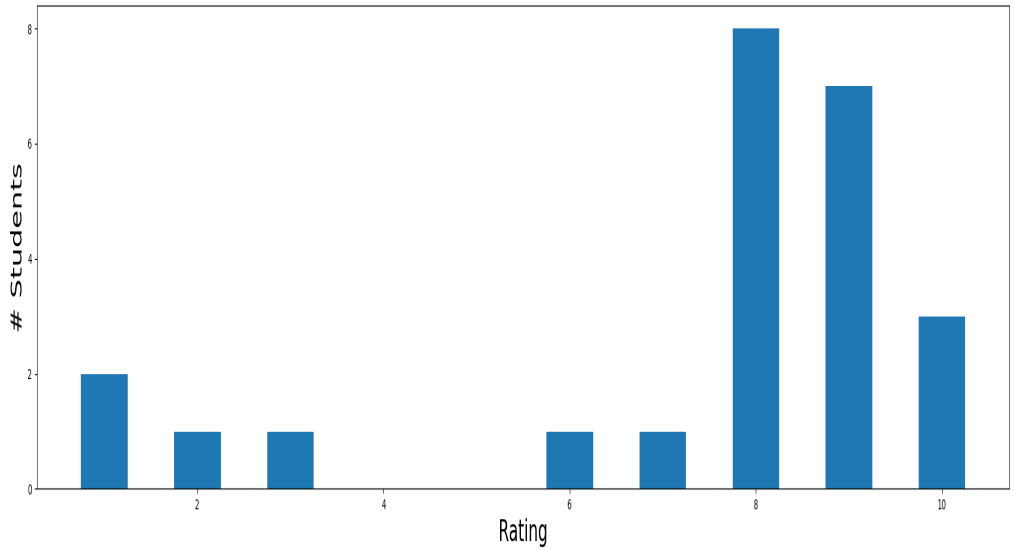
Part IV

Evidence of the Effects of Teaching

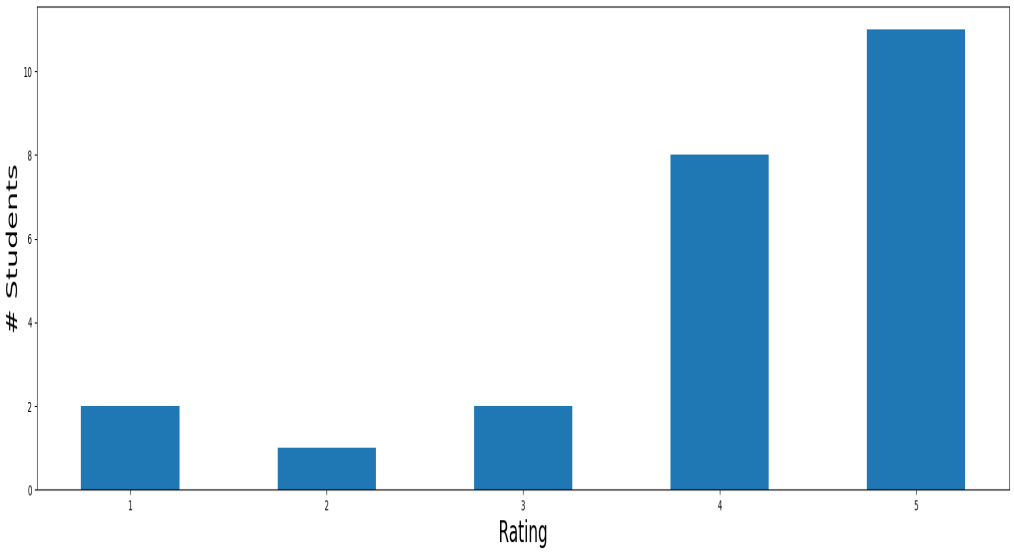
Teaching Evaluations

CS 1XA3 Winter 2019 (Combined C01/C02 Sections)

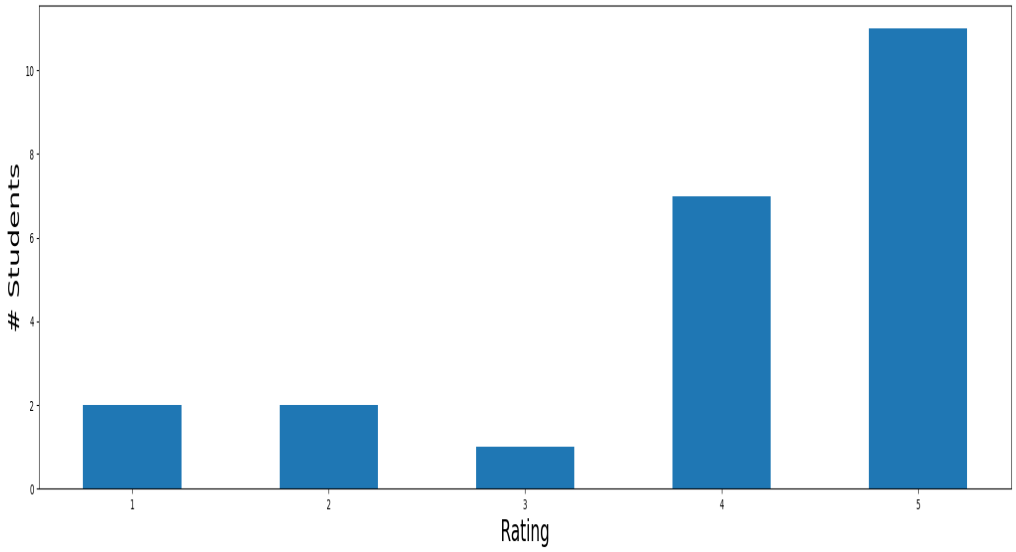
Overall for this course, what is your opinion of the effectiveness of the instructor? (Scale: 1 Very Poor to 10 Excellent)



Independent critical judgement was encouraged (Scale: 1 Very Poor to 5 Excellent)



The instructor's response to students (Approachability, attitude, availability, well-explained answers) (Scale: 1 Very Poor to 5 Excellent)



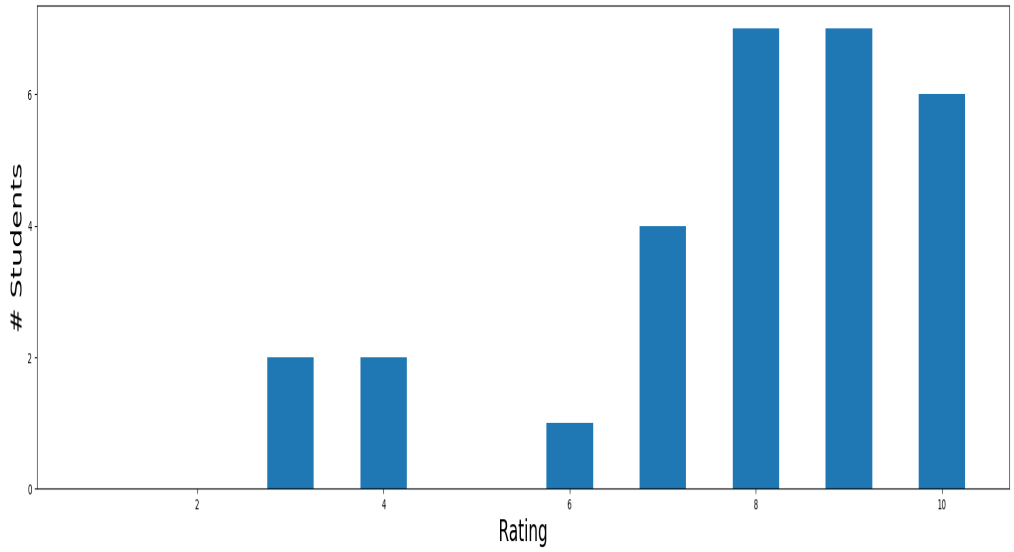
CS 1XA3 was the first time I developed a course completely on my own. I was employed as a sessional instructor and given full control of what content to teach, and choose to develop a fully original curriculum. The curriculum was ambitious, covering a full stack development (Javascript - Python - Django - MySQL) range of skills in a first year computer science program. I found myself particularly proud of the amount of independent critical judgement I encouraged, as was reflected in my course evaluations.

Not all students found my approach effective though. It would be convenient for me to hand-wave these criticisms as students who weren't ambitious enough to be in a program as difficult as Computer Science at McMaster. However it's evident that some of the same students that gave me a low overall effectiveness still recognized that the class was good for encouraging independent critical judgement. Being critical of myself, part of the problem was my lack of flexibility and demeanor when dealing with students who struggle to initially grasp concepts more than the average student. In later sessional positions I've taught (such as CS 1JC3), I've continued to try improve how I approach challenging students without discouraging them.

