BDSI: A Validated Concept Inventory for Basic Data Structures

Leo Porter
University of California, San Diego

Daniel Zingaro
University of Toronto Mississauga

Soohyun Nam Liao
University of California, San Diego

Cynthia Taylor
Oberlin College

Kevin C. Webb
Swarthmore College

Cynthia Lee
Stanford University

Michael Clancy
University of California, Berkeley

ABSTRACT

A Concept Inventory (CI) is a validated assessment to measure student conceptual understanding of a particular topic. This work presents a CI for Basic Data Structures (BDSI) and the process by which the CI was designed and validated. We discuss: 1) the collection of faculty opinions from diverse institutions on what belongs on the instrument, 2) a series of interviews with students to identify their conceptions and misconceptions of the content, 3) an iterative design process of developing draft questions, conducting interviews with students to ensure the questions on the instrument are interpreted properly, and collecting faculty feedback on the questions themselves, and 4) a statistical evaluation of final versions of the instrument to ensure its internal validity. We also provide initial results from pilot runs of the CI.

CCS CONCEPTS

• **Social and professional topics → Computing Education**.

KEYWORDS

concept inventory, assessment, data structures

ACM Reference Format:
Leo Porter, Daniel Zingaro, Soohyun Nam Liao, Cynthia Taylor, Kevin C. Webb, Cynthia Lee, and Michael Clancy. 2019. BDSI: A Validated Concept Inventory for Basic Data Structures. In *International Computing Education Research Conference (ICER '19), August 12–14, 2019, Toronto, ON, Canada.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3291279.3339404

1 INTRODUCTION

A Concept Inventory (CI) is a validated assessment of student conceptual knowledge for a particular topic. Validation is the process through which it is determined that a CI measures what the designers intended it to measure [1]. CIs first appeared in physics with the Force Concept Inventory [14], an assessment designed to measure student knowledge of Newtonian Mechanics, and have since become popular within STEM disciplines for establishing what students know. The value of such an assessment includes facilitating educational research by providing a common benchmark for student knowledge, and supporting instructors looking to diagnose areas of strength (or weakness) in their particular courses.

In computer science, there have been two instruments created for use in such a manner: the Second CS1 instrument [20] and the Digital Logic Concept Inventory [13]. Our work complements the field by adding the Basic Data Structures Inventory (BDSI), a new CI that can be used in conjunction with a CS2 course (or the corresponding course that teaches Basic Data Structures).

In this work, we describe our process in creating and validating the CI. We developed questions that address both the Learning Goals [22] and common difficulties with Basic Data Structures [33]. The question development (and CI development) was iterative. Questions were developed, students were interviewed on those questions, and, based on their interpretation of the questions, questions were revised. Periodically, feedback from faculty at multiple institutions was solicited. As the questions became more refined, a draft of the CI was used at multiple institutions to gauge student performance and interpretation of the questions. Then, a near-complete version of the CI was used at multiple institutions to statistically test the instrument. A final round of revision was then performed along with a final administration and statistical analysis.

In this paper, we present the process by which the CI was developed and validated. Specifically, in accordance with modern assessment design practices, we make claims about the instrument's validity and support those claims with corresponding methods and results. In the final step, the statistical validation, we provide results on the CI based on administration at multiple institution types. We end with a discussion of some challenges in CI development, and encourage faculty and researchers to begin using this CI to measure student knowledge of Basic Data Structures.

2 BACKGROUND

A Concept Inventory (CI) is a multiple-choice assessment designed to measure student understanding of the core concepts of a course. The goal of a CI is not to summatively assess a particular student, but rather to create a standardized instrument that can be used to compare across instructors, courses, and pedagogies [1].

2.1 CIs in STEM and Computer Science

CIs arose from observations by physics educators in the mid 1980s that conventional instruction in Newtonian physics was not working. It appeared that everyday beliefs about motion and force were incompatible with Newtonian concepts; moreover, conventional instruction produced little change in these beliefs [14]. This led to

ICER '19, August 12–14, 2019, Toronto, ON, Canada

ACM ISBN 978-1-4503-6185-9/19/08…$15.00
https://doi.org/10.1145/3291279.3339404

the design of the Force Concept Inventory (FCI) [14] to measure these student misconceptions. The FCI led to the adoption of Peer Instruction and other active learning techniques within physics education [4]. A later study [11] administered the FCI to 6542 students in 62 different courses, and found that all traditional courses had lower learning gains than those using active learning. The impact of the FCI on the physics community led to the design of other CIs both in physics [3, 19] and in numerous other scientific areas including chemistry [17], genetics [25], statistics [26], calculus [8], geoscience [18], oceanography [2], nursing [23], astronomy [24, 32], and biology [6].

Within computer science, there have been relatively few CIs developed [27]. Tew and Guzdial developed the FCS1, an assessment for CS1 that uses a pseudocode to measure students' basic programming knowledge [28]. Due to the unavailability of the FCS1, the Second CS1 Assessment (SCS1), an isomorphic version of the FCS1, was developed and made available to instructors [20]. A CI has also been developed for Digital Logic [13]. Preliminary work towards CIs is available for a number of other computer science topics, including algorithms [5, 9], recursion [12], advanced data structures [16], architecture [21], and operating systems [30].

2.2 CI Creation Process

Adams and Wieman [1] list several requirements for a useful CI:

- It must be easy to administer and grade without any training, and in the context of a normal course schedule.
- The instrument must test 'expert thinking' of obvious value to teachers.
- It needs to be demonstrated that it measures what it claims (evidence of validity).

