Fundamentals CQI

Analyzed Semester: Fall 2022

During Fall of 2022 there were 8 primary instructors (and two Googlers in Residence) teaching 10 sections of the 3 fundamentals courses:

- CS1301/1101: Akbar/Garrett, Gurijala, Jimenez Velasco, and Mejia;
- CS 2401: Akbar, Ceberio/Gamez, and DeBlasio;
- CS 2302: Aguirre, Fuentes, and Mejia.

In general, most instructors continue to see some recovery from online versions of the courses. It is noted that in general the DF rate seems to be around 20% (a bit higher for CS1).

1 General Recommendations

While each instructor made their own individual recommendations, below are some common themes that appeared in multiple analyses.

- All courses moved away from the zyBook this semester and elected the Pearson Revel book as a general adoption, and while this seems to be a good thing, some instructors noted we may want to revisit this closer to the next CQI cycle. In general, we still encourage new instructors and in early fundamentals courses (i.e. CS1) to heavily rely on a published textbook, but individuals may choose to use their own resources.
- Continue to include review of the previous material early in the semester, in most cases the recommendation was to do this outside class. It is recommended that these reviews be coordinated between sections to reduce per-instructor effort and increase availability.
- The committee would like the curriculum committee to evaluate if it is feasible to re-include a 1 credit hour lab section in CS3 (making it CS 2402).
- Consult with other committees to move the teamwork outcomes (ABET required) to upper-level courses (such as Databases).

2 Course Change Recommendations

It is noted that several of the individual reports include some suggestions for changes to outcomes. Due to time constraints, we do not propose the faculty vote on these now, as we will being the changes indecently at another time. These suggestions have not been vetted by the entire committee and thus will be tabled until they have been. We iterate them here as a summary of the instructors current thoughts.

Individual justifications can be found in the summary reports for the respective courses. Note some of these changes relate to reverting changes made only recently.

2.1 CS 1301

- Replace outcome 2.7 with a level 1 outcome: "Understand the use of hexadecimal and binary in problem solving and computer science in general".
- ~~Remove "Use Linked List" from level 2 and add the following as a level 1 outcome: "Understand basic linked list representation and manipulation"~~

2.2 CS 1101

- Remove outcome 2.7 (binary, etc).
- Remove using linked lists as a level 2 outcome

3 Proposal to Aquire Longitudinal Analysis Data

For the purposes of improving our ability to adjust the fundamentals courses to the needs of our students the committee would like to request information in order to do a longitudinal study. We request the following information about students over the past 5 years:

1. grades, semesters, sections, and instructors for each attempt of each of the fundamentals classes.
2. grades in the CS4s (CS 3331, CS 3350, CS 3360, CS3432).
3. which classes (if any) did a student get credit for taking at another institution.
4. student entry point/previous institution and program (community college, high school, etc)

We understand these results will likely need to be anonymized to protect student information. We think this will allow us to make additional recommendations in future CQI cycles. Some of the questions we would like answered are:

- Does taking fundamentals classes over the summer correlate in any way with progress in future classes?
- Does this change if those classes are taken in the summer as a second or third attempt?
- Does receiving credit from another institution impact later progress?
- Does CS2101/2202 vs MATH2300 impact outcomes in other cases?
- Are repeated attempts of fundamentals classes indicative if grades in CS4s?

One of the reasons we would like to do this is to help adjust decision making in final grades, it may also help us identify students who may need additional help early in a semester.

CS 1301 & CS 1101

The University of Texas at El Paso
Department of Computer Science
Fall 2022 CQI
CS 1301 & CS 1101 Summary

CS 1301 & CS 1101 were evaluated during the Fall 2022 semester. In general, all sections of CS 1301 & CS 1101 were successful. In comparison to all the sections, a majority of all the sections had an acceptable passing rate – two sections of CS 1301 and CS 1101 had a higher-than-expected failure rate. Below is a breakdown of all the sections as a whole as well as course recommendations.

The instructors for the semester were:

- Akbar, Monika
- Gurijala, Bhanukiran
- Jimenez Velasco, Maria
- Mejia, Daniel

CS 1301:

- 5 Sections
- 197 students were issued a grade
  - 145 students passed (A, B, C) – 73.6%
  - 52 students failed (D, F) – 26.4%
- 26 students withdrew (W)

CS 1101

- 5 Sections
- 201 students were issued a grade
  - 128 students passed (A, B, C) – 63.7%
  - 73 students failed (D, F) – 36.3%
- 27 students withdrew (W)

Grade Distribution:

				A	B	C	D	F	W	TOTAL EARNING GRADES	A	B	C	D	F	ABC	DF
Akbar	18414	CS	1301	22	12	7	1	6	3	48	45.8%	25.0%	14.6%	2.1%	12.5%	85.4%	14.6%
Akbar	18685	CS	1101	22	7	5	3	11	3	48	45.8%	14.6%	10.4%	6.3%	22.9%	70.8%	29.2%
Gurijala	15665	CS	1301	5	6	9	5	15	8	40	12.5%	15.0%	22.5%	12.5%	37.5%	50.0%	50.0%
Gurijala	15662	CS	1101	9	7	2	4	18	8	40	22.5%	17.5%	5.0%	10.0%	45.0%	45.0%	55.0%
Jimenez	20440	CS	1301	3	3	3	6	2	9	17	17.6%	17.6%	17.6%	35.3%	11.8%	52.9%	47.1%
Jimenez	20441	CS	1101	4	2	3	8	1	10	18	22.2%	11.1%	16.7%	44.4%	5.6%	50.0%	50.0%
Jimenez	15666	CS	1301	22	8	5	2	7	5	44	50.0%	18.2%	11.4%	4.5%	15.9%	79.5%	20.5%
Jimenez	16430	CS	1101	14	9	8	1	15	5	47	29.8%	19.1%	17.0%	2.1%	31.9%	66.0%	34.0%
Mejia	15619	CS	1301	14	13	13	3	5	1	48	29.2%	27.1%	27.1%	6.3%	10.4%	83.3%	16.7%
Mejia	15637	CS	1101	21	11	4	1	11	1	48	43.8%	22.9%	8.3%	2.1%	22.9%	75.0%	25.0%
			Total 1301	66	42	37	17	35	26	197	**33.5%**	**21.3%**	**18.8%**	**8.6%**	**17.8%**	**73.6%**	**26.4%**
			Total 1101	70	36	22	17	56	27	201	**34.8%**	**17.9%**	**10.9%**	**8.5%**	**27.9%**	**63.7%**	**36.3%**

Outcomes:
A majority of the course outcomes were assessed in all the sections. However, there were a few that were not assessed in some sections. In general, all course outcomes were met with 70% or above.

CS 1301

- Level 1 outcomes appear to be the least met
    - Purpose of multi-dimensional arrays
    - Classes of programming languages
- Level 1 outcomes tend to be on the edge of the scope of the course
- Level 2 outcomes are mostly met except for
    - String manipulations
    - Binary arithmetic
- Level 3 outcomes are well met and assessed by all sections

CS 1101

- Level 2 outcomes were mostly met except
    - Binary arithmetic
    - Usage of linked lists
- Level 3 outcomes are well met and assessed by all sections

Observations:

1. Some level 1 outcomes are difficult to integrate into the course when attempting to focus on other outcomes (i.e., level 3)
2. Students spend a significant amount of time learning level 3 outcomes, as such level 1 & 2 outcomes are discussed less
    a. Students need the additional time working on level 3 outcomes (i.e., practicing loops, methods, and arrays)

Recommendations:

General

1. Continue monitoring the textbook effectiveness
    a. Revisit the textbook recommendation during the next CQI cycle
2. Continue monitoring the following outcomes to determine how they fit within the scope of the course
    a. Compilation & Interpretation
    b. Classes of programming languages

CS 1301

1. Remove binary arithmetic outcomes from the course
    a. Level 2:
        i. Apply Binary arithmetic to solve problems. This includes:
            1. Conversion between binary, decimal, and hexadecimal numbers,
            2. Application of arithmetic operations on binary and hexadecimal numbers
        ii. RATIONALE: This outcome does not align to the scope of the course
2. Move Linked List to a level 1 outcome
    a. Describe the purpose of linked lists
        i. RATIONALE: This outcome comes near the very end of the course and often does not have enough time to be reasonably covered as a level 2 outcome

CS 1101

1. Add file reading as a Level 2 outcome:
    a. Level 2:
        i. Read/write files (e.g., .txt, .csv)
        ii. RATIONALE: Reading files is critical for lab assignments and is considered a fundamental skill for introductory students
2. Remove using linked lists as a level 2 outcome
    a. RATIONALE: Linked lists appears near the end of the course and does not have enough time to be implemented.
3. Remove level 2-11 outcome
    a. Use teamwork roles and strategies in the classroom
    b. RATIONALE: This cannot be appropriately and objectively assessed in the course as students do not engage in paired programming. Students work with one another in informal settings, however, it is not practical to assess in an introductory course.

Course Number: CS 1301
Course Title: Intro to Computer Science
Course Instructor: Daniel Mejia
Term: Fall 2022

Grade Distribution
There were 49 students registered for credit on census day – 1 student dropped the course after census day up until the course drop deadline and received an automatic W. 48 students earned a grade in the course.

Grade	Pass			Fail	
	A	B	C	D	F
No. of students	14/48	13/48	13/48	3/48	5/48
Percentage	29.1%	27.1%	27.1%	6.3%	10.4%
Total	83.3%			16.7 %	

*C is a passing grade in CS 1301.

Failing Students

- S1 did not take any exams and turned in one assignment the entire semester. Stopped accessing the course September 29, 2022.
- S2 took Exam 1 & 2 and did not do well. Completed only about half of the assignments/quizzes. Stopped accessing the course November 20, 2022.
- S3 took Exam 1, 2, 3, and Final Exam and did poorly. Completed the entire course but missed several assignments.
- S4 took Exam 1, 2, 3, and Final Exam and did poorly. Completed the entire course but missed several assignments.
- S5 took Exam 1, 2, 3, and Final Exam and did poorly. Completed the entire course but missed several assignments.
- S6 took Exam 1, 2, 3, and Final Exam and did poorly. Completed the entire course but missed several assignments.
- S7 took Exam 1, 2, 3, and Final Exam and did poorly. Completed the entire course but missed several assignments.
- S8 took Exam 1, 2, 3, and Final Exam and did poorly. Completed the entire course but missed several assignments.

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. Quizzes were given each week and students were given 36 hours to complete the quiz through Blackboard. 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.

1. The major advances in the history of computing
    a. FC11 – 91%
2. The relation between computing and society, including social, ethical, and legal issues
    a. History & Professionalism – 97.2%
3. The importance of computing in a variety of professions: required knowledge and skill sets for major career options
    a. History & Professionalism – 97.2%
4. Classes of programming languages, including:
    a. Imperative,
    b. Object oriented,
    c. Declarative, and
    d. Functional
    FC3 – 84.7%
5. The purpose of multi-dimensional arrays (dimension 3 and above)
    a. Not assessed
6. The purpose of and relationship between classes and objects
    a. FC9 – 89%
    b. FC14 – 91.3%
    c. FC15 – 89%
7. The purpose of pre/post conditions, in particular as related to verification
    a. FA20 – 97.8%
8. Compilation and interpretation
    a. Not assessed

Level 2: Application and Analysis.

1. Analyze problems, design and implement solution algorithms, including correct use of:
    a. User-defined types and their implementation as classes
    b. Basic string manipulation techniques using language functions, including:
        i. Traversing strings,
        ii. Accessing characters,
        iii. Comparing strings,
        iv. Concatenating strings
        FB5 – 76.1%
        Exam Three 6 – 69.7%
        Quiz 6 #7 – 95.4%

2. Algorithm-tracing techniques to ensure solution correctness including method calls
    a. FA11 – 89.1%

    b. FC7 – 86.9%
3. Use testing and debugging strategies to identify software faults by creating test suites that include:
    a. Black-box test cases
    b. Basic white-box test cases
       FC17 – 84.7%
4. Use general software engineering principles, including abstraction and problem decomposition in problem and solution analysis
    a. FA17 – 93.4%
5. Use informal pseudocode to describe algorithms
    a. FB10 – 73.9%
6. Use 2D arrays
    a. FC22 – 89.1%
7. Apply Binary arithmetic to solve problems. This includes:
    a. Conversion between binary, decimal, and hexadecimal numbers,
    b. Application of arithmetic operations on binary and hexadecimal numbers
       FC27 – 84.7%
       FC29 – 67.3%
       Exam Three 1 – 74.4%
8. Use recursion for solving simple problems
    a. FC16 – 82.6%
9. Use linked lists
    a. FC24 – 86.9%
    b. Exam Three 31 – 74.4%

Level 3: Synthesis and Evaluation.