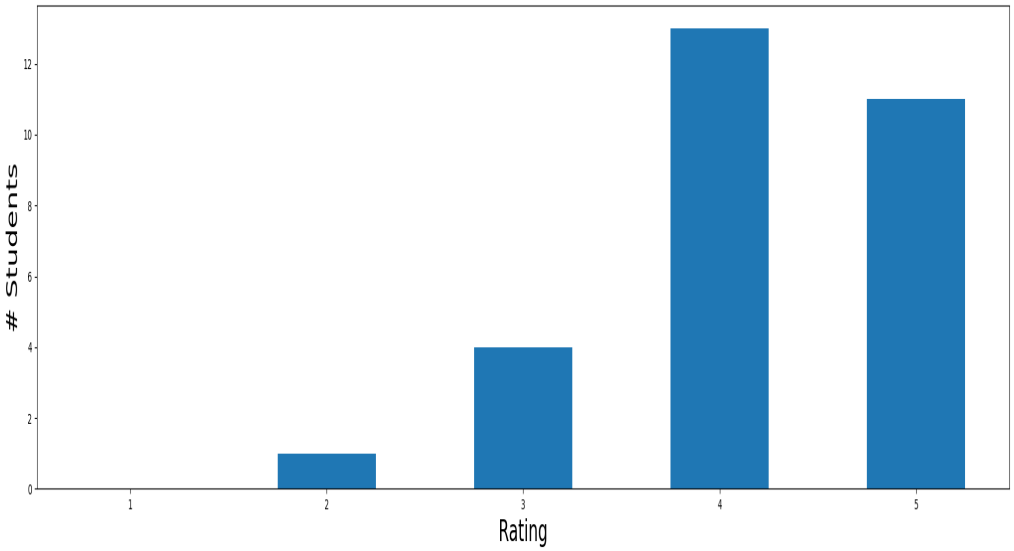
See the full evaluations for CS 1XA3 2019 C01 at Appendix C and C02 at Appendix C

CS 1JC3 C02 Fall 2019

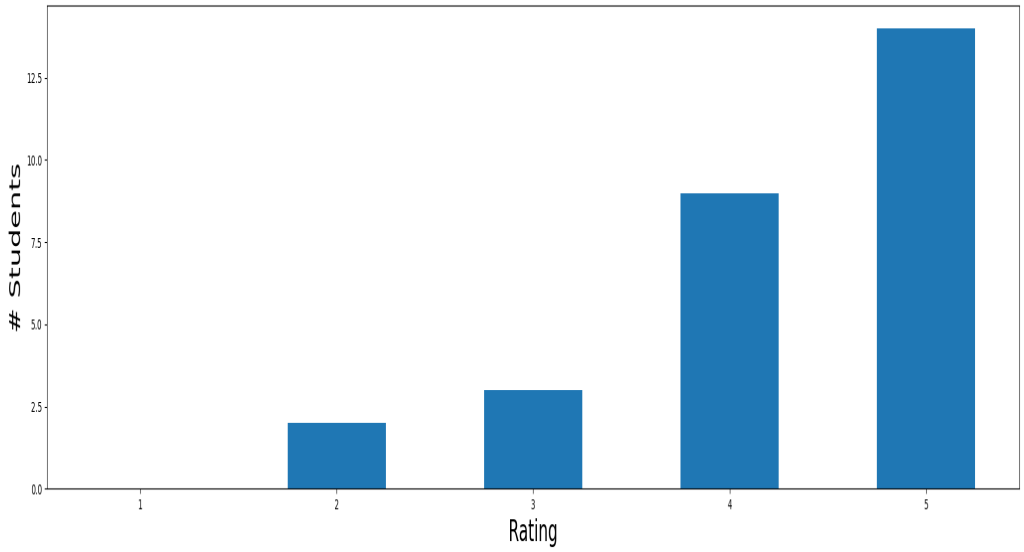
Overall for this course, what is your opinion of the effectiveness of the instructor? (Scale: 1 Very Poor to 10 Excellent)



Independent critical judgement was encouraged (Scale: 1 Very Poor to 5 Excellent)



The instructor's response to students (Approachability, attitude, availability, well-explained answers) (Scale: 1 Very Poor to 5 Excellent)



After my winter 2019 session teaching CS 1XA3, I taught CS 1JC3 C02 as a sessional instructor (alongside another professor who taught CS 1JC3 C01). The course was split up into two courses, C01 being students enrolled in the Computer Science program at McMaster, and C02 for students outside the program (this was done due to the high demand of students wishing to take an intro programming course).

Despite being an introductory Computer Science course, CS 1JC3 was a challenging course covering a wide variety of concepts in computer science and functional programming. McMaster has high expectations of students enrolled in Computer Science, and I was tasked with teaching the same content to students outside of the program for whom computer programming was not their forte. I did my best to continue to challenge students the way I did in CS 1XA3, but be more flexible and approachable. Although I still have room to improve in this respect, by year end it seems less students found me unapproachable than in CS 1XA3 while still recognizing I valued independent critical judgement.

See the full evaluations for CS 1JC3 2019 C02 at Appendix E

Observations of Teaching

Below are a selection of comments provided by the instructors I have worked with

Dr. Christopher Anand, McMaster Outreach, Supervisor

Dr. Anand was my Ph.D supervisor and organizer of McMaster Outreach. In the future I will include a statement from him about my work with outreach

Dr. William Farmer, CS 1JC3 C01 Fall 2019, Instructor

I taught CS 1JC3 alongside Dr. Farmer, in the future I will include a statement from him

Part V

Teaching Development

Programs and Certificates

MacPherson Institute EDUCATN 750/751

- A graduate course offered by the MacPherson Institute at McMaster University. The focus is on honing essential pedagogical and practical teaching skills. This includes sessions on curriculum design, teaching strategies, assessment strategies, and developing a teaching portfolio.
- Completed in Winter Semester 2021

Conferences

Trends in Functional Programming in Education (2017)

- Attendant and presenter at TFPIE 2017 (held at Kent University)
- The goal of TFPIE is to gather researchers, teachers and professionals that use, or are interested in the use of, functional programming in education. TFPIE aims to be a venue where novel ideas, classroom-tested ideas and work-in-progress on the use of functional programming in education are discussed.

Papers

Co-Author: Using Elm to Introduce Algebraic Thinking to K-8 Students

- Research paper analyzing the use of functional programming to teach K-8 students topics in mathematics
- Published in EPTCS volume 270 http://eptcs.web.cse.unsw.edu.au/paper.cgi?TFPIE2017.2

Part VI

Future Goals

Short-Term Goals

In the immediate future, my goals for teaching include:

- ◇ Finish the rest of the MacPherson Institute courses and earn their Teaching & Learning Certificates of Completion Program
- ◇ Publish another conference paper on the use of computer science in education
- ◇ Teach CS 1JC3 during the Spring/Summer 2021 session (already underway)

Long-Term Goals

In the next few years, my goals for teaching include:

- ◇ Continue teaching sessional positions as a post-graduate fellow
- ◇ Apply for (and eventually get hired) teaching track (or possibly tenure track) positions at Colleges/Universities

Part VII

Appendix

Appendix A: Sample Course Syllabus

The following document (see next page) is a sample course syllabus I originally developed for use in a first year computer science course (CS 1XA3) I taught as a sessional instructor in 2018/2019 and have since refined

CS 1XA3: Course Syllabus

Curtis D'Alves

March 18th, 2021

Course Description

Computer Science 1XA3 is an experiential learning based course designed to teach first year students common tools/skills utilized in the practice of software engineering while learning about underlying theoretical computer science concepts. Students are expected to already have a beginners knowledge of the python programming language (previous or concurrent enrollment in CS 1MD3 is recommended)

Instructor (Contact Info)

- Curtis D'Alves, Ph.D Candidate @ McMaster University
- Office Hours: in ITB 229 (no specific hours, but please email before dropping by)
- Email: curtis.dalves@gmail.com

Lecture/Lab Sessions

Lectures	Fr 9:30AM	10:20AM in BSB B135
Lab Section L01	TuFr 2:30PM	4:30PM in BSB 249
Lab Section L02	TuTh 9:30AM	11:30AM in BSB 244
Lab Section L03	MoWe 9:30AM	11:20AM in KTH B123
Lab Section L04	MoTh 12:30PM	2:30PM in KTH B123

Lecture Sessions

There will be a single Lecture session each week presenting concepts in computer science interleaved with discussion sessions on these topics. Students will be broken up into groups for guided discussion sessions and brainstorm answers that will be presented in class by a volunteer

Lab Sessions

There will be two Lab sessions each week. The first session each week will be a Lab Activity session that students will complete with TA assistance as needed and the second session will present practical programming content with live coding demonstrations (that students will be encouraged to follow along with).