In order to meet these criteria, a CI must go through a rigorous development process. We adopted the development process established by Adams and Wieman [1], used for at least nine prior concept inventories, and consisting of the following steps:
(1) Work with instructors to establish which topics are important to cover. This can be done through the Delphi process [10], surveying instructors' syllabi and final exams, or discussing with instructors.
(2) Identify important topics on which students have misconceptions, and develop open-ended questions on these topics. This can be done via interviews with instructors or students.
(3) Identify particular student misconceptions when answering open-ended questions, and use these as distractor answers when developing multiple-choice versions of the open-ended questions. This can be done by administering open-ended versions of the CI and coding student wrong answers, or by having students perform think-alouds while working through the open-ended test.
(4) Validate multiple-choice test questions through think-aloud interviews with students to ensure that questions are not being misinterpreted, and answers are not chosen for the wrong reasons.
(5) Statistically validate the test. After administering the test to a large sample of students (preferably across multiple different types of institutions), statistically assess the results.

There are a number of measures of statistical validity. The SCS1 was originally validated using Classical Test Theory (CTT) to measure the difficulty and discrimination of each question [20]. (Discrimination measures how likely students who correctly answer a particular problem are to correctly answer other problems.) Later work by Xie et al. [31] used Item Response Theory (IRT) to evaluate the SCS1, and found that four questions on the test were too difficult when it was used as a pre-test. Other commonly used statistical measurements include Cronbach's alpha, used to measure internal consistency [13, 20, 31]; and Ferguson's Delta, a measure of the discrimination of the test as a whole [3].

3 BDSI CONTENT AND DEVELOPMENT

The 13 questions from the final version of the instrument are summarized in Table 1. The questions cover a variety of data structures topics, including lists, trees, stacks, and sets, with a heavy emphasis on lists and trees. While all of the questions are multiple-choice, they are presented in two formats—some ask that students select only one option, whereas others ask students to select all the options that apply. Each question has a primary learning goal according to the goals described by Porter et al. [22]; the relevant table is reproduced in Table 2.

In developing the questions for the BDSI, we faced a challenge due to variations in course content, particularly surrounding the choice of programming language. To make the BDSI accessible to as many courses as possible, we aimed to develop questions that are agnostic to the choice of programming language. Thus, we augmented the pseudocode language originally used in the FCS1 [28] and SCS1 [20] with minor modifications to include objects. When presenting the BDSI to students, we provide a short reference packet that encompasses:

- Question answering instructions that highlight the differences between the select-one and select-all question styles.
- Brief definitions for the interfaces and data structures used in the BDSI. We provide these definitions to help account for students who may have learned different terminology depending on their programming language (e.g., array lists in Java, lists in Python, and vectors in C++).
- A concise summary of the BDSI's pseudocode language with several examples.

Regardless of the language in which students learned the material (Java, Python, or C++), students rarely identified programming language differences, the pseudocode language, or data structure terminology as sources of confusion in the course of our validation work, as discussed in Section 4.4. While our pseudocode was tested with students using a variety of imperative coding languages, we do not address Basic Data Structures in functional paradigms such as Scheme, whose teaching and learning may be different.

3.1 BDSI Development Team

There were two main groups of faculty participants: the BDSI Project Team of six faculty (the authors), and the Expert Panel of nine faculty. The Project Team consists of faculty with teaching backgrounds from both large research-intensive universities and small liberal arts colleges. We met frequently during the development of BDSI, authored questions, conducted (or analyzed) interview data from students, and analyzed results from pilot runs of the instrument. Throughout the CI development process, we consulted with an Expert Panel: CS2 instructors from a broad set

Table 1: Summary of BDSI's questions, goals, and data structures.

	Primary Goal	Structures	Question Description
q1	4	LL	Correctly add an element to the tail of a linked list.
q2	1	LL	Compare list performance with and without a tail pointer.
q3	1	LL	Without looking at the code, experimentally determine how an ADT was implemented.
q4	1	LL	Analyze performance of implementing an interface with a linked list.
q5	5	BST	Identify BST insertion orders that lead to balanced trees.
q6	5	LL	Explain the behavior of a function that uses a linked list.
q7	4	BT	Write a function to compute a property of a binary tree.
q8	3	Many	Compare the performance of satisfying an abstract interface with several structures.
q9	5	BST	Analyze performance and effect on tree properties of adding elements to a BST.
q10	2	LL, Stack	Evaluate the performance of implementing 'undo' functionality with a linked list.
q11	6	BST	Determine appropriate cases for testing the correctness of a BST implementation.
q12	4	BT	Compute the width of a binary tree.
q13	2	List, Set	Compare solving a problem with a list interface versus a set interface.

Table 2: Learning Goals Adopted from Porter et al. [22]

Goal#	Learning Goal
1	Analyze runtime efficiency of algorithms related to data structure design.
2	Select appropriate abstract data types for use in a given application.
3	Compare data structure tradeoffs to select the appropriate implementation for an abstract data type.
4	Design and modify data structures capable of insertion, deletion, search, and related operations.
5	Trace through and predict the behavior of algorithms (including code) designed to implement data structure operations.
6	Identify and remedy flaws in a data structure implementation that may cause its behavior to differ from the intended design.

of North American institutions including research-intensive universities, public regional universities, liberal arts colleges, and community colleges. The Expert Panel provided feedback periodically during the development process and were extremely helpful in contributing their institutional context to the work. We expand on the role that each group played in Section 4.