1. Basic variable types including Booleans, integers, real numbers, characters, strings
    a. FA7 – 97%
    b. FA16 – 98.7%
    c. FA19 – 100%
2. 1-D arrays
    a. FB7 – 86.9%
    b. FB9 – 76.1%
3. Assignment and arithmetic
    a. FA14 – 94.8%
4. Logical propositions to define conditional and loop statements
    a. FA20 – 97.8%
5. For-loops
    a. FA9 – 93%
    b. FB5 – 76.0%
6. While-loops
    a. FB1 – 86.9%
7. Methods/functions, parameter passing, return values
    a. FB15 – 80.4%
8. Algorithmic building blocks including
    a. Min

i. Exam Three 28 – 91%

b. Max
    i. Exam Three 28 – 91%
c. Average
    i. FB10 – 73.9%
d. Summation
    i. FB7 – 86.9%
e. Linear search
    i. FB7 – 86.9%
    ii. FB14 – 95%

Analysis of the Results, and Instructor Recommendations

The results of the learning outcomes were all met with at least a satisfactory level, with. Most being good/exceptional. Except for compilation, interpretation, and purpose of multidimensional arrays (3+), 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

Observations:

3. Some level 1 outcomes are difficult to integrate into the course when attempting to focus on other outcomes (i.e., level 3)
4. Students spend a significant amount of time learning level 3 outcomes, as such level 1 & 2 outcomes are discussed less
5. Binary Arithmetic appears to be outside (or very close to) the scope of the course considering time constraints

Instructor recommendations:

1. Have more discussion about level one outcomes to determine the scope within the class
    a. Programming language paradigms does not easily fit within the course to be assessed
    b. Compilation and interpretation do not easily fit within the course to be assessed
    c. Purpose of multi-dimensional arrays does not easily fit within the course to be assessed
    d. Review these outcomes in the upcoming CQI to determine if they are necessary

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

Grade Distribution
There were 49 students registered for credit on census day – 1 student dropped the course after census day up until the course drop deadline and received an automatic W. 48 students earned a grade in the course.

Grade	Pass			Fail	
	A	B	C	D	F
No. of students	21/48	11/49	4/49	1/49	11/49
Percentage	43.8%	22.9%	8.3%	2.1%	22.9%
Total	75%			15%	

*C is a passing grade in CS 1101.

Failing Students

- S1-S3 did not turn in any comprehensive labs (3)
- S4 did not turn in 1 comprehensive labs (3) and 2 Comprehensive Labs had an average of 8/100
- S5 did not turn in 1 comprehensive labs (3) and 2 Comprehensive Labs had an average of 27.5/100
- S6 did not turn in 1 comprehensive labs (3) and 2 Comprehensive Labs had an average of 38.5/100
- S7 did not turn in 1 comprehensive labs (3) and 2 Comprehensive Labs had an average of 58.5/100
- S8-10 did not turn in 2 comprehensive labs (3)
- S11 earned an average of 55 of all comprehensive labs
- 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
Learning Outcomes

Level 2: Application and Analysis.

1. Analyze problems, design and implement solution algorithms, including correct use of:
    a. Simple I/O operations (reading from and printing to the terminal)
        i. CL1 – 88.7%
        ii. CL2 – 77.1%
        iii. CL3 – 77.8%
    b. User-defined types and their implementation as classes
        i. CL3 – 77.8%
    c. Basic string manipulation techniques using language functions, including:
        i. Traversing strings,
        ii. Accessing characters,
        iii. Comparing strings,
        iv. Concatenating strings

        CL2 – 77.1%
2. Algorithm-tracing techniques to ensure solution correctness including method calls
        i. CL1 – 88.7%
        ii. CL2 – 77.1%
        iii. CL3 – 77.8%
3. Use testing and debugging strategies to identify software faults by creating test suites that include:
    a. Black-box test cases
    b. Basic white-box test cases
        i. CL3 – 77.8%
4. Use general software engineering principles, including abstraction and problem decomposition in problem and solution analysis
    a. CL1 – 88.7%
    b. CL2 – 77.1%
    c. CL3 – 77.8%
5. Use informal pseudocode to describe algorithms
        i. CL1 – 88.7%
        ii. CL2 – 77.1%
6. Use 2D arrays
    a. CA3 – 84.9%
7. Apply Binary arithmetic to solve problems. This includes:
    a. Conversion between binary, decimal, and hexadecimal numbers,
    b. Application of arithmetic operations on binary and hexadecimal numbers

    Not Assessed
8. Use recursion for solving simple problems
    a. Quiz 5 – 97.5%
9. Use linked lists
    a. Not assessed
10. Instead of IDEs, use a command line interface (terminal) to compile and execute programs
    a. CL1 – 88.7%
    b. CL2 – 77.1%
    c. CL3 – 77.8%
11. Use teamwork roles and strategies in the classroom

    Not Formally Assessed, but witnessed

Level 3: Synthesis and Evaluation.

1. Basic variable types including Booleans, integers, real numbers, characters, strings,
    a. CL1 – 88.7%
    b. CL2 – 77.1%
    c. CL3 – 77.8%
2. 1-D arrays
    a. CL2 – 77.1%
3. Assignment and arithmetic
    a. CL1 – 88.7%
4. Logical propositions to define conditional and loop statements
    a. CL1 – 88.7%
    b. CL2 – 77.1%
    c. CL3 – 77.8%
5. For-loops
    a. CL1 – 88.7%
    b. CL2 – 77.1%
    c. CL3 – 77.8%
6. While-loops
    a. CL1 – 88.7%
    b. CL2 – 77.1%
    c. CL3 – 77.8%
7. Methods/functions, parameter passing, return values
    a. CL2 – 77.1%
    b. CL3 – 77.8%
8. Algorithmic building blocks including:
    a. Min
    b. Max
    c. Average
    d. Summation
    e. Linear search

    CA3 – 84.9%

9. Coding and documentation standards
    a. CL1 – 88.7%
    b. CL2 – 77.1%
    c. CL3 – 77.8%

Analysis of the Results, and Instructor Recommendations

The results of the learning outcomes were all met with a satisfactory (excellent) level, except for 2.7 & 2.9 which were not formally assessed. A majority of the outcomes are covered by the comprehensive labs as they encompass the majority of the topics that are discussed.

Recommendations

Instructor recommendations:

1. 2.7 – Binary arithmetic and conversions does not appear to fit within the scope of the course and is recommended to be removed

2. 2.9 – LinskedList 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.
    a. Move to level 1 outcome or remove it completely

Course Outcomes: CS 1301, CRN: 18414
Fall 2022

February 16, 2023

Course Number: CS 1301
Course Title: Introduction to Computer Science
Course Instructor: Monika Akbar
Instructional Assistant: Alejandro (Alex) Vargas Acosta (20 hours/week)

Course Description
This is the first course of the CS fundamental course series. Students are expected to 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.

This report describes the outcome section taught by Monika Akbar.

Grade Distribution

The class started with 50 students enrolled, and one student dropped the course. The table below summarizes the distribution of final letter grades that the remaining 49 students earned. Out of the seven students who failed, six did not take the final exam.

Grade	Pass				Fail
	A	B	C*	D	F
No. of students	22/49	12/49	7/49	1/49	7/49
Percentage	44.89%	24.48%	14.28%	2.04%	14.28%
Total	83.67%			2.04%	14.28%

*C is the passing grade.

Assessment Method

The formal assessment was done using homework, in-class activities, quizzes, midterms, and final exam. The course had eight graded quiz, four in-class activities, five homework, three midterms, and one final exam. This report lists the average score for the question used to assess the outcome.

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:

1. The major advances in the history of computing
   Met (In-class activity: Computing and Linked list, q1, 78.95%)
2. The relation between computing and society, including social, ethical, and legal issues
   Met (Homework-moral machine, 83.67%)
3. The importance of computing in a variety of professions: required knowledge and skill sets for major career options
   Met (In-class activity: Computing and Linked list, q2, 78.95%)
4. Classes of programming languages, including:

a. Imperative,
b. Object oriented,
c. Declarative, and
d. Functional

Met (In-class activity: Computing and linked list, q3, 78.95%)

5. The purpose of multi-dimensional arrays (dimension 3 and above)
   Not assessed.
6. The purpose of and relationship between classes and objects
   Met (Final exam, q16, 79.77%)
7. The purpose of pre/post conditions, in particular as related to verification
   Met (In-class: Testing, q1, 91.36%)
8. Compilation and interpretation
   Met (Midterm 1, q15, 94.02%)

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. Analyze problems, design and implement solution algorithms, including correct use of:
   a. User-defined types and their implementation as classes
   Met (Final exam, q14, 91.67%)
   b. Basic string manipulation techniques using language functions, including:
      i. Traversing strings,
      ii. Accessing characters,
      iii. Comparing strings,
      iv. Concatenating strings

      Not assessed.
2. Algorithm-tracing techniques to ensure solution correctness including method calls
   Met (Final exam, q13, 89.29%)
3. Use testing and debugging strategies to identify software faults by creating test suites that include:
   a. Black-box test cases
   Met (Final exam, q24, 80.96%)
   b. Basic white-box test cases
   Met (Final exam, q23, 85.72%)
4. Use general software engineering principles, including abstraction and problem decomposition in problem and solution analysis
   Met (Midterm 3, q4, 78.62%)
5. Use informal pseudocode to describe algorithms
   Met (Midterm 1, q18, 88.63%)
6. Use 2D arrays
   Not Met (Final exam, q19, 61.91%)
7. Apply Binary arithmetic to solve problems. This includes:
   a. Conversion between binary, decimal, and hexadecimal numbers,
   Met (Final exam, q12, 82.74%)
   b. Application of arithmetic operations on binary and hexadecimal numbers
   Not assessed
8. Use recursion for solving simple problems
   Met (Midterm 3, q8, 69.34%, q9, 93.34%, q10, 84.45%)
9. Use linked lists

Not Met (Final exam, q25, 62.95%, homework:linked list q1, 84.87%, q2, 81.58%)

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 high-level language to express solutions to programming problems, including the pseudocode correct use of:

1. Basic variable types including Booleans, integers, real numbers, characters, strings
   Met (Midterm 1, q12, 89.22%)
2. 1-D arrays
   Met (Midterm 2, q3, 76.39%)
3. Assignment and arithmetic
   Met (Final exam, q1, 97.62%)
4. Logical propositions to define conditional and loop statements
   Met (Final exam, q9, 95.24%)
5. For-loops
   Met (Final exam, q18, 85.72%)
6. While-loops
   Met (Midterm 2, q8, 81.25%)
7. Methods/functions, parameter passing, return values
   Met (Midterm 3, q12, 75.56%)
8. Algorithmic building blocks including
   a. Min
   b. Max
   c. Average
   d. Summation
   e. Linear search

   Met (Midterm 2, q2, 88.46%, Midterm 3, q4, 78.62%)

Not assessed:

- Level 1, 5: The purpose of multi-D arrays was discussed in class but not assessed.
- Level 2, 1b: String manipulations. Strings were introduced in class and used for comparison and other string operations, but String-related proficiency was not assessed.
- Level 2, 7b: Arithmetic operations of binary or hexadecimal numbers. This was not introduced or assessed in the class due to time constraints.

Observations

1. Students get the concept of recursion, but they struggle to use recursion to solve a problem. Students also struggle to trace recursive methods.
2. 2D array is another concept that students struggled with.
3. Students showed proficiency in linked list when it was given as homework. A question in the final exam shows students did not meet the proficiency level.
4. Level 2, item 4 needs to be clarified. I am using proficiency in writing methods as an indication of competency in this item. However, methods usually include other items (e.g., branching, loop), which should also be considered relevant for this item. So, the data for this item would be linked to multiple CS1 concepts.
5. Level 2, 1b. The class focuses on the foundations of computing concepts and their use in solving problems. The class is thus suitable for introducing Strings, and the lab is ideal for in-depth coverage of Strings.
6. Level 2, 7b, arithmetic operations in the context of CS1 may take time away from other important topics that students struggle with (e.g., loop, 2D array, recursion).
7. Level 1, item 7 is similar to Level 2, item 3.