Course Schedule with topics

Week	Lecture	Activity	Lab
Jan 6 - 12	McMaster ECCS / Intro Un*x Systems	Installation Session	Course Overview
Jan 13 - 19	Shells, FileSystems and Trees	Terminus Game	SSH Activity
Jan 20 - 26	Version Control, Git and Graphs	FileSystems	Git Repo Creation
Jan 27 - Feb 2	Streams, Processes and Permission Systems	Git	Intro Shell Scripting
Feb 3 - 9	Regular Expressions	Scripting 1	Shell Scripting Cont.
Feb 10 - 16	UI Design	Proj.1 Help	HTML/CSS
Feb 17 - 23	**Midterm Recess!**	**Midterm Recess!**	**Midterm Recess!**
Feb 24 - Mar 2	Object Oriented Programming	Simple Webpage	JavaScript
Mar 3 - 9	Model Driven Development	JavaScript	JavaScript Cont.
Mar 10 - 16	Client-Server Model	Proj. 2 Help	Http Requests Django
Mar 17 - 23	Network Protocols / Routing	Django 1	Django Cont.
Mar 24 - 30	Relational Algebra	Django 2	Databases
Mar 31 - Apr 6	Cryptography / Signing / Key Exchange	Proj. 3 Help	User Authentication
Apr 7 - 13	**CLASSES END**	**CLASSES END**	**CLASSES END**

Important Deadlines

	Date Due (by 11:59pm)
GitHub Repo Registration	Jan 26
Proj. 1 Part 1	Feb 2
Proj. 1 Part 2	Feb 16
Proj. 2 Part 1	Mar 2
Proj. 2 Part 2	Mar 16
Proj. 3	Apr 20

NOTE structure and proper organization (including tentative deadlines) is important for maximizing your learning experience in this course. However should you require an extension please don't hesitate to contact the me (curtis.dalves@gmail.com) to work out a more flexible schedule

Assessment and evaluation structure

Grades will be calculated using the following criteria

Project 1	15%
Project 2	15%
Project 3	20%
Labs Activities	40%
Participation	10%

Projects

You will submit 3 projects each in 2 parts:

- Project 1 will be a Bash Scripting activity
- Project 2 will have you design a client side web app using Javascript/HTML/CSS
- Project 3 a "full-stack" web app using Python Django and SQL.

You will host the projects on GitHub and your commit history will be analyzed as part of your final mark (frequent, well spaced out commits will gain you more marks). For further details refer to the project outlines that will be posted on Avenue to Learn.

Lab Activities

There are a total of 8 Lab Activities that will be completed synchronously (in-class) in the first lab session of each week. TA's will be available to assist you through the activities, and full marks will be given for each successful completion. Should you miss a lab activity, you may contact a TA about completing it asynchronously over Discord.

Participation

Participation marks can be earned in the following ways:

- Sharing notes on a lecture session (upload to avenue)
- Sharing notes on a (non-activity) lab session (upload to avenue)
- Answering a live lecture/lab question
- Creating and sharing a coding activity relevant to the lab content

If you consider each of the above to be worth 1 "Participation Mark", there are a total of 5 Participation Marks to earn throughout the semester. This makes each Participation Mark worth 2% of your final grade. If you have another idea for how you would prefer to earn Participation Marks, please to hesitate to contact the instructor about it.

Intended Learning Outcomes

CS 1XA3 has the following Intended Learning Outcomes, i.e.

- students should have the practical skills to
  - Use Un*x command line interfaces to navigate, create and manipulate filesystems, ssh/scp to access remote servers and the Un*x commands top*/ps/kill to manage system processes
  - Investigate network connections using the Un*x command netstat
  - Manage a code base using git version control
  - Design a simple webpage using HTML, CSS and Javascript, then a web server application using the python Django framework, and a simple SQL database
- and students should have enough working knowledge of computer science concepts to
  - Recognize corresponding tree data structures in file systems, directed graph data structures in git revision commit history and basic UI principals used in webpage design
  - Define regular expressions for string enumeration/matching, conceptual models (using UML Diagrams) corresponding to module relationships in python code and basic relational algebra equations corresponding to SQL queries

Learning outcomes for practical skills are largely cummulative, i.e. use of command line tools will be necessary to use version control which will be necessary to manage your code to build web pages, which is in turn necessary to have a functioning server which is also in turn necessary to implement a database. Therefore evaluation of skills acquired will be reinforced by evaluation of subsequent skills. Furthermore learning outcomes for computer science concepts underly the use of corresponding practical skills

Course Requirements

Course Prerequisites

- High school calculus
- Current or previous enrollment is CS 1MD3 **OR** sufficient knowledge of the python programming language (instructors permission required for the latter)

Course Expectations

Throughout the course, students are expected to

- get to know and use your peers (fellow classmates) as a resource
- seek out ways to extend the core objectives of a project to suit your learning needs
- consult the instructor before attempting to implement features that may be too difficult
- organize your work so that your assignments are done in a timely manner (as will be evident by your GitHub commit history)
- follow instructions for submitting assignments included in the assignment outlines
- students are responsible to ensure all code they submit compiles (on the mac1xa3.ca server) before submission **FAILURE TO SUBMIT CODE THAT COMPILES CAN RESULT IN A MARK OF ZERO**

Resources

Mandatory

You are required to create specialized accounts for the following services

- Avenue to Learn: will serve as the course webpage. All announcements and course content (including slides, grades, project templates) will be posted there. Please check it regularly
- GitHub: a free, public repository with version control you will be using to store your source code for your assignments
  - Click the link above, sign up for an account (if you don't already have one)
  - Sign up for **GitHub Student Developer Pack** by following the instructions here
    - `https://help.github.com/en/github/teaching-and-learning-with-github-education/applying-for-a-student-developer-pack`)
  - Create a **private repository** named CS1XA3
  - From the repositories main page, go to Settings / Collaborators
  - Search for me, dalvescb, and add me as a collaborator
  - Message me on Discord with your Repo URL and Mac ID
  - **YOU MUST REGISTER YOUR GITHUB REPO BEFORE THE END OF THE WEEK 2**
- Discord: an instant messaging service that's very popular among gamers but also to some extent project developers
  - Join the CS1XA3_{2020} Discord with the following link `https://discord.gg/PjGEY3f`
- `https://mac1xa3.ca`: a Ubuntu based server, you can **ssh** into it using your Mac Id and the first 8 characters of your student number as password (message me on discord if you cannot connect). Once you've logged into the server, used the command **passwd** to change your password

Recommended

- Ubuntu: a very popular linux OS
- Windows Subsystem for Linux: a new and very convenient way to run Linux from within Windows
- Spacemacs: a distribution of emacs that combines the power of vim, my editor of choice
- StackOverflow: every hackers best friend, a Q&A forum for all things coding
- GitHub Desktop: a graphical app that makes doing basic operations with GitHub quick and painless
- Suggest other resources on discord for participation marks

University Policies

Missed Work / MSAF Policy

In the event of an absence for medical or other reasons, students should review and follow the Academic Regulation in the Undergraduate Calendar "Requests for Relief for Missed Academic Term Work".

Students are expected to contact the instructor (see Contact Info) to request relief for missed marked work, as well as follow the official McMaster policy for submission of MSAF's (see `https://www.mcmaster.ca/msaf/`).

Relief for the following works can be provided:

- **Projects** will receive a 3 day extension
- **Lab Activities** will have their marks shifted onto the rest of the labs

Usage of On-Line Resources Policy

In this course we will be using Avenue to learn and Discord messaging system. Students should be aware that, when they access the electronic components of this course, private information such as first and last names, user names for the McMaster e-mail accounts,and program affiliation may become apparent to all other students in the same course. The available information is dependent on the technology used. Continuation in this course will be deemed consent to this disclosure. If you have any questions or concerns about such disclosure please discuss this with the course instructor.

