CQI Report
CS Fundamentals
Fall 2020

Summary

The CS fundamentals committee met about once a semester to share their approaches and general concerns. The instructors of each course also regularly met with instructors of other sections of the same courses. In summer 2020 and well into fall 2020, the committee also met once a week to work on the CS1 outcomes and review the articulation with CS2 (expectations of CS1). The outcomes of CS1 were revised and new outcomes approved by the department in fall 2020. The committee also met with the Software Engineering CQI committee to review whether the outcomes of CS123 were suited to the needs to CS3331, and reversely, ensure that CS3331 instructors knew what to expect from students coming from CS3.
During fall 2020, the instructors in charge of CS1, 2, and 3, collected data about outcomes attainment: this report summarizes the data and provides some recommendations for the next 2-year cycle. Overall, we conclude that all outcomes were covered and met. Some recommendations are made about the need to review the wording of some outcomes.

Reflection on 2018 Recommendations

1/ The granularity of details of our outcomes sometimes hinders our ability to fully diagnose our students' ability. We recommend looking at the outcomes and identify how to address this problem.
→ CS1 outcomes were revisited in 2020. CS2 and CS3 should be revisited next: starting in summer 2021.

2/ We recommend ensuring a better communication with EPCC so that any changes in instruction be discussed and reflected in both institutions. A meeting a semester should take place.
→ A meeting a semester did not occur. However, an alignment meeting took place to review that outcomes were matching. More recommendations regarding our collaboration with EPCC are made in this report.

2020 Recommendations

Outcomes. We recommend reviewing the outcomes of CS2 and CS3, following the process used for reviewing the outcomes of CS1. We anticipate that summer 2021 should be a good time to do that.
EPCC. We recognize that establishing regular meetings can be taxing and is therefore not practical. We also recognize that meetings only are not productive enough. We suggest that we restart sharing material with EPCC (we should all share all assessment instruments, and EPCC would do the same) so we can better diagnose whether there are any discrepancies in coverage. Also, we suggest, in this collaboration fashion, encouraging EPCC faculty to conduct CQI reports like we do and sharing them with us (like we would then share ours with them).
Material sharing. More generally, we recommend that each group of instructor in charge of each course put together a set of problems students should be able to complete as they start their course. This should

help students prepare in between sessions. That can also help keep the articulation between courses clearer.

Instructional Support. We recognize the quality of the training of our TAs / IAs / PLs. We do need to point out that course enrollment should be kept at 40 to 50 students per section. At this size, the amount of instructional support is deemed fine (1 TA, 2 IAs, 1 PL per section). However, if a section has to be open to more students, we want to make clear that the level of support then becomes too little and insufficient to ensure proper student support.

Textbook. All three courses have used an online textbook for a few years now. The level of satisfaction with such book varies across courses. For some courses (CS1 in particular), the granularity of the textbook coverage is not consistent with that of the actual lectures. We suggest spending the rest of spring and summer (2021) to explore other textbook options (possibly still online but not necessarily).

Programming Language. After successfully transitioning CS3 from Java to Python, the committee suggests reconsidering the programming language for CS1 and CS2.

Student Tracking. We observe that some students who pass one course struggle at the next level (e.g., CS1 to CS2, CS2 to CS3). We suggest that instructors communicate across sections often and clearly about such struggles using the concrete cases of students experiencing difficulty as a way to improve our our teaching and our students' success.

Content of this Report

- CS1301/1101 report:
  - Summary of outcomes and recommendations
  - Individual reports (2)
- CS2401 report:
  - Summary of outcomes and recommendations
  - Individual reports (3)
- CS2302 report:
  - Summary of outcomes and recommendations
  - Individual reports (2)

CS 1301/1101 – Intro to Computer Science, Fall 2020 CQI Report

Course Description and Outcomes

Course Description
Students will learn to be active learners, understand the motivations for computing, basic concepts of algorithms, basic computer organization, and impacts of computing. They will develop problem-solving skills, implement solutions to computing problems in a high- level programming language, and build team skills, critical-thinking skills, and professionalism.

Knowledge and Abilities Required Before Entering the Course
Students entering the course are not required to have a background in Computer Science or programming. They should be familiar with topics from Pre-calculus, including algebraic functions, proofs, and base representations of numbers.

Prerequisites:
MATH 1508 or MATH 1411 with a grade of C or better

Textbook:
Online text: Intro to Computer Science, Zybooks, available at zybooks.zyante.com.

Course Outcomes

CS 1301 – Intro to Computer Science

Level 1: Knowledge and Comprehension. Level 1 outcomes are those in which the student has been exposed to the terms and concepts at a basic level and can supply basic definitions. On successful completion of this course, students will be able to describe, at a high level:

1. The history of computing
2. The relation between computing and society, including social, ethical, and legal issues
3. Computing as a profession, from required knowledge and skills to major career options
4. Computer representation of simple data types and operations, including operations with binary numbers
5. Differences among programming languages
6. Pseudocode of the use of Multi-D arrays
7. Pseudocode of the use of Linked lists

Level 2: Application and Analysis. Level 2 outcomes are those in which the student can apply the material in familiar situations, e.g., can work a problem of familiar structure with minor changes in the details. Upon successful completion of this course, students will be able:

1. To analyze problems and express solution algorithms in pseudocode, including the correct use of:
    a. Arithmetic and logical expressions
    b. Simple I/O operations
    c. User-defined subprograms, including recursive methods
    d. User-defined types
2. To use testing and debugging strategies, including black-box and white-box testing, test drivers, stubs and test suites, to identify software faults
3. Use teamwork roles and methods in the classroom

Level 3: Synthesis and Evaluation. Level 3 outcomes are those in which the student can apply the material in new situations. This is the highest level of mastery. On successful completion of this course, students will be able to use the syntax and semantics of a higher-level language to express solutions to programming problems, including the pseudocode correct use of:

1. Basic variable types such as integer, real number, character, string, 1-D array
2. Assignment, arithmetic, and logical operations

Basic control structures: if-then, for-loop, while-loop

CS 1101 – Intro to Computer Science Lab

Level 1: Knowledge and Comprehension. Level 1 outcomes are those in which the student has been exposed to the terms and concepts at a basic level and can supply basic definitions. On successful completion of this course, students will be able to describe, at a high level:

1. Computer representation of simple data types and operations, including operations with binary numbers
2. Technical aspects of computing, including memory, operating systems, editors, interpreters, compilers, debuggers, and virtual machine
3. Differences among programming languages
4. The purpose and use of exceptions
5. Pseudocode and implementation in a programming language of the use of Multi-D arrays
6. Pseudocode and implementation in a programming language of the use of Linked lists

Level 2: Application and Analysis. Level 2 outcomes are those in which the student can apply the material in familiar situations, e.g., can work a problem of familiar structure with minor changes in the details. Upon successful completion of this course, students will be able:

1. To analyze problems and express solution algorithms in pseudocode
2. To implement pseudocode algorithms in a high-level language, including the correct use of:
    a. Arithmetic and logical expressions
    b. Simple I/O operations
    c. User-defined subprograms, including recursive methods
    d. User-defined types
3. To use testing and debugging strategies, including black-box and white-box testing, test drivers, stubs and test suites, to identify software faults
4. Use teamwork roles and methods in the classroom

Level 3: Synthesis and Evaluation. Level 3 outcomes are those in which the student can apply the material in new situations. This is the highest level of mastery. On successful completion of this course, students will be able to use the syntax and semantics of a higher-level language to express solutions to programming problems, including the correct use of:

1. Basic variable types such as integer, real number, character, string, 1-D array
2. Assignment, arithmetic, and logical operations

Basic control structures: if-then, for-loop, while-loop

Summary of Outcomes Coverage and Attainment

CS 1301
Nearly all outcomes were covered and assessed in two sections of CS 1301.

Section 1:

- Outcome 2.1.b (Simple I/O Operations) were not met.
- Outcome 2.3 (teamwork roles & methods) was not assessed through graded assessment, but was observed by instructor in breakout rooms
- All other outcomes were met satisfactorily

Section 2:

- Outcomes 1.7, 2.1D (user-defined types; 63%), 2.2 (testing) were not met.
- Outcomes 2.3 (using teamwork roles) were not assessed for grading.
- All other outcomes were met, marginally (70%-74%) or fully (75% or above).

In general, the instructors are satisfied with the success of the outcome results for the course.

CS 1101
Nearly all outcomes were covered and assessed in two sections of CS 1101.

Section 1:

- Outcome 1.6 (Pseudocode of Linked Lists) was not assessed
- Outcome 2.4 (teamwork roles) was not assessed

Section 2:

- All outcomes except 2.4 were met, either marginally (70%-74%) or fully (75% or above).
- Outcome 2.4 was not assessed even though students worked in groups for in-lab activities.

In general, the instructors are satisfied with the success of the outcome results for the course.

Students' Success: Passing and Failure Rates

CS 1301						
A	B	C	D	F	Total Earned Grade	W
Mejia, Section 1						
16	12	7	1	9	45	12
Akbar, Section 2						
16	12	4	4	7	43	4
Reynolds (Google), Section 3						
					0	2
Total						
32	24	11	5	16	88	18
36.4%	27.3%	12.5%	5.7%	18.2%	100.0%	
67			21			
76.1%			23.9%			

The passing and failure rates for Sections 1 & 2 are very similar. Section 1 had a passing rate for credit (C or better) of 77.8% and section 2 had a passing rate of 74.4%. Moreover, the failure rate for credit (D or worse) Section 1 was 22.2% and section 2 was 25.6%. Overall, the pass/fail rate for Sections 1 & 2 of CS 1301 was 76.1% & 23.9%, respectively.

CS 1101						
A	B	C	D	F	Total Earned Grade	W
Mejia, Section 1						
23	6	6	1	9	45	12
Akbar, Section 2						
26	7	2	2	9	46	6
Reynolds (Google), Section 3						
					0	2
Total						
49	13	8	3	18	91	20
53.8%	14.3%	8.8%	3.3%	19.8%		
70			21			
76.9%			23.1%			

The passing and failure rates for Sections 1 & 2 are very similar. Section 1 had a passing rate for credit (C or better) of 77.8% and section 2 had a passing rate of 76.1%. Moreover, the failure rate for credit (D or worse) Section 1 was 22.2% and section 2 was 23.9%. Overall, the pass/fail rate for Sections 1 & 2 of CS 1301 was 76.9% & 23.1%, respectively.

Sections 1 & 2 had very similar passing and failure rates for both CS 1301/1101. The overall failure rate appears to be higher than in some semesters. Though there is no documentation to support this claim, it appears that learning introductory material through an online platform with limited interaction between other students played a significant role in the higher failure rates. Moreover, many students who failed the course stopped attending or stopped submitting assignments completely. The overall drop rates for CS 1301/1101 were significantly higher than usual. Additionally, most of the drop cases can be specifically attributed to circumstances outside of class (COVID-19 related for most drops).

<u>Summary of Observations and Recommendations</u>

Lessons Learned
Online learning is difficult for both instructors and students. A lot of time was spent preparing the logistical aspects of the course in an attempt to reduce academic dishonesty. Additionally, not having physical access to work out and trace problems on a piece of paper/whiteboard during office hours between students and the instructional team caused difficulties in catching potentially simple mistakes. As a result, it is important to continue being innovative in delivering one-on-one help to students in an online environment.

Virtual Positives
Some live interaction in class was improved through polling, quick surveys, and chat messages. Additionally, quick questions were easily asked outside of class to the instructor and instructional team – this has improved the accessibility of the instructional team as a whole. Some virtual office hours may still be appropriate when moving back to a full face to face format, but in person office hours would be preferred.

Instructional Team

The student instructional team is critical for the success of the course, specifically the TA/IA. Students attended office hours with the instructional team more than the instructor. Students tend to be more comfortable with TA/IA than with the instructor.

Recommendations

1. Textbook
    a. The textbook for CS 1301/1101 is lacking comprehensive coverage of what is expected for this course, and an alternative should be considered. The activities provided in the textbook tend to be more tedious than useful for students. The animated visualization can help some students, but the tradeoff of not having a better depth of subject matters may be hurting more students.
    b. It is recommended to begin looking for an alternative text for this class and promote reading habits through reading quizzes, or similar.
2. Class Size
    a. The size of the classes is increasing, and it would be useful to hold the class size between 40-50 students. Specially for online classes, a class of size 50 can be challenging to monitor closely and respond accordingly.
    b. One suggestion to address the large class size is to increase the number of sections of CS 1301/1101.
3. Order of Topics
    a. Based on the order taught in Sections 1 & 2, the order is clear and makes sense.
    b. Continue teaching the topics in the same order.
4. Identify Students at Risk of Failing
    a. Provide more formal training to TA/IAs regarding how to identify struggling students, how to communicate, and work with those students.
    b. Reach out earlier and more frequent to students who are struggling to master concepts.
5. Increase the size of the Instructional Team
    a. With the large number of students enrolled in the course, it would be useful to increase the size of the TA/IA team.
    b. Possibly increase the number of IAs from 2 to 3 qualified students.
    c. Encourage upper division faculty to encourage the graduate students in their classes to apply for TA positions, especially female students.
6. Outcomes and Assessment
    a. Provide information on techniques to help improve assessment for outcomes that are not easily assessed.
7. Assignments
    a. Continue to create new practice problems and interactions for students.
    b. Create additional innovative assignments that are interesting for students to remain engaged and connection to real-world problems in Computer Science.

Course Number: CS 1301
Course Title: Intro to Computer Science
Course Instructor: Daniel Mejia

Grade Distribution
There were 57 students registered for credit on census day – 12 students dropped the course after census day up until the course drop deadline and received an automatic W. 45 students earned a grade in the course.

Grade	Pass			Fail	
	A	B	C	D	F
No. of students	16/45	12/45	7/45	1/45	9/45
Percentage	35.6%	26.7%	15.5%	2.2%	20%
Total	77.8%			22.2%	

*C is a passing grade in CS 1301.