3.2 BDSI Versions

The BDSI went through many versions prior to being finalized. Early in development, there was simply a collection of questions used during interviews to elicit student thinking. Later, a cohesive sequence of questions emerged and continued to be refined. There are two key versions worth noting for our purposes here. A near-final version that we call the *Beta* version was used in runs of the instrument in Fall 2018. Small changes to two questions (q4 and q5) were subsequently made based on feedback and analysis. The result was the version we call the *Final* version of the BDSI; it was used in the final pilot run. We provide analysis for both of these versions of the instrument in Section 5.

3.3 Pilot Administration

To pilot the BDSI, we organized "final exam review sessions" to be held in conjunction with the end of the term for courses that teach Basic Data Structures. These sessions were led by course instructors, TAs, or other faculty in the department and involved students individually taking a one-hour "practice exam" that was the current draft of the BDSI. Once complete, the exam booklets were collected by the staff administering the exam, and the staff then went over the answers to the BDSI as well as other course topics not addressed by the CI. Students who attended the final exam review session were allowed to opt-in or opt-out from participating per our approved Human Subject Board protocol (if they opted out, they could complete the practice exam but we did not use their data).

4 MEASURES OF VALIDITY

Modern assessment design involves marshalling evidence to support claims/interpretations [15]. As such, Table 3 provides an overview of the claims of validity and the methods and evidence in support of those claims. Each claim is discussed in detail in the following corresponding subsections.

As previously mentioned, a number of institutions participated in the project at various points in the development process. A summary of institutions and their involvement can be found in Table 4. The table contains the institution type: Community College (CC), Primarily Undergraduate Institution (PUI), Liberal Arts Institution (LAI), and Research-Intensive University (RIU); and the size of the institution: (S)mall (< 5000), (M)edium ($5000 \leq n < 10000$), or (L)arge (≥ 10000 students). Whether there was a Project Team member and/or Expert Panel member at each institution is also denoted. (Note that one Project Team member changed institutions during the project, and that both institutions are reflected; and that at I-M there were two Expert Panel members due to coverage of topics across multiple classes.) Finally, the table indicates the institutions where interviews were conducted, and how many students were involved in piloting the Beta and Final versions of the instrument. For institution I-F, there were two separate runs of the Beta version during two separate terms.

Table 3: Summary of Claims of Validity and Corresponding Methods

Section	Validity Claim	Methods Supporting Validity
4.1	C1. CI addresses topics that matter to instructors	• Reached agreement among a Project Team and Expert Panel, representing a diverse set of institutions, about the desired learning goals and topics [22]
4.2	C2. CI questions are meaningful to instructors	• Developed questions to address Learning Goals • Derived many of the initial questions from exercises that had been used in a classroom setting • Held meetings with Project Team and Expert Panel to gain agreement that questions were valued • Adapted questions based on feedback from Expert Panel and from instructors piloting the instrument
4.3	C3. CI questions address key concepts and student difficulties	• Aligned questions with Learning Goals • Identified potential misconceptions and developed questions by conducting 50 total interviews with students at 3 different institutions where students learned Basic Data Structures in Java, C++, or Python • Piloted open-ended version of CI with 408 students at 2 institutions to identify common difficulties and construct multiple-choice distractors. • Published the student difficulties with Basic Data Structures that we identified [33]
4.4	C4. Students properly interpret questions, including questions written in pseudocode	• Conducted 49 additional think-aloud interviews with students across 5 different institutions as they worked through the questions • Modified questions to address student understanding, and interpretation by non-native English speaking students • Used item discrimination (as shown in C5) to demonstrate that students are correctly interpreting questions
5	C5. Instrument is internally consistent	• Conducted a statistical evaluation using Classical Test Theory and Item Response Theory to determine if the instrument is internally consistent.

Table 4: Summary of Institutions and Respective Project Involvement.

	I-A	I-B	I-C	I-D	I-E	I-F	I-G	I-H	I-I	I-J	I-K	I-L	I-M	I-N
School Type	CC	PUI	LAI	RIU	LAI	RIU	RIU	RIU	CC	PUI	LAI	LAI	RIU	RIU
School Size	M	L	S	L	S	L	L	L	L	L	M	M	L	L
Project Team			X	X	X	X	X	X		X				
Expert Panel	X	X				X			X		X	X	X	X
Interviews				X			X	X	X	X				
# Beta Participants	16	17	12	14	63	82 / 408	110							
# Final Participants			57	11			164							

4.1 C1: Targets Learning Goals

Our prior work surveyed a group of faculty across a diverse set of North American institutions to determine which topics and concepts are common to CS2 courses [22]. We found that CS2 courses across institutions and instructors commonly included Basic Data Structures such as stacks, queues, array-based lists, linked lists, and binary search trees. Working with our Expert Panel at multiple institutions, we produced a set of learning goals for Basic Data Structures [22] (articulated in Table 2). That work is critical to claim C1 as it established the necessary consensus among faculty about which learning goals were most important for Basic Data Structures. We used the learning goals when building the CI.

4.2 C2: Questions Accepted by Instructors

To ensure that the questions on the instrument would be accepted by instructors, we engaged instructors throughout the question development process. The first drafts of our questions were made to align with the Learning Goals, and were either taken from prior studies [5, 16, 33] or from exercises that had previously been used in CS2 classes at multiple institutions represented by the Project Team.

The Project Team (the authors) were deeply involved in the process of question development and refinement through student interviews (described in Sections 4.3 and 4.4). The Expert Panel, consisting of faculty from a broader and more diverse set of institutions than the Project Team, were engaged at multiple points. Early in the question development process, we met with a number

of Expert Panel members at a conference to solicit feedback on the early drafts. This feedback was used to refine and re-focus the questions. A year later, questions had been heavily refined through interviews, and the Expert Panel was given a nearly-final version of the questions for their review. In an engaged discussion between the Project Team and Expert Panel, the Expert Panel gave the team some minor suggestions for ways to improve the questions. More importantly, during that meeting, members of the Expert Panel agreed that the questions were meaningful to them as instructors and that they would want to use the instrument in their courses (subsequently, a few members volunteered to use the instrument to assist with our validation).