Academic Ethics

You are expected to exhibit honesty and use ethical behaviour in all aspects of the learning process. Academic credentials you earn are rooted in principles of honesty and academic integrity. Academic dishonesty is to knowingly act or fail to act in a way that results or could result in unearned academic credit or advantage. This behaviour can result in serious consequences, e.g. the grade of zero on an assignment, loss of credit with a notation on the transcript (notation reads: "Grade of F assigned for academic dishonesty"), and/or suspension or expulsion from the university. It is your responsibility to understand what constitutes academic dishonesty. For information on the various types of academic dishonesty please refer to the Academic Integrity Policy, located at `http://www.mcmaster.ca/academicintegrity`. The following illustrates only four forms of academic dishonesty:

1. Plagiarism, e.g. the submission of work that is not one's own or for which other credit has been obtained.
2. Collaboration where individual work is expected. You are allowed, and encouraged, to collaborate on the exercise questions. (The tutorials are typically not expected to cover all exercise questions.)

3. Improper collaboration in group work.

4. Copying or using unauthorised aids in tests and examinations.

Note: Although you may use code found online for your assignments, you must reference your source in the projects README. Grades for assignments that are entirely or mostly others code will reflect such.

Academic Accommodation of Students with Disabilites

Students who require academic accommodation must contact Student Accessibility Services (SAS) to make arrangements with a Program Coordinator. Academic accommodations must be arranged for each term of study. Student Accessibility Services can be contacted by phone 905-525-9140 ext. 28652 or e-mail sas@mcmaster.ca. For further information, consult McMaster University's Policy for Academic Accommodation of Students with Disabilities

Academic Accommodation For Religious, Indigenous Or Spiritual Observances (RISO)

Students requiring academic accommodation based on religious, indigenous or spiritual observances should follow the procedures set out in the RISO policy. Students requiring a RISO accommodation should submit their request to their Faculty Office normally within 10 working days of the beginning of term in which they anticipate a need for accommodation or to the Registrar's Office prior to their examinations. Students should also contact their instructors as soon as possible to make alternative arrangements for classes, assignments, and tests

Discrimination

"The Faculty of Engineering is concerned with ensuring an environment that is free of all adverse discrimination. If there is a problem that cannot be resolved by discussion among the persons concerned, individuals are reminded that they should contact the Department Chair, the Sexual Harassment Office or the Human Rights Consultant, as soon as possible."

Disclaimer / Course Changes

The instructor and university reserve the right to modify elements of the course during the term. The university may change the dates and deadlines for any or all courses in extreme circumstances. Changes will be communicated through regular McMaster communication channels, such as McMaster Daily News, A2L and/or McMaster email. It is the responsibility of the student to check their McMaster email and course websites weekly during the term and to note any changes.

Appendix B: Sample Assessment

The following document (see next page) is a sample project I originally developed for use in a first year computer science course (CS 1XA3) I taught as a sessional instructor in 2018/2019 and have since refined

CS 1XA3 Project 01

Created by: Curtis D'Alves

Due Date: Part 1 Feb 2, Part 2 Feb 16

Contents

1 Project Setup

1.1 Clone Your Repo Into Your Private Directory

You must keep your repo inside of your directory located in `$HOME/private` on the `mac1xa3.ca` server

- WARNING not doing so or changing the default permissions on the private directory will be considered academic dishonesty
- If you accidentently delete the directory, contact me or one of your TA's (preferably me / dalvescb on discord)

1.2 Create a new folder in your repo

On the master branch, create a new folder `CS1XA3/Project01` (where `CS1XA3` is the already existant root of repo)

1.3 Add the folowing files

- Add `CS1XA3/Project01/project_analyze.sh`
- Add `CS1XA3/Project01/README.md`

1.4 Commit and Push

- Add and commit with the following message EXACTLY "Initial Project01 Commit"
- Push to GitHub

1.5 Create a Branch

- Create a branch called project01
- Push the branch to github (i.e git push origin project01)
- Work from the project01 branch and only merge with master when your ready to submit Part 1 or Part 2

2 Project Submission Part 1 (Due Feb 2 at 11:59pm)

Merge to the master branch a WORKING SCRIPT that:

- implements at least 20 points worth of 5 point features including the Script Input feature
- documents the features you implement and script usage in the README.md
- documents the two custom features you plan to implement (although you don't need to implement them yet)
- has a commit/merge message "Submitting Project 01 Part 1" EXACTLY

WARNING make sure to push to GitHub to complete your submission

3 Project Submission Part 2 (Due Feb 16 at 11:59pm)

Merge to the master branch a WORKING SCRIPT that:

- implements at least 20 points more worth of features including at least one 10 point feature
- implements 2 custom features that should be around the same level of difficulty as the 10 point features
- has completed documentation of all features in the README.md
- has a commit/merge message "Submitting Project 01 Part 2" EXACTLY

WARNING make sure to push to GitHub to complete your submission

4 README

You MUST document your code in a file `CS1XA3/Project01/README.md`

- The document should be styled with MarkDown (see https://guides.github.com/features/mastering-markdown/)
- The document should describe usage of the script (i.e how to execute, with what arguments, under what conditions)
- The document should contain a section (header) for each feature (including custom features)
- You must reference any code used that was found online with a link to the url (failing to do so will be considered academic dishonesty)

An example of the general outline of the document would be as follows (this is not an exact template you need to follow, you are encouraged to use your best judgment for constructing a useful README):

```
#  CS 1XA3 Project01 - <MyMacId>

## Usage
   Execute this script from project root with:
        chmod +x CS1XA3/Project01/project_analyze.sh
        ./CS1XA3/Project01/project_analyze arg1 arg2 ...
   With possible arguments
        arg1: ....
        arg2: ....
   ....
## Feature 01
 Description: this feature does ....
 Execution: execute this feature by ...
 Reference: some code was taken from [[https://someurl.com]]

## Feature 02
  ...
## Custom Feature SomeFeature
  ...
```

5 Grading Scheme

README Documentation	**20%**
Custom Features	**20%**
Other Features	**60%**

WARNING failure to properly follow instructions (including not cloning your repo to the proper directory, not pushing to GitHub, not using the correct commit message, etc) will result in A MARK OF 0

5.1 Grading: README Documentation

The README file submission is worth a total of 15 points that is projected onto 20% of the overall grade, please refer to the following rubric for details on what criteria you will be marked upon

	Criteria	**Points**
Style	Use markdown where appropriate	2 points
	Readable formatting	2 points
	Broken up into an appropriate amount of sections	1 point
Correctness	Instructions for execution are correct	1 point
	Description of each feature is correct	4 points
Detail	Instructions for execution have appropriate detail	1 point
	Description of each feature has appropriate detail	4 points

5.2 Grading: Custom Features

You will implement two custom features, each worth 5 points (so the two together will be worth a total of 10 points which will be projected onto 20% of the overall grade), please refer to the following rubric for details on what criteria each individual custom feature will be marked upon

	Criteria	**Points**
Creativity	Is unique, substantially different from other features	1 point
Applicability	Has an actually useful application	1 point
	Incorporates skills taught in course	2 points
Correctness	Works as expected/described in README	1 point

5.3 Criteria: Other Features

You will implement 40 points of features corresponding to 60% of your overall project mark. Reference the Features section for a list of all possible features and their corresponding point worth. Please refer to the following rubric for details on what criteria each 5 point feature will be marked upon (double each point for features worth 10 points)

	Criteria	Points
Correctness	Works for at least one use case	1 point
	Accounts for directories/files with special characters	1 point
	AND/OR not existing AND/OR accounts for command IO failure	
	Is executable as described in README	1 point
	Works for all use cases	2 points

5.4 Plagiarism / Academic Dishonesty

Tools will be used to compare your code with your peers (including previous years of this course)

- Stealing a custom feature idea will be considered plagiarism
- Using code without referencing the source in your README will be considered plagiarism.
- Any account of plagiarism will result in an automatic grade of 0

6 Features

6.1 Script Input (Mandatory) (5 Points)

- Make the script interactive (i.e select which feature(s) are executed) either by providing script arguments or by user prompted input
- Describe this feature in the Usage section of the README.md document rather than in it's own header

6.2 FIXME Log (5 points)

- Find every file in your repo that has the word #FIXME in the last line
- Put the list of these file names in `CS1XA3/Project01/fixme.log` with each file separated by a newline
- Create the file `CS1XA3/Project01/fixme.log` if it doesn't exist, overwrite it if it does

6.3 Checkout Latest Merge (5 points)

- Find the most recent commit with the word merge (case insensitive) in the commit message
- Automatically checkout that commit (so that you're in a detached head state)