Passing Students

- Of the passing students, 3 opted into the S/U option: 2 B's and 1 C

Failing Students

- Of the failing students, 6 opted into the S/U option: 6 F's to U's
- S1 did not take any exams and turned in one assignment the entire semester. Stopped accessing the course October 8, 2020.
- S2 took only Exam 1 and completed only about half of the assignments/quizzes. Stopped accessing the course October 8, 2020.
- S3 took Exam 1 & 2 and did not do well. Completed half of the assignments/quizzes. Stopped accessing the course November 3, 2020.
- S4 took Exam 1 and completed about half of the assignments. Stopped accessing the course October 16, 2020.
- S5 took Exam 1, 2, 3 and did poorly. Completed about 60% of the assignments. Student completed the entire course.
- S6 took Exam 1, 2, 3 and did poorly. Student was unable to master the material and opted to remain in the course. Student completed the entire course.
- S7 took Exam 1, 2 and did poorly. Student completed the entire course.
- S8 took Exam 1 & 2 and did poorly. Student completed the entire course.
- S9 took Exam 1, 2, 3, and Final Exam. Student was offered an Incomplete, student did not respond.
- S10 took Exam 1, 2, 3, and Final Exam – Earned D. Student did not master the material.

All students were contacted after poor performance on an exam or if they did not take an exam. A majority of the students did not respond to email or Blackboard messages. On occasion students would respond and appropriate action would be taken on a case-by-case basis. A majority of students were able to complete the missing work prior to the end of the semester.

One student was given an Incomplete, which was later changed to a C. Students were advised prior to the end of the semester advising them of the option of the S/U grade mode or to withdraw. S1-S4 did not participate throughout the semester and S5-10 did not master the concepts of the course.

The final exam was separated into three parts to map with each of the midterm exams. The exam grades were replaced if the corresponding section on the final exam was higher than the original exam grade. Homework was collected throughout the semester and a majority of the homework was accepted late, with minimal to no penalty.

Learning Outcomes

Assessment was done using exams, in-class quizzes, and one research discovery homework only. Homework practice problems were assigned as part of the class but was intended for students to use as practice and not as an assessment.

For each outcome I measured the average score obtained by students on the exam or quiz questions. The nominal acceptance value was 70%; Most outcomes were assessed multiple times, however, most outcomes were practiced and mastered later in the semester (i.e. Exam 2 & Final Exam), thus these scores are representative of what the student has learned.

Learning Outcomes

Level 1: Knowledge and Comprehension.

8. The history of computing
   F3-Q12 – 97.3%
   F3-Q16 – 96%
   Hist of Computing/Profession HW – 94.5%
   Excellent

9. The relation between computing and society, including social, ethical, and legal issues
   F3-Q25 – 100%
   Hist of Computing/Profession HW – 94.5%
   Excellent

10. Computing as a profession, from required knowledge and skills to major career options
    Hist of Computing/Profession HW – 94.5%
    Excellent

11. Computer representation of simple data types and operations, including operations with binary numbers

    F3-Q18 – 82%
    F3-Q19 – 98%
    F3-Q20 – 100%
    Excellent

12. Differences among programming languages
    E3-Q29 – 93%

E3-Q29 – 98%
F3-Q17 – 98%
Excellent

13. Pseudocode of the use of Multi-D arrays
    E2-Q2 – 80.6%
    F2-Q4 – 82.4%
    F2-Q12 – 81.8%
    Good

14. Pseudocode of the use of Linked lists
    E3-Q2 – 85.5%
    F3-Q5 – 61.4%
    Satisfactory

Level 2: Application and Analysis.

4. To analyze problems and express solution algorithms in pseudocode, including the correct use of:
    a. Arithmetic and logical expressions
       F1-Q2 – 87%
       F1-Q5 – 78.4%
       F2 -Q10 – 89.2%
       Good

    b. Simple I/O operations
       Quiz 2-Q8 – 57.7%
       Unsatisfactory

    c. User-defined subprograms, including recursive methods
       E2-Q7 – 83%
       F1-Q17 – 74.2%
       F2-Q7 – 83.4%
       F2-Q11 – 83.4%
       F2-Q15 – 76.8%
       F3-Q23 – 92%
       Good

    d. User-defined types
       E3-Q4 – 86%
       E3-Q5 – 75.6%
       E3-Q15 – 85.4%
       F3-Q26 – 70.3%
       Good

5. To use testing and debugging strategies, including black-box and white-box testing, test drivers, stubs and test suites, to identify software faults
   F3-Q7 – 78.4%
   F3-Q24 – 95%
   Satisfactory

6. Use teamwork roles and methods in the classroom

   Breakout room role assignments to complete practice exercises. Observed by instructor and instruction team that students participated and followed assignments satisfactorily.

   Satisfactory

Level 3: Synthesis and Evaluation.

3. Basic variable types such as integer, real number, character, string, 1-D array
   E1-Q4 – 89.5%
   E2-Q1 – 88%
   E2-Q2 – 80.6%
   E2-Q4 – 71.4%
   E2-Q7 – 83%
   E2-Q15 – 94%
   F1-Q13 – 97.4%
   F2-Q5 – 83.2%
   Good

4. Assignment, arithmetic, and logical operations
   F1-Q12 – 94.6%
   F1-Q14 – 100%
   F1-Q16 – 94.6%
   Excellent

5. Basic control structures: if-then, for-loop, while-loop
   F1-Q1 – 100%
   F1-Q6 – 71.2%
   F1-Q9 – 83.4%
   F1-Q18 – 91.4%
   Good

Analysis of the Results, and Instructor Recommendations

The results of the learning outcomes were all met with at least a satisfactory level, except for 2.1.b. Although it was discussed in great detail, this was not heavily assessed in the lecture section. The assessment made was prior to a lot of practice in the corresponding lab. In the future, more assessment should be made in this area.

Additionally, more emphasis of testing and all level one outcomes should be made to ensure that the students have enough exposure to these areas. Moreover, teamwork in the classroom proved to be difficult for the instructor and students in the virtual environment as there were some students not actively participating; though a majority of students were able to incorporate simple team strategies into their practice exercises.

Except for the abovementioned, all outcomes were assessed several times with satisfactory (or better) performance in each area. Most of the assessments were made multiple times, however, a majority of the results reflected in this report come from the last portion of the class where students have deepened their understanding of the material. Students greatly improved in their understanding of topics from the beginning of the course (i.e. Exam 1) to the end of the course (i.e. Final Exam).

Recommendations

Instructor recommendations:

1. Increase exposure to testing methodologies

Course recommendations:

1. Develop clear ways to measure teamwork in the classroom
2. Provide a sample schedule of topics to new instructors

Course Outcomes: CS 1301, Fall 2020

Course Number: CS 1301
Course Title: Introduction to Computer Science
Course Instructor: Monika Akbar
Instructional Assistant: Erik Macik (20 hours/week)

Course Description
Students will learn to be active learners, understand the motivations for computing, basic concepts of algorithms, basic computer organization, and impacts of computing. They will develop problem-solving skills, implement solutions to computing problems in a high- level programming language, and build team skills, critical-thinking skills, and professionalism.

Grade Distribution
The class started with 40+ students enrolled, and several students dropped before the course drop date. The table below summarizes the distribution of final letter grades that the remaining 43 students earned.

Grade	Pass				Failure
	A	B	C	D	F
No. of students	16	12	4	4	7
Percentage	37.2%	27.9%	9.3%	9.3%	16.27%
Total	74.41% (32/43)			8% (4/43)	16.27% (7/43)

Learning Outcomes

Level 1: Knowledge and Comprehension. Level 1 outcomes are those in which the student has been exposed to the terms and concepts at a basic level and can supply basic definitions. On successful completion of this course, students will be able to describe, at a high level:

15. The history of computing
    Met (Average 100%)

16. The relation between computing and society, including social, ethical, and legal issues
    Met (Average 100%)

17. Computing as a profession, from required knowledge and skills to major career options
    Met (Average 97.3%)

18. Computer representation of simple data types and operations, including operations with binary numbers
    Met (Average 93%)

19. Differences among programming languages
    Met (Average 93.1%)

20. Pseudocode of the use of Multi-D arrays
    Met (Average 87.5%)

21. Pseudocode of the use of Linked lists
    Not met (Average 58.7%)

Level 2: Application and Analysis. Level 2 outcomes are those in which the student can apply the material in familiar situations, e.g., can work a problem of familiar structure with minor changes in the details. Upon successful completion of this course, students will be able:

7. To analyze problems and express solution algorithms in pseudocode, including the correct use of:
    a. Arithmetic and logical expressions
       Met (Average 89.7%)

    b. Simple I/O operations
       Met (Average 86.7%)

    c. User-defined subprograms, including recursive methods
       Met (Average 78.6%)

    d. User-defined types
       Not Met (Average 63.7%)

8. To use testing and debugging strategies, including black-box and white-box testing, test drivers, stubs and test suites, to identify software faults
   Not Met (Average 63.7%)

9. Use teamwork roles and methods in the classroom
   Not assessed. Students worked in teams for in-class activities.

Level 3: Synthesis and Evaluation. Level 3 outcomes are those in which the student can apply the material in new situations. This is the highest level of mastery. On successful completion of this course, students will be able to use the syntax and semantics of a higher-level language to express solutions to programming problems, including the pseudocode correct use of:

6. Basic variable types such as integer, real number, character, string, 1-D array
   Met (Average 89.3%)

7. Assignment, arithmetic, and logical operations
   Met (Average 93.4%)

8. Basic control structures: if-then, for-loop, while-loop
   Met (Average 80.9%)

Outcomes 1.7, 2.1D (user-defined types; 63%), 2.2 (testing) were not met.

Outcomes 2.3 (using teamwork roles) were not assessed for grading.

All other outcomes were met, either marginally (70%-72%) or fully (72% or above).

Observations

1. Outcomes related to user-defined types (class and objects) were not met (1.7, 2.1D). Students have difficulty converting their understanding of user-defined types into programming. They perform well in basic questions (90% median for creating objects of a class) but struggle when asked to manipulate object-specific functions.
2. Some of the outcomes for the class are more suitable for the lab (I/O: 2.1B, teamwork: 2.3).
3. Learning outcome related to testing (2.2) was not met. Testing is ingrained with much of the concepts introduced in the class. I will develop formal assessment tools for measuring specific types of testing skills (blackbox-testing, whitebox-testing).
4. The instructional team with the TA and IAs is instrumental in providing a supportive environment for the students. This team helps monitor at-risk students and reach out to them on a regular basis.
5. The background of the students is mixed in this introductory class. It remains a challenge to engage all students, with varying expertise, with the class. Specially, given the COVID-19 situation, the online class of Fall 2020 made it difficult to keep students engaged and motivated. Also, it is not clear what is the impact of the measures taken to keep the integrity of the online assessments (e.g., creating question sets for a single question).

Recommendations

1. Continue providing examples and practice opportunities (in-class activities, homework) to build expertise on the basic concepts.
2. Continue having frequent assessments (quiz, homework) to help students learn.
3. Develop effective assessment instruments for assessing teamwork.

Course Number: CS 1101
Course Title: Intro to Computer Science
Course Instructor: Daniel Mejia

Grade Distribution
There were 58 students registered for credit on census day – 13 students dropped the course after census day up until the course drop deadline and received an automatic W. 45 students earned a grade in the course.

Grade	Pass			Fail	
	A	B	C	D	F
No. of students	23/45	6/45	6/45	1/45	9/45
Percentage	51.1%	13.4%	13.3%	2.2%	20%
Total	77.8%			22.2%	

*C is a passing grade in CS 1101.

Passing Students

- Of the passing students, 2 opted into the S/U option: 1A and 1 C

Failing Students

- Of the failing students, 7 opted into the S/U option: 7 F's to U's
- S1 did not turn in any comprehensive lab (3) and submitted 1/13 assignments. Last accessed the course October 6, 2020.
- S2 did not turn in any comprehensive lab (3) and submitted 2/13 assignments. Last accessed the course October 8, 2020.
- S3 did not turn in any comprehensive lab (3) and submitted 2/13 assignments. Last accessed the course October 5, 2020.
- S4 did not turn in any comprehensive lab (3) and submitted 6/13 assignments. Last accessed the course November 23, 2020.
- S5 turned in comprehensive lab 1 and did poorly. Submitted 5/13 assignments. Last accessed course October 16, 2020.
- S6 did not turn in any comprehensive lab (3). Submitted 9/13 assignments. Last accessed the course November 3, 2020.
- S7 turned in comprehensive lab 1 and did poorly. Submitted 10/13 assignments. Student completed the course and opted not to drop the course. Student did not make progress mastering the topics.
- S8 turned in comprehensive lab 1 & 2 and did poorly. Submitted 8/13 assignments. Student completed the course. Student did not make progress mastering the topics.
- S9 turned in comprehensive lab 1 & 2 and did poorly. Submitted 5/13 assignments. Last accessed course November 19, 2020.
- S10 turned in comprehensive lab 1, 2 & 3 and did not progress in mastering the topics. Submitted 9/13 assignments. Student completed course but did not successfully demonstrate mastery of topics.

- One student was given an Incomplete, which was later changed to a C. Students were advised prior to the end of the semester advising them of the option of the S/U grade mode or to withdraw. S1-S6 did not turn in many major and minor assignments throughout the semester and S7-10 did not master the concepts of the course.

Learning Outcomes

Assessment was done using the comprehensive labs, class assignments, and quizzes. Homework was assigned insofar as an opportunity to complete the assigned class assignment that was begun in lab session.

The nominal acceptance value was 70%; Most outcomes were assessed multiple times, however, most outcomes were practiced and mastered later in the semester (i.e. Comprehensive Lab 2-3), thus these scores are representative of what the student has learned. The main focus of assessment was through the Comprehensive Labs.