After this positive feedback from the Expert Panel and the validation interviews with students described in Section 4.4, the Project Team began using the instrument in pilot runs at a number of institutions. During the pilot runs (including the large pilot runs described in detail in Section 5), instructors at a number of institutions were engaged in administering the CI. Those instructors gave us feedback, mostly minor suggestions, that we incorporated into the final version of the CI.

4.3 C3: Questions Address Key Concepts

To better understand student thinking about key concepts, our team conducted a total of 50 interviews at 3 different institutions (I-D, I-G, and I-H). These interviews consisted of asking students to solve sample questions that targeted the learning goals for Basic Data Structures [22]. The interviewers asked students to think aloud when solving the problems, and occasionally asked follow-up questions to clarify student thinking on particular questions. In addition to the one-on-one interviews, we piloted open-ended versions of the exam between winter 2017 and summer 2017 with 408 students at 2 institutions (I-G, I-J). The open-ended version asked students to solve the problems either without multiple choice options or, when multiple choice options were available, to justify their selection in text. Overall, this process of interviews and open-ended pilots identified a number of student difficulties with Basic Data Structures, previously published in Zingaro et al. [33], and were the foundation of our multiple choice distractors. Question design involved an interleaving of misconception discovery and question improvement, ultimately leading to a set of questions that probe student thinking.

4.4 C4: Questions Are Interpreted Properly

Through the earlier interviews described in the prior subsection, we had some confidence that the questions were being interpreted properly. To bolster this interpretation, we used a draft of the full instrument in 26 interviews at 3 institutions (I-G, I-H, and I-J) in the summer and fall of 2017. These interviews were think-alouds where students worked through the questions on their own without any intervention from the interviewer except to prompt them to continue thinking aloud if they went silent.

Those interviews identified misinterpretation by students because of ambiguity in the questions, key details of the questions not being highlighted sufficiently for students to recognize their importance, and difficulties with terminology by non-native English speakers. Questions were revised in collaboration with the Project Team over the course of these interviews.

After gaining further confidence that the questions were being properly interpreted, a second round of interviews in the spring of 2018 was conducted with 23 students at 5 institutions (I-D, I-G, I-H, I-J, and I-I). Again, the format was think-alouds with minimal intervention from the interviewer. These interviews produced a few minor wording changes to some questions, but the key takeaway was that the questions were being interpreted properly.

Throughout these interviews, the interviewers were careful to note any confusions that might occur by students in using pseudocode rather than the programming language they used in their particular classes. Somewhat surprisingly to the team, the pseudocode was not a barrier to the student interpretation of the questions in any significant way. Students expressing confusion about the language was extremely rare, but when they did (e.g., how to create a new object in the pseudocode), they would refer to the provided example code and quickly resolve the confusion.

A final indication of proper interpretation is that questions across the instrument should be internally consistent. Such consistency tells us that when a student answers a particular question correctly, they are likely to understand that question and are more likely to answer other questions on the instrument correctly. Further discussion of internal consistency appears in the following section.

5 C5: BDSI IS INTERNALLY CONSISTENT

In this section, we describe our analysis of the questions themselves based on pilot runs conducted at a number of institutions. In order to score the questions, we graded each question as either correct or incorrect (no partial credit). For multiple-choice questions with a single correct response, the correct response is required. For select-all questions, all selections must be marked properly to receive credit (e.g., if the correct answer is options B, C, and D, and a student were to select either just options B and C or options B, C, D, and E, they would be graded as incorrect).

As the BDSI was run at a number of schools in its Beta and Final versions, we report summaries of these results where appropriate. For some measures, we merge results across similar institutions to simplify comparisons. For the bulk of the evaluation, we used statistical techniques requiring large sample sizes. To perform this analysis, while also including as many institutions as possible, we use two primary datasets:

- ***Beta.*** Four pilot runs of the Beta version of the instrument included more than 50 participants. The smallest of these four pilot runs included 63 participants. To include more than one institution in the analysis, but to not overweigh the larger samples (one run at an institution had 408 participants), we selected all 63 participants from the smallest group and then randomly selected 63 students from each of the other three runs. This dataset of 252 students from 4 different classes at 3 different institutions is referred to as our *Beta* dataset.
- ***Final.*** One institution had 164 participants take the Final version of the BDSI.

5.1 CTT vs. IRT

Classical Test Theory (CTT) is commonly used to validate CIs [20, 29]. It provides useful measures, such as question difficulty, question

Table 5: Summary of Question Correctness

	Beta				Final		
	CC	PUI	LAI	RIU	LAI	RIU	avg.
q1	0.81	0.41	0.87	0.5	0.67	0.61	0.65
q2	0.31	0	0.37	0.12	0.14	0.09	0.18
q3	0.81	0.71	0.65	0.7	0.81	0.8	0.75
q4	0.75	0.76	0.95	0.85	0.79	0.72	0.81
q5	0.62	0.35	0.4	0.27	0.33	0.33	0.39
q6	0.56	0.47	0.63	0.66	0.51	0.61	0.58
q7	0.31	0.18	0.33	0.45	0.33	0.39	0.33
q8	0.88	0.47	0.75	0.67	0.58	0.72	0.68
q9	0.38	0.06	0.31	0.22	0.18	0.21	0.23
q10	0.56	0.47	0.66	0.48	0.6	0.48	0.54
q11	0.69	0.71	0.78	0.69	0.77	0.7	0.73
q12	0.5	0.35	0.45	0.46	0.53	0.48	0.46
q13	0.94	0.65	0.91	0.83	0.82	0.78	0.82

discrimination, and internal reliability, and these measures can be used to ferret out questions that are performing poorly. At the same time, it has several key drawbacks that make it unsuitable as the sole source of quantitative validation data. First, CTT yields measures that are dependent on the particular people taking the test. For every administration of a test, one must re-run the CTT analysis to check that the administration is "valid" [29]. Second, CTT scores conflate ability with properties of questions; the scores cannot be taken as uncontaminated measures of ability.