6.4 File Size List (5 points)

- List all files in the repo (just files not directories) and their sizes in a human readable format (i.e KB, MB, GB, etc)
- List the files sorted from largest to smallest

6.5 File Type Count (5 points)

- Using the read command (with a prompt), prompt the user for an extension (i.e txt, pdf, py, etc)
- Output the number of files in your repo with that extension

6.6 Find Tag (5 points)

- Using the read command (with a prompt, prompt the user for a Tag (any single word)
- Create a log file `CS1XA3/Project01/Tag.log` (where Tag is the name of the Tag provided) if it doesn't already exist, overwrite it if it does
- For each python file (i.e ending in .py) in the repo, find all lines that begin with a comment (i.e `#`) and include Tag and put them in `CS1XA3/Project01/Tag.log`

6.7 Switch to Executable (10 points)

- Find all shell scripts (i.e ending in .sh) in the repo
- Create a file `CS1XA3/Project01/permissions.log` if it doesn't already exist
- Using the read command, prompt the user to Change or Restore (use a prompt that tells the user what to do)
- If the user selects Change:
  - For each shell script, change the permissions so that only people who have write permissions also have executable permissions (i.e if only user has write permissions, then only user has executable permissions)
  - Store a log of the file and it's original permissions in `CS1XA3/Project01/permissions.log` (overwrite it if it already exists)
- If the user selects Restore
  - Restore each file to its original permissions (as specified in `CS1XA3/Project01/permissions.log`)

6.8 Backup and Delete / Restore (10 points)

- Using the read command, prompt the user to Backup or Restore (use a prompt that tells the user what to do)
- If the user selects Backup:
  - Create an empty directory `CS1XA3/Project01/backup` if it doesn't exit
  - Empty the directory `CS1XA3/Project01/backup` if it does exist
  - Find all files that end in the .tmp extension
    - copy them to the `CS1XA3/Project01/backup` directory
    - delete them from their original location
    - create a file `CS1XA3/Project01/backup/restore.log` that contains a list of paths of the files original locations
- If the user selects Restore:
  - use the file `CS1XA3/Project01/backup/restore.log` to restore the files to their original location
  - if the file does not exist, through an error message

Appendix C: Course Evaluation CS 1XA3 2019 C01

The following document is the course evaluations for CS 1XA3 Winter 2019 section C01 (students enrolled in the Computer Science program at McMaster University). I taught this section alongside section C02, whose evaluations are in the proceeding appendix

Full Report

Evaluation for 2019 - Term 2 (Winter)

Course Code	Instructor	Response Rate (Respondants/Enrolled)
COMPSCI 1XA3 (C01)	D'Alves, Curtis	35.29% (18/51)

1. Overall for this course, what is your opinion of the effectiveness of the instructor?
(*Scale: 1 Very Poor to 10 Excellent*)

7 Students (38.89%) said: 9
4 Students (22.22%) said: 8
2 Students (11.11%) said: 1
1 Students (5.56%) said: 3
1 Students (5.56%) said: 10
1 Students (5.56%) said: 6
1 Students (5.56%) said: 2
1 Students (5.56%) said: 7

Median: 8.00	Mean: 6.94	StDev: 3.0190	Variance: 9.1144	Not Responded: 0

2. The timing and appropriateness of feedback on your progress:
Receiving assignments back in a reasonable time frame, clear explanation of grade
(*Scale: 1 Very Poor to 5 Excellent*)

9 Students (50.00%) said: 1
5 Students (27.78%) said: 3
2 Students (11.11%) said: 4
1 Students (5.56%) said: 2
1 Students (5.56%) said: 5

Median: 1.50	Mean: 2.17	StDev: 1.3394	Variance: 1.7941	Not Responded: 0

3. Independent critical judgement was encouraged:
(*Scale: 1 Very Poor to 5 Excellent*)

9 Students (50.00%) said: 5
5 Students (27.78%) said: 4
2 Students (11.11%) said: 3
2 Students (11.11%) said: 1

Median: 4.50	Mean: 4.06	StDev: 1.3048	Variance: 1.7026	Not Responded: 0

4. OVERALL, how do you rate the value of this course compared with others you have taken at McMaster?
(*Scale: 1 Very Poor to 5 Excellent*)

9 Students (50.00%) said: 5

3 Students (16.67%) said: 1
3 Students (16.67%) said: 4
2 Students (11.11%) said: 3

Median: 5.00	Mean: 3.88	StDev: 1.5363	Variance: 2.3603	Not Responded: 1

5. The organization of this course:

Progression of learning material, resource availability, professor was timely and prepared
(*Scale: 1 Very Poor to 5 Excellent*)

5 Students (27.78%) said: 5
4 Students (22.22%) said: 4
3 Students (16.67%) said: 3
3 Students (16.67%) said: 1
2 Students (11.11%) said: 2

Median: 4.00	Mean: 3.35	StDev: 1.4975	Variance: 2.2426	Not Responded: 1

6. The instructor's response to students:

Approachability, attitude, availability, well-explained answers
(*Scale: 1 Very Poor to 5 Excellent*)

7 Students (38.89%) said: 5
5 Students (27.78%) said: 4
2 Students (11.11%) said: 2
2 Students (11.11%) said: 1
1 Students (5.56%) said: 3

Median: 4.00	Mean: 3.76	StDev: 1.4374	Variance: 2.0662	Not Responded: 1

7. The coverage and fairness of tests:

Material coverage, mark distribution, difficulty level
(*Scale: 1 Very Poor to 5 Excellent*)

5 Students (27.78%) said: 4
5 Students (27.78%) said: 5
3 Students (16.67%) said: 3
2 Students (11.11%) said: 1
2 Students (11.11%) said: 2

Median: 4.00	Mean: 3.53	StDev: 1.3747	Variance: 1.8897	Not Responded: 1

8. Please comment on the quality of the TA's in this course:

- Never met a TA But the professor was so good that he doesn't require a TA
- It is nearly the end of the term, we've had 7 labs thus far, and only 3 have been marked. However, Chris has been active on the class chat, providing helpful resources and insights.
- There were no tutorials and we were not allowed to contact the TAs.
- The TAs are always available for assistance which is provided in a satisfactory manner. They do it quickly and it is done well.
- I didn't know there was TA until like April. There's no TA office hour, it's always Curtis who has

one every Tuesday. The TA is quite helpful on Slack though; is very knowledgeable about the topic.
- Not much interaction with the TA's. Many marks from previous assignments have not been posted for student viewing. Whether this is the responsibility of the TA's or the instructor is unclear. TA Chris has always been available to answer questions on the courses' slack however the other the TA has not really been active.
- n/a
- Very helpful, and approachable TA’s.
- due to the nature of this course, we don't really see the TA's that often
- The TAs helped with every question I had. They provided good feedback and it was apparent that they liked doing what they were doing.
- The TA's were very helpful all in all!
- N/A
- The TA's in this course were very good. They are available on a class Slack channel and almost always respond to questions immediately. Participation of the students is encouraged, allowing students to engage in discussions and co-operate to create a supportive and constructive learning environment.
- Not alot of interactions with the TA
- Chris was a very helpful TA, as he often shared resources and answered questions on slack.
- Professional, somewhat low presence, but effective when active.

9. Please list aspects of this course that you found valuable and should be continued:
- Nice Labs and Lecture schedule and pattern
- I found front-end development with Elm an interesting topic of study.
- Encouraged to work on our own.
- Git, Django
- The use of slack to connect the students. the encouragement to do things my own way, as long as it is not bad/negative.
- This course was perhaps the most useful course I've taken in first year as a Computer Science major; it teaches you all the essential tools programmers need such as using Github, server, and giving practical hands on experience in doing a project
- The course offers material used in real-world work environments that is really helpful. Concepts like source control, http requests and responses, front-end development and back-end scripting are very useful to learn as well. The emphasis on job-ready concepts and applicable projects such as the online CV make for a great teaching approach. The teaching style that encourages freedom of choice and student independence creates in the student a sense of real-world environments and problem solving skills that are not cultivated by other courses with set problems and solutions.
- Content of the course is a great way for students to quickly learn basic full-stack developer skills, very useful for the real world.
- I really appreciate being taught industry demanding technologies like GIT -content of the Course is well structured -I really like how the projects can help when applying to COOP
- I think everything I've learned in this course is extremely valuable. I believe that if we had not lost a week to the snow day, the overall pace would've been a lot more smoother.
- The Slack group for the class is arguably the greatest asset employed in a classroom environment. Getting help from classmates, TAs and the instructor all happen within at most a few hours. It allows for sharing resources that are beneficial to the class and a professional workspace to chat and discuss work. Regarding the instructor and TAs, they are online at reasonable times so that you can ask whenever you need help or need to talk about important matters regarding late work etc. Slack is naturally appealing to students nowadays because of its