8. We used the Pearson Revel book for the first time. This ebook was linked with Blackboard. Student scores were automatically populated in the Grade center. Student response was mixed. An initial study into students' scores in the book revealed that most students with the highest score in the book homework also received an 'A' in the course.
   A different trend is seen when I sorted the list for time spent in the book. Students who spent more time in the book did not necessarily get good grades. The five students who spent the most time in the book had grades ranging from D, C, B, and A. Two of these students spent more than 70 hours reading the book; rest three spent more than 30 hours. The student who had the second highest time spent in the book received a D. The book does list this student as 'Struggling'. The choice of a book and how the book can be used better to support student learning, needs more analyses and discussion.

Recommendations

- Branching and loops have the property to be nested. Currently, the learning outcomes are missing nested items. These proficiencies are necessary for doing well with advanced concepts (e.g., recursion, 2D array manipulation). We should consider including nested items in the learning out, at least at level-2.

Course Outcomes: CS 1101, CRN: 18685
Fall 2022

February 17, 2023

Course Number: CS 1101
Course Title: Introduction to Computer Science
Course Instructor: Monika Akbar
Instructional Assistant: Alejandro (Alex) Vargas Acosta (20 hours/week)

Course Description: CS 1101 is lab course for CS 1301. In this lab, 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.

This report describes the outcome section taught by Monika Akbar.

Grade Distribution

The lab had 49 students. The following table shows the summary of the final grades of these students.

Grade	Pass				Fail
	A	B	C*	D	F
No. of students	22/49	7/49	5/49	3/49	12/49
Percentage	44.89%	14.28%	10.20%	6.12%	24.48%
Total	69.38%			6.12%	24.48.8%

*C is the passing grade.

Assessment Method

The formal assessment was done using lab assignments and lab quizzes. The course had 11 lab assignments and nine quizzes. Aside from these, the lab had in-lab activities and homework, some of which were graded. This report lists the average score for the lab assignment or the quiz question.

Learning Outcomes

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. Analyze problems, design and implement solution algorithms, including correct use of:
    a. Simple I/O operations (reading from and printing to the terminal)
    Met (Lab assignment 2, 84.71%)

    b. User-defined types and their implementation as classes
    Met (Lab assignment 9, 96.53%)

    c. Basic string manipulation techniques using language functions, including:
        i. Traversing strings,
        ii. Accessing characters,
        iii. Comparing strings,
        iv. Concatenating strings

    Met (Lab assignment 5, 86.83%)

2. Algorithm-tracing techniques to ensure solution correctness including method calls
   Met (Lab assignment 5, 86.83%)

3. Use testing and debugging strategies to identify software faults by creating test suites that include:
   a. Black-box test cases
   b. Basic white-box test cases

   Met (Lab assignment 3, 91.04%)

4. Use general software engineering principles, including abstraction and problem decomposition in problem and solution analysis
   Met (Lab assignment 7, 90.25%)

5. Use informal pseudocode to describe algorithms
   Met (Lab assignment 4, 93.3%)

6. Use 2D arrays
   Not assessed

7. Apply Binary arithmetic to solve problems. This includes:
   a. Conversion between binary, decimal, and hexadecimal numbers,
   b. Application of arithmetic operations on binary and hexadecimal numbers

   Not assessed.

8. Use recursion for solving simple problems
   Met (Lab assignment 8, 84.43%)

9. Use linked lists
   Met (Lab assignment 9, 96.53%)

10. Instead of IDEs, use a command line interface (terminal) to compile and execute programs.
    Met (All students were expected to use a text editor for writing their code and a command line interface for compiling and running their code. Students submitted only the .java file as their answer to the lab assignments. Quiz 1 and 2 went over debugging using command line errors. Students received more than 80% on all the questions in these two quizzes. There were eight questions in total.)

11. Use teamwork roles and strategies in the classroom
    No 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 high-level language to express solutions to programming problems, including the pseudocode correct use of:

1. Basic variable types including Booleans, integers, real numbers, characters, strings,
   Met (Lab assignment 3, 91.04%)

2. 1-D arrays
   Met (Lab assignment 6, 90.54%)

3. Assignment and arithmetic
   Met (Lab assignment 3, 91.04%)

4. Logical propositions to define conditional and loop statements
   Met (Lab assignment 4, 93.3%)

5. For-loops
   Met (Lab assignment 5, 86.83%)

6. While-loops
   Met (Lab assignment 5, 86.83%)

7. Methods/functions, parameter passing, return values
   Met (Lab assignment 7, 90.25%, comprehensive lab 2, 82.72%)

8. Algorithmic building blocks including:
   a. Min
   b. Max
   c. Average
   d. Summation
   e. Linear search

   Met (Lab assignment 6, 90.54%, Lab assignment 8, 84.43%)

9. Coding and documentation standards
   Met (Coding standard was part of every lab. For example, good documentation, meaningful variable names, etc., were given certain points in grading each lab. Lab assignment description included how many points were assigned to such items, so students were aware of the expectations).

Observations

1. File reading is an expected skill for many of the lab assignments. This is not included in the learning outcomes.
2. Level 2, item 6 - 2D array was given as an extra credit/optional in-class activity. It was not assessed through a lab assignment due to time constraints.
3. Level 2, item 7 – binary and hexadecimal numbers were not assessed due to time constraints.
4. Level 3, item 11 - while some group activities are included as lab activities, it is difficult to measure those. Especially since the lab assignments are individual, the scope of emphasizing teamwork is limited in CS1. Maybe this can move to later classes when students have already built their individual skills.

Recommendations

1. Including file reading/writing as a learning outcome.
2. Removing binary-related items from the learning outcome.
3. Passing arrays to a method could be a separate item to indicate the importance of knowing the difference between passing variables and arrays in methods.
4. Reducing the number of comprehensive labs to two (instead of three). These labs take a significant amount of time, which takes time away from regular labs.

Course Number: CS 1301
Course Title: Intro to Computer Science
Course Instructor: Bhanukiran Gurijala

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 some students dropped out before the course drop date. The table below summarizes the distribution of final letter grades that the remaining 40 students earned at the end of the semester.

Grade	Pass			Fail	
	A	B	C	D	F
No. of students	5/40	6/40	9/40	5/40	15/40
Percentage	12.5%	15%	22.5%	12.5%	37.5%
Total	50%			50%	

*C is a passing grade in CS 1301.

Failing Students

Students who received F in the course

- 1 student did not submit anything in the entire semester and did not show up to the class as well.
- 2 students did not any of the exams (Exam 1, Exam2, Exam 3, and Final Exam)
- 2 students did not take 3 out of 4 exams (Exam 2, Exam 3, and Final Exam)
- 4 students did not take 2 out of 4 exams (Exam 3 and Final Exam)
- 1 student did not take 1 out of 4 exams (Exam 1)
- 3 students did not submit any of the assignments.
- 2 students did not submit 3 out of 4 assignments.
- 2 students did not submit 2 out of 4 assignments.
- 7 students did not submit 1 out of 4 assignments.
- In total, 9 students missed one or more exams and 14 students did not submit one or more assignments in the semester.
- 6 of the students failing the course stopped accessing and participating in the course from August 26, September 23, October 23, October 27, November 9, and November 23, 2022, respectively.

Students who received D grade in the course

- Student S1 did poorly (< 60) on Exams 2 and 3 and got a D in Exam 1 and C in Final Exam.
- Student S2 did poorly (< 55) on Exams 1, 2, and 3 and got a B in Final Exam.
- Student S3 did not submit any of the assignments but received A, D, C, and B on Exams 1, 2, 3, and final exam.

- Student S4 did poorly on all exams with highest being < 60 on Exam 2 and < 50 on all other exams.
- Student S5 did poorly on all exams with highest being borderline D on Final Exam and all other exams < 55.

All students were contacted after poor performance on an exam or if they did not take an exam. Most 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. Most students were able to complete the missing work prior to the end of the semester.

Exam 3 had some questions on concepts that were assessed in Exam 2 and if students demonstrated their mastery on those concepts and if they had a lower grade on those questions in Exam 2 then the grade for that question was replaced by the higher grade, they received on those questions in Exam 3. Homework was collected throughout the semester and a majority of the homework was accepted late, with minimal penalties.

Learning Outcomes
The assessment was done using exams, in-class quizzes, and homework.

For each outcome, I measured the average score obtained by students on the exam or quiz questions. The nominal acceptance value was 70% or above; 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. The percentage does not include student grades who did not submit an assessment item and/or did not attempt a particular question in an exam or assignment.

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:

1. The major advances in the history of computing.
   Hist of Computing/Profession HW – 80%
   Met (Good)

2. The relation between computing and society, including social, ethical, and legal issues.
   Hist of Computing/Profession HW – 80%
   Met (Good)

3. The importance of computing in a variety of professions: required knowledge and skillsets for major career options.
   Hist of Computing/Profession HW – 80%
   Met (Good)

4. Classes of programming languages, including Imperative, Object oriented, Declarative, and Functional
   Quiz 8 (Q1) – 46.43%
   Not Met

5. The purpose of multi-dimensional arrays (dimension 3 and above).
   Quiz 8 (Q4) – 92.86%

Met (Excellent)

6. The purpose of and relationship between classes and objects.
   Quiz 7 (Q1) – 88.89%
   Quiz 7 (Q2) – 88.89%
   Quiz 7 (Q3) – 77.78%
   Quiz 7 (Q4) – 70.37%
   Quiz 7 (Q7) – 85.19
   Average: 82.22%
   Met (Good)

7. The purpose of pre/post conditions, in particular as related to verification.
   Quiz 8 (Q3) – 100%
   Average: 100%
   Met (Excellent)

8. Compilation and interpretation.
   Exam 2 (Q11) – 48.65%
   Not Met

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. Analyze problems, design and implement solution algorithms, including correct use of:
   Homework 1 (Q2) – 63.63%
   Homework 1 (Q3) – 63.63%
   Homework 2 (Q2) – 91.18%
   Homework 2 (Q4) – 88.24%
   Average: 76.67%
   Met (Okay)
   a. User-defined types and their implementation as classes
      Exam 3 (Q5) – 84.38%
      Final Exam (Q12) – 75%
      Average: 79.7%
      Met (Good)
   b. Basic string manipulation techniques using language functions, including:
      i. Traversing strings
         Exam 3 (Q6) – 71.88%
         Homework 3 (Q2) – 74.19%
         Average: 73.03%
         Met (Okay)
      ii. Accessing characters
         Exam 3 (Q7) – 34.38%
         Final Exam (Q4) – 56.25%
         Average: 45.32%
         Not Met
      iii. Comparing strings
         Final Exam (Q4) – 56.25%

Quiz 6 (Q1) – 89.66%
Average: 72.96%
Met (Okay)

iv. Concatenating strings
Exam 3 (Q4) – 82.14%
Final Exam (Q4) – 56.25%
Average: ~ 70%
Met (Okay)

2. Algorithm-tracing techniques to ensure solution correctness including method calls.
Final Exam (Q8) – 75%
Final Exam (Q9) – 53.57%
Quiz 3 (Q1) – 79.41%
Quiz 5 (Q5) – 76.47%
Average: 71.11
Met (Okay)

3. Use testing and debugging strategies to identify software faults by creating test suites that include:
    a. Black-box test cases
    b. Basic white-box test cases

    Quiz 8 (Q2) – 66.67%

    Not Met
4. Use general software engineering principles, including abstraction and problem decomposition in problem and solution analysis.
Exam 1 (Q1) – 86.49%
Homework 2 (Q2) – 91.18%
Homework 2 (Q4) – 88.24%
Homework 4 (Q2) – 90.32%
Average: 89.1%
Met (Good)
5. Use informal pseudocode to describe algorithms.
Exam 1 (Q24) – 61.76%
Teamwork – 100%
Average: 80.88%
Met (Good)
6. Use 2D arrays.
Exam 2 (Q3) – 54.84%
Not Met
7. Apply Binary arithmetic to solve problems. This includes:
    a. Conversion between binary, decimal, and hexadecimal numbers
    Exam 2 (Q10) – 71.43%
    Homework 3 (Q2) – 74.19%
    Average: 72.81%
    Met (Okay)

    b. Application of arithmetic operations on binary and hexadecimal numbers

Exam 2 (Q10) – 71.43%
Met (Okay)