By contrast, Item Response Theory (IRT) is more robust: assuming the data meets IRT's assumptions, its parameters are less sensitive to the particulars of the sample taking the test. In the following subsections, we perform both CTT (Sections 5.2–5.4) and IRT analyses (Sections 5.5–5.6) in order to learn about the CI from both perspectives.

5.2 Correctness

A summary of the correctness per question is provided in Table 5. One key observation from these results is that correctness varied by institution type; it varied across individual institutions, too, but this is not shown here to avoid conclusions being drawn about particular participating institutions. Identifying such variability in student performance per question and institution is precisely the point of such an instrument, but is not part of the validation. The second observation is that questions varied in difficulty. Some were easier for students in general (e.g., q4 and q13) and others were more difficult (e.g., q2 and q9). A wide range of difficulty was a goal of the project to ensure that ranges of student abilities were captured and that faculty would want to use the instrument in their classes. The latter point was brought up early in the development process, with both the Project Team and Expert Panel expressing concern that the instrument should neither be too difficult nor too easy overall.

5.2.1 Point-Biserial Correlations. The point-biserial correlation can be used to measure the extent to which an item discriminates between high-performing and low-performing students. It is a number between -1 and 1: the higher the value, the greater the positive relationship between correctness on the item and correctness on the overall instrument. Values of at least 0.2 are desirable [29]. In the left half of Table 6, we provide the point-biserial correlation for each question. For each dataset, we find that each measure is well above the 0.2 cutoff, offering evidence that each question is discriminating.

Table 6: Point-biserial Correlations, Cronbach Alpha Drops, and Factor Loadings

	Point-biserial		Alpha Drop		Factor Loading	
	Beta	*Final*	*Beta*	*Final*	*Beta*	*Final*
q1	0.34	0.52	0	-0.03	0.1	0.22
q2	0.52	0.31	-0.03	-0.02	0.2	0.09
q3	0.32	0.46	0	-0.03	0.07	0.17
q4	0.43	0.31	-0.02	-0.02	0.11	0.08
q5	0.44	0.42	-0.01	-0.01	0.17	0.15
q6	0.46	0.47	-0.02	-0.01	0.16	0.19
q7	0.53	0.5	-0.03	-0.02	0.24	0.2
q8	0.47	0.51	-0.02	0	0.19	0.21
q9	0.49	0.49	-0.02	-0.01	0.2	0.17
q10	0.52	0.5	-0.03	-0.02	0.23	0.22
q11	0.45	0.52	-0.02	-0.02	0.16	0.21
q12	0.52	0.46	-0.03	-0.03	0.22	0.19
q13	0.4	0.41	-0.01	-0.02	0.11	0.15

5.3 Cronbach's Alpha

CI researchers often include a measure of internal reliability [7, 13, 20]. If the reliability is high, then a significant amount of variation in test scores is a result of variation in the sample taking the test (rather than a result of variability within the responses of particular people). A common measure of internal reliability is Cronbach's alpha, and values above 0.7 are taken as sufficient evidence for internal reliability [13]. Values for *Beta* and *Final* were 0.67 and 0.68, respectively, and are near this threshold.

Cronbach's alpha can also be used to determine whether any individual question is negatively affecting the test's internal reliability. Specifically, we can separately drop each item from the test, and calculate Cronbach's alpha on the test excluding that item [13]. In each case, we hope that the alpha value either stays as-is or falls: if the alpha value were to increase, then we would have evidence that the test is more reliable without the question than with it. The center two columns of Table 6 provide the alpha drops that we obtained when separately dropping each question from the test. We see that the alpha drops are either zero or negative, so no item is adversely affecting the test reliability. Moreover, the vast majority of questions have a negative drop, demonstrating that the question is contributing to the reliability of the assessment.

5.4 Ferguson's Delta

Unlike point-biserial correlations that give a score to each question, Ferguson's delta is a single score for the test as a whole. It measures the discriminatory power of a test by quantifying the amount of spread that is observed in respondent scores [7]. Ferguson's delta yields a value between 0 and 1, with scores of at least 0.9 offering

evidence that the test is discriminating. For our datasets, Ferguson's delta was 0.96 for the *Beta* version and also 0.96 for the *Final* version. In both cases, we can interpret the result as evidence that the assessment is discriminating.

5.5 Item Response Theory: Preliminaries

When modeling with IRT, it is necessary to choose among several competing models. 1PL, 2PL, and 3PL are often the models of interest [31]. A 1PL model gives each question a difficulty parameter; a 2PL model adds a discrimination parameter; and a 3PL model adds a guessing parameter. We tested each of these models, and settled on a 2PL model. The 1PL measures of fit were the strongest of the three models, but three questions did not fit this model. All questions fit the 2PL and 3PL models; we chose the 2PL model as it had stronger measures of fit than the 3PL model.