similarity to social messaging platforms such as Instagram and Snapchat. Slack was more effective in this specific course because the course material was programming: bugs and errors are often tricky and it just may take a fresh set of eyes to help find the bug or error. Due to its instant messaging capability, ability to share media and private message capability, there's no real reason students will have a hard time communicating with their instructor and TAs. A project-based course ensures that the theory we learn is actually used in practice. Instead of just memorizing how the model/update/view architecture works in Elm and regurgitating that information on a midterm, producing a project (whether an app, game etc.) using Elm ingrains that information into the student's mind by making the student think about what they've learned and how to use it. The trial and error helps students remember what they've done because it takes time and effort to fix the errors. Memorization is sometimes given a hollow effort that doesn't always work, especially for subjects like programming, databases, severs, etc. You can only learn these subjects if you actually code, if you try and set up a server etc. Personally, I had trouble with Elm (even with my familiarity with Haskell) because of its different nature and the way I was applying it (the architecture and making a game). After finishing the game project, I've now understood the architecture and have a much better understanding of Elm and how it works. I will almost never make the same mistakes and struggles. The lab assignments were another aspect which helped teach the material, they were small and doable. Although, I would say the Django lab assignment was the trickiest one.
- The course content was very useful in terms of what is used in industry.
- Use of slack is good. Freedom in projects is good. The course being centered around projects is good.
- The project oriented nature of this course is very important, as it not only provides us with marketable experience that we can use when applying to co-ops, but also allows us to learn practical applications of what we have learned in the real world.
- The content was good
- I found the content covered in this course to be very valuable and one of the most useful out of all the other courses here at McMaster, as it was practical. The project based approach was also refreshing and rewarding.
- I have to get a bit personal this time, since majority of this course is designed by the instructor himself... This course taps into every aspect of the IT industry, and points us to the right direction for future study.

10. Please list aspects of this course that might be improved:
- Please lessen the amount you teach.... not everyone is as brilliant as you are. Don't expect that if you can do it, everyone can do it. Instead of going for 10 languages or types, you can teach 5 but teach them nicely so that we can remember what you taught rather than cramming up just to clear out the labs or projects..... make your teaching worthwhile and remembered forever and thanked forever. You need to slow down the pace (had the problem with you in 1JC3 as well) And try to teach the class, not yourself..... at times you give the feeling of teaching yourself no matter how much you interact, the pace gets so much that only if you knew the things, then you can understand otherwise you can not. Slight modifications to the difficulty and lowering it down can help us perform better.
- The instructor is intimidating and can come off very condescending.
- The back-end development section of this course has been taught sloppily and without sufficient depth. We were told to simply use Django following step-by-step instructions from Curtis, without understanding the inner workings at all. It would be much better if this course simply focused on front-end development without trying to make us fake full stack developers.
- Class needs much more structure and planning.
- The timeframe which the grades are given back.

- There needs to be more lectures per week. There's only one 1-hour lecture and one 2-hr lab per week, which so students are expected to do their own learning outside of class. It would have been more useful to help students actually understand the topic if there were more lecture periods so that we can go through topics in detail. Also, more TA office hours please. There's a limit to how much we can ask/understand on Slack. Please grade our labs/projects it's April now and we've not been getting our marks back since like December. Also I feel like the participation mark is quite heavy; for beginner programmers like me it's hard to contribute in online discussions :/
- The lab timing and room are not ideal for this course. The 7:00 pm lab can be slow, boring and tedious especially when the instructor does not give time to actually work on a lab. The course may be improved by providing the instructor with supplementary lecture times, such as other classes have. 2-3 lectures a week would greatly improve student comprehension and provide students with time to actually complete the lab with instructor supervision. This would give time for questions instead of supplementary teaching time in labs. Marks were especially slow coming out and feedback was sketchy.
- Pretty badly organized course. Marks/feedback were received much too late (at the time of writing this I have not received any feedback since February). Also I wasn't a fan of Elm, I think it would've been more useful for students to learn a popular front-end framework such as React.js or Angular.js.
- I think some people might find it easier to follow if the pace was it a bit slower, but overall it was a pretty good course.
- Honestly not sure. Maybe more time for lectures because it can be hard to keep up with all the material in the lectures/labs
- Prof says he'll correct assignments this week, or he'll post lecture recordings this week, but it doesn't happen. Haven't gotten any marks back in over a month.
- We do not get feedback for our work in a timely manner. For example, we don't have any feedback for our first project, which may be valuable when working on our current project.
- Marking and feedback still havent recieved any on the work done
- The marking was very slow, which is somewhat understandable as I would expect that projects would take longer to mark than tests, but I still feel like feedback for smaller things could've been a little faster (ie labs).
- Coverage is wide but shallow to say the least. Absurd extra-curriculum knowledge is expected. But all that pushes us to strive so I'm fine with it.

11. Additional comments:

- don't say that if you are not worth the course and if I think you can not perform in the upcoming years, I'll fail you..... you never know when somebody might rise and shine.... everyone has their own time to get the hang of it..... You need to be one with the class. and favouring smart students won't help the world..... it would only help your selfish purpose TEACH EVERYONE You give such weird vibes that I don't even prefer asking you doubts. You are scary when you should be FRIENDLY. Dr.Farmer was scary and hard but he was approachable and friendly (always had a positive vibe and smilled)..... the things you said in the first few days makes me stay away from you rather than getting in the student teacher relationship..... Give positive vibes instead of making students scared about their future..... You professors and teachers are there to make students learn and make them capable of doing stuff but instead you say I'll fail you..... that shows how negative and pessimistic you are..... you are not a visionary at all
- Pretty awful. Instructor was rude and bad at relating to beginners, being patient, and empathetic. He also stopped marking anything in February.
- Our instructor for this course does not seem to care a lot about his job. His marking is very lazy, at the point of writing this course evaluation, he still hasn't given our marks back for the lab we completed about 6 weeks ago. Therefore, the students have no idea what our marks are like,

how we're doing and are worried of the mark we'll get in the end.
- I really enjoyed this course though; although the project was hard I actually enjoyed doing them and I feel I learned a lot from this course. Thanks Curtis :)
- The slides can be titled to better help find the right slides. Some of the slides could use more description regarding certain concepts
- Overall I think the course is a good idea but definitely needs to be re-organized/re-structured.
- I enjoy the content covered in the course, keep up the good work!
- This is probably one of the better courses I've taken and I thoroughly enjoyed it.
- His style of teaching includes keeping the lecture slide on for a while, but mainly focusing on in-class examples to follow and learn from. This is much more receptive to students as they can learn how to do things as they are being taught, and can pose questions when something goes wrong. On the other hand, just reading from a lecture slide can be sometimes boring and information may not be absorbed fully by students, which is something I have not seen him do. Curtis himself has a good attitude and is approachable. This is mainly due to him being younger than most instructors, and therefore more relatable. He interacts with students in a professional yet fun manner, cracking jokes, providing advice from experience and guidance when needed. He loves what he does, and it makes students enjoy their learning. It is apparent that Curtis wants us to truly understand the concepts in class by his attitude towards projects and assignments. He tells us to not worry about marks as if you only care about the marks (and work hard enough to only get marks) you will have a hard time working in the real world. The grading of projects/assignments are fair as they do employ course material, but they also require understanding of all concepts used in the project. You can't just have good code and lack proper documentation: both are equally important in the software development process and knowing how to create good code and good documentation is essential in a real development environment. His passion for computing and desire for us to learn good practices inspires me to learn about these concepts myself and make sure I'm actually learning instead of just memorizing it (it also helps that this course is project-based, so it is easier to learn through practice). This kind of attitude and position on projects/work is not explicitly evident in most other instructors I have been taught by. This attitude is then therefore rare, and is what makes him a good instructor. He truly cares for our learning and wants us to be prepared for the working world.
- Curtis is a good teacher. He provides good content and feed back and doesn't hold back the truth. Although that might upset some students I think the pros outweigh the cons with that attitude. All in all great course, very useful.
- N/A
- Overall, this is a great course, and I strongly believe that the way this course is structured should be kept the same.