8. Use recursion for solving simple problems.
   Homework 3 (Q2) – 90.32%
   Homework 3 (Q3) – 87.1%
   Homework 3 (Q5) – 96.77%
   Average: 91.4%
   Met (Excellent)

9. Use linked lists.
   Final Exam (Q11) – 41.94%
   Quiz 8 (Q6) – 85.72%
   Quiz 8 (Q7) – 60.71%
   Average: 62.79%
   Not Met

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 including Booleans, integers, real numbers, characters, strings.
   Exam 1 (Q3) – 83.72%
   Exam 1 (Q5) – 30.23%
   Exam 1 (Q9) – 93%
   Exam 1 (Q14) – 97.67%
   Homework 1 (Q2) – 80%
   Quiz 1 (Q8) – 79.36%
   Average: 77.33%
   Met (Okay)

2. 1-D arrays
   Exam 2 (Q4) – 41.18%
   Final Exam (Q2) – 83.87%
   Final Exam (Q3) – 71.87%
   Homework 3 (Q1) – 96.77%
   Homework 3 (Q3) – 80.65%
   Average: 74.87%
   Met (Okay)

3. Assignment and arithmetic.
   Homework 1 (Q2) – 80%
   Met (Good)

4. Logical propositions to define conditional and loop statements.
   Exam 1 (Q16) – 69.77%
   Exam 2 (Q1) – 54.29%
   Final Exam (Q1) – 90.63%
   Final Exam (Q3) – 71.88%
   Homework 1 (Q3) – 75.68%

Average: 72.45%
Met (Okay)

5. For loops
   Exam 1 (Q17) – 60.47%
   Exam 1 (Q20) – 27.91%
   Exam 2 (Q1) – 54.29%
   Exam 1 (Q4) – 41.18%
   Final Exam (Q3) – 71.88%
   Homework 2 (Q2) – 91.18%
   Homework 2 (Q4) – 88.24%
   Homework 3 (Q1) – 96.77%
   Homework 3 (Q3) – 80.65%
   Homework 3 (Q5) – 87.1%
   Average: 70%
   Met (Okay)

6. While loops
   Final Exam (Q3) – 71.88%
   Average: 71.88%
   Met (Okay)

7. Methods/functions, parameter passing, return values
   Exam 2 (Q1) – 54.29%
   Exam 2 (Q4) – 41.18%
   Exam 2 (Q5) – 55.88%
   Exam 2 (Q6) – 58.1%
   Exam 3 (Q6) – 76.67%
   Exam 3 (Q7) – 40.74%
   Exam 3 (Q8) – 55.17%
   Homework 2 (Q2) – 91.18%
   Homework 2 (Q4) – 88.24%
   Homework 3 (Q2) – 90.32%
   Homework 3 (Q3) – 87.1%
   Homework 3 (Q5) – 96.77%
   Average: ~70%
   Met (Okay)

8. Algorithmic building blocks including
   a. Min
      Exam 2 (Q4) – 41.18%
      Not Met
   b. Max
      Homework 3 (Q2) – 90.32%
      Met (Excellent)
   c. Average
      Extra credit – 66.67%
      Not Met
   d. Summation
      Exam 3 (Q8) – 55.17%

Homework 3 (Q3) – 87.1%
Average: 71.14%
Met (Okay)

e. Linear Search
Final Exam (Q3) – 71.88%
Average: 71.88%
Met (Okay)

Analysis of the Results, and Instructor Recommendations

The results of the learning outcomes were all met with at least a satisfactory level, except for the following:

- **Level 1 (outcomes 4 and 8)**:
  - Topics related to outcomes 4 and 8 were discussed in detail.
    - However, I think they are not heavily assessed, and it may be a good idea to assess them in more than once as this may give students opportunities to demonstrate their mastery.
    - In future, more assessment should be made in this area.
- **Level 2 (outcomes 1.b.ii, 3, 6, and 9)**:
  - Topics related to outcomes 1.b.ii were discussed in good detail.
    - However, I think giving students more practice exercises (non-graded) can help them practice and master the material better.
  - Topics related to outcome 3 were discussed in good detail.
    - However, I think they are not heavily assessed in the lecture section, but students applied and used those concepts while working testing their lab assignments in their lab section.
    - In future, more assessment should be made in this area in lecture section and give students multiple opportunities to demonstrate their mastery.
  - Topics related to outcome 6 were discussed in good detail.
    - However, I think they are not heavily assessed, and it may be a good idea to assess them in more than once as this may give students opportunities to demonstrate their mastery.
    - In future, more assessment should be made in this area.
  - Topics related to outcome 9 were discussed in good detail.
    - However, I think giving students more practice exercises (non-graded) can help them practice and master the material better.
- **Level 3 (outcomes 8a and 8c)**:
  - Topics related to outcome 8a and 8c were discussed in good detail.
    - However, I think they are not heavily assessed in the lecture section, but students applied and used those concepts while working testing their lab assignments in their lab section.
    - In future, more assessment should be made in this area in lecture section and give students multiple opportunities to demonstrate their mastery.

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 and assess them more than once in lecture section.
2. Provide non-graded practice exercises for students to practice complex concepts more to enhance learning and demonstrate mastery of those by end of the course.

Course recommendations:

1. Develop clear and structured ways to measure and assess teamwork in the classroom.

Course Number: CS 1101
Course Title: Intro to Computer Science
Course Instructor: Bhanukiran Gurijala

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 some students dropped out before the course drop date. The table below summarizes the distribution of final letter grades that the remaining 40 students earned at the end of the semester.

Grade	Pass			Fail	
	A	B	C	D	F
No. of students	9/40	7/40	2/40	4/40	18/40
Percentage	22.5%	17.5%	5%	10%	45%
Total	45%			55%	

*C is a passing grade in CS 1101.

Failing Students
Students who received F in the course

- All students who failed the course are missing comprehensive labs and other lab submissions.

Students who received D grade in the course

- All students are missing some lab assignments submissions and/or performed poorly on comprehensive labs.

Learning Outcomes

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

For each outcome, I measured the average score obtained by students on the assessment items used to assess the outcome. 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. The percentage does not include student grades who did not submit an assessment item and/or did not attempt a particular question in an assignment or quiz.

Learning Outcomes

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:

12. Analyze problems, design and implement solution algorithms, including correct use of:
    a. Simple I/O operations (reading from and printing to the terminal)
       Lab 3 – 94.44%
       Lab 4 – 97.14%
       Lab 5 – 91.43%
       Comprehensive Lab 1 – 88.89%
       Average: 92.98%
       Met (Excellent)
    b. User-defined types and their implementation as classes
       Comprehensive Lab 3 – 81.81%
       Exam 3 Review – 92.31%
       Average: 87.06%
       Met (Good)
    c. Basic string manipulation techniques using language functions, including:
        i. Traversing strings,
           Comprehensive Lab 2 – 72%
           Lab 11 – 95.45%
           Lab 12 – 96.3%
           Average: 87.92%
           Met (Good)
        ii. Accessing characters,
           Exam 3 Review – 92.31%
           Lab 11 – 95.45%
           Lab 12 – 96.3%
           Average: 94.69%
           Met (Excellent)
        iii. Comparing strings,
           Comprehensive Lab 1 – 88.89%
           Exam 3 Review – 92.31%
           Lab 11 – 95.45%
           Lab 12 – 96.3%
           Average: 93.24%
           Met (Excellent)
        iv. Concatenating strings
           Lab 4 – 97.14%
           Lab 5 – 91.43%
           Lab 8 – 86.11%
           Lab 9 – 68.97%
           Average: 85.91%
           Met (Good)
13. Algorithm-tracing techniques to ensure solution correctness including method calls.
    Lab 12 – 96.3%
    Met (Excellent)
14. Use testing and debugging strategies to identify software faults by creating test suites that include:
    a. Black-box test cases
    b. Basic white-box test cases

    Not Assessed.
15. Use general software engineering principles, including abstraction and problem decomposition in problem and solution analysis.
    Lab 4 – 97.14%

Lab 5 – 91.43%
Lab 8 – 86.11%
Lab 9 – 68.97%
Average: 85.91%
Met (Good)

16. Use informal pseudocode to describe algorithms.
    Comprehensive Lab 1 – 88.89%
    Lab 4 – 97.14%
    Lab 5 – 91.43%
    Lab 6 – 96.88%
    Average: 93.59%
    Met (Excellent)
17. Use 2D arrays.
    Comprehensive Lab 3 – 81.81%
    Quiz 3 – 69.23%
    Average: 75.52%
    Met (Okay)
18. Apply Binary arithmetic to solve problems. This includes:
    a. Conversion between binary, decimal, and hexadecimal numbers,
       Quiz 3 – 69.23%
       Lab 11 – 95.45%
       Average – 82.34%
       Met (Good)
    b. Application of arithmetic operations on binary and hexadecimal numbers
       Not Assessed
19. Use recursion for solving simple problems.
    Lab 9 – 68.97%
    Lab 12 – 96.3%
    Quiz 2 – 45.71%
    Exam 3 Review – 92.31%
    Average: 75.82%
    Met (Okay)
20. Use linked lists.
    Comprehensive Lab 3 – 81.81%
    Met (Good)
21. Instead of IDEs, use a command line interface (terminal) to compile and execute programs.
    Comprehensive Lab 1 – 88.89%
    Lab 4 – 97.14%
    Lab 8 – 86.11%
    Lab 9 – 68.97%
    Average: 85.28%
    Met (Good)
22. Use teamwork roles and strategies in the classroom.
    Lab 6 – 96.88%
    Lab 12 – 96.3%
    Average: 96.59%
    Met (Excellent)

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 high-level language to express solutions to programming problems, including the pseudocode correct use of:

10. Basic variable types including Booleans, integers, real numbers, characters, strings,
    Comprehensive Lab 2 – 72%
    Lab 6 – 96.88%
    Lab 11 – 95.45%
    Lab 12 – 96.3%
    Average: 90.16%
    Met (Excellent)
11. 1-D arrays
    Comprehensive Lab 2 – 72%
    Exam 3 – 92.31%
    Lab 10 – 77.78%
    Average: 80.7%
    Met (Good)
12. Assignment and arithmetic
    Comprehensive Lab 2 – 72%
    Lab 4 – 97.14%
    Lab 5 – 91.43%
    Lab 8 – 86.11
    Average: 86.67%
    Met (Good)
13. Logical propositions to define conditional and loop statements.
    Lab 5 – 91.43%
    Lab 6 – 96.88%
    Lab 8 – 86.11%
    Lab 9 – 68.97%
    Average: 85.85%
    Met (Good)
14. For-loops
    Comprehensive Lab 2 – 72%
    Lab 6 – 96.88%
    Lab 10 – 77.78%
    Lab 11 – 95.45%
    Average: 85.53%
    Met (Good)
15. While-loops
    Comprehensive Lab 2 – 72%
    Lab 6 – 96.88%
    Lab 9 – 68.97%
    Lab 11 – 95.45%
    Average: 83.33%
    Met (Good)
16. Methods/functions, parameter passing, return values.
    Lab 8 – 86.11%
    Lab 9 – 68.97%
    Lab 10 – 77.78%
    Lab 11 – 95.45%
    Average: 82.08%
    Met (Good)
17. Algorithmic building blocks including:

a. Min
   Lab 10 – 77.78%
   Met (Okay)
b. Max
   Lab 10 – 77.78%
   Met (Okay)
c. Average
   Lab 6 – 96.88%
   Lab 10 – 77.78%
   Average: 87.33%
   Met (Good)
d. Summation
   Comprehensive Lab 1 – 88.89%
   Lab 4 – 97.14%
   Quiz 2 – 45.71%
   Average – 77.25%
   Met (Okay)
e. Linear search
   Comprehensive Lab 2 – 72%
   Exam 3 Review – 92.31%
   Average: 82.16%
   Met (Good)

18. Coding and documentation standards
    Comprehensive Lab 2 – 72%
    Lab 5 – 91.43%
    Average: 81.72%
    Met (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 the following:

- **Level 2 (outcomes 3a, 3b, and 7b)**:
  - Topics related to outcomes 3a and 3b were discussed in good detail in class and lab sections.
    - However, they were not formally assessed. Students used the black-box and white-box testing techniques to check their lab assignments and comprehensive labs for correctness of their implementation.
    - In future, it may be a good idea to ask students to come up with test cases that they used to test their program as one of the deliverables along with their source code.
  - Topics related to outcome 7b were discussed in good detail in class and lab sections.
    - However, I think they are not formally assessed in the lab section.
    - In future, assessment should be made in this area in lab section and give students multiple opportunities to demonstrate their mastery.