In order to perform our analysis, we must establish that the instrument is unidimensional. This is an important assumption: IRT assumes the data reflects one underlying trait [31]—in our case, a student's achievement level of Basic Data Structures. One way to test this unidimensionality is through a Confirmatory Factor Analysis (CFA). In a CFA, the researcher posits a number of underlying factors that are hypothesized to explain relationships between variables. As we are testing whether a single factor (dimension) explains the pattern of data, we ran a CFA with just 1 factor. Our results confirm that the model fits the data well, suggesting that we are measuring one underlying trait. Specifically [31]:

- The model chi-square value is an overall indication of whether the model fits the data. A large p-value, at least .05, suggests that we should reject the hypothesis that the model does not fit the data. Here, we have evidence of model fits: for *Beta*, chisquare(252) = 71.6, p = 0.29; for *Final*, chisquare(164) = 52.7, p=0.8.
- We also look at other measures of fit, and compare to the cutoff values from the literature. We require a Comparative Fit Index (CFI) of at least 0.9; our value was 0.98 for *Beta* and 1.0 for *Final*. Root Mean Square Error of Approximation (RMSEA) should be at most 0.1; ours was 0.02 for *Beta* and 0.00 for *Final*.

In the right two columns of Table 6, we provide the factor loading for each item. The factor loading is the proportion of variance in the item that the factor explains (between 0 and 1). All items had a p value of < 0.01, strong evidence that each item's variance is significantly explained by the factor.

All of these measures point to a unidimensional dataset, and so we proceed with IRT analysis.

5.6 Item Response Theory: Discrimination

An IRT model plots each question based on student relative performance on the rest of the test, and in doing so, produces the question's discrimination and difficulty. For example, in Figure 1 we have the IRT curve for Q1 for the *Final* dataset. On the x-axis, we have the z-score for student ability based on their performance on other questions on the test (all other questions than Q1) where a higher score means better performance. The y-axis is the probability that a student at that ability level will answer q1 correctly. As student ability improves, their likelihood of answering Q1 improves. Ideally, the curve on the graph is steep as this denotes the question

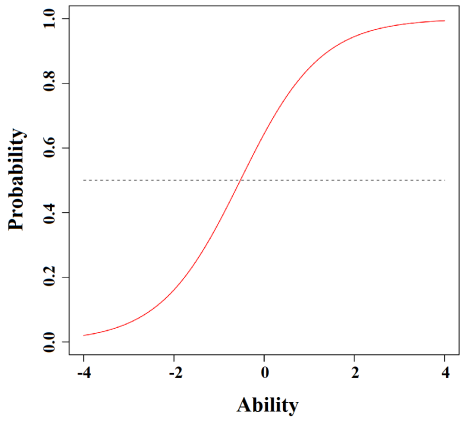

Figure 1: IRT curve for Q1 on *Final*

discriminates well between populations of students (Q1 does in fact discriminate well). The corresponding x-value for when the curve crosses the 0.5 probability threshold denotes the difficulty of the question; for Q1, the difficulty is -0.53. More negative values denote easier questions whereas higher values denote harder questions.

For each question to differentiate among students, we require that its discrimination parameter is acceptable. But the cutoff for what is acceptable is not agreed-upon in the literature. Values of 0.7 or 0.8 are typical [31].

Table 7 provides the difficulty and discrimination of each question for the two datasets. We find that the discrimination for nearly every question is above 0.8 for both datasets. A few questions (q1 and q3 for *Beta* and q4 for *Final*) are below 0.8 for one dataset but not for the other. The variation in student populations may help explain this difference. For example, q4 has the second highest discrimination for *Beta* yet has the lowest discrimination for *Final*. In speaking with the instructor of *Final*, we learned that the instructor had taught this very question to their students.

We find that the test has a mix of high and low difficulty questions. Supporting the correctness percentages in Table 5, q2 and q9 are quite difficult and q4 and q13 are fairly easy.

Overall, these results confirm that the BDSI's questions discriminate well and that they have a range of difficulty.

6 DISCUSSION

In this section, we discuss two examples of the subtle challenges involved in validating that students interpret questions as intended, our confidence in the present wording, and how faculty can access and use the instrument.

6.1 Changes to the BDSI: A Cautionary Tale

To give a sense of the kinds of issues that arise in validating questions, we present two illustrative anecdotes from the experiences of the Project Team.

Question 2 asks students about the operations whose performance improves when adding a tail pointer to a Singly-Linked List. In think-aloud interviews, students often initially said that it improved both addition at the tail (correct) and removal from the tail (incorrect). But for roughly half of those students, as they talked through their choices, many self-corrected, realizing that removal

Table 7: Item (Diff)iculty and (Discr)imination by Question

	Beta		*Final*	
	Diff	Discr	Diff	Discr
q1	-1.7	0.38	-0.53	1.12
q2	1.1	2.03	3	0.86
q3	-3.03	0.33	-1.38	1.31
q4	-1.69	1.7	-2.71	0.4
q5	0.88	0.89	1.08	0.74
q6	-0.86	0.8	-0.66	0.88
q7	0.41	1.27	0.53	1.02
q8	-1.1	1.11	-1	1.33
q9	1.1	1.36	1.25	1.42
q10	0.01	1.33	0.14	1.05
q11	-1.13	0.95	-0.92	1.25
q12	0.31	0.93	0.1	0.85
q13	-2.02	0.98	-1.69	0.86

from the tail will not be faster as there is no way to quickly set the tail to the preceding element. However, during large-scale runs of the instrument, we found that the percentage of students correctly solving the question without the think-alouds dropped. There was variation by institution, but an average (across institutions) correctness of 17% is below our experience in think-alouds. We suspect the act of talking through their choice changed their answer.