Appendix D: Course Evaluation CS 1XA3 2019 C02

The following document is the course evaluations for CS 1XA3 Winter section C02 (students NOT enrolled in the Computer Science program at McMaster University). I taught this section alongside section C01, whose evaluations are in the previous appendix

Full Report

Evaluation for 2019 - Term 2 (Winter)

Course Code	Instructor	Response Rate (Respondants/Enrolled)
COMPSCI 1XA3 (C02)	D'Alves, Curtis	46.15% (6/13)

1. Overall for this course, what is your opinion of the effectiveness of the instructor?
(*Scale: 1 Very Poor to 10 Excellent*)

4 Students (66.67%) said: 8
2 Students (33.33%) said: 10

Median: 8.00	Mean: 8.67	StDev: 1.0328	Variance: 1.0667	Not Responded: 0

2. The timing and appropriateness of feedback on your progress:
Receiving assignments back in a reasonable time frame, clear explanation of grade
(*Scale: 1 Very Poor to 5 Excellent*)

2 Students (33.33%) said: 1
2 Students (33.33%) said: 3
1 Students (16.67%) said: 2

Median: 2.00	Mean: 2.00	StDev: 1.3229	Variance: 1.7500	Not Responded: 1

3. Independent critical judgement was encouraged:
(*Scale: 1 Very Poor to 5 Excellent*)

3 Students (50.00%) said: 4
2 Students (33.33%) said: 5
1 Students (16.67%) said: 2

Median: 4.00	Mean: 4.00	StDev: 1.0954	Variance: 1.2000	Not Responded: 0

4. OVERALL, how do you rate the value of this course compared with others you have taken at McMaster?
(*Scale: 1 Very Poor to 5 Excellent*)

3 Students (50.00%) said: 4
3 Students (50.00%) said: 5

Median: 4.50	Mean: 4.50	StDev: 0.5477	Variance: 0.3000	Not Responded: 0

5. The organization of this course:
Progression of learning material, resource availability, professor was timely and prepared
(*Scale: 1 Very Poor to 5 Excellent*)

3 Students (50.00%) said: 4
2 Students (33.33%) said: 5
1 Students (16.67%) said: 3

Median: 4.00	Mean: 4.17	StDev: 0.7528	Variance: 0.5667	Not Responded: 0

6. The instructor's response to students:

Approachability, attitude, availability, well-explained answers
(*Scale: 1 Very Poor to 5 Excellent*)

4 Students (66.67%) said: 5
2 Students (33.33%) said: 4

Median: 5.00	Mean: 4.67	StDev: 0.5164	Variance: 0.2667	Not Responded: 0

7. The coverage and fairness of tests:

Material coverage, mark distribution, difficulty level
(*Scale: 1 Very Poor to 5 Excellent*)

4 Students (66.67%) said: 4
2 Students (33.33%) said: 5

Median: 4.00	Mean: 4.33	StDev: 0.5164	Variance: 0.2667	Not Responded: 0

8. Please comment on the quality of the TA's in this course:

- The only TA I had any contact with was Chris and he was very helpful and clear in explaining anything I had a question about.
- Although we did not interact with the TA too much in the course, the one time I did talk to the TA they were able to answer my questions which helped me finish my project. The TA was available on Slack whenever we needed them. One suggestion to make the course better could be to have the TA's be present during the lab to help up so we wouldn't have to pause the lesson to ask the Prof. a question.
- TA's were helpful, and any errors were unexpected
- TA's were not utilized much as the course was mostly ran by the instructor, but when asking questions on the slack discussion board, TA Chris was usually very helpful and quick to respond with resources if needed.
- Very good. Assisted on slack very often. Provides useful resources for the course and further exploration on topics covered.

9. Please list aspects of this course that you found valuable and should be continued:

- This course had a lot of valuable examples needed to properly learn how to write code. Rather than just learning what some command does we are forced to use it and find creative uses for it which is how you really learn the depth of how a language can be used. The example based learning is more difficult but more beneficial. Lecture notes are in a very convenient and easy to use format.
- I found that most of the material presented in this course was very useful and valuable overall. Most of the things taught are stuff that we will continue to use throughout our studies and even in the work force. I found the CV website to be especially useful because I feel it gives us an edge over more reputable schools like Waterloo when searching for a co-op.

- The depth to which the topics were covered
- GitHub and version control - keep this, very useful tool to teach regardless of what the programming platform is. Also, having the github with some examples and the code done in class was useful if we wanted to use it as a template, or to play around with quickly.
- Slack Usage was great along with website. lab instruction and notes posted on time and available when needed. The two hour lab was nice to have to all at once.

10. Please list aspects of this course that might be improved:
- Sometimes it's not 100% clear on how to do something and difficult o follow examples in class. If a mistake is made along the way we really fall behind and it's difficult to catch back up. It would be nice if the examples created in class could be available online afterwards. Also the lectures are available online could be titled with the subject matter for reference rather than just numbered.
- I think the time in which we receive our grades should be drastically improved. We have not received any grades past the first couple of weeks of school. I would have preferred to receive these grades in a timely manner so that we can bring up any discrepancies and just have some perspective on how we are doing in the course. I also think the slides should be improved a bit. When doing projects and labs, it is hard to remember what certain things mean in the slides. So I think it would help to have explanation on the slides themselves on what certain things mean.
-
- Further in-depth tutorials on the server-side/django programming.
- Better room with a clear projector.

11. Additional comments:
- Really the course was good and Curtis was great as an instructor. The only real problem was that Curtis said he would post something or give grades back or a solution or something but never did. His responsiveness was great but didn't always follow up on it.
- Overall I enjoyed the course and the instructor was very good. The instructor was approachable and was able to answer any questions we had. I feel that this course should continue to be taught in the future.
- If any example code is potentially malicious, the instructor should attempt to make that fact very apparent to ensure to their best attempt a student does not unintentionally suffer.
- Keep this instructor, the final third section of the course about django and server-side programming could use a little bit more refining however. Version control, bash, elm, and html sections were all very thorough though.
- Very enjoyable learned a lot from the course including many employable aspects. Prof D’Alves was great along with the teaching he did.

Appendix E: Course Evaluation CS 1JC3 2019 C02

The following document is the course evaluations for CS 1JC3 Fall 2019 C02. (students NOT enrolled in the Computer Science program at McMaster University). I taught this section separate from section C01, which was taught by Dr. William Farmer

Full Report

Evaluation for 2019 - Term 1 (Fall)

Course Code	Instructor	Response Rate (Respondants/Enrolled)
COMPSCI 1JC3 (C02)	D'Alves, Curtis	47.54% (29/61)

1. Overall for this course, what is your opinion of the effectiveness of the instructor?
(*Scale: 1 Very Poor to 10 Excellent*)

7 Students (24.14%) said: 9
7 Students (24.14%) said: 8
6 Students (20.69%) said: 10
4 Students (13.79%) said: 7
2 Students (6.90%) said: 3
2 Students (6.90%) said: 4
1 Students (3.45%) said: 6

Median: 8.00	Mean: 7.83	StDev: 2.0714	Variance: 4.2906	Not Responded: 0

2. The timing and appropriateness of feedback on your progress:
Receiving assignments back in a reasonable time frame, clear explanation of grade
(*Scale: 1 Very Poor to 5 Excellent*)

11 Students (37.93%) said: 4
10 Students (34.48%) said: 5
6 Students (20.69%) said: 2
1 Students (3.45%) said: 3
1 Students (3.45%) said: 1

Median: 4.00	Mean: 3.79	StDev: 1.2358	Variance: 1.5271	Not Responded: 0

3. Independent critical judgement was encouraged:
(*Scale: 1 Very Poor to 5 Excellent*)

13 Students (44.83%) said: 4
11 Students (37.93%) said: 5
4 Students (13.79%) said: 3
1 Students (3.45%) said: 2

Median: 4.00	Mean: 4.17	StDev: 0.8048	Variance: 0.6478	Not Responded: 0

4. OVERALL, how do you rate the value of this course compared with others you have taken at McMaster?
(*Scale: 1 Very Poor to 5 Excellent*)

12 Students (41.38%) said: 4
8 Students (27.59%) said: 5

6 Students (20.69%) said: 3
2 Students (6.90%) said: 1
1 Students (3.45%) said: 2

Median: 4.00	Mean: 3.79	StDev: 1.1142	Variance: 1.2414	Not Responded: 0

5. The organization of this course:

Progression of learning material, resource availability, professor was timely and prepared
(*Scale: 1 Very Poor to 5 Excellent*)

