Studied Questions in Data Structures and Algorithms Assessments

Iris Gaber
The Academic College of Tel-Aviv Yaffo
Tel-Aviv Yaffo, ISRAEL
gaber@mta.ac.il

Amir Kirsh
The Academic College of Tel-Aviv Yaffo
Tel-Aviv Yaffo, ISRAEL
amirk@mta.ac.il

David Sttater
Afeka College of Engineering
Tel-Aviv Yaffo, ISRAEL
davids@afeka.ac.il

ABSTRACT

Designing a proper exam that accurately evaluates students' knowledge and skills is one of the important tasks of every teacher. The format of the exams affects the way students learn throughout the course, and a well-designed exam can enhance meaningful learning. In this paper, we address this topic in the context of Data Structures and Algorithms courses, and argue that a good exam should contain questions that students have seen during the semester, and that the grading of those questions should be strict. We describe a case study which, over three semesters, supports the claim that answering these questions require the "Understand" level of Bloom's taxonomy, and that this strategy fosters more meaningful learning and better assesses students' knowledge.

CCS CONCEPTS

• **Social and professional topics** → **Computing Education**; **Student Assessment**.

KEYWORDS

Exams, Data Structures, Algorithms, Proofs, Bloom's taxonomy

ACM Reference Format:
Iris Gaber, Amir Kirsh, and David Sttater. 2023. **Studied Questions in Data Structures and Algorithms Assessments** . In *Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 1 (ITiCSE 2023), July 8–12, 2023, Turku, Finland.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3587102.3588843

1 INTRODUCTION

Composing good exams is one of the difficult challenges faced by teachers in any domain. The goal of exam is not only to assess students' capabilities; it also serves as a key element in the learning process, shaping the students' learning through their perceptions of the content and cognitive skills needed to do well in the exam [14], [13]. Students tend to selectively study material and acquire skills they believe will be useful to them in the exam [17]. Moreover, the exam format affects their learning, since they tend to strategically study in a manner that best matches their expectations of the exam format. For example, if an exam is likely to be filled with multiple-choice questions, primarily testing recall and basic comprehension, students might choose a surface-learning approach, spending study time memorizing facts and formulas [11].

ITiCSE 2023, July 8–12, 2023, Turku, Finland

ACM ISBN 979-8-4007-0138-2/23/07.
https://doi.org/10.1145/3587102.3588843

An unfortunate side effect of an exam is anxiety. Students are particularly likely to experience anxiety when they see a new and sometimes unclear question in the exam, which also affects their self-efficacy [4]. The anxiety from the exam, and the lack of self-efficacy, do not occur only while taking the exam, but also prior to it, while preparing for the exam. A student's level of self-efficacy and test anxiety directly impacts their academic success [5]. When students doubts themselves and their own ability to test well, their sole focus becomes worrying about poor grades, preventing them from focusing [4]. One attribute of a good exam is therefore lowering the students' anxiety level.

Another question that relates to exams is their fairness. Having an exam that is objectively fair means that success in the exam is well correlated with mastering the course's material. The students' subjective perception of the exam's fairness is almost as important. This feeling is based on the alignment between the exam and students' expectations from it [22]. Felder [12] claims that "What students hate more than anything else are examinations that they perceive as unfair" (p.1). He names the characteristics of an unfair exam, as perceived by students, among which are problems based on content not covered in lectures or homework assignments, and problems the students consider tricky, with unfamiliar twists that must be worked out on the spur of the moment. Students can better plan their learning for the exam when they have clear expectations, or instructions from their teachers, regarding how they should prepare themselves for the exam.

Graduate students pass dozens of exams throughout their studies, and therefore we expect their knowledge to reach at least some basic level in their domain, as we assume the exams they have passed guarantee it. However this is not always the case. Bodner's classic study in the field of chemistry asked incoming chemistry graduate students what was in the bubbles that are created when water boils. The responses would send shivers down the spine of any chemistry educator: while 70% of the students did correctly answer that the water molecules were entering the gas phase, 20% of students claimed the bubbles were air or oxygen. Most disturbingly, a full 5% of chemistry graduates claimed the bubbles contained hydrogen and oxygen gas [7].

Bodner's paper presents a list of ridiculous answers given by chemistry majors to simple questions. This shows that it is indeed possible to pass through an entire undergraduate chemistry program carrying substantial misconceptions, which most chemistry educators would consider worrisome.

In what follows, we make a suggestion regarding exams in the field of Data Structures and Algorithms, and address the aspects of a good exam we have discussed above.

2 COMPUTER SCIENCE THEORY STUDIES

In Computer Science (CS) studies we distinguish between