Question 9, asking about the impact of adding a number of values to a BST whose root presently holds its median value (where all new values are greater than the largest presently in the BST), went through many iterations of wording. An option choice once said "The root holds the median of the tree." Students struggled with whether we meant the median of the original tree (a true claim) or if we meant the median of the new tree (a false one). We then added the nomenclature T_0 to refer to the original tree, and T_1 to refer to the tree after the additions. After that change, students properly interpreted the options (though many still answered it incorrectly).

These experiences illustrate how subtle the question wording and presentation format issues can be. As such, we caution those attempting to amend the instrument as this bypasses the validation process we have undertaken to develop the BDSI. It was only with the full combination of elements in Table 3 that we obtained confidence in the question wording. As such, even well-meaning attempts to improve the test are likely to introduce unforeseen problems and will limit the ability to perform cross-comparisons against results from the unmodified BDSI.

Faculty should also be sensitive to how teaching examples inspired directly by the BDSI may vitiate the instrument's measurement capability. We already experienced one such accidental instance during one of our pilots (*Final*), as mentioned in Section 5.6. The instructor had lectured on the performance implications of using a LinkedList to create a key-value store, diminishing the measured discrimination power for that question relative to identical wording of that question in the previous pilot at the same institution. (We note that the instructor did so without any regard to the BDSI; they simply always taught this concept explicitly.)

6.2 How to Use the BDSI

With that caution out of the way, we now address how faculty can adopt the BDSI as an instrument to compare aggregate student knowledge of Basic Data Structures across institutions, instructors, courses, and pedagogies.

The BDSI is available to faculty by joining the BDSI group online[1]. The format of the instrument is a PDF in two parts: the introductory material (including pseudocode guide), and the test itself. It should be printed for students to complete on paper. There are some diagrams in the BDSI, and an accessible, though not validated, version with text descriptions in place of the diagrams is also available for visually-impaired students to complete on a device with screen reader software. Faculty should plan on giving students about 1 hour to complete the CI. Because CIs are not intended to be used for summative assessment of individual students, the BDSI should not be used on, or in place of, a comprehensive final exam. However, students may find it useful as a supervised "dress rehearsal" practice final exam, which faculty can use as a mutually beneficial opportunity to obtain student results.

To preserve the usefulness and validity of the BDSI, faculty must never publicly release the test or its solutions, and students should not be permitted to take their copy from the room during or after administration.

7 CONCLUSION

In this work, we present BDSI—a Concept Inventory for Basic Data Structures—developed through consultations with faculty and interviews with learners. Questions on the CI were developed through an iterative process of revision and feedback until faculty at a variety of institutions reported valuing the content and students consistently and correctly interpreted the questions. A large-scale pilot run of the instrument followed by a smaller run based on final revisions enabled statistical evaluation of the instrument. Using Classical Test Theory and Item Response Theory, the CI was shown to be internally consistent using a variety of statistical measures.

The value of a CI is not in the questions themselves, but in its capacity to drive pedagogical change. As such, along with reporting on the validity of the instrument, we have made the CI available to the community. Educational researchers may now use this validated assessment of student knowledge in their research, and instructors can use it to gauge strengths and weaknesses in their own curriculum. Ultimately, we hope that this CI will be taken up and deployed by the CS education research community to further enhance student learning.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the contributions of the following collaborators: Meghan Allen, Owen Astrachan, John Bell, Darci Burdge, Stephanie Chasteen, Rajendra Chattergoon, John Donaldson, Maureen Doyle, John Glick, Paul Hilfinger, Josh Hug, Marina Langlois, Kevin Lin, Lisa Meeden, Zach Palmer, Mahika Phutane, Joe Politz, Kate Rydberg, Samar Sabie, Kate Sanders, Jacqueline Smith, Marty Stepp, Paramsothy Thananjeyan, Steve Wolfman, Anna Zaitsev, and Christine Zhou. This work was supported in part by NSF award 1505001.

[1] https://groups.google.com/forum/#!forum/cs2-bdsi-concept-inventory

REFERENCES

[1] W. Adams and C. Wieman. Development and validation of instruments to measure learning of expert-like thinking. *International Journal of Science Education*, 33(9):1289–1312, 2011.

[2] L. Arthurs, J. F. Hsia, and W. Schweinle. The oceanography concept inventory: A semicustomizable assessment for measuring student understanding of oceanography. *Journal of Geoscience Education*, 63(4):310–322, 2015.

[3] S. V. Chasteen, R. E. Pepper, M. D. Caballero, S. J. Pollock, and K. K. Perkins. Colorado upper-division electrostatics diagnostic: A conceptual assessment for the junior level. *Physical Review Special Topics - Physics Education Research*, 8(2), 2012.

[4] C. H. Crouch and E. Mazur. Peer instruction: Ten years of experience and results. *American journal of physics*, 69(9):970–977, 2001.

[5] H. Danielsiek, W. Paul, and J. Vahrenhold. Detecting and understanding students' misconceptions related to algorithms and data structures. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education*, pages 21–26, 2012.

[6] C. D'Avanzo. Biology concept inventories: overview, status, and next steps. *BioScience*, 58(11):1079–1085, 2008.

[7] L. Ding, R. Chabay, B. Sherwood, and R. Beichner. Evaluating an electricity and magnetism assessment tool: Brief electricity and magnetism assessment. *Physical review special Topics-Physics education research*, 2(1):010105, 2006.