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 to the end of the course.

Recommendations

1. Formally assess introduced and used testing methodologies and assess them more than once. Include submitting test cases into lab grading rubrics.
2. Assess arithmetic operations on binary and hexadecimal numbers and not just conversions.

CS 1301/1101 – Intro Computer Science, Spring 2023 CQI Report

Course Description and Outcomes

Course Description

Students will learn to be active learners and understand the motivations for computing, basic concepts of algorithms, basic computer organization, and the 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. 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.

Textbook:

"Revel Introduction to JAVA Programming and Data Structures, 12 e".

2020 Pearson, Daniel Lang. Platform: Revel. 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 describe, at a high level:

1) The major advances in the history of computing
2) The relation between computing and society, including social, ethical, and legal issues
3) The importance of computing in a variety of professions: required knowledge and skill sets for major career options
4) Classes of programming languages, including:
    a) Imperative,
    b) Object-oriented,
    c) Declarative, and
    d) Functional
5) The purpose of multi-dimensional arrays (dimension 3 and above)
6) The purpose of and the relationship between classes and objects
7) The purpose of pre/post conditions, in particular as related to verification
8) Compilation and interpretation

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. Analyze problems, design and implement solution algorithms, including correct use of:

   a. Simple I/O operations (reading from and printing to the terminal)
   b. User-defined types and their implementation as classes
   c. Basic string manipulation techniques using language functions, including:
2. Traversing strings,
      i. Accessing characters,
      ii. Comparing strings,
      iii. Concatenating strings
3. Algorithm-tracing techniques to ensure solution correctness, including method calls
4. Use testing and debugging strategies to identify software faults by creating test suites that include:
   a. Black-box test cases
   b. Basic white-box test cases
5. Use general software engineering principles, including abstraction and problem decomposition in
6. problem and solution analysis
7. Use informal pseudocode to describe algorithms
8. Use 2D arrays
9. Apply Binary arithmetic to solve problems. This includes:
   a) Conversion between binary, decimal, and hexadecimal numbers,
   b) Application of arithmetic operations on binary and hexadecimal numbers
10. Use recursion for solving simple problems
11. Use linked lists
12. Instead of IDEs, use a command line interface (terminal) to compile and execute programs.
13. Use teamwork roles and strategies in the classroom

Level 3 Outcomes: 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 high-level language to express solutions to programming problems, including the correct use of:

1. Basic variable types including Booleans, integers, real numbers, characters, strings,
2. 1-D arrays
3. Assignment and arithmetic
4. Logical propositions to define conditional and loop statements
5. For-loops
6. While-loops
7. Methods/functions, parameter passing, return values
8. Algorithmic building blocks include:
   a) Min
   b) Max
   c) Average
   d) Summation
   e) Linear search
9. Coding and documentation standards

Course Number: CS 1301
Course Title: Introduction to Computer Science
Course Instructor: Maria G. Jimenez
Instructional Assistant: Ahnaf Farhan (20 hours/week) Summary of Outcomes Coverage and Attainment

CS 1301

All outcomes were covered and assessed in two sections of CS1301. Sections no 1 and 2:

Nearly the outcome 1.7 was covered and assessed in two sections of CS1301.

- All the outcomes, except the 1.7, were assessed through quizzes and exams. Some outcomes were also assessed through graded assessment.

As an instructor, I am satisfied with the success of the outcome results for the course.

CS 1101

Nearly all outcomes were covered and assessed in two sections of CS1101. Sections 1 and 2:

- All the outcomes, except the 1.3, 1.7 and 2.13, were assessed through quizzes and exams. Some outcomes were also assessed through graded assessment.
- For Outcome 2.13, the instructor asked students to work in teams in the classroom.
  For outcome 1.3, the class watched some videos related to the topics, and we discussed the topic in the classroom.

As an instructor, I am satisfied with the success of the outcome results for the course.

Students' Success: Passing and Failure Rates

CS 1301-15666						
A	B	C	D	F	Total Earned Grade	W
22	8	4	2	7	45	1
50%	18.8%	9.09%	4.55%	16.28%	100%	
34			9			
=79.07%			20.93%			

CS 1301-20440						
A	B	C	D	F	Total Earned Grade	W
3	3	3	1	5	15	0
20%	20.0%	20.0%	6.67%	33.33%	100%	
9			6			
=60.00%			40.00%			

Observations:

1. Regarding outcomes 3.5, 3.6, and 3.7, some students had difficulty implementing loops and recursion methods. They performed well when solving simple problems showing a better understanding of the implementation of 1-D arrays than 2D arrays.
2. Students had more difficulty understanding how to return than passing parameters.
3. Working in teams works better for students than working individually.
4. Outcome 1.6 was also barely met. Students understood the concepts but had difficulty implementing classes based on a given problem. More practice was required.
5. Asking students to write an algorithm or pseudocode of any problem before writing a java code was useful for most of the students.
6. The main reasons students failed the course were:
    a. Many absences because of illness conditions (mental health, stress, covid19, and accidents), work, and family responsibilities. Then, they did not have enough time to understand the material.
    b. Students arriving late to class (no responsibility to arrive on time because of the early morning schedule).
    c. A low number of students attending office hours or asking for help.
    d. Some students did not purchase the textbook.
7. The at-risk students were emailed and tracked, but they did not show interest in getting help until the last month.
8. The background of the students varies. Most graduate students with no background in Computer Science get much stress because of their research duties, partial work, and a high number of credit hours.
9. Having a tentative schedule of the activities was useful.
10. Most of the students required more time to complete comprehensive labs. We noticed that some students started working on them the last week.
11. The TA teaching style in one of the Lab sections was not so successful. Some students had difficulty understanding how to transfer their algorithm to Java code. Again, the number of absences and punctuality were relevant factors for understanding.
12. Getting the TA to update the grades on time was hard. Also, he did not assess demonstrations of comprehensive labs despite my instructions.
13. I have one TA as a grader to grade the two courses, only one instructional assistant to assist with one lecture section, and one instructional assistant that supports my two lab sections.

Recommendations

1. Implement better strategies to improve students' understanding of loops, methods, and classes.
2. Ask students to provide feedback about what did and did not work in the classroom to improve our teaching style.
3. Have more hands-on activities.
4. Increment teamwork activities.
5. Include more testing strategies.
6. Improve the teaching style in Lab sections.

7. As an instructor, improve my leadership on Teaching Team and supervise closer the activities done in the lab. Be more supportive.
8. Implement a better strategy to help students with recurrent absences. Make-up assignments and exams are not enough sometimes.
9. Drop students with more than consecutive unexcused absences.
10. Improve the strategy to assess the learning progress in the lab.
11. Develop a strategy to assess teamwork.
12. Work with the teaching team to implement more effective weekly reviews.
13. Provide training to Teaching Assistants to support the instructor, track at-risk students, and collect evidence for ABET.
14. Provide training to newbie instructors about how is expected to teach the courses and how deeply the concept should be covered. Mainly core courses. Also, explain about the CQI report to keep in consideration the way to assess each outcome.

CS 2401

CS 2401

A summary of the overall observations and recommendations (in blue) are below. Specific observations from each section are included after.

Outcomes. All outcomes were met in all sections of the course.

Textbook. All 3 sections of the course used a new textbook this round, though with three very different approaches: (1) required with assigned homework, (2) optional but with a single recommended text, and (3) no specific textbook recommended, just something relevant. In all cases, this change seemed to be an improvement from the previous use of a zyBook. Continued disuse of the zyBook is recommended.

Review. All instructors noted the needed to heavily review CS1 concepts. While this is typical, this is a theme that occurs on a regular basis. It is highly recommended that instructors of CS2 begin doing review sessions outside class time early in the semester, as this tends to bring everyone up to skill level before covering new material.

DF Rate. Compared to previous semesters, the DF rates seem to be about the same, being between 13% and 24%, compared to the about 18% in 2020. A summary can be found in Table 1.

Table 1: Summary of grades

	A	B	C	D	F
Akbar	21	11	6	3	3
Ceberio/Gamez	27	9	3	12	1
DeBlasio	6	17	10	4	4
Total	54 39.4%	37 27.0%	19 13.9%	19 13.9%	8 5.8%
	Pass 80.3%			Did not pass 19.7%	

Course Outcomes: CS 2401, CRN: 18415
Fall 2022

February 11, 2023

Course Number: CS 2401
Course Title: Elementary Data Structures and Algorithms
Course Instructor: Monika Akbar
Instructional Assistant: Anindita Nath (20 hours/week)

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.

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.

The course provides a review and deeper study of arrays, objects, linked lists, and recursion; introduction to algorithm analysis and rigorous study of searching and sorting algorithms; and new data structures, such as, binary trees (including binary search trees), stacks, and queues, along with their implementations.

This report describes the outcome section taught by Monika Akbar.

Grade Distribution

The class had 50 students at the beginning of the semester. Along the way, six students dropped out. The following table shows the summary of the final grades of the 44 students.

Grade	Pass				Fail
	A	B	C*	D	F
No. of students	21/44	11/44	6/44	3/44	3/44
Percentage	47.7%	25%	13.6%	6.8%	6.8%
Total	86.36%			6.8%	6.8%

*C is the passing grade.

Assessment Method

The formal assessment was done using midterms, final exam, and lab assignments. The course had seven graded in-class activity/quiz, five homework, 3 midterms, one final exam, and 10 lab assignments. This report lists the average score for exam questions (midterm or final) or lab assignments.

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:

1. Demonstrate an understanding of the features of object-oriented languages (related to the implementation of data structures discussed in other objectives)

   Met (Midterm 1, q11, 96%; final exam, q1, 97.5%,)

2. Articulate the notion of average-case time complexity.

   Met (Final exam, q3, 95%)

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 algorithms for:

   a. Searching: linear and binary search.
   Met (Lab assignment 7, 84.2%)

   b. Sorting: merge sort, quicksort, and at least one quadratic sorting algorithm.
   Met (Midterm 3, q7, 76.2%; final exam, q18, 80.7%)

2. Use basic notions of algorithm complexity:

   a. Use Big-O notation to describe the running time and memory requirements for an algorithm.
   Met (Midterm 3, q1, 76%)

   b. Determine the best- and worst-case behaviors of a simple algorithm.
   Met (Final exam, q18, 80.7%)

3. Use standard problem-solving techniques such as: problem decomposition, iteration, and recursion.
   Met (Midterm 2, q8, 76.96%, q9, 95.75%, q10, 93.62%)

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 identify, implement and use the following data structures as appropriate for a given problem:

1. Implement and use the following data structures to solve computational problems:

   a. Linked lists
   Met (Final exam, q17, 78.2%)

   b. Binary trees as linked structures
   Met (Lab assignment 10, 84.55%)

   c. Binary search trees as linked structures
   Met (Final exam, q7, 83.93%)

   d. Stacks
   Met (Final exam, q10, 78.83% (average of the 3 versions of the question))

   e. Queues

Met (Final exam, q9, 87.9%)

2. Trace code that uses:

a. Strings
Met (Lab assignment 4, 85.86%)

b. Single-value data types
Met (Lab assignment 1, 94.57%)

c. 1D and 2D arrays
Met (Final exam, q19, 90.3%)

d. Reference-based data structures
Met (Lab assignment 2, 87.02%)

e. Iterative methods
Met (Lab assignment 5, 87.02%)

f. Recursive Methods
Met (Lab assignment 6, 96.9%)

g. Primitive vs. non-primitive data types
Met (Midterm 1, q7, 71.43%, midterm 2, q4, 79.15)

Observations

- Reviewing CS1 concepts takes a significant amount of time at the beginning of the semester. This was during a Fall semester. It is unclear whether the learning loss is due to the long break between a Spring and a Fall semester.
- Having one TA and two undergraduate TAs made it easier to provide better assistance to students.
- Surveys taken during the lab revealed that students struggle to convert their understanding of a concept into codes. Students mentioned a number of ways to overcome the struggle, including talking to someone from the instructional team (i.e., TA, IA, instructor), talking to their peers, and searching online.
- We used Pearson Revel ebook in class for the first time. We assigned reading homework from the book. The book can be integrated with Blackboard, making it easier to sync student grades.
- The classroom was not suitable for group activities or in-class activities. There were rows of chairs with one big table stretched from one side of the room to another. The setting made it difficult to see or provide help to students who sat in the middle.