Learning Outcomes

Level 1

7. Computer representation of simple data types and operations, including operations with binary numbers
   BMICalulator – 99.7%
   Music – 98.8%
   LeapYear – 99.1%
   PasswordChecker – 94.7%
   NameSorter – 86.7%
   FanClass – 94.7%
   Excellent

8. Technical aspects of computing, including memory, operating systems, editors, interpreters, compilers, debuggers, and virtual machine
   In-lab demos
   Quiz 1-Q5 –95.9%
   Quiz 1-Q6 – 91.7%
   Excellent

9. Differences among programming languages
   Quiz 1-Q4 – 85.5%
   Quiz 1-Q7 – 85.5%
   Excellent

10. The purpose and use of exceptions
    CL3 – 91.2%
    Converter – 93.6%
    GPA – 92.5%
    Excellent

11. Pseudocode and implementation in a programming language of the use of Multi-D arrays
    CL2 – 92.3%
    Excellent

12. Pseudocode and implementation in a programming language of the use of Linked lists
    Pseudocode was shown and discussed; however, it was not formally assessed.
    Not Assessed

Level 2

5. To analyze problems and express solution algorithms in pseudocode
   CL1 – 84.5%
   CL2 – 92.3%
   Excellent

6. To implement pseudocode algorithms in a high-level language, including the correct use of:
   a. Arithmetic and logical expressions
      CL1 – 84.5%
      CL2 – 92.3%
      CL3 – 91.2%
      Excellent

   b. Simple I/O operations
      CL1 – 84.5%
      CL2 – 92.3%
      CL3 – 91.2%
      Excellent

   c. User-defined subprograms, including recursive methods
      CL2 – 92.3%
      CL3 – 91.2%
      Excellent

   d. User-defined types
      CL3 – 91.2%
      Excellent

7. To use testing and debugging strategies, including black-box and white-box testing, test drivers, stubs and test suites, to identify software faults
   ArrayManipulator – 88.4%
   CL2 – 92.3%
   CL3 – 91.2%
   Excellent

8. Use teamwork roles and methods in the classroom
   Team strategies were introduced and conducted; however, it was not formally assessed.
   Not Assessed

Level 3

3. Basic variable types such as integer, real number, character, string, 1-D array
   CL1 – 84.5%
   CL2 – 92.3%
   CL3 – 91.2%
   Excellent

4. Assignment, arithmetic, and logical operations
   CL1 – 84.5%
   CL2 – 92.3%
   CL3 – 91.2%
   Excellent

5. Basic control structures: if-then, for-loop, while-loop
   CL1 – 84.5%
   CL2 – 92.3%
   CL3 – 91.2%
   Excellent

Analysis of the Results, and Instructor Recommendations

The results of the learning outcomes were all met with a satisfactory (excellent) level, except for 1.6 & 2.4 which were not formally assessed. Both 1.6 and 2.4 were assessed in the lecture portion of the course, but not in the lab. A majority of the outcomes are covered by the comprehensive labs as they encompass the majority of the topics that are discussed. Due to being online, some topics took longer than expected to cover; thus, less time was spent on linked lists.

1.6 - LinkedList code was seen, but students were never asked to write code with LinkedList. LinkedList was demo'd in lecture/lab for students to understand what it fundamentally is.

2.4 - Due to the online environment it was difficult to develop reasonable teams that enhance learning in code. Students were encouraged to break out into rooms with the TA/IAs to discuss coding problems collaboratively. TA/IAs reported successful discussion and increased understanding of concepts being covered.

Recommendations

Instructor recommendations:

1. Include more lab quizzes to formally measure level 1 outcomes outside of coding
2. Introduce more testing strategies

Course recommendations:

1. Develop clear ways to measure teamwork in the classroom
2. Provide a sample schedule of topics to new instructors

Course Outcomes: CS 1101, Fall 2020

Course Number: CS 1101
Course Title: Introduction to Computer Science
Course Instructor: Monika Akbar
Instructional Assistant: Erik Macik (20 hours/week)

Course Description
Students will learn the foundations of algorithmic thinking and algorithm development, and learn how to implement them in a variety of languages. They will also learn to be active learners. They will develop problem-solving skills and build team skills, critical- thinking skills, and professionalism.

Grade Distribution
The class started with 40+ students enrolled, and several students dropped before the course drop date. The table below summarizes the distribution of final letter grades that the remaining 46 students earned.

Grade	Pass				Failure
	A	B	C	D	F
No. of students	26	7	2	2	9
Percentage	56.5%	15.2%	4.3%	4.3%	19.5%
Total	76.08% (35/46)			4.3% (2/46)	19.5% (9/46)

Learning Outcomes

Level 1: Knowledge and Comprehension. Level 1 outcomes are those in which the student has been exposed to the terms and concepts at a basic level and can supply basic definitions. On successful completion of this course, students will be able to describe, at a high level:

13. Computer representation of simple data types and operations, including operations with binary numbers
    Met (Average 71%, Median 85%)

14. Technical aspects of computing, including memory, operating systems, editors, interpreters, compilers, debuggers, and virtual machine
    Met (Average 86.7%)

15. Differences among programming languages
    Met (Average 86.7%)

16. The purpose and use of exceptions
    Met (Average 83.4%)

17. Pseudocode and implementation in a programming language of the use of Multi-D arrays
    Met (Average 77.4%)

18. Pseudocode and implementation in a programming language of the use of Linked lists
    Met (Average 100%)

Level 2: Application and Analysis. Level 2 outcomes are those in which the student can apply the material in familiar situations, e.g., can work a problem of familiar structure with minor changes in the details. Upon successful completion of this course, students will be able:

9. To analyze problems and express solution algorithms in pseudocode
   Met (Average 87.7%)

10. To implement pseudocode algorithms in a high-level language, including the correct use of:
    a. Arithmetic and logical expressions
       Met Marginally (Average 73.2%)

    b. Simple I/O operations
       Met (Average 77.9%)

    c. User-defined subprograms, including recursive methods
       Met (Average 87.1%)

    d. User-defined types
       Met (Average 81.1%)

11. To use testing and debugging strategies, including black-box and white-box testing, test drivers, stubs and test suites, to identify software faults
    Met (Average 100%)

12. Use teamwork roles and methods in the classroom
    Not assessed.

Level 3: Synthesis and Evaluation. Level 3 outcomes are those in which the student can apply the material in new situations. This is the highest level of mastery. On successful completion of this course, students will be able to use the syntax and semantics of a higher-level language to express solutions to programming problems, including the correct use of:

6. Basic variable types such as integer, real number, character, string, 1-D array
   Met (Average 90.7%)

7. Assignment, arithmetic, and logical operations
   Met (Average 82.1%)

8. Basic control structures: if-then, for-loop, while-loop
   Met (Average 78.9%)

All outcomes except 2.4 were met, either marginally (70%-74%) or fully (75% or above). Outcome 2.4 was not assessed even though students worked in groups for in-lab activities.

Observations

1. The in-lab team activities (e.g., Kahoot) were engaging. Students had a chance to review code, identify bugs in code, and assess their understanding. Such activities should continue.

2. The Peer leading activity is also helpful for reinforcing student learning.
3. Students require significant support to complete the comprehensive labs. Structuring the labs so that students submit the pesudocodes well-ahead of the due date and receive feedback makes it helpful for students to complete these labs.
4. The online format poses a challenge in observing student work and providing instant feedback.

Recommendations

1. Outcome related to teamwork, 2.4, can be more specific.
2. TA for the class+lab should have a strong CS background and should be a PhD student. It would be helpful to retain the same TAs over semesters. The same goes for the IAs. They make a great impact only when we recruit the best TA, train them, and retain them for the following semesters.
3. Testing can be integrated with some of the labs as deliverables. Currently, the lab rubric we use includes elements for best programming practices, along with other lab requirements. We can incorporate testing as part of that rubric.

CS2401 Elementary Data Structures, Fall 2020 CQI Report

Course Description and Outcomes

Course Description

This is the second course for students majoring in Computer Science. Students will learn about fundamental computing algorithms including searching and sorting; recursion; elementary abstract data types including linked lists, stacks, queues and trees; and elementary algorithm analysis.

Knowledge and Abilities Required Before the Students Enter the Course: Students are assumed to be comfortable programming in Java. Students should be able to code basic arithmetic expressions, define simple classes, use strings, code loops and conditional statements, write methods, create objects from classes, invoke methods on an object, perform basic text file input and output, and use arrays.

Prerequisite: CS 1301 and CS 1101 with a grade of C or better in both.
Mandatory Textbook: Online text: Elementary Data Structures/Algorithms, zybooks, available at zybooks.zyante.com.

Course Outcomes

Level 1: Knowledge and Comprehension: Level 1 outcomes are those in which the student has been exposed to the terms and concepts at a basic level and can supply basic definitions. On successful completion of this course, students will be able to:

1. Explain the concept of polymorphism

Level 2: Application and Analysis: Level 2 outcomes are those in which the student can apply the material in familiar situations, e.g., can work a problem of familiar structure with minor changes in the details. Upon successful completion of this course, students will be able to:

1. Describe, implement, and use the following concepts:
    1. classes, subclasses, and inheritance
    2. encapsulation and information hiding
2. Describe, implement, and use the following algorithms:
    1. sequential and binary search
    2. quadratic and O(n log n) sorting
    3. string manipulation and parsing
3. Describe and trace computer representation and memory allocation of:
    1. integers, real numbers, arrays, and objects

    2. methods, including recursive methods and the use of activation records
4. Use basic notions of algorithm complexity:
    1. use Big-O notation to express the best-, average- and worst-case behaviors of an algorithm
    2. determine the best, average and worst-case behaviors of a simple algorithm
    3. assess basic time and space trade-offs in algorithms
5. Use recursion and iteration as problem solving techniques

Level 3: Synthesis and Evaluation: Level 3 outcomes are those in which the student can apply the material in new situations. This is the highest level of mastery. On successful completion of this course, students will be able to:

1. Design and implement solutions to computational problems using the following data structures:
    1. multi-dimensional arrays;
    2. lists implemented as arrays or linked lists;
    3. stacks;
    4. queues;
    5. binary trees and binary search trees.

Summary of Outcomes Coverage and Attainment

All outcomes were covered and assessed in all three sections of CS2.

If we look at the % of success of our students on each of the CS2 outcomes, it appears that some outcomes may not have been met. A deeper look at these seemingly unmet outcomes shows that the lower % of success are usually due to modes of evaluations that contribute to low grades (and therefore bring down the average), but do not necessarily mean that students' work is not satisfactory (see individual sections' reports for more details). So overall, the instructors are satisfied with the attainment of all outcomes.

Students' Success: Passing and DFW rates

A	B	C	D	F / U	Drop	I
Ceberio, Section 1						
22	7	1	1	6	11	3
Hossain, Section 2						
22	11	6	1	4	2	0
DeBlasio, Section 3						
8	4	11	6	3	11	1
Total						
52	22	18	8	13	24	4
36.8%	15.6%	12.8%	5.7%	9.2%	17%	2.8%

The passing and DFW rates vary significantly across sections. In one section, the passing rate is about 88%, while in the other two, it is respectively about 65% and 55%. The overall passing/DFW rates are: 65.2% / 31.9%.

The lower passing rates for the first and third sections is due to a larger than usual drop rate. Most of the drop cases can be specifically attributed to circumstances outside of class (COVID-related for the majority).

Summary of Observations and Recommendations

Lessons learned from teaching online: Anecdotally, it seems that some students enjoy the online setting for learning. We suggest keeping this in mind and possibly polling the students to know what works best for them, and then possibly to think of creative ways to accommodate these students even when we are back in person.

Cheating is rampant: (not a majority but an issue anyway) Many of our current students have never been on campus. The lack of bond with the department may be a reason for lower ethics and a propension in cheating. We recommend thinking of ways to engage these students once back on campus, to increase their feeling of belonging to the CS department and identity as a computer science student.

Textbook: It is not clear whether the current textbook really supports our students' success. We suggest that over the next semesters, the CS2 instructors explore alternative options or get a better understanding of the value of the current one.

Outcomes: In light of summer 2020's review of the CS1 outcomes, it appears that the CS2 outcomes need revamping. We suggest conducting such a review in summer 2021.

Reflections on Fall 2018 Recommendations

Recommendation 1. Based on the third section of CS2401 experience in fall 2018, we recommend to try and accommodate struggling students in smaller groups and see if this increases their retention and success consistently over time.
→ The online nature of our work, over 2020, has made working in small group and organizing group meeting much easier. However, student engagement is hard to guarantee. Overall, students are getting used to working in groups and we are getting used to organizing them, so things have improved in this respect.

Recommendation 2. We recommend to keep using an online textbook with reading and homework tracking.
→ We have kept using an online textbook: zybook. It is not clear how much students retain / learn with this book, but the ability to track their study habit is very useful.

Recommendation 3. We suggest adopting a way of assessing students based on skill acquisition since it is more attuned to our students' life circumstances (heavy work load, children, family commitments, etc.).
→ The sections of Dr. Ceberio have been assessed based on skill acquisition rather than using traditional grading. It turns out that even with traditional grading, instructors do use qualitative approach to their decision making about final grade. However, making it explicit from the start of the semester proves to be very helpful for retention.

Recommendation 4. We propose to reconsider the sequence in which topics are introduced in this course to make it flow better for students (to be tested in spring 2019).
→ Since Spring 2019, we have swapped the coverage order of trees and stacks/queues. We now cover trees (binary trees, BSTs) right after lists, and we close the semester on stacks and queues.

Recommendation 5. We recommend removal of outcome 2.5c to level 1. The topic is covered in the next course.
→ Outcomes changes were proposed, approved, and effected.

Recommendation 6. While the dropping and failing rates are typical, there is a scope of improvement by reducing these rates. A recommendation is to check if there is a correlation between the dropped or failed students with their grades in earlier courses in Computer Science. Possible interventions include mandatory tutoring for students and membership in programming clubs who do not do well at the beginning of the semester. TAs, peer leaders, and the Head TA identified the students at risk in the Fall of 2018 semester but not all the students who ended up with D and F took the advantage of the additional support the department provided. Counseling from the department about challenges of the struggling students is another possible direction.
→ We have not acted on this.