[8] J. Epstein. Development and validation of the calculus concept inventory. In *Proceedings of the ninth international conference on mathematics education in a global community*, volume 9, pages 165–170, 2007.

[9] M. F. Farghally, K. H. Koh, J. V. Ernst, and C. A. Shaffer. Towards a concept inventory for algorithm analysis topics. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, pages 207–212, 2017.

[10] K. Goldman, P. Gross, C. Heeren, G. Herman, L. Kaczmarczyk, M. C. Loui, and C. Zilles. Identifying important and difficult concepts in introductory computing courses using a delphi process. *SIGCSE Bulletin*, 40(1):256–260, 2008.

[11] R. R. Hake. Interactive-engagement versus traditional methods: A six-thousand-student survey of mechanics test data for introductory physics courses. *American journal of Physics*, 66(1):64–74, 1998.

[12] S. Hamouda, S. H. Edwards, H. G. Elmongui, J. V. Ernst, and C. A. Shaffer. A basic recursion concept inventory. *Computer Science Education*, 27(2):121–148, 2017.

[13] G. L. Herman, C. Zilles, and M. C. Loui. A psychometric evaluation of the digital logic concept inventory. *Computer Science Education*, 24(4):277–303, 2014.

[14] D. Hestenes, M. Wells, and G. Swackhamer. Force concept inventory. *The Physics Teacher*, 30(1):141–158, 1992.

[15] N. Jorion, B. D. Gane, K. James, L. Schroeder, L. V. DiBello, and J. W. Pellegrino. An analytic framework for evaluating the validity of concept inventory claims. *Journal of Engineering Education*, 104(4):454–496, 2015.

[16] K. Karpierz and S. A. Wolfman. Misconceptions and concept inventory questions for binary search trees and hash tables. In *Proceedings of the 45th ACM Technical Symposium on Computer Science Education*, pages 109–114, 2014.

[17] S. Krause, J. Birk, R. Bauer, B. Jenkins, and M. J. Pavelich. Development, testing, and application of a chemistry concept inventory. In *34th Annual Frontiers in Education Conference*, 2004.

[18] J. Libarkin, E. Ward, S. Anderson, G. Kortemeyer, and S. Raeburn. Revisiting the geoscience concept inventory: A call to the community. *GSA Today*, 21(8):26–28, 2011.

[19] D. P. Maloney, T. L. O'Kuma, C. J. Hieggelke, and A. Van Heuvelen. Surveying students' conceptual knowledge of electricity and magnetism. *American Journal of Physics*, 69(S1):S12–S23, 2001.

[20] M. C. Parker, M. Guzdial, and S. Engleman. Replication, validation, and use of a language independent CS1 knowledge assessment. In *Proceedings of the 12th ACM Conference on International Computing Education Research*, 2016.

[21] L. Porter, S. Garcia, H.-W. Tseng, and D. Zingaro. Evaluating student understanding of core concepts in computer architecture. In *Proceedings of the 18th ACM conference on Innovation and technology in computer science education*, pages 279–284, 2013.

[22] L. Porter, D. Zingaro, C. Lee, C. Taylor, K. C. Webb, and M. Clancy. Developing course-level learning goals for basic data structures in CS2. In *Proceedings of the 49th ACM technical symposium on Computer Science Education*, pages 858–863, 2018.

[23] C. Y. Read and L. D. Ward. Faculty performance on the genomic nursing concept inventory. *Journal of Nursing Scholarship*, 48(1):5–13, 2016.

[24] P. M. Sadler, H. Coyle, J. L. Miller, N. Cook-Smith, M. Dussault, and R. R. Gould. The astronomy and space science concept inventory: development and validation of assessment instruments aligned with the k–12 national science standards. *Astronomy Education Review*, 8(1):010111, 2010.

[25] M. K. Smith, W. B. Wood, and J. K. Knight. The genetics concept assessment: a new concept inventory for gauging student understanding of genetics. *CBE—Life Sciences Education*, 7(4):422–430, 2008.

[26] A. Stone, K. Allen, T. R. Rhoads, T. J. Murphy, R. L. Shehab, and C. Saha. The statistics concept inventory: A pilot study. In *33rd Annual Frontiers in Education Conference*, 2003.

[27] C. Taylor, D. Zingaro, L. Porter, K. C. Webb, C. B. Lee, and M. Clancy. Computer science concept inventories: past and future. *Computer Science Education*, 24(4):253–276, 2014.

[28] A. E. Tew and M. Guzdial. Developing a validated assessment of fundamental CS1 concepts. In *Proceedings of the 41st ACM Technical Symposium on Computer Science Education*, pages 97–101, 2010.

[29] C. S. Wallace and J. M. Bailey. Do concept inventories actually measure anything. *Astronomy Education Review*, 9(1):010116, 2010.

[30] K. C. Webb and C. Taylor. Developing a pre-and post-course concept inventory to gauge operating systems learning. In *Proceedings of the 45th ACM technical symposium on Computer science education*, pages 103–108, 2014.

[31] B. Xie, M. J. Davidson, M. Li, and A. J. Ko. An item response theory evaluation of a language-independent CS1 knowledge assessment. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, pages 699–705, 2019.

[32] M. Zeilik. Birth of the astronomy diagnostic test: Prototest evolution. *Astronomy Education Review*, 1(2):46–52, 2002.

[33] D. Zingaro, C. Taylor, L. Porter, M. Clancy, C. Lee, S. Nam Liao, and K. C. Webb. Identifying student difficulties with basic data structures. In *Proceedings of the 14th ACM Conference on International Computing Education Research*, pages 169–177, 2018.