Recommendations:

- TA/IAs should hold CS1 review sessions early in the semester, reviewing the concepts students identified as needing review.
- Such review sessions should be held regularly to help students who fall behind on certain CS2 topic.
- The Pearson ebook costs more than zybook, but it has better content coverage and longer access time (one year compared to six months for zybook). This ebook is a better alternative than zybook, at this moment.

CS2401 Elementary Data Structures, Fall 2022 CQI Report

Martine Ceberio (UTEP) & Marlon Gamez (Google)

Course Approach

The course was taught, entirely in person, by two co-instructors as part of the Google Engineer In Residence (GIR) program. Additionally, one graduate Teaching Assistant and 2 Instructional Assistants (undergraduate TAs) were part of the instructional team.

MS Teams: Although lectures were held in person, all students had access to an MS team for our course. The lecture notes were written on a OneNote notebook that was accessible to all students through the MS team of the course. Announcements were shared on our team, which made the course more lively and students more engaged.

Textbook: The instructional team of all CS fundamental courses instructors decided to use a new textbook from Pearson starting in Fall 2022. Marlon Gamez and I decided to assign homework following the structure of the Pearson book, but not requiring that students actually use this book. All students did have to have a textbook and we had to accept it: some used the Pearson book, others used other books, which we checked and accepted as equivalent. As a result, assigned homework was on a topic, not on a specific set of pages from a given book. Students' homework was not graded on their completion of the homework, but as it showed on their performance on quizzes for instance.

Office hours: Each member of the instructional team held regularly scheduled office hours weekly. Although the instruction was entirely in person, 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).

Written examinations: We gave 2 exams instead of the usual 3 from previous semesters, and one final exam. Quizzes were given weekly and make-up quizzes offered to help students gain back some points for each exam (for a total of 13 quizzes).

Lab assignments: Students had to complete 10 lab assignments over the course of the semester. They received support for their labs during closed lab hours (3 hours a week) and outside of class/lab during office hours of anyone in the instructional team. Given that the course was co-taught by a Googler in Residence, students had access to a large number of office hours, spanning pretty much the whole week (aside from scheduled labs and lectures).

Weekly workshops: Each week, the instructional team offered workshops on topics best suited to support the students' learning and success at that time in the semester. Topics were usually discussed at our weekly team meetings.

Attendance / Participation / Communication: We expected students to attend lectures and labs sessions. We took attendance but did not count attendance towards the final grade: instead, we used attendance (including lateness or leaving early) to detect any issue students may have and to be able to offer help or mentoring to students about strategies in attending school. Students were usually appreciative of our care.

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 or exam question grade would be replaced for the better performance. This approach to grading was meant to provide a failure-safe environment since experimenting and failing are part of the learning process. We believe that students did not drop because they knew they could still improve their grades and did not give up.

Observations and Instruments Mapping to Outcomes

Participation / DFW:

There were 53 students at the very start of the semester. 2 dropped during the first week, and 2 more joined (one of them simply auditing). 10 were female students (18.9%).

- No student dropped beyond the first week of classes (which was well before census day)
- Out of the 53 students on the roster until the end of the semester and who obtained a grade:
    - 1 obtained an F because he stopped attending in late October and did not respond to any of our attempts to contact him and offer our help: he did not drop though.
    - 12 students (2 of them female) who attended the course until the end of the semester failed the course with a D.
    - 39 students passed the class (73.6%). Out of these:
        - 27 A's: (23 male/4 female students) – 51%
        - 9 B's: (6 male/3 female students) – 17%
        - 3 C: (1 female student) – 5.7%

The DFW rate for this course is: 24.5%, which is about 10 points lower than in Fall 2020.

A	B	C	D	F	Drop
27	9	3	12	1	0
51%	17%	5.7%	22.6%	1.9%	0%

Assessment instruments:

Students were evaluated throughout the semester via quizzes, two mid-term exams, one comprehensive final exam. Students were also evaluated based on their successful completion of lab assignments (10 lab assignments).

The mapping of instruments to outcomes is shown below:

	EXAMS	EXAMS																						
		FINAL EXAM									EXAM 2							EXAM 1						
		Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q1	Q2.1	Q2.2	Q2.3	Q3	Q4	Q5	Q1	Q2	Q3.1	Q3.2	Q4	Q5	Q6
Level 1: Knowledge and Comprehension																								
1/ Demonstrate their understanding of the features of object-oriented languages (related to the implementation of the Data Structures discussed through this course)																								
2/ Articulate the notion of average-case time complexity																								
Level 2: Application and Analysis																								
1/ Describe, implement, and use the following algorithms for:																								
a/ Searching: linear and binary search	89	86			92													77	75	93	75			
b/ Sorting: merge sort, quick sort, and at least one quadratic sorting algorithm	92				92																	77		
2/ Use basic notions of algorithm complexity:																								
a/ Use Big-O notation to describe the running time and memory requirements of an algorithm	92				92															93	75			
b/ Determine the best- and worst-case behaviors of simple algorithms	92				92															93	75			
3/ Use standard problem-solving techniques such as: problem decomposition, iteration, recursion.	88	86	89	89	92	84					67							77	75				77	79
Level 3: Synthesis and Evaluation																								
1/ Implement and use the following data structures to solve computational problems:																								
a/ Linked lists	93.67			89	92		100				67				47									
b/ Binary trees as linked structures	85.33					84				76		96												
c/ Binary search trees as linked structures	89.00					84							92	91										
d/ Stacks	75.50								75	76							51							
e/ Queues	75.50								75	76							51							
2/ Trace code that uses:																								
a/ Strings	88.00	86	89	89							67					74		77					77	79
b/ Single-value data types	84.80			89			100	84	75	76								77	75					
c/ 1D and 2D arrays	85.50	86		89	92				75								51	77	75			77		
d/ Reference-Based Structures	84.80			89		84	100		75	76	67	96	92	91	47		51							
e/ Iterative methods	86.33	86		89	92		100		75	76					47		51	77	75					
f/ Recursive methods	85.67		89			84		84			67					74							77	79
g/ Primitive vs non-primitive data types	89.00			89																				

	QUIZZES	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11
Level 1: Knowledge and Comprehension												
1/ Demonstrate their understanding of the features of object-oriented languages (related to the implementation of the Data Structures discussed through this course)	1								1			
2/ Articulate the notion of average-case time complexity	1							1				
Level 2: Application and Analysis												
1/ Describe, implement, and use the following algorithms for:												
a/ Searching: linear and binary search	6					1	1	1	1		1	1
b/ Sorting: merge sort, quick sort, and at least one quadratic sorting algorithm	6			1	1	1	1				1	1
2/ Use basic notions of algorithm complexity:												
a/ Use Big-O notation to describe the running time and memory requirements of an algorithm	5					1	1	1			1	1
b/ Determine the best- and worst-case behaviors of simple algorithms	5					1	1	1			1	1
3/ Use standard problem-solving techniques such as: problem decomposition, iteration, recursion.	11	1	1	1	1	1	1	1	1	1	1	1
Level 3: Synthesis and Evaluation												
1/ Implement and use the following data structures to solve computational problems:												
a/ Linked lists	2							1	1			
b/ Binary trees as linked structures	1									1		
c/ Binary search trees as linked structures	1										1	
d/ Stacks	0											
e/ Queues	0											
2/ Trace code that uses:												
a/ Strings	4	1	1	1	1							
b/ Single-value data types	8	1	1	1	1	1	1			1		1
c/ 1D and 2D arrays	6		1	1	1	1	1					1
d/ Reference-Based Structures	2							1	1			
e/ Iterative methods	8	1	1	1	1		1	1	1	1		
f/ Recursive methods	0											
g/ Primitive vs non-primitive data types	1								1			

		LABS									
		Lab 1	Lab 2	Lab 3	Lab 4	Lab 5-6	Lab 7-8	Lab 9-10	Lab 11	Lab 12-13	Lab 14-15
Level 1: Knowledge and Comprehension											
1/ Demonstrate their understanding of the features of ol	93.3	1	1	1	1	1	1	1	1	1	1
2/ Articulate the notion of average-case time complexity	91			1							
Level 2: Application and Analysis											
1/ Describe, implement, and use the following algorithms for:											
a/ Searching: linear and binary search	86						1				
b/ Sorting: merge sort, quick sort, and at least one quadr	90.5					1	1				
2/ Use basic notions of algorithm complexity:											
a/ Use Big-O notation to describe the running time and n	91			1							
b/ Determine the best- and worst-case behaviors of sim	91			1							
3/ Use standard problem-solving techniques such as: pr	93.3	1	1	1	1	1	1	1	1	1	1
Level 3: Synthesis and Evaluation											
1/ Implement and use the following data structures to solve computational problems:											
a/ Linked lists	92.5							1			1
b/ Binary trees as linked structures	95								1		
c/ Binary search trees as linked structures	90									1	
d/ Stacks	95										1
e/ Queues	95										1
2/ Trace code that uses:											
a/ Strings	92.875	1	1	1	1		1	1	1	1	
b/ Single-value data types	93.3	1	1	1	1	1	1	1	1	1	1
c/ 1D and 2D arrays	93.286	1	1	1	1	1	1	1			
d/ Reference-Based Structures	93.4		1					1	1	1	1
e/ Iterative methods	93.111	1	1	1		1	1	1	1	1	1
f/ Recursive methods	93.75				1				1	1	1
g/ Primitive vs non-primitive data types	93.667		1			1		1	1	1	1

So we observe that we covered and met all outcomes in assessment tools (labs, quizzes, exams).

Analysis by Instructors and Recommendations

What do we think about the DFW? Why? The DFW rate of 24.5%, which is lower than in Fall 2020. In particular, it is notable that we had no drop and only 1 student stopped attending. We believe that these results are due to our grading philosophy that encourages students to focus on their learning even while they are still struggling a little. We see that the number of As is high too: this is the result of students applying themselves despite early struggles.

Were outcomes met? All outcomes were met. For some of the outcomes, the percentage of attainment is around 75% on exams: the two outcomes that were only met at 75% are stacks and queues, which are the last two topics covered in this class. On the other hand, students demonstrated mastery in these topics in lab, achieving 95% for both topics.

Recommendations:

- We found that not enforcing a specific textbook was not detrimental to the learning, perseverance, and performance of our students in the course. We suggest exploring this strategy further.
- We found that students liked having access on MS Teams after class to the lectures put together interactively during the class. We suggest using either this tool (MS Team and OneNote) or another that allows students to have access to lectures after class.

Reflections on Recommendations from Fall 2020

Assignment Structure

We moved away from short labs and a few longer ones. We focused on important topics and even gave the students a breather around week 10.

Textbook

"We are not convinced of the efficacy of the ZyBooks homework assignments." "**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." → We tested an approach with no prescribed textbook, with success.

Grading

"We suggest that this approach be more widely considered in the fundamentals at least." → We kept using the same grading strategy. To the best of our knowledge, it is not widely adopted.

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." → Early labs and quizzes were meant to diagnose students' issues as early as possible.

Subject Matter

Java vs Algorithms. I don't think we do a great job at teaching students Java tools as we focus more on the CS aspect of what we teach, independently from the language. I do not have a solution about how to proceed, yet.

Time complexity. "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." → Students were given a lot of exercises about tracing for time complexity (in quizzes and exams, as well as in class during lecture)

Unit Testing. "The recommendation would be to begin the class by exposing unit tests to students." → This semester, we felt that students would be overwhelmed if they had to write tests right away. We pushed testing a little later in the semester.

CS 2401: Elementary Data Structures and Algorithms

Dan DeBlasio

Fall 2022

1 Course Overview and Approach

Detailed learning outcomes are shown in Table 1 and correspond to the labels in the later alignments. This year we used a new book from Pearson publishing, though in my section it was optional and anecdotally not many students purchased it.