Recommendation 7. We acknowledge the quality of the TA training and recommend the TA organization with a head TA be continued. We only suggest to focus more of the TA training on time management so that they can do well both in teaching and in their own studies.
→ We renew our appreciation for TA / IA / PL training as their quality and readiness is very satisfactory.

CS2401 Elementary Data Structures, Fall 2020 CQI Report

Martine Ceberio (UTEP) & Daniel Shanker (Google)

Course Approach

The course was taught by two co-instructors as part of the Google Faculty In Residence program. Additionally, one Teaching Assistant, 2 Instructional Assistants (undergraduate TAs), and one Peer Leader were part of the instructional team.

This course (lecture and lab) was taught entirely online.

Lectures: Synchronous lectures on Teams. Mostly lecture mode with interaction through the chat (and oral questions allowed at any time, but in fact only few oral questions). Lectures were led by either Dr. Ceberio or Daniel, alternating by topic, with both instructors present at every lecture to provide additional commentary (in the chat mostly). Lecture homework was ZyBooks reading with occasional additional practice problems.

Labs: Synchronous lectures on Teams and Gather.Town. Each session consisted of an intro to the lab topic then individual or collaborative work + peer led activity once a week. Students were split out into small groups led by members of the lab team (Daniel and the TA/IAs/PL). Lab homework was whatever portion of the lab assignment not finished in lab (for most students this was usually the majority of the lab assignment).

Exams: Exams were administered on Blackboard as “take-home” exams, typically with several sections (~30 minutes each) to be completed any time over the course of a week. Questions were mostly multiple choice, involving many non-trivial tracing questions, intertwined with some short coding problems.

Lecture notes and recorded sessions: were all available on Teams.

Office hours: each member of the instructional team held regularly scheduled office hours weekly. We also addressed students questions via private message on Teams and committed to replying to each question within a business day (most of the time, we answered immediately or within a couple of hours).

Weekly workshops: Each week, the instructional team offered 2 to 3 workshops on topics best suited to support the students’ learning and success at that time in the semester.

Attendance / Participation / Communication: We expected students to attend lectures and labs sessions. We took attendance, but did not count attendance towards the final grade (since some – few – students had changing work hours and genuinely could not attend): instead, we used attendance to track who could not attend and be able to follow

up with each student missing lecture / lab and check whether they were sick / needed any help catching up / etc.

Assessment of Performance: Our grading focused on the demonstration of skills. Although we shared with students the usual percentage make of a final grade, we also informed them that, for instance, if they were to fail at a quiz or an exam question on a given topic, but later perform much better on that same topic, the previous quiz of exam question grade would be forgiven (replaced for the better performance). This, in our opinion, provided a safe environment for experimenting and failing as part of the learning process.

Observations and Instruments Mapping to Outcomes

Participation / DFW:
There were 51 students originally registered for the class at the time of the first day of classes: 10 were female students (19.2%).

- 11 students dropped (1 female student, 10 male students).
    - 1 male student dropped without attending
    - 1 female student dropped less than a week into the semester
    - 1 male student was dropped for failure to pay for the course
    - 4 students faced excruciating circumstances related to COVID
- 3 male students (5.8% of the original roster of students, 7.5% of the students who did not drop) are receiving an incomplete and should complete their coursework by the end of spring semester. They are excluded from the below statistics (pass/fail) since we cannot predict their pass/rate outcome yet.
- Out of the 37 students remaining on the roster until the end of the semester and who obtained a grade:
    - 4 faced struggle related to COVID (including getting sick) and stopped attending the course early in the semester but did not drop (we did not drop them as we were hoping they would come back and knew the lack of effort was not due to carelessness): none of these students passed the class (10% of the students who did not drop, 7.8% of the original roster of 51 students).
    - 2 students (both male) who attended the course until the end of the semester failed the course (one with a U, one with a D). The student who received a U struggled all semester with extended work hours due to COVID, specifically during our scheduled lectures and labs (5% of the students who did not drop, 3.9% of the original roster).
    - 31 students passed the class (77.5% of the students who did not drop, 60.8% of the original roster of students). Out of these:
        - 22 A's: (14 male/6 female students) – 55% or 43.1%
        - 7 B's: (5 male/2 female students) – 17.5% or 13.7%
        - 1 C: (1 female student) – 2.5% or 1.9%

Counting all students who dropped the course (11) and those who failed the course (4), the DFW rate for this course is: 35.3%.

A	B	C	D	F / U	Drop	I
22	7	1	1	6	11	3
43.1%	13.7%	2%	2%	11.7%	21.6%	6%

Assessment instruments:

Students were evaluated throughout the semester via quizzes (mostly online, using Socrative.com and then MS Teams), homework assignments, four mid-term exams, one comprehensive final preparation exam and a comprehensive final exam (only for those who had not yet fully demonstrated mastery of the topics covered in the class after the final prep exam). Students were also evaluated based on their participation in labs and successful completion of lab assignments (10 short assignments, 3 comprehensive ones).

The mapping of instruments to outcomes was as follows: (P stands for Primary, S for Secondary)

	1.1	2.1.1	2.1.2	2.2.1	2.2.2	2.2.3	2.3.1	2.3.2	2.4.1	2.4.2	2.5	3.1.1	3.1.2	3.1.3	3.1.4	3.1.5
Small Exam 1						S	P	S	P	P		P				
Small Exam 2				P	P			P	P	P	P					
Small Exam 3				S	S	S		P	S	S	P		P			P
Small Exam 4													S	P	P	P
Final Exam prep	P	P		P	P	P		S	P	P	P	P	P	P	P	P
Final exam	P	P		P	P	P		S	P	P	P	P	P	P	P	P
MiniLab 0						S	S									
MiniLab 1			S									P				
MiniLab 2											P	S				
MiniLab 3		P	P	S			S	S				S				
MiniLab 4											P	S				
MiniLab 5				P	P				S							
MiniLab 6			S						S	P						
MiniLab 7		S											P			
MiniLab 8						S	S				P					P
Tree Hackathon								P			P					P
MiniLab 9											P					P
MiniLab 10		S						P		S				P	P	
Lab 1		S						P			P	P				
Lab 2									S	S			P			
Lab 3								P				S		S	P	
Homework	HW2	HW2	HW2	HW5	HW5	HW0	HW0	HW3	HW4	HW4	HW4	HW1	HW6	HW8	HW8	HW7
Quiz week 1												P				
Quiz week 2												P				
Quiz week 3	P	P	P													
Quiz week 4												P				
Quiz week 5				P	P											
Quiz week 6	P				P											
Quiz week 7											P					
Quiz week 8												P				
Quiz week 9																P
Quiz week 10																P
Quiz week 11																P
Quiz week 12																P
	5	8	5	8	8	7	5	11	9	9	12	14	7	6	6	12

So we observe that we covered all outcomes in assessment tools (labs, quizzes, homework, exams). The one we covered the least is Outcome 1.1. The one we covered most was Outcome 3.1.1 on Multi-dimensional Arrays. It makes sense to have covered it most since it is one of the earliest topics covered in this class.

The success for each outcomes was as follows:

	LABS	OTHER ASSESSMENTS
1.1	N/A	81.84%
2.1.1	86.18%	86.69%
2.1.2	97.97%	81.84%
2.2.1	86.31%	70%
2.2.2	80.97%	70%
2.2.3	87.16%	71.50%
2.3.1	84.43%	76.54%
2.3.2	73.55%	72.97%
2.4.1	83.12%	78.83%
2.4.2	85.14%	78.83%
2.5	81.82%	78.62%
3.1.1	89.72%	73.49%
3.1.2	81.47%	75.52%
3.1.3	82.57%	79.10%
3.1.4	82.57%	79.10%
3.1.5	71.17%	92.66%

Overall, we met all outcomes, but we can see that usually, except for outcome 3.1.5, the success in labs outdid the success in other assessment instruments. An explanation for the lower success of 3.1.5 in lab is that one major instrument to assess students on 3.1.5 was the tree hackathon we conducted in lab: the grading was either integers between 1 and 6 out of 6, hence leading to lower grades quite easily as many students obtained 3 or 4 out of 6.

Analysis by Instructors

Attendance and dedication: Overall, despite the current situation, most students attended class regularly. Some students did struggle with changing working hours or having to attend to sick family members. Some students even became sick.

What do we think about the DFW? Why? The DFW rate of 29.4% is high in comparison to previous semesters. However, 2/3 of the Drops were either at the onset of the semester, due to finances, or due to COVID, and among those who got an F, most had stopped attending becaused of COVID as well. Among those who were still attending the class at the end of the semester (31 students), only one failed the class.

Were outcomes met? All outcomes were met. For some of the outcomes, the percentage of attainment is in the lower 70's: this can be explained by the fact that the grading scale for some of the assignments to assess these outcomes made it easy to lose a lot of points (which brings the average down). Yet the performance of students on such assignments was deemed satisfactory for a large majority of students.

Online textbook: The online setting, with all lecture notes available, made us more accutely aware of the fact that students do not really access or use our notes. This was unfortunate as we knew these notes were more detailed than the textbook we used. We started wondering about the merit of this book in the success of our students. **We**

suggest not continuing with the online textbook and asking students to read our material instead and the extra time for practice on sites like codingbat or other tailored activities.

Grading: See our approach to assessing our students' success earlier in this document. **We suggest** that this approach be more widely considered in the fundamentals at least.

Sequence of topics: Based on our assessment and recommendations about this course, **we suggest** that the CS2 instructors discuss best order practices.

Recommendations

Assignment Structure

In this section, the instructors attempted to create lab assignments that were more applied in nature, some resembling end-to-end projects. Top-performing students reported that these assignments were fun and challenging, but the consistent cadence -- initially every session, but then reduced to weekly -- of complex lab assignments proved to be too much for students who struggled with more fundamental computer science concepts, and they were never really able to catch up. Longer labs throughout the semester should focus on more complex, more applied problems, but some of the most successful smaller labs were:

- A partially guided introductory lab assignment -- at Google we would call this a CodeLab, but it is essentially a step-by-step walkthrough of how to complete the assignment, with portions left for the student to complete along the way (but always with a detailed description of what the code's output should look like so a student can be confident that they are on track).
- An in class Binary Tree "hackathon" where students worked in groups to solve recursive tree problems. Students worked in groups of 4 or 5 to solve a slate of problems -- as soon as they finished one, they checked their work with an instructor and were given the next problem in the set. All but 2 groups were able to complete at least 4 problems, which was lower than expected, but still satisfactory. Even students who performed poorly on most labs were contributing to group problem solving.

Textbook

We are not convinced of the efficacy of the ZyBooks homework assignments. The material of the book is known to be shallow and mostly introductory to the topics covered more in depth in class. Although we find it useful for tracking students' work habit and potentially catching poor practice early on, an actual analysis of its effectiveness is needed, as we worry that any correlation between ZyBooks performance and course performance are simply correlation and not causation. The risk of it is that it is busy work and time that could be otherwise spent on more meaningful assignments. The greater risk is that it may create the false impression in a student that they fully understand the material.

CS2 Preparedness

Some students entering CS2 had trouble writing basic code that compiled, which is expected to be a solid skill coming out of CS1. Not discussing potential modifications to assessment of readiness post CS1 or modifications to CS1 itself, CS2 needs to swiftly identify these students as early as possible and work with them to get them up to speed. CS2 moves quickly and focuses on genuinely complex data structures, and without a solid understanding of the fundamentals, students will struggle for the entire semester. The majority of students told to practice on their own on Coding Bat or using similar resources simply will not, so any catch-up learning should be a mandatory part of class for struggling students.

Among students more comfortable with coding, the weakest spots from CS1 material were methods (mainly method parameters) and objects (constructors, how to call the methods of an object instance, the difference between an instance and the class itself, etc.). Not even considering more advanced applications of those topics, such as recursion or inheritance, students struggled to use method parameters properly or create and use an instance of an object. These topics either need to be assessed more thoroughly in CS1 or taught more explicitly in CS2. One other potential topic in this category would be any non-traditional for-loop -- that is, any for-loop that did not go from i=0 to a.length, such as for-loops that decrement instead of increment, for loops that start at a number other than 0, or for-loops that increment by a number other than 1.

Subject Matter

Java vs Algorithms. CS2 takes the stance that the course does not teach Java, but instead teaches computer science. [Daniel] While I think the primary goal should be to teach computer science through data structures and algorithmic problem solving, the two do not have to be mutually exclusive, and putting more emphasis on Java - examples given in the context of compilable Java code, more emphasis on coding in the lecture - may reduce the burden on students as they learn genuinely complex data structures and algorithms. Java is just a tool to be used to learn data structures and algorithms, but students need to learn how to properly use that tool.

[Daniel] The CS2 curriculum does not emphasize Java's built-in data structures - LinkedLists, Stacks, Queues, etc. from the java.util library. This skill is necessary for success on coding interviews, and while this class is by no means meant to funnel students directly into software engineering positions, the reality is that many students are interested in software engineering internships. Most students who interview are surprised that they do not feel prepared for the types of questions asked. Some lab assignments should focus on using the Java-implemented data structures, as is already being done in some CS3 sections, to continue to build general technical problem solving skills.

Time complexity. Students were able to memorize the Big O runtimes of the searching and sorting algorithms, but could not explain them. Although students are going through

the tracing and discussion of time complexity in class, it is important to put more emphasis on tracing code to determine the runtime.

Unit Testing. Students were asked to write their own unit tests for nearly every assignment, but many students wrote incomplete test suites or did not write tests at all, leading them to the false impression that they had completed the assignment correctly -- some students consistently turned in code that would not even compile. Assignments that had instructor-written unit tests exposed to students had higher quality lab assignments submissions and students worked more independently, as the tests served as guideposts during development. In addition, they reinforced good coding practices like (explicitly) iterative development and (implicitly) test-driven development. The recommendation would be to begin the class by exposing unit tests to students. Then, depending on how students are performing, spend some time in lab discussing and working on good unit test practices, and only then require students to write their own unit tests.