13 Students (44.83%) said: 5
12 Students (41.38%) said: 4
2 Students (6.90%) said: 2
2 Students (6.90%) said: 3

Median: 4.00	Mean: 4.24	StDev: 0.8724	Variance: 0.7611	Not Responded: 0

6. The instructor's response to students:

Approachability, attitude, availability, well-explained answers
(*Scale: 1 Very Poor to 5 Excellent*)

14 Students (48.28%) said: 5
9 Students (31.03%) said: 4
3 Students (10.34%) said: 3
2 Students (6.90%) said: 2

Median: 4.50	Mean: 4.25	StDev: 1.2285	Variance: 1.5093	Not Responded: 1

7. The coverage and fairness of tests:

Material coverage, mark distribution, difficulty level
(*Scale: 1 Very Poor to 5 Excellent*)

10 Students (34.48%) said: 5
10 Students (34.48%) said: 4
8 Students (27.59%) said: 3
1 Students (3.45%) said: 1

Median: 4.00	Mean: 3.97	StDev: 0.9814	Variance: 0.9631	Not Responded: 0

8. Please comment on the quality of the TA's in this course:

- I though the TA was good at explaining the material of the course but I had trouble applying the material to the assignment given due to lack of class time.
- The TAs were really good at going in detail when trying to explain a concept in the tutorials, especially when they wrote out code in real-time. They were also readily available on Slack, and were open to answering any questions people had. Overall, very helpful TAs.
- Jack was amazing at clearly explaining topics that were covered in the SlideSets prepared. The only thing is the fast laser nature didn’t allow as much practice through exercises. It would also help (and this is not a blame), that some of the slides had different methods than the ones used in class, so it was slightly confusing at times. But overall Jack was good at teaching the tutorial materials.

- TAs are very helpful, and will do their best to make sure you understand everything. I like how Jack adds more to the tutorial slides so that the students better understand.
- Tutorials were harder to follow than lectures.
- Jack was an outstanding ta. My greatest regret in this course was not utilizing the tutorials to the full extent. He is approachable and probably the best comp sci ta I have ever had and this is not an exaggeration.
- Jack does an excellent job explaining how to program in Haskell and offers pointers on further learning outside of the classroom.
- very good and clear
- I had Jack. He was very good, don't know what else to say lol.
- Jack and Julie were both amazing!
- TA’s were pretty good.
- TAs were amazing for this course. Especially Jack.
- Jack was the best TA I have had at McMaster
- The TA's weren't that helpful. The tutorial class sessions were too big and the screen was not visible.
- TA's seemingly had a difficult time keeping up with grades on assignments.
- The TA's did a great job of teaching the course
- My TA's are very helpful and are very willing to help students. They almost instantly reply to queries I have.
- I found the TA to be very helpful, compared to others the TA was very helpful in terms of guiding me to the solution. They made sure not to give me the solution right away and helped me understand the process thoroughly.
- jack is a very helpful and good TA
- The TA's were inconsistent with marking assignments and midterms on time.
- The TA Jack was very helpful and friendly. He created a learning environment fostered by a friendly nature and atmosphere. it made it very easy for us to understand the key concepts and practice coding.
- TA understood the content fully and was able to teach the material to students.
- The TA is very helpful and explain course material well

9. Please list aspects of this course that you found valuable and should be continued:
- I believe the knowledge I gain within this course is valuable for my future career because we learned the fundamentals of programming and how programs and computers work this will help whenever i will work with a computer.
- I found the actual concepts learned in class to be very valuable. I found it very helpful when Curtis typed out examples, dot jots and code onto the big screen. That made it easier to follow along. I also found Slack to be very useful. Just having an online platform to ask questions on was very convenient for me, and I found that I would always get an answer back no more than a day after.
- Mr. D'Alves was an amazing instructor and explained things perfectly. He was really passionate about a lot of topics taught so it was easier to learn from him. Also his 2-part midterms are fun and really helpful, cause in CS people gotta communicate with team members a lot so that helped a lot.
- weekly discussion sessions, no matter how pressurising they were at times - more practical aspects of the more theoretical materials (helps understand their application a bit more) - examples in lecture (and exercises in the tutorial) Slides
- instructor and TAs adding additional information outside of what is written on the Slides - ways to message instructor and TA outside of class with questions that will be answered in a very timely manner - discussion sessions: help cover important material

- M&Ms Discussion sessions Test discussion time
- a strong appreciation for functional programming - how computers input/output data and display this on an interface
- the way we take midterm is very interesting.
- Discussion sessions are awesome. Assignments were also good, but maybe some too hard. Of course, that might be the point. Oh and the midterms having an individual part and a group/anyone part was nice.
- I really enjoyed the second part of the test. I've noticed that this makes the test more memorable, to the point where I still remember many of the questions I ended up switching during the second part.
- The discussion sessions were really helpful for reviewing content.
- I found the tutorials extremely useful
- The slides were interesting.
- The tutorials were super useful in learning actual functional coding.
- M&M's were a great idea it made me look back at what I learned that week.
- Recursion.
- I enjoyed the first half of the course where we talked about floating-point numbers and functions and whatnot.
- Nothing
- I liked the M&M's and the Bonus Marks since they brought my mark up.
- the programming language Haskel was learnt and was a different experience as Haskel does not usually follow the set of commands like the other major computer languages. I personally likes the discussions about the various software developments and internet facilities that we use on a daily basis.
- weekly M&Ms - discussion sessions

10. Please list aspects of this course that might be improved:
- I believe the tutorials should be more oriented to content in the assignments, such as examples the resemble the assignment that are explained by the TA.
- Learn how to pronounce names. Also, for the assignments, written feedback would be much appreciated, so that I know what in particular to focus on for next time.
- SlideSet examples could provide more than one way of doing some examples (or offer more explanations to the methods in the Slides as Jack sometimes doesn’t utilise the same method) - slowing the pace a bit? (I felt some topics were a bit more rushed and jumped over more than others) - examples on lecture Slides Providing a bit more of explanation rather than just a slide of a piece of code (or at least organising the Slides a bit better rather than having to jump back and forth between Slides)
- more examples - more explanation of examples
- Tutorials catered to help with assignments More Haskell learning in class. I felt like induction should have been talked about more since recursion in Haskell is essentially induction lol
- - Curtis could time the lectures to fit within the 50 minute limit and not go over. This could be done by having diagrams done beforehand instead of drawing them in class. - A better understanding of where Haskell is implemented in the real world could be
- Discuss section hope can post answer on avenue.
- I'd like for tutorials to go over the assignments. Not to be solved in the tutorial, just to explain what's going on, maybe what should/shouldn't be done, or an example of what should be the result for some formulas. Oh also M&Ms are fine, but it's very easy to forget to do them (I missed 2). So maybe take like 9/10 M&Ms for full marks? It's not a huge deal, but it kinda frustrated me a lot when I remembered.
- The Haskell textbook is kinda bad. I'm pretty sure we can all agree on that.

- Going over some concepts over in a slower or more effective manner. Sometimes a lot of time was not spent on the content and it was difficult to grasps concepts since most of us had no background knowledge.
- I don’t know how this could change, but I felt like the content in this course was too much, compared to other course. And especially for those who don’t have any programming or comp sci background, they would find it really hard
- The discussion session was unfair as the difficulty of the questions varied.
- the jump in difficulty between the general material and coding assignments is very high given the beginner nature of the course.
- Since it is a course that some people have a lot of difficulty understanding. It could use some more examples.
- Tutorial rooms are way too far away.
- The second half of the course seems really useless, also the discussion lectures were challenging as we are put on the spot to answer certain questions. Some people are not comfortable with that.
- why haskell, its useless.
- I despised the objective marking portion of the assignments, and found it abhorrent that the objective marks were based on a pass or fail basis. I hated working with Haskell as a programming language. I didn't mind the functional programming language function of it, but I kept getting parse errors which forced me to comment out some of my code, otherwise I would've got a zero on the objective marking scheme.
- the theoretical part like the case studies and some topics like logic and recursion taught in the lectures could be improved for better understanding by the students.
- Handing back assignment grades within a more reasonable time frame rather than all at once near the end of the Term - Providing solutions to the assignments

11. Additional comments:
- Overall, I learned a lot from this course
- N/A
- This has been by far the best course I've ever taken in my four years at McMaster. Curtis is an amazing prof! I really appreciate the way he also acts as a mentor, by going on tangents, explaining the basics and special cases, and by also being very approachable. I really like the idea of using Slack.
- None
- overall the course was very nice and helpful. the knowledge I gained from it will surely stay with me for a long time
- I found that the 2-part midterm was unfair to those writing as a SAS student because they didn't have the opportunity to compare their answers for a chance to raise their marks.