The course utilized of 4 major resources:

- **blackboard** — used primarily as a one-directional information system for the instructional staff to post announcements and grades.
- **replit** — an online IDE that allowed for the integration of assignment dissemination, submission, unit testing, and grading.
- **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 online office hours are conducted.
- **prerecorded content** — videos posted by the instructor that correspond to the topics covered in class, and utilizing the same visuals; originally developed while teaching this course online the videos are available on demand to all students to use as review.

2 Assessment Instruments

The two primary sources of assessment came from exams (4 mid-term "assessments", and an optional final, the word assessment was used in place of test as an experiment) and hands-on labs (12 labs aligning to learning objectives). Smaller contributors not included in this report were: in-class quizzes and participation. These smaller assessments were not graded for correctness, the participation quizzes are used in place of taking attendance.

If a student fails to receive a "C" on either the lab component (lab assignments and participation) or the final exam (or in this case the average of assessments in its place), a student will receive a "D". This semester no students fell into this category.

2.1 Exams

All exams were graded by the instructor, feedback was given individually physically. Additionally, a postmortem discussion was provided in class after each exam.

Delivery method Due to exam security issues in previous semesters several tactics were used to ensure assessment was taken of the individual student and no one else:

Table 1: Detailed Outcome Numbering Used

1.1	Demonstrate their understanding of the features of object-oriented languages (related to the implementation of the Data Structures discussed through this course)
1.2	Articulate the notion of average-case time complexity
Describe, implement, and use the following algorithms for:	
2.1.a	Searching: linear and binary search
2.1.b	Sorting: merge sort, quicksort, and at least one quadratic sorting algorithm
Use basic notions of algorithm complexity:	
2.2.a	use Big-O notation to describe the running time and memory requirements of an algorithm
2.2.b	determine the best- and worst-case behaviors of simple algorithms
2.3	Use standard problem-solving techniques such as: problem decomposition, iteration, and recursion.
Design and implement solutions to computational problems using the following data structures:	
3.1.a	Linked lists
3.1.b	Binary trees as linked structures
3.1.c	Binary search trees as linked structures
3.1.d	Stacks
3.1.e	Queues
Trace code that uses:	
3.2.a	Strings
3.2.b	Single-value data types
3.2.c	1- and 2-dimensional arrays
3.2.d	Reference-based structures
3.2.e	Iterative methods
3.2.f	Recursive methods
3.2.g	Primitive vs. non-primitive types

- assessments were given in person, on paper, in the classroom
- some, but not all, assessments were shuffled in order
- assessments were not printed until the day of the exam, and each was printed with a unique identifying number
- assessments were close book

In addition, to help ensure unbiased grading, all assessments were separated and graded with only the number available to the grader. The front page of the assessments was the only place the student's name was present.

Scores from the mid-term assessments, and their alignment to the outcomes are listed in Table 5. Alignment and scores for the final exam, optional if the average assessment grade was above a "C", are shown in Table 2.

2.2 Laboratories

Labs were assigned on Mondays and typically due the following Monday before class (an extension from the previous Sunday at Midnight used in the past). 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 `repl.it`. Labs were graded by the TA and IAs for the course given a rubric provided by the instructor. Labs 7 and 10 were group assignments (given the weeks of assessments).

3 Analysis and Observations

Initially 46 students registered for the course, 41 were enrolled at the time of the final exam. The final letter grades assigned and scores needed for each letter are shown in Table 4.

Table 2: Concepts on Final Exam

Objective	Times	All	Passed	Question	1	2	3	4	5	6	7	8	9	10
				Result	4.38	4.65	10.73	13.46	5.00	7.85	12.15	1.72	14.73	10.92
					9.00	9.00	18.00	25.00	7.00	12.00	16.00	4.00	25.00	18.00
1.1	0													
1.2	1	65.38%	65.97%							1				
2.1.a	2	67.31%	69.36%				1				1			
2.1.b	2	58.24%	60.32%		1					1				
2.2.a	2	59.52%	61.31%			1				1				
2.2.b	1	51.71%	55.09%			1								
2.3	3	61.13%	62.50%					1			1		1	
3.1.a	1	59.62%	63.43%				1							
3.1.b	1	60.68%	63.43%											1
3.1.c	1	60.68%	63.43%											1
3.1.d	1	53.85%	56.67%					1						
3.1.e	1	60.68%	63.43%											1
3.2.a	2	61.10%	61.27%						1			1		
3.2.b	2	69.38%	69.53%								1	1		
3.2.c	2	56.22%	57.58%		1								1	
3.2.d	3	58.77%	61.17%					1	1					1
3.2.e	5	57.30%	59.62%				1	1				1	1	1
3.2.f	1	59.62%	63.43%				1							
3.2.g	2	57.69%	59.90%					1	1					

Notes: – Final was options, and includes only students who needed to take the final to pass or wanted to increase their final grade.
– The "all" and "passed" columns are the weighted total score of the questions for that objective of all students, and those who passed the course.
– Objectives with scores lower than a C are marked in yellow.

Table 3: Concepts on Labs

Objective	Times	Score	0	1	2	3	4	5	6	8	9	11	12
1.1	11	52.76%	1	1	1	1	1	1	1	1	1	1	1
1.2	0	—											
2.1.a	2	56.75%				1					1		
2.1.b	1	72.00%						1					
2.2.a	0	—											
2.2.b	1	72.00%						1					
2.3	10	57.68%		1	1	1	1	1	1	1	1	1	1
3.1.a	1	72.5%									1		
3.1.b	1	51.61%										1	
3.1.c	1	51.61%										1	
3.1.d	0	—											
3.1.e	1	61.25%											1
3.2.a	3	56.67%		1			1		1				
3.2.b	0	—											
3.2.c	3	65.33%		1	1			1					
3.2.d	2	48.31%								1		1	
3.2.e	3	56.67%		1	1				1				
3.2.f	3	54.87%					1		1			1	
3.2.g	5	57.40%		1	1			1	1	1			

Notes: – Labs 7 & 10 were group assignments and are not included.
– A lab 13 was given as an optional lab to emphasize missed concepts.

Table 4: Letter Grades Assigned

Letter	Cutoff	Count
A	$\geq 87\%$	6
B	$\geq 77\%$	17
C	$\geq 62\%$	10
D	$\geq 50\%$	4
F		4
W		5
I		0

The DF rate was 19.51% (from the students who did not withdraw), and the DFW rate was 28.26% (counting those who completed the welcome survey).

Note that the threshold used to decide a student passed the lab was decided to be a 59%. Typically I move the bars for letter grades based on the histogram of scores of students, in most situations a clear quad-modal distribution (or close to) appears as long as enough students are enrolled where separations can be made, this allows me to adjust to the difficulty of assessments in any given semester.

3.1 Observations

1. With the adjusted outcomes list developed between the 2022 CQI and now, we better met the outcomes list and had positive scores in most cases.
2. The lowest scored outcomes seems to mostly be an accounting issue. While there were only a couple questions that directly focused on recursive and iterative methods those concepts are generally included in other questions.
3. In general, lab scores were lower than test scores for topics covered in deeper detail. This may also be an issue with terminology, for instance in outcome 3.2, the phrase is "trace code" but my attribution on labs was to "write code" which *includes* tracing logically, but goes further. Since its difficult to extract those two concepts from the lab grade this is likely an influence.
4. In general, all concepts tested at or near passing on tests, since my test scores tend to be low. A few concepts were slightly lower than passing on average for those who moved into CS3.
5. The idea of having the asynchronous videos available to students was positively received, though as we are back in-person are under utilized.
6. Using participation quizzes in place of attendance was not well received by the students, but was helpful.
7. Strict policies on testing resulted in 0 reports to OSCCR, though anecdotally there seems to be some evidence showing collaboration is happening at a more than acceptable amount in the lab. During this semester not enough evidence was present to make any reports.

3.2 Recommendations

1. The choice of moving away from the zyBook was mostly a good thing, but there seems to be a vacuum in reinforcement without the required readings and homework assignments. While it would be ideal to have the same number/level/detail of homework assignments as the zyBook, creating such material as an individual section is not really reasonable. It is recommended to follow though on the much-discussed department question repository that can be utilized for such assignments (as well as tests). With the move to a single section, less of a need for this for tests exists but that also means compiling the database

from the historical instructors of the course can easily lead to such a resource as their material may not be used in the form in which it existed previously.

Action items:

- develop a department question databank
- create a unified homework set that can be re-used in any year

2. The use of repl.it for assignments was useful, and there was more positive feedback about the system than negative. Again, some extra work would help make the automated feedback more informative, though again more effort than was available previously may be needed. Once again this is a problem that may be solved in the single-section model where more teaching/instructional assistance is available to devote to this single resource. The major point of feedback received from students was the lack of feedback that they received, more training is needed of the IAs and TAs on grading.

   Action items:

   - introduce better feedback in unit tests when being created in the repl.it auto-grader
   - provide rigorous training to instructional staff on how feedback should be given to students when final grades are assigned

3.3 Response, Action, and Update on Previous Recommendations

Below are my personal recommendations from the previous CQI and how they has been addressed.

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. This has been addressed in the most recent update to the objectives. We covered it at a level 1, and most students understood it in the assessments asked.
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. This has been addressed in the most recent update to the objectives. This was again not included in the material of the course.
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. While this is not directly a problem now that courses are back in person, several online-style elements are still included such as an anonymous questions sheet though this has been used less frequently than before.
4. In general, exam questions in my section tend to be overly complicated (in particular questions highlighted in Table 5 in yellow). It may be better to provide questions for each topic at various levels to get a better sense of understanding. This is still a problem, as mentioned above in the recommendations from this semester.
5. 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. Again with the new objectives this is not as much of a problem, though we do briefly discuss it when talking about different ways to loop though arrays. It is not assessed.
6. Improved class-specific instructional staff training, and better communicate my expectations for the feedback they provide students. Similarly clarify the grading workflow. This has been improved, I now meet

with the entire instructional staff at least once a week and we keep an ongoing dialogue in a private channel on Teams.

7. 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. I now make the objectives an open part of my presentation of the course. The students (as mentioned in the review) also appreciate that they can keep track of upcoming and past assessment items.

Table 5: Concepts on Exams

				Assessment 1					Assessment 2						Assessment 3								Assessment 4				
			Question	1	2	3	4	5	1	2	3	4	5	6	1	2	3	4	5	6	7	8	1	2	3	4	5
			Result	8.5	7.5	6.7	7.3	11.4	16.4	15.4	3.4	0.7	3.2	11.6	4.2	13.7	17.0	6.4	3.1	7.4	13.6	8.9	12.9	3.9	15.1	12.5	6.6
			Out Of	12	15	9	16	18	20	28	4	4	5	18	5	20	20	12	5	12	25	10	18	5	30	21	10
Objective	Times	All	Passed																								
1.1	15	62%	65%		1	1		1				1		1	1	1	1	1	1	1	1	1			1	1	
1.2	1	77%	81%																					1			
2.1.a	4	68%	71%					1			1			1									1				
2.1.b	2	78%	82%						1				1														
2.2.a	3	80%	83%						1													1	1				
2.2.b	2	58%	61%							1																	1
2.3	6	63%	66%	1		1		1						1		1									1		
3.1.a	3	64%	67%													1					1		1				
3.1.b	3	62%	62%												1			1		1							
3.1.c	2	80%	81%														1		1								
3.1.d	2	58%	60%																				1		1		
3.1.e	2	65%	67%																				1			1	
3.2.a	2	56%	58%		1																					1	
3.2.b	6	60%	64%	1		1		1								1					1				1		
3.2.c	6	61%	65%		1	1		1				1		1		1											
3.2.d	3	60%	63%																	1		1			1		
3.2.e	3	55%	57%									1				1									1		
3.2.f	2	56%	59%			1								1													
3.2.g	1	59%	61%																							1	

Notes:

– Includes only students who were assigned final grades.

– The “all” and “passed” columns are the weighted total score of the questions for that objective of all students, and those who passed the course.

– The final exam is not included since it was optional and does not reflect all of the students, but covered all objectives.

– Objectives with scores lower than a C are marked in yellow.