Reflection on Recommendations from Fall 2018

1. We should try to accommodate struggling students in smaller groups and see if we can increase their retention and success this way.
   → This is made easier online. However, student engagement is hard to guarantee.
2. We should keep using an online textbook with reading and homework tracking.
   → We kept using an online textbook since Fall 2018. However, (see above recommendations), it is not clear whether it is useful in the student learning or simply busy work.
3. We should adopt a way of assessing students based on skill acquisition since it is more attuned to our students' life circumstances (heavy work load, children, family commitments, etc.).
   → In Dr. Ceberio's sections, this has been done and, based on interactions with students, has often made the difference between a student dropping and staying in the class.
4. We should reconsider the sequence in which topics are introduced in this course to make it flow better for students.
   → The coverage of Trees and Stacks/Queues has been swapped.

CS 2401 Elementary Data Structures and Algorithms
Course Review
Fall 2020
Mahmud Shahriar Hossain

Course Description

CS 2401 focuses on basic data structures and fundamental algorithms. The course is the second fundamental course for students majoring in Computer Science. Students learn about basic algorithms, including searching and sorting; elementary abstract data types including linked lists, stacks, queues and trees; and basic algorithm analysis. Students complete a series of programming assignments using Java programming language to practice and enhance their understanding about the concepts taught in the lecture sessions. Laboratory assignments are designed to develop skills in problem solving, implementation, and debugging.

Prerequisite: CS 1301 and CS 1101, or equivalents of CS 1301 and CS 1101. Students must secure C or better in the prerequisites to continue with CS 2401.

Text

Zybook

Learning outcomes are provided in the combined summary page for three sections.

Instruments
The primary instruments of evaluating student performance in the course were four midterm exams, one comprehensive prefinal exam, and a comprehensive final exam. A small portion of the grades was based on in-class quizzes and lab attendance. There were also lab assignments, which were not used as instruments of this evaluation given that students received ample time to solve the assignment problems and they tend to perform better in lab assignments compared to exams and quizzes.

All exams were given using Blackboard. The outcome were set for the exam and quiz questions.

Outcome	Level	Result
3.1a Design and implement solutions to computational problems using the following data structure: multi-dimensional arrays;	3	Satisfactory (Avg. score: 88.3%)
3.1b Design and implement solutions to computational problems using the following data structure: list implementation of arrays or linked lists;	3	Satisfactory (Avg. score: 77.5%)
3.1c Design and implement solutions to computational problems using the following data structure: stacks;	3	Satisfactory (Avg. score: 75.5%)

Outcome	Level	Result
3.1d Design and implement solutions to computational problems using the following data structure: queues;	3	Satisfactory (Avg. score: 83.3%)
3.1e Design and implement solutions to computational problems using the following data structure: binary trees and binary search trees;	3	Satisfactory (Avg. score: 80.2%)
2.1a Describe, implement, and use the following concepts: classes, subclasses, and inheritance	2	Satisfactory (Avg. score: 81.5%)
2.1 b Describe, implement, and use the following concepts: encapsulation and information hiding	2	Satisfactory (Avg. score: 73.5%)
2.2a Describe, implement, and use the following algorithms: sequential and binary search	2	Satisfactory (Avg. score: 87.6%)
2.2b Describe, implement, and use the following algorithms: quadratic and O(n log n) sorting	2	Satisfactory (Avg. score: 94.1%)
2.2c Describe, implement, and use the following algorithms: string manipulation and parsing	2	Satisfactory (Avg. score: 85%)
2.3a Describe and trace computer representation and memory allocation of: integers, real numbers, arrays and objects	2	Satisfactory (Avg. score: 91.8%)
2.3b Describe and trace computer representation and memory allocation of: methods, including recursive methods and the use of activation records	2	Satisfactory (Avg. score: 74.1%)
2.4a Use basic notions of algorithm complexity: use Big-O notation to express the best-, average- and worst-case behaviors of an algorithm	2	Not Satisfactory (Avg. score: 64.2%)
2.4b Use basic notions of algorithm complexity: determine the best, average and worst-case behaviors of a simple algorithm	2	Satisfactory (Avg. score: 80.5%)
2.4c Use basic notions of algorithm complexity: assess time and space trade-offs in algorithms	2	Not Satisfactory (Avg. score: 64.6%)
2.5 Use recursion and iteration as problem solving techniques	2	Not Satisfactory (Avg. score: 67.7%)
1.1 Explain the concept of polymorphism	1	Satisfactory (Avg. score: 73.3%)

Further breakdown of questions are provided in appendix for “Not Satisfactory” outcomes.

Analysis by Instructor:

The table above presents the results of mapping educational outcomes to assessment. My observations and recommendations are as follows.

Observations:

1. All outcomes, except for outcomes 2.4a, 2.4c, and 2.5 are satisfied. Further breakdown for outcomes 2.4a, 2.4c, and 2.5 in the Appendix provides an idea about what types of questions students struggle in. Analyzing complexity given a formula or code and writing recursive methods from scratch are area of struggle to most students. This observation matches with my past experience in teaching this course many times.
2. Number of students at the end of the semester: 44.
   Number of students in the beginning of the semester: 46.
   The grade distribution was as follows:
   A: 22 (22/44=50%)
   B: 11 (11/44=25%)
   C: 6 (6/44=13.6%)
   D: 1 (1/44=0.02%)
   F: 4 (4/44=0.09).
   The passing rate (C or better grades) is 39/44 = 88.63%.
3. Three of the four students who received F stopped attending the class from the middle of the semester. They did not respond to emails from the instructional team and the department front office.
4. The instructional team noticed a drastic surge in academic dishonesty. Ten cases were filed to UTEP OSCCR from this course. The surge in academic dishonesty might be resultant from the fact that many of these students never had physical classroom experience, where they can directly communicate with the instructor, TA's, IA's and PL's better. The connection with the instructional team in a physical classroom environment is stronger than the connection built in classes delivered online.

Recommendations:

1. Tutoring on recursion and complexity analysis might help improve student performance.
2. Some students stopped responding. A mechanism from the department to communicate with these students might help.
3. A combined effort from the department is required to increase ethical standards in terms of academic environment from the beginning when students enter the Computer Science bachelors program. After the pandemic is over, it is essential to welcome these students in the department so that they feel it is their home and not just a place from where they will get their degree by hook or by crook.

Appendix:

Instruments for Outcome 2.4 A:

Content List

Course Name	Elem. Data Struct./Algorithms (Lecture) (CS-2401-009 CRN:18737)
Goal	UTEP (US): CO-CS-2401-2.4.A: Students will be able to use basic notions of algorithm complexity. In particul...
Content Type	Questions
Performance Target	70.0%
Performance Range	5.0%

Report Options | Back

Questions

Performance Average Below 65.0%	Average	Median	Mode
The time complexity for the Towers of Honoi algorithm in the text is _____.	97.6%	100.0%	100.0%
The worst-time complexity for bubble sort is _____.	90.2%	100.0%	100.0%
The worst-time complexity for quick sort is _____.	95.1%	100.0%	100.0%
What are the best, average, and worst-case behaviors of linear search algorithm?	85.4%	100.0%	100.0%
What is the average time complexity of the following method in terms of the paramter n.	51.2%	100.0%	100.0%
What is the time complexity of the following method in terms of the input parameter n? Selec...	46.3%	0.0%	0.0%
What is the time complexity of the following method in terms of the input parameter n? Selec...	85.4%	100.0%	100.0%
What is the time complexity of the following method in terms of the input parameter n? Selec...	34.1%	0.0%	0.0%
What is the time complexity of the following method in terms of the input parameter n? Selec...	56.1%	100.0%	100.0%
What is the time complexity of the following method in terms of the input parameter n? Selec...	0.0%	0.0%	0.0%
What is the time complexity resulting from the following time function: T(n)=T(n-1)+c	73.2%	100.0%	100.0%
What is the time complexity resulting from the following time function: T(n)=T(n-2)+c	73.2%	100.0%	100.0%
You have two algorithms. After counting the instructions, you figure out that Algorithm...	46.3%	0.0%	0.0%

Instruments for Outcome 2.4 C:

Content List

Course Name	Elem. Data Struct./Algorithms (Lecture) (CS-2401-009 CRN:18737)
Goal	UTEP (US): CO-CS-2401-2.4.C: Students will be able to use basic notions of algorithm complexity. In particul...
Content Type	Questions
Performance Target	70.0%
Performance Range	5.0%

Report Options | Back

Questions

Performance Average Below 65.0%	Average	Median	Mode
Which of the following operations is performed more efficiently by a linked list than by an array?	82.9%	100.0%	100.0%
You have two algorithms. After counting the instructions, you figure out that Algorithm...	46.3%	0.0%	0.0%

Instruments for Outcome 2.5:

Content List

Course Name	Elem. Data Struct./Algorithms (Lecture) (CS-2401-009 CRN:18737)
Goal	UTEP (US): CO-CS-2401-2.5: Students will be able to use recursion and iteration as problem solving techniques.
Content Type	Questions
Performance Target	70.0%
Performance Range	5.0%

Report Options | Back

Questions

Performance Average Below 65.0%	Average	Median	Mode
Consider the following recursive method.What value is returned when invoking m(5)?	80.0%	100.0%	100.0%
You have the following class that represents a Rectangle. In another class, you have the f...	78.0%	100.0%	100.0%
You have the following method, inside which there is a loop. Rewrite the method using recurs...	28.3%	20.0%	10.0%
You have the following method: public static String mystery5(String str, int num) { ...	85.0%	100.0%	100.0%

CS 2401: Elementary Data Structures and Algorithms

Dan DeBlasio

Fall 2020

1 Course Approach

Being online the course consisted of 4 major interaction methods:

- **prerecorded content** — videos posted by the instructor inline with the current topic; these videos replace a typical in-person lecture
- **synchronous class meetings** — interactive sessions with the instructor and the students, supplementing the videos; these typically consisted of problem solving sessions either on a virtual whiteboard or using one of the recommended IDEs
- **synchronous laboratory meetings** — interactive sessions with the other instructional staff where the laboratory assignments are discussed; time is also dedicated to making progress on the assignment with others
- **online class "team"** — all students and staff were part of a university approved MS Team where students could ask questions or start discussions 24/7, the instructor and other staff could monitor and intervene; this is also where review sessions were conducted.

2 Assessment Instruments

The two primary sources of assessment came from exams (4 mid-term exams, and 2 finals) and hands-on labs (12 labs aligning to learning objectives). Smaller contributors not included in this report were: in-class quizzes, participation, as well as homework integrated into the online textbook. These smaller assessments were not graded for correctness.

2.1 Exams

All exams were graded by the instructor, feedback was given individually using Blackboard. Additionally, a postmortem discussion was provided both in class and via a pre-recorded video.

Delivery method Two delivery methods were used for exams this semester:

- **Asynchronous** — students were allowed to take the exam at any time in a prescribed amount of time (i.e. 2x 30-min sessions within a week).
- **Synchronous** — students were required to take the exam all at the same time (typically during a lab meeting period)

Initially all exams were planned to be asynchronous to allow students flexibility, but due to some student conduct issues, the latter two exams were delivered synchronously. While the average grades were similar between the two delivery methods (for those that took all 4 midterms), there seemed to be a pattern that students either did

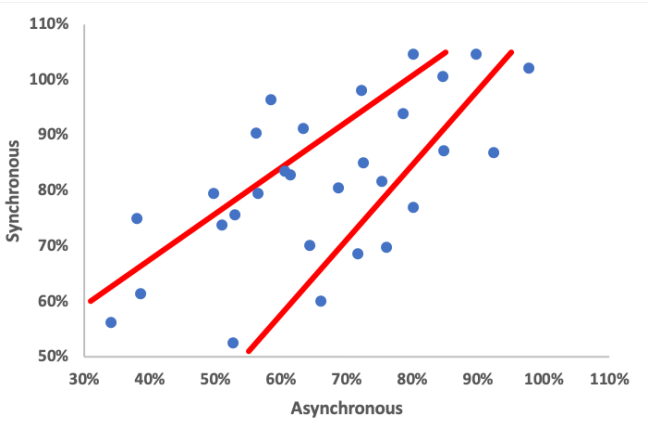

Figure 1: Grade analysis by exam delivery method.

better on one versus the other. Figure 1 shows the scores for all of the students who took all 4 midterms. Each point represents a single student. The two red lines are added manually for effect (i.e they are not computed regressions). Because of this deviation, two final exams were given, one in each delivery method and the higher of the two grades was given more weight (this weight though is not reflected in the right most column of Table 1).

2.2 Laboratories

Labs were assigned on Mondays and typically due the following Sunday evening. All but the last lab were only assigned for 1 week, the final lab was assigned for 2 weeks. Each lab was assigned and submitted using `Github`. Labs were graded by the TA and IAs for the course given a detailed rubric provided by the instructor.

3 Analysis and Observations

Initially 48 students registered for the course, 4 dropped before census day, 33 were enrolled at the time of the final exam. The final letter grades assigned are below:

A	8
B	4
C	11
D	6
F	3
W	11
I	1

This means the DF rate was 27% (assuming the student who was assigned an I passes), and the DFW rate was 45% (only counting those who made it to census day).

Of those that dropped after the census day only one provided a reason to the instructor, citing increased work commitments; two students dropped because no assignments were turned in and had not attended class; and one other withdrew from the entire semester. The other 8 have not responded to contact regarding this.

Of the students who received Fs: one student was assigned an F by OSCCR, one was diagnosed with COVID early in the semester (early October), and the third provided no information.

Table 1: Concepts on Exams

Objective	Midterm 1	Midterm2	Midterm 3	Midterm 4	Asynch. Final	Synch. Final	Total
1.1					4 (75)		75
2.1.1 (classes)	5 (73)	3 (34)	5 (61)		3 (55)	13, 14 (65)	57
2.1.1 (subclasses)							—
2.1.1 (inheritance)					3 (55)		55
2.1.2 (encapsulation)	5 (73)				3 (55)		63
2.1.2 (hiding)	5 (73)					13 (65)	69
2.2.1 (seq.)	1 (68)	3, 8* (37)				12, 14 (69)	56
2.2.1 (binary)					6 (50)	6B (60)	56
2.2.2 (n^2)		4, 5, 6 (53)			8 (60)	10 (97)	61
2.2.2 ($n \log n$)		4, 7 (47)			8 (60)	10 (97)	59
2.2.3 (manipulation)	3 (61)	3 (34)				2, 12 (65)	52
2.2.3 (parsing)	3, 4, 6 (64)					2 (55)	63
2.3.1 (int)	2 (70)		2, 4 (79)		3 (55)	11, 12 (69)	70
2.3.1 (real)		8* (42)			1, 3, 6 (53)	6B, 14 (62)	54
2.3.1 (arrays)					5 (26)	12 (76)	39
2.3.1 (objects)	6 (49)				3 (55)		53
2.3.2 (methods)		8* (42)	5 (61)		1 (54)	2 (55)	51
2.3.2 (recursion)		8* (42)	5 (61)		1 (54)	2 (55)	51
2.4.1 (best)		5, 6, 7 (79)			7 (41)	7B, 10 (81)	75
2.4.1 (worst)			4 (62)		7 (41)	7B, 10 (81)	69
2.4.1 (average)							—
2.4.2 (best)					7 (41)	7B (66)	56
2.4.2 (worst)			4 (62)		7 (41)	7B (66)	58
2.4.2 (average)							—
2.5 (recursion)		8* (42)				11 (66)	54
2.5 (iteration)	1 (79)				5 , 9 (43)	14 (65)	55
3.1.1	1 (79)				9 (60)	12 (76)	70
3.1.2 (arrays)		8* (42)			5 (26)	12 (76)	40
3.1.2 (linked lists)		3 (34)	5 (61)			14 (65)	48
3.1.3				3, 6 (62)		15 (82)	67
3.1.4				3, 6 (62)		15 (82)	67
3.1.5 (binary tree)			1 (87)	1, 4 (76)			80
3.1.5 (BST)			2, 3 (86)	2, 5 (91)	5 (26)		70

Notes:
– Each cell in the middle 6 columns show the question numbers that address the objective, the numbers in parenthesis are the scores as a percentage (%).
– The column "total" is weighted total score of the questions for that objective.
– Midterm 2, Question 8 – Questions 1 and 2 were not counted due to errors, Q8 was given as a follow up quiz and counted in the test grade.
– Both final exams share a numbering scheme, questions that were slightly modified and reused are labeled as "B".

Table 2: Concepts on Labs

Objective	Lab 2	3	4	5	6	7	8	9	10	11	12	14/15	Overall Score
Avg. Score	66.2	58.2	84.6	70.0	76.2	67.7	69.1	91.1	68.5	72.6	66.0	122.4	
Weight	100	100	150	100	100	100	100	160	100	100	125	190	64%
1.1						✓							68%
2.1.1 (classes)			✓				✓		✓		✓		61%
2.1.1 (subclasses)													—
2.1.1 (inheritance)													—
2.1.2 (encapsulation)						✓							68%
2.1.2 (hiding)			✓										56%
2.2.1 (seq.)	✓	✓	✓						✓				62%
2.2.1 (binary)							✓				✓		60%
2.2.2 (n^2)					✓								76%
2.2.2 ($n \log n$)													—
2.2.3 (manipulation)				✓				✓				✓	63%
2.2.3 (parsing)				✓	✓	✓		✓				✓	66%
2.3.1 (int)	✓	✓									✓		59%
2.3.1 (real)													—
2.3.1 (arrays)	✓	✓										✓	63%
2.3.1 (objects)			✓				✓			✓			65%
2.3.2 (methods)	✓	✓	✓	✓	✓	✓				✓		✓	66%
2.3.2 (recursion)				✓	✓	✓		✓	✓			✓	66%
2.4.1 (best)													—
2.4.1 (worst)													—
2.4.1 (average)													—
2.4.2 (best)													—
2.4.2 (worst)													—
2.4.2 (average)													—
2.5 (recursion)				✓	✓	✓		✓	✓				67%
2.5 (iteration)	✓	✓	✓										60%
3.1.1	✓	✓	✓										60%
3.1.2 (arrays)	✓	✓	✓	✓	✓							✓	65%
3.1.2 (linked lists)						✓	✓	✓					63%
3.1.3												✓	64%
3.1.4												✓	64%
3.1.5 (binary tree)									✓				68%
3.1.5 (BST)										✓	✓		62%

Notes: Lab 1 did not align to any objectives

3.1 Observations

1. While average-case time analysis is discussed in class in passing, it not assessed (rows highlighted in Table 1 in blue). Due to the more detailed outcome breakdown I have made, this becomes more obvious.
2. Similar to the previous, subclasses are discussed in class (rows highlighted in Table 1 in green). This idea was also included in an in-class activity but was not assessed.
3. Unlike other sections this semester my section had a peak in both As and Cs, other sections saw a mono-modal distribution on As.
4. In general, though the scores on individual question was low, those concepts that were tested were demonstrated correctly by most students. Some exceptions remain such as Objectives 2.3.1 (description and tracing of arrays), 3.1.2 (use of arrays), and 3.1.2 (use of linked lists) where less than 50% of points were awarded across all exams (rows highlighted in Table 1 in red).

3.2 Recommendations

Recommendations related to the observations above:

1. The actual practice of average-case analysis is not as strait forward as best- and worst-case, and given the issues present with those concepts, I would recommend that average-case be moved to a level 1 outcome.
2. The use of subclasses is an oversight on the side of the instructor, and should be included. That said, 2401 is not an object oriented programming course, so the inclusion of the entire objective 2.1 is only a consequence of using Java, my preference would be use use a non-objective language.
3. Being online presented major challenges in reaching out to students who might be struggling at one time or another. The communication could be improved. Some students mentioned in their comments that it took some time to adjust to my methodology for communication of ideas and question answering, so this could be improved.

Other recommendations I would make for myself in future semesters:

1. In general, exam questions in my section tend to be overly complicated (in particular questions highlighted in Table 1 in yellow). It may be better to provide questions for each topic at various levels to get a better sense of understanding.
2. We did not discuss the memory allocation of various (the second part of Outcome 2.3). This seems beyond the scope of this course and additionally could be moved to Level 1.
3. Improved class-specific instructional staff training, and better communicate my expectations for the feedback they provide students. Similarly clarify the grading workflow.
4. It would be good to consult the learning outcomes more frequently to ensure all of the topics are being covered. Many times I find myself going back at the end of the semester to fill in points I may have missed.

3.3 Non-actionable observations

- The idea of having the asynchronous videos available to students was positively received, as was the use of class time for problem solving.
- Unlike in-person classes, the correlation between attending and success in the course was not strong. My assumption is that some students connected but didn't pay attention. While there is a trend (as shown in Figure 2) many students who attended all sessions still got low grades.

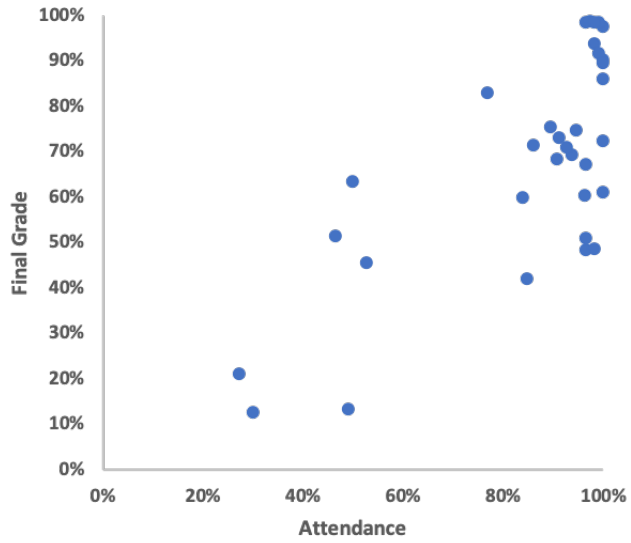

Figure 2: Relationship between percentage of sessions attended and final grade.

- Having taught this course twice online, once fully online and the other partially, I think I am more prepared for in-person delivery than I was previously. Planning this course online seemed to make it so more preparation and a better understanding of the whole course was needed.
- I think the current analysis of the course outcomes will be helpful in addressing some of the issues above.

Fall 2020 CQI Report
CS 2302 Data Structures

Diego Aguirre and Olac Fuentes

1. Course Description and Outcomes

1.1 Course Description

Data Structures is the third and final course in the fundamental computer science sequence. Students will learn about fundamental data structures and analysis and design of algorithms.

Knowledge and Abilities Required Before the Students Enter the Course:
Students entering this course are expected to 1) possess fundamental problem-solving skills; 2) implement solutions to computing problems in a high-level programming language; and 3) know about elementary data structures (lists, stacks, and queues). Students must also be familiar with topics from Discrete Math.

Prerequisites: MATH 2300 (Discrete Math) and CS 2401 (Elementary Data Structures and Algorithms)

Textbook:
Online: Data Structures Essentials (Python) - zyBooks

1.2 Course Outcomes

Level 3: Synthesis and evaluation:
Level 3 outcomes are those in which the student can apply the material in new situations. This is the highest level of mastery. On successful completion of this course students will be able to:

1. Given a problem, judge which data structures are required to solve it efficiently and justify the selection.
2. Given a non-recursive algorithm examine its loop structure, assess its asymptotic running time, and express it using big-O notation.
3. Given a recursive algorithm, examine its structure, formulate and solve a recurrence equation defining its running time, and express it using big-O notation.
4. Design and implement solutions to computational problems based on iteration and recursion.
5. Trace the behavior of non-trivial methods and algorithms.

Level 2: Application and analysis:
Level 2 outcomes are those in which the student can apply the material in familiar situations, e.g., can work a problem of familiar structure with minor changes in the details.
Upon successful completion of this course, students will be able to:

1. Describe, implement, and use the following data structures:
    a) Heaps
    b) Hash tables
    c) Balanced trees
    d) Graphs
    e) Disjoint set forests

2. Describe, implement, and apply the following graph algorithms:
    a) Connected components
    b) Breadth-first search

c) Depth-first search
d) Topological sorting
e) Minimum spanning trees (Kruskal's and Primm's)
f) Single-source shortest paths (Dijkstra's algorithm)

3. Trace the behavior of recursive programs using activation records.
4. Reason about the running times of algorithms in relation to the size of their inputs.

Level 1: Knowledge and comprehension:
Level 1 outcomes are those in which the student has been exposed to the terms and concepts at a basic level and can supply basic definitions. On successful completion of this course, students will be able to:

1. Identify and explain the following algorithm design techniques:
   a) Greedy algorithms
   b) Divide and conquer
   c) Dynamic programming
   d) Backtracking

2. Explain the concept of NP completeness.
3. Explain the utility of randomized algorithms.

2. Summary of Observations and Recommendations

2.1 Grade distribution

A	B	C	D	F/U	Drop	I
Aguirre, Section 1 (MW 10:30am – 11:50am)						
15	**8**	**10**	**3**	**3**	**12**	**0**
Aguirre, Section 2 (TR 10:30am – 11:50am)						
10	**3**	**2**	**1**	**2**	**7**	**0**
Fuentes, Section 3 (TR 10:30am – 11:50am)						
9	**6**	**3**	**1**	**5**	**16**	**0**
Total						
34	**17**	**15**	**5**	**10**	**35**	**0**
29.3%	**14.7%**	**12.9%**	**4.3%**	**8.6%**	**30.2%**	**0%**

2.2 Observations

- The number of students that dropped the class (combining the three sections) was 35, which represents a 30% of the students that originally registered for the class. This is an unprecedented event. Since this occurred in other fundamental courses as well, the global pandemic and the wide variety of diverse problems that it brings seem to be cause.
- Of the students who finished the course, 81.5% obtained a passing grade, which is slightly higher than in previous semesters.
- Four of the five level three outcomes were satisfied. Outcomes 3.4 and 3.5 are considered the most important in terms of helping students develop problem-solving skills and thus were evaluated multiple times, often in combination with the more fact-based material covered in class. Outcome 3.3, related to analysis of recursive algorithms was not satisfied. Students had difficulty understanding the mathematical tools required, recurrence relations. Given the circumstances, the instructor decided not to spend extra time trying to address this issue, instead focusing on problem solving.

- Level two outcomes were the most problematic. Understanding of specific algorithms is usually tested by having student trace them for particular inputs. As the exams were administered remotely, testing by means of tracing became unreliable, as it was possible for students to simply run the code provided instead of tracing.
- All level one outcomes were satisfied, with the exception of explaining NP completeness, which was not tested.
- The use of Google Colab helped students get started with Python more quickly. They did not need to install any software and could use any device with browsing capabilities to complete assignments.
- Automated grading, either through Google Colab or by special-purpose programs, was very helpful, especially considering the large number of students. It allowed feedback to be given instantly for many problems and freed our TAs and IAs to provide more specialized feedback where it was needed most.
- Even though some students were concerned with the transition from Java to Python, it did not represent a problem, even though the class was taught online. An *Intro to Python* assignment was given at the beginning of class which allowed students to get more comfortable with the language early in the class.
- The use of pre-recorded lectures was successful in general.