CS 2302

CS2302 Data Structures
Course Review
Fall 2022
Instructor: Olac Fuentes

Outcome Assessment

Learning 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. Solve problems using arrays and lists.
3. Given a non-recursive algorithm examine its loop structure, assess its asymptotic running time in relation to the size of the input, and express it using big-O notation.
4. Given a recursive algorithm, examine its structure, formulate, and solve a recurrence equation defining its running time in relation to the size of the input, and express it using big-O notation.
5. Design and implement solutions to computational problems based on iteration and recursion.
6. Trace the behavior of functions and algorithms involving iteration and recursion.

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) Balanced search trees
   c) Graphs
2. Solve problems using hashing, specifically using language-specific data structures (e.g., sets and dictionaries in Python)
3. Describe, implement, and apply the following graph algorithms:
   a) Breadth-first search
   b) Depth-first search
   c) Topological sorting
   d) Minimum spanning trees (Kruskal's and Prim's)
   e) Single-source shortest paths (Dijkstra's algorithm)
4. Assess space requirements 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
e) Randomized algorithms

Additional Course Information

Instructional Staff:
Md. Mahmudulla Hassan (TA)
Andres Segura (IA)
Alec Tellez (IA)

Textbook:
Introduction to Python Programming and Data Structures, Y. Daniel Liang, Pearson (eText), Second Edition (recommended)

Most materials were presented using a set of Colab notebooks developed by the instructor.

Grading: Final grades were computed using the following weights:

- 20% - Lab projects (3 projects)
- 12% - Quizzes, homework, attendance, and in-class exercises
- 48% - Partial Exams (4 exams)
- 20% - Final exam

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

Outcome	Level	Exam 1	Exam 2	Exam 3	Exam 4	Other	Final Exam	Result
3.1 Given a problem, judge which data structures are required to solve it efficiently and justify the selection.	3					Lab1:70% Lab2:86% Lab3:94%		Satisfactory Median: 86%
3.2 Solve problems using arrays and lists.	3	Q2:71% Q3:41%				Lab1:70%	QA3:78% QA6:78% Q1:97% Q3: 75% Q4: 100% Q8:69%	Satisfactory Median: 75%
3.3 Given a non-recursive algorithm examine its loop structure, assess its asymptotic running time in relation to the size of the input, and express it using big-O notation.	3	QA2-25:90%					QA1: 78%	Satisfactory Median: 84%
3.4 Given a recursive algorithm, examine its structure, formulate, and solve a recurrence equation defining its running time in relation to the size of the input, and express it using big-O notation.	3		QA8-16: 86%	QA12-14: 81%		QuizRec:63%	QA2:78%	Satisfactory Median:80%
3.5 Design and implement solutions to computational problems based on iteration and recursion.	3	Q1:100% Q2:71% Q3:41% Q4:55% Q5:87% Q6:52%	Q1:100% Q2:100% Q3:80% Q4:81% Q5:78% Q6:57%				Q5: 38% Q6: 83%	Satisfactory Median:79%
3.6 Trace the behavior of functions and algorithms involving iteration and recursion.		QA1-17: 90%	QA1-7: 84%	QA1-4: 81%		QuizBST:72%	QA3-15:78%	Satisfactory Median:81%
2.1.a Describe, implement, and use the following data structures: Heaps	2			QA9-11,15: 81% Q3: 58%			QA7:78% Q10:97%	Satisfactory Median:80%
2.1.b Describe, implement, and use the following data structures: Balanced trees	2		Q2:100% Q4:81% Q6:57%			QuizBTrees:79% Lab2	QA9:78% Q7: 38%	Satisfactory Median: 79%
2.1.c Describe, implement, and use the following data structures: Graphs	2			QA1-5: 81% Q1: 86% Q2: 98% Q4: 100% Q5: 98%			QA10: 78% Q8:69%	Satisfactory Median: 86%
2.2. Solve problems using hashing, specifically using language-specific data structures (e.g., sets and dictionaries in Python)	2	Q4:55% Q5:87% Q6:52%		QA1-5: 81% Q1: 86%			Q2: 56% Q3: 75%	Satisfactory Median: 75%
2.3.a Describe, implement, and apply the following graph algorithms: Breadth-first search	2				QA4: 79%		QA11:78%	Satisfactory Median: 79%
2.3.b Describe, implement, and apply the following graph algorithms: Depth-first search	2				QA4: 79%			Satisfactory Median: 79%

2.3.c Describe, implement, and apply the following graph algorithms: Topological sorting	2				QA4: 79%		QA13: 78%	Satisfactory Median: 79%
2.3.d Describe, implement, and apply the following graph algorithms: Minimum spanning trees (Kruskal's and Prim's)	2			QA1-5: 81% Q5:98%			QA12: 78% Q9: 85%	Satisfactory Median: 83%
2.3.e Describe, implement, and apply the following graph algorithms: Single-source shortest paths	2				QA4: 79%		QA14: 78% Q11: 67%	Satisfactory Median: 78%
2.4 Assess space requirements of algorithms in relation to the size of their inputs.	2							Not evaluated
1.1.a Identify, explain and apply the following algorithm design techniques: Greedy algorithms	1			QA1-5: 81% Q5:98%	QA4: 79%	Lab3:94%		Satisfactory Median: 88%
1.1.b Identify, explain and apply the following algorithm design techniques: Divide and conquer	1				QA4: 79% Q1:75%	Lab3:94%		Satisfactory Median: 79%
1.1.c Identify, explain and apply the following algorithm design techniques: Dynamic programming	1				QA5: 79% Q3: 80%	Lab3:94%	QA15: 78%	Satisfactory Median: 87%
1.1.d Identify, explain and apply the following algorithm design techniques: Backtracking	1				QA6: 79% Q4: 84%	Lab3:94%		Satisfactory Median: 84%
1.1.3 Identify, explain and apply the following algorithm design techniques: Randomized algorithms	1				Q2: 100%	Lab3:94%	Q12: 43%	Satisfactory Median: 94%

Analysis by Instructors:

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

Observations:

1. All level-three outcomes were satisfied. Outcomes 3.5 and 3.6 are considered the most important in terms of helping students develop problem-solving skills and thus were evaluated multiple times, often in combination with more fact-based material covered in class.

2. All level-two outcomes that were evaluated were satisfied. Special emphasis was placed on outcome 2.2, as dictionaries are a very important data structure in Python and results in early assessments were not good. However, outcome 2.4 was only briefly covered and not evaluated due to lack of time.

3. All level-one outcomes were satisfied.

4. The use of paper-only tracing exercises (which was impossible during the pandemic), seemed to have a positive effect on learning.

5. 31 students took the final exam. The grade distribution was as follows: A: 14, B: 7, C: 4, D: 1, F: 5. This yields a passing rate of 25/31 =81%, which is slightly higher than that of previous semesters. However, an

unusually high number of students dropped the class, a total of 19, thus only half of the students who originally enrolled in the class completed it successfully. This is among the lowest percentages we have had in recent years.

6. It is difficult for underprepared students to catch up in the course. A total of 17 students scored less than 50% on Exam 1 and only three of them passed the course. Conversely, out of 17 students who scored over 75% on that exam, 15 passed the class with A or B.

7. Very few students submitted course evaluations.

8. We tried sharing lab assignments among the three instructors with modest success.

Recommendations:

1. In-class problem solving is an effective teaching tool, as it allows instructors to identify and correct individual students' misunderstandings and knowledge gaps. However, some students feel uncomfortable receiving feedback and constructive criticism. We should look into ways to mitigate this problem.

2. Students' course evaluations are a valuable tool to improve instruction. We must make an effort to increase response rates. Perhaps we should also share evaluations among instructors.

3. While there was a small reduction in the number of topics covered, it is still difficult to cover all topics presented in the outcomes.

4. The large number of drops is still a concern. Perhaps re-introducing peer leaders would help.

Analysis by Committee:

Pending

Recommendations from previous reports and actions taken:

Recommendation: 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.

> **Action taken:** We tried several strategies, including pairing stronger and weaker students together in team exercises, providing extra credit for attending office hours, subtly paying more attention to struggling students during in-class work, providing review sessions for CS2401 topics, but none appeared to be particularly successful.

Recommendation: 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.

> **Action taken:** We performed a large number of tracing exercises where students were only allowed to use paper and we also included a tracing part in every exam. It seems like this was helped for students.

Recommendation: 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.

Action taken: None.

Recommendation: 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.

Action taken: We eliminated some outcomes, including disjoint set forests and the connected components algorithm.

CS2302 Data Structures
Course Review
Fall 2022
Instructor: Daniel Mejia

Grade Distribution
There were 55 students registered for credit on census day – 3 students dropped the course after census day up until the course drop deadline and received an automatic W. 52 students earned a grade in the course.

Grade	Pass			Fail	
	A	B	C	D	F
No. of students	18/52	11/52	9/52	3/52	11/52
Percentage	34.6%	21.2%	17.3%	5.7%	21.2%
Total	73.1%			26.9 %	

*C is a passing grade in CS 1301.

Failing Students
- S1-S5 did not take the final exam
- S6-S7 scored less than 15 percent on their final exam and did poorly on all of the exams
- S8-S15 scored less than 50 percent on their final exam and did poorly on all of the exams

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. 3 Lab assignments were also assigned.

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:

2. Identify and explain the following algorithm design techniques:
   f) Greedy algorithms

        a. E3 #13 – 91.1%
    g) Divide and conquer
        a. E3 #10 – 97.7%
    h) Dynamic programming
        a. E3 #12 – 91.1%
    i) Backtracking
        a. E3 #14 – 91.1%
    j) Randomized algorithms
        a. E3 #11 – 97.7%

Level 2: Application and analysis:

5. Describe, implement, and use the following data structures:
    d) Heaps
        a. Lab 3 – 78.2%
    e) Balanced trees
        a. F12 – 82.6%
        b. F13 – 82.6%
    f) Graphs
        a. F35 – 64.5%
        b. F36 – 69.3%
        c. F37 – 59.4%
        d. F38 – 57.8%
6. Solve problems using hashing, specifically using language-specific data structures (e.g., sets and dictionaries in Python)
    a. HW 2 – 79.9%
7. Describe, implement, and apply the following graph algorithms:
    f) Breadth-first search
        a. F28 – 93.5%
    g) Depth-first search
        a. F29 – 78.3%
    h) Topological sorting
        a. E3 #1 – 71.1%
    i) Minimum spanning trees (Kruskal's and Prim's)
        a. F32 – 93.5%
    j) Single-source shortest paths (Dijkstra's algorithm)
        a. F30 – 63.3%
8. Assess space requirements of algorithms in relation to the size of their inputs
    a. Not Assessed

Level 3: Synthesis and evaluation:

1. Given a problem, judge which data structures are required to solve it efficiently and justify the selection
    a. Lab 1 – 81.1%

    b. Lab 2 – 85.1%
    c. Lab 3 – 78.2%
2. Solve problems using arrays and lists
    a. Lab 1 – 81.1%
    b. Lab 2 – 85.1%
    c. Lab 3 – 78.2%
3. Given a non-recursive algorithm examine its loop structure, assess its asymptotic running time in relation to the size of the input, and express it using big-O notation
    a. F1 – 95.7%
    b. F3 – 93.5%
    c. F4 – 82.6%
4. Given a recursive algorithm, examine its structure, formulate, and solve a recurrence equation defining its running time in relation to the size of the input, and express it using big-O notation
    a. F15 – 65.2%
    b. F18 – 78.3%
    c. F20 – 67.4%
5. Design and implement solutions to computational problems based on iteration and recursion
    a. F11 – 53%
    b. Lab 1 – 81.1%
    c. Lab 2 – 85.1%
    d. Lab 3 – 78.2%
6. Trace the behavior of functions and algorithms involving iteration and recursion
    a. F7 – 71.7%

Analysis of the Results, and Instructor Recommendations

The results of the learning outcomes were mostly met at a satisfactory level, with a few not meeting a 70% threshold. Several outcomes were met with 80%+. 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).

Some of the lower results (graphs) may be lower due to being done towards the end of the semester with less opportunity for assessment. Students struggle with developing recursive solutions for computational problems.

Recommendations

Instructor recommendations:

i. Review basic recursion topics earlier in the semester (first week)
ii. Introduce space complexity analysis early in the semester
iii. Provide additional review sessions (TA/IA) for students on a “per topic” level