2.3 Recommendations

- It is very difficult to cover all topics presented in the outcomes. The committee should find, in consultation with the rest of the faculty, one or more non-essential outcomes to eliminate in order to allow more flexibility to review and provide additional practice in the topics that students find most challenging.
- It has been observed that most students who pass CS2401 with a C do not pass CS2302. CS2401 instructors should consider whether borderline cases would benefit more from repeating CS2401 than from taking, and likely failing, CS2302.
- In-class work was emphasized, and it resulted on improved learning for many students. However, it seemed to widen the gap between the stronger students and those who were struggling, as the latter tended to disengage from activities and miss class more often. The return to face to face interaction should help reduce this problem, but instructors should work on finding ways of keeping unprepared students engaged.
- Many students had difficulty tracing code, particularly recursive functions, even though a lot of class time was spent on it. This may be a consequence of the fact that in online learning students are always in front of the computer and develop a tendency to program by trial and error. Emphasizing paper-only exercises should help with this.
- The TA, PL, and IAs were key to overcome the problems imposed by on-line learning and increased class sizes. We should continue to support the programs that fund them.
- Students actually read the zyBook, we should continue using it.

3. Recommendations from previous reports and actions taken

In general, remote instruction presented an additional set of challenges that made addressing previous recommendations difficult and, in some cases, unfeasible.

Recommendation: Outcome 2.1.c was not satisfied. For the past few years, we have used B-trees as the instance of balanced search tree. We recommend considering switching to a simpler data structure, perhaps AVL trees.

Action taken: We provided an extensive review of binary search trees, which are only superficially covered in CS2401, to help students prepare for the topic. We also spent more time on writing and tracing of recursive functions. Unfortunately, results were still unsatisfactory. We will consider switching to 2-3 or AVL trees.

Recommendation: Outcome 2.1.e was not satisfied. Either more time should be spent on this topic, or perhaps only the basic version of disjoint set forests should be presented, omitting path compression, union by height, and union by size.

Action taken: The recommendation was implemented, but, unfortunately, little time was available to cover disjoint set forests, even in the simplified version.

Recommendation: Outcome 2.4 was not satisfied. A large amount of time was spent working on this topic with unsatisfactory results. A large fraction of the students have very weak mathematical skills, including elementary algebra, and understanding of exponents, and logarithms. Many also do not understand scientific notation and the use of time units. We should consider whether going over these topics is a good use of the time in a Data Structures course.

Action taken: We focused on more informal reasoning, relying less on mathematical background and focusing more on developing an intuitive understanding of the topic. Students were more successful under this approach.

Recommendation: The course covers a large number of topics. It would be desirable to find one or more non-essential topics to eliminate in order to allow more flexibility to review and provide additional practice in the topics that students find difficult.

Action taken: In general, less time was spent on topics considered non-essential, in order to focus on problem solving skills – understanding solutions given and formulating novel solutions. However, this, together with the limitations imposed by online teaching, resulted on several outcomes not being satisfied. The course outcomes were not modified.

4. Reports by Instructor

CS 2302 Data Structures, Fall 2020 CQI Report

Diego Aguirre – Sections 1 and 2

Students were evaluated throughout the semester via zyBook assignments, quizzes, homework assignments, three partial exams and a comprehensive final examination. In addition to this, students were assigned 7 labs where they had to apply the concepts learned in class.

Details about the class

- All lectures were pre-recorded. Live sessions were used to practice and build a sense of community.

  To make the class more accessible, all lectures were pre-recorded. Students were asked to watch the lectures before joining our synchronous sessions. When joining a synchronous session, students were given a quick ice-breaker question to socialize for a small amount of time, and wait for others to join. After the social activity, students were given time to ask question about the lecture they watched before joining. Once all questions were answered, students were given a quiz where they had to apply the concepts learned in the lecture. Students were placed in teams, and the instructor and TAs/IAs/PL moved between teams to answer questions and make sure everyone made progress.

- Quizzes were graded differently than in previous semesters. Students were given full-credit just for attempting to solve all problems.

  The idea behind this decision was to shift the students' focus from "getting the right answer" to valuing practice and the development of better problem-solving strategies. Google Colab was used to write and submit quizzes.

- Exams were graded automatically using unit tests.

  All exams were administered through Google Colab. All exams had a combination of multiple-choice, fill in the blank, and coding problems. All coding problems came with a set of unit tests to assess students' answers in real time. Students were encouraged to run the provided unit tests as many times as they saw fit during the exam, and fix their code if necessary. The code was not trivial, so students really needed to understand the material to get a good grade.

  All three partial exam grades could be made up when taking the final. Students were told since the first day that the final exam would be divided in three sections, and that each of these sections could serve as a make up for the section's corresponding partial exam.

- All reading material came from zyBooks. The pre-recorded lectures covered the same content.

Learning outcomes mappings

The following table shows the mapping between learning outcomes and exam questions.

LO	Exam 1	Exam 2	Exam 3	Final Exam
L1.1.a			P13	P38
L1.1.b			P10	P38
L1.1.c			P5, P12	P38
L1.1.d			P13	P38
L1.2			P14	P37

L1.3			P11	P38
L2.1.a		P3, P5, P6, P19, P20, P21, P22		P19, P21, P29, P30
L2.1.b		P4, P15, P16, P17, P18		P19, P20, P27, P28
L2.1.c		P1, P2, P7, P8, P9, P10, P11, P12, P13, P14		P17, P18, P19, P23, P24, P25, P26
L2.1.d			P15, P16, P17, P18, P19, P20, P21, P22	P19, P39, P40, P41, P42
L2.1.e			P23, P24	P19, P22, P43, P44
L2.2.a				
L2.2.b			P2, P8	P32, P36
L2.2.c			P3, P9	P32, P36
L2.2.d			P1	P31
L2.2.e			P6, P7	P35
L2.2.f			P4	P33
L2.3	P12, P13, P14, P17			P11, P12, P13
L2.4	P1, P2, P3, P4, P5, P6, P7, P8, P9, P10, P11			P1, P2, P3, P4, P5, P6, P7, P8, P9, P10
L3.1	P17		P8, P9	P36
L3.2	P1, P2, P3, P4			P1, P2, P3, P4
L3.3	P5, P6, P7, P8, P9, P10, P11			P5, P6, P7, P8, P9, P10
L3.4	P15, P16, P17	P7, P8, P9, P10, P11, P12, P13, P14, P15, P16, P17, P18, P19, P20, P21, P22	P15, P16, P17, P18, P19, P20, P21, P22, P23, P24	P14, P15, P16, P23, P24, P25, P26, P27, P28, P29, P30, P39, P40, P41, P42, P43, P44
L3.5	P12, P13, P14, P15, P16	P7, P8, P9, P10, P11, P12, P13, P14, P15, P16, P17, P18, P19, P20, P21, P22	P15, P16, P17, P18, P19, P20, P21, P22, P23, P24	P11, P12, P13, P14, P15, P16, P23, P24, P25, P26, P27, P28, P29, P30, P39, P40, P41, P42, P43, P44

The following table shows the mapping between learning outcomes and lab assignments.

LO	**Lab 1**	**Lab 2**	**Lab 3**	**Lab 4**	**Lab 5**	**Lab 6**	**Lab 7**
L1.1.a						X	
L1.1.b				X			
L1.1.c							X
L1.1.d	X						
L1.2							
L1.3							
L2.1.a							
L2.1.b					X		
L2.1.c				X			
L2.1.d						X	

L2.1.e						X	
L2.2.a						X	
L2.2.b						X	
L2.2.c						X	
L2.2.d							
L2.2.e						X	
L2.2.f							
L2.3	X						
L2.4	X	X	X	X	X	X	X
L3.1	X	X	X	X	X	X	X
L3.2							
L3.3							
L3.4	X	X	X	X	X	X	X
L3.5	X	X	X	X	X	X	X

The following tables show the mapping between learning outcomes and quiz problems.

LO	**Quiz 1**	**Quiz 2**	**Quiz 3**	**Quiz 4**	**Quiz 5**	**Quiz 6**	**Quiz 7**
L1.1.a							
L1.1.b					P1, P2, P3, P4, P5	P1, P2, P3, P4, P5	
L1.1.c							
L1.1.d	P1, P2	P2					
L1.2							
L1.3							
L2.1.a							
L2.1.b							P1, P2, P3, P4, P5, P6, P7
L2.1.c					P1, P2, P3, P4, P5	P1, P2, P3, P4, P5	
L2.1.d							
L2.1.e							
L2.2.a							
L2.2.b							
L2.2.c							
L2.2.d							
L2.2.e							
L2.2.f							
L2.3		P1		P8			

L2.4			P1, P2, P3, P4, P5, P6, P7, P8, P9, P10	P1, P2, P3, P4, P5, P6	P1, P2, P3, P4, P5		
L3.1				P8			
L3.2			P1, P2, P3, P4, P5, P6, P7, P8, P9, P10				
L3.3				P1, P2, P3, P4, P5, P6			
L3.4	P1, P2	P2, P3		P7, P8	P1, P2, P3, P4, P5	P1, P2, P3, P4, P5	P2, P3, P4, P5, P6, P7
L3.5	P1, P2	P2, P3		P7, P8	P1, P2, P3, P4, P5	P1, P2, P3, P4, P5	P2, P3, P4, P5, P6, P7

LO	**Quiz 8**	**Quiz 9**	**Quiz 10**	**Quiz 11**	**Quiz 12**	**Quiz 13**	**Quiz 14**
L1.1.a				P1	P1		
L1.1.b							
L1.1.c						P3	P1, P2
L1.1.d						P1, P2	
L1.2							
L1.3							
L2.1.a	P1, P2, P3, P4, P5, P6, P7, P8, P9, P10, P11, P12						
L2.1.b							
L2.1.c							
L2.1.d			P1, P2, P3, P4, P5, P6, P7, P8, P9, P10, P11, P12	P1, P2	P1, P2		
L2.1.e		P1, P2, P3, P4, P5, P6, P7, P8, P9					
L2.2.a			P11				
L2.2.b				P2			
L2.2.c				P2			
L2.2.d					P2		

L2.2.e				P1			
L2.2.f					P1		
L2.3						P2	
L2.4							
L3.1							P1, P2
L3.2							
L3.3							
L3.4	P6, P7, P8, P9, P10, P11, P12	P2, P4, P5, P6, P7, P8, P9	P3, P4, P5, P6, P7, P8, P9, P10, P11, P12			P1, P3	P1, P2
L3.5	P7, P8, P9, P10, P11, P12	P2, P4, P5, P6, P7, P8, P9	P3, P4, P5, P6, P7, P8, P9, P10, P11, P12			P1, P2, P3	P1, P2

Exam 1 statistics

Number of questions: 17
Number of students that took the exam (MW section): 44
Number of students that took the exam (MW section): 24
Average score per question (0 to 1)

Problem #	MW Section	TR Section	Section Average
1	0.72727273	0.83333333	0.78030303
2	0.86363636	0.875	0.86931818
3	0.65909091	0.875	0.76704545
4	0.45454545	0.83333333	0.64393939
5	0.98484848	0.91666667	0.95075758
6	0.96969697	0.88888889	0.92929293
7	0.96969697	0.95833333	0.96401515
8	0.81818182	0.625	0.72159091
9	0.90909091	0.95833333	0.93371212
10	0.97727273	0.875	0.92613636
11	0.77272727	0.79166667	0.78219697
12	0.93181818	1	0.96590909
13	0.77272727	0.95833333	0.8655303
14	0.81818182	0.83333333	0.82575758
15	0.38636364	0.33333333	0.35984848
16	0.20454545	0.375	0.28977273
17	0.81818182	0.79166667	0.80492424

Exam 2 statistics

Number of questions: 22
Number of students that took the exam (MW section): 41
Number of students that took the exam (TR section): 21
Average score per question (0 to 1)

Problem #	MW Section	TR Section	Section Average
1	0.95121951	1	0.97560976
2	0.65853659	0.75	0.70426829
3	0.92682927	0.8	0.86341463
4	0.92682927	0.85	0.88841463
5	0.90243902	0.9	0.90121951
6	0.68292683	0.7	0.69146341
7	0.82926829	0.75833333	0.79380081
8	0.70487805	0.62	0.66243902
9	0.70325203	0.75833333	0.73079268
10	0.59044715	0.4625	0.52647358
11	0.73170732	0.71666667	0.72418699
12	0.22439024	0.37	0.29719512
13	0.32195122	0.48	0.40097561
14	0.50304878	0.35625	0.42964939
15	0.92682927	0.85	0.88841463
16	0.69686411	0.71071429	0.7037892
17	0.21747967	0.7	0.45873984
18	0.67682927	0.675	0.67591463
19	0.24796748	0.23333333	0.24065041
20	0.87054409	0.80769231	0.8391182
21	0.32012195	0.43125	0.37568598
22	0.35033259	0.49090909	0.42062084

Exam 3 statistics

Number of questions: 24
Number of students that took the exam (MW section): 38
Number of students that took the exam (TR section): 17
Average score per question (0 to 1)

Problem #	MW Section	TR Section	Section Average
1	0.72368421	0.82352941	0.77360681
2	0.87280702	0.92156863	0.89718782
3	0.6754386	0.76470588	0.72007224
4	0.85526316	0.68823529	0.77174923
5	0.88980263	0.87132353	0.88056308

6	0.92105263	0.94117647	0.93111455
7	0.52631579	0.64705882	0.58668731
8	0.97368421	1	0.98684211
9	0.94736842	1	0.97368421
10	0.97368421	1	0.98684211
11	0.97368421	1	0.98684211
12	1	1	1
13	0.89473684	0.76470588	0.82972136
14	0.5	0.64705882	0.57352941
15	0.94736842	0.76470588	0.85603715
16	0.8	0.88235294	0.84117647
17	0.45614035	0.62745098	0.54179567
18	0.46491228	0.52941176	0.49716202
19	0.89473684	0.76470588	0.82972136
20	0.6	0.76470588	0.68235294
21	0.49122807	0.68627451	0.58875129
22	0.37719298	0.52941176	0.45330237
23	0.97368421	0.41176471	0.69272446
24	0.09649123	0.64705882	0.37177503

Final Exam statistics

Number of questions: 45
Number of students that took the exam (MW section): 37
Number of students that took the exam (TR section): 17
Average score per question (0 to 1)

Problem #	MW Section	TR Section	Section Average
1	0.78378378	1	0.89189189
2	0.94594595	0.88235294	0.91414944
3	0.94594595	0.88235294	0.91414944
4	0.75675676	0.88235294	0.81955485
5	0.97297297	0.88235294	0.92766296
6	0.67567568	0.82352941	0.74960254
7	0.94594595	0.94117647	0.94356121
8	1	0.94117647	0.97058824
9	0.54054054	0.64705882	0.59379968
10	0.91891892	0.88235294	0.90063593
11	1	0.88235294	0.94117647
12	0.83783784	0.88235294	0.86009539
13	0.94594595	0.94117647	0.94356121
14	0.7027027	0.70588235	0.70429253
15	0.7027027	0.70588235	0.70429253
16	0.83783784	0.82352941	0.83068362
17	0.22972973	0.29411765	0.26192369

18	0.89189189	0.88235294	0.88712242
19	0.83783784	0.64705882	0.74244833
20	0.43243243	0.64705882	0.53974563
21	0.91891892	0.82352941	0.87122416
22	0.94594595	0.82352941	0.88473768
23	0.67567568	0.76470588	0.72019078
24	0.91891892	0.86470588	0.8918124
25	0.91891892	0.61176471	0.76534181
26	0.14864865	0.74264706	0.44564785
27	0.26486486	0.8	0.53243243
28	0.59459459	0.88235294	0.73847377
29	0.81081081	0.91176471	0.86128776
30	0.54954955	0.32352941	0.43653948
31	0.24324324	0.35294118	0.29809221
32	0.85585586	0.82352941	0.83969263
33	0.81531532	0.94117647	0.87824589
34	0.89527027	0.81617647	0.85572337
35	0.90202703	0.84926471	0.87564587
36	0.86486486	0.94117647	0.90302067
37	0.97297297	1	0.98648649
38	0.43243243	0.70588235	0.56915739
39	0.78378378	0.82352941	0.8036566
40	0.95945946	0.86764706	0.91355326
41	0.27027027	0.35294118	0.31160573
42	0.93918919	0.88235294	0.91077106
43	0.43243243	0.47058824	0.45151034
44	0.40540541	0.80392157	0.60466349
45	0.83783784	0.41176471	0.62480127

The following table shows the averages (0 to 1) of all lab assignments (counting only those students that submitted the labs and not including late submission penalties):

-	MW Section	TR Section	Section Average
Lab 1	0.7	0.85847059	0.77923529
Lab 2	0.83	0.86411765	0.84705882
Lab 3	0.78	0.77617529	0.77808765
Lab 4	0.85	0.85308824	0.85154412
Lab 5	0.78	0.84360133	0.81180067
Lab 6	0.73	0.82642857	0.77821429
Lab 7	0.79	0.83861538	0.81430769

Analysis by the instructor

Some observations of the information presented above:

- The exam averages were better than in previous iterations of the course.
- All partial exams had more than 115 points available (and they were graded out of 100), so some students chose not to answer some questions. This resulted in some exam problems having an average lower than 0.5.
- Even though the final exam was comprehensive and long, students did great.
- The areas where the students struggled the most are: running time analysis (big-O for both recursive and non-recursive algorithms), balanced trees, disjoint set forests, and some graph algorithms.
- Based on the final exam results, and the average of lab grades, most outcomes were attained. The only learning outcome that had an average lower than 0.5 (for one of the sections) was L2.2.d (topological sort). This algorithm (and the connected components algorithm) need to be given a little more attention in the next iteration of the course.
- Overall, the skills that labs aimed to tackle were attained.
- The number of students that did not pass the class (combining the two sections) was 9, which represents a 15.7% of the students that completed the class.
- The number of students that dropped the class (combining the two sections) was 19, which represents a 25% of the students that originally registered for the class. This is an unprecedented event. Since this occurred in other fundamental courses as well, the global pandemic and the wide variety of diverse problems that it brings seem to be cause. Despite our best efforts, we lost a significant amount of students.
- The use of Google Colab and automated grading really helped. Students did not need to install anything to get going. They could use any device with browsing capabilities to complete assignments. This allowed feedback to be given instantly for many problems, and freed our TAs and IAs to provide more specialized feedback where it was needed most.
- Even though some students were concerned with the transition from Java to Python, it did not represent a problem, even when the class was taught online. An *Intro to Python* assignment was given at the beginning of class which allowed students to get more comfortable with the language early in the class.
- Students like the fact that they can take make-up for their exam grades when they take the final. Many of them greatly improved their grades because of this.
- Grading coding exams at test time with the use of unit tests was challenging, but it worked great.

Recommendations

- The TA, PL, and IAs were key to make the class work well online. Our live sessions were spent solving problems. Students worked in small teams (breakout rooms) while the TA, IAs, PL, and myself moved around to assist anybody that needed help. Without the assistants' help, this would not have been as efficient and effective as it was. We should continue to support the programs that fund TAs/IAs/PLs.
- Students actually read the zyBook, we should continue using it.
- Some of the pre-recorded lectures were long. It may be beneficial to split long videos into smaller, more manageable pieces.
- A lot of content is taught in this course, we should have a discussion on this topic.

Percentage of Success – Learning outcomes

Only exams and lab assignments were used to calculate the following statistics. Quizzes were excluded since students were awarded full-credit for attempting to solve all problems.

Outcome	% Success (MW Section)	% Success (TR Section)	Section Average
L1.1.a	0.68572309	0.76567227	0.72569768
L1.1.b	0.75203888	0.8529902	0.80251454
L1.1.c	0.77805877	0.85395532	0.81600704

L1.1.d	0.67572309	0.77635294	0.72603802
L1.2	0.73648649	0.82352941	0.78000795
L1.3	0.70305832	0.85294118	0.77799975
L2.1.a	0.67438894	0.64264246	0.6585157
L2.1.b	0.63545613	0.76057862	0.69801738
L2.1.c	0.64946227	0.66291803	0.65619015
L2.1.d	0.68229425	0.71064126	0.69646775
L2.1.e	0.63168529	0.66147859	0.64658194
L2.2.a	0.73	0.82642857	0.77821429
L2.2.b	0.85944239	0.90254062	0.8809915
L2.2.c	0.81470555	0.87116807	0.84293681
L2.2.d	0.48346373	0.58823529	0.53584951
L2.2.e	0.76984886	0.81598214	0.7929155
L2.2.f	0.83528924	0.81470588	0.82499756
L2.3	0.85308661	0.89346078	0.8732737
L2.4	0.82330525	0.85913423	0.84121974
L3.1	0.82400903	0.87212184	0.84806543
L3.2	0.76712224	0.88296569	0.82504396
L3.3	0.88119763	0.856272	0.86873481
L3.4	0.61891552	0.67409562	0.64650557
L3.5	0.64294572	0.69711983	0.67003277

CS 2302 Data Structures, Fall 2020 CQI Report

Olac Fuentes – Section 3

Outcome Assessment

1. **Table 1: Summary of Outcomes and Assessment.**

Outcome	Level	Exam 1	Exam 2	Exam 3	Other	Final Exam	Result
3.1 Given a problem, judge which data structures that are required to solve it efficiently and justify the selection.	3				Lab1:91% Lab2:80% Lab3:88% Lab4:89% Lab5:82% Lab6:86%		Satisfactory Median: 87%
3.2 Given a non-recursive algorithm, examine its loop structure, assess its asymptotic running time, and express it using big-O notation.	3	Q7:79% Q8:92% Q9:100% Q10:83% Q11:100%				Q9:43% Q10:91% Q11:57%	Satisfactory Median: 88%
3.3 Given a recursive algorithm, examine its structure, formulate and solve a recurrence equation defining its running time, and express it using big-O notation	3	Q12:67% Q13:100% Q14:63% Q15:38% Q16:75%				Q12:43% Q13:22% Q14:30% Q15:22%	Unsatisfactory: Median:63%
3.4 Design and implement solutions to computational problems based on iteration and recursion	3	Q17:67% Q18:42% Q19:80% Q20:86% Q21:51%	Q1:96% Q2:78% Q3:83% Q4:60% Q5:58% Q6:71% Q7:92% Q8:61%	Q1:96% Q2:88% Q3:96% Q4:49% Q5:64%	Lab1:91% Lab2:80% Lab3:88% Lab4:89% Lab5:82% Lab6:86%	P1:95% P2:62% P3:76%	Satisfactory Median:82%
3.5 Trace the behavior of non-trivial methods and algorithms.	3	Q1:95% Q2:76% Q3:85% Q5:41% Q6:41%				Q1:61% Q2:70% Q3:61% Q4:70% Q5:65% Q6:83% Q7:57%	Satisfactory Median:70%
2.1.a Describe, implement, and use the following data structures: Heaps	2		Q4:60% Q8:61%			P6:58%	Unsatisfactory Median:60%
2.1.b Describe, implement, and use the following data structures: Hash tables	2		Q3:83% Q7:92%		Lab4:89%	P7:57%	Satisfactory Median: 86%
2.1.c Describe, implement, and use the following data structures: Balanced trees	2		Q1:96% Q2:78% Q5:58% Q6:71%		Lab3:88% P4:44% P5:29%	P4:53% P5:53%	Unsatisfactory Median:58%
2.1.d Describe, implement, and use the following data structures: Graphs	2			Q1:96% Q2:88% Q3:96% QA1:78% QA2:53% QA3:66%		Q16:70% Q17:78% Q18:48% Q19:91% Q20:57% P8:74% P9:78% P10:53%	Satisfactory Median: 76%
2.1.e Describe, implement, and use the following data structures: Disjoint set forests	2			Q4:49%	Lab5:82%	P12:43%	Unsatisfactory Median: 59%
2.2.a Describe, implement, and apply the following graph algorithms: Connected components	2				Lab5:82%		Satisfactory Median: 82%
2.2.b Describe, implement, and apply the following graph algorithms: Breadth-first search	2			QA18:71%		Q21:83%	Satisfactory Median: 77%
2.2.c Describe, implement, and apply the following graph algorithms: Depth-first search	2			QA19:79%		Q22:57%	Unsatisfactory Median: 68%
2.2.d Describe, implement, and apply the following graph algorithms: Topological sorting	2			QA20:54%			Unsatisfactory Median: 54%
2.2.e Describe, implement, and apply the following graph algorithms: Minimum spanning trees (Kruskal's and Primm's)	2			QA16:71% QA17:29%			Unsatisfactory Median: 50%
2.2.f Describe, implement, and apply the following graph	2			QA20:63%		Q23:13%	Unsatisfactory

algorithms: Single-source shortest paths							Median: 33%
2.3 Trace the behavior of recursive programs using activation records	2	Q4:38%			Q8:22%		Unsatisfactory Median: 30%
2.4 Reason about the running times of algorithms in relation to the size of their inputs.	2				Lab2:80% Lab3:88% Lab4:89%		Satisfactory Median: 88%
1.1.a Identify, explain and apply the following algorithm design techniques: Greedy algorithms	1					Q24:78% Q25:70% Q26:87% Q27:57% Q28:83%	Satisfactory Median: 78%
1.1.b Identify, explain and apply the following algorithm design techniques: Divide and conquer	1					Q24:78% Q25:70% Q26:87% Q27:57% Q28:83% P11:55%	Satisfactory Median: 78%
1.1.c Identify, explain and apply the following algorithm design techniques: Dynamic programming	1			Q5:64%		Q24:78% Q25:70% Q26:87% Q27:57% Q28:83%	Satisfactory Median: 78%
1.1.d Identify, explain and apply the following algorithm design techniques: Backtracking	1			QA23:42%		Lab1:91% Q24:78% Q25:70% Q26:87% Q27:57% Q28:83%	Satisfactory Median: 78%
1.2 Explain the concept of NP completeness	1						Not evaluated
1.3 Explain the utility of randomized algorithms	1			Q24:75%		Q24:78%	Satisfactory Median: 77%

Analysis by Instructor:

Table 1 presents the results of mapping educational outcomes to assessment. We make following observations and recommendations:

Observations:

1. Four of the five level three outcomes were satisfied. Outcomes 3.4 and 3.5 are considered the most important in terms of helping students develop problem-solving skills and thus were evaluated multiple times, often in combinations the more fact-based material covered in class. Outcome 3.3, related to analysis of recursive algorithms was not satisfied. Students had difficulty understanding the mathematical tools required, recurrence relations. Given the circumstances, the instructor decided not to spend extra time trying to address this issue, instead focusing on problem solving.
2. Level two outcomes were the most problematic. Understanding of specific algorithms is usually tested by having student trace them for particular inputs. As the exams were administered remotely, testing by means of tracing became unreliable, as it was possible for students to simply run the code provided instead of tracing.
3. All level one outcomes were satisfied, with the exception of explaining NP completeness, which was not tested.
4. 24 students took the final exam. The grade distribution was as follows: A: 9, B: 6, C: 3, D: 1, F:5. This yields a passing rate of 18/24 =75%, which is similar to that of previous semesters. However, an unusually high number of students dropped the class, a total of 16, thus less than half of the students who originally enrolled in the class completed it successfully. This is, by far, the lowest percentage we have had in recent years.

Recommendations:

1. In-class work was emphasized, and it resulted on improved learning for many students. However, it seemed to widen the gap between the stronger students and those who were struggling, as the latter tended to disengage from activities and miss class more often. The return to face to face interaction should help reduce this problem, but instructors should work on finding ways of keeping unprepared students engaged.

2. Many students had difficulty tracing code, particularly recursive functions, even though a lot of class time was spent on it. This may be a consequence of the fact that in online learning students are always in front of the computer and develop a tendency to program by trial and error. Emphasizing paper-only exercises should help with this.
3. It has been observed that most students who pass CS2401 with a C do not pass CS2302. CS2401 instructors should consider whether borderline cases would benefit more from repeating CS2401 than from taking, and likely failing, CS2302.
4. It is very difficult to cover all topics presented in the outcomes. The committee should find, in consultation with the rest of the faculty, one or more non-essential outcomes to eliminate in order to allow more flexibility to review and provide additional practice in the topics that students find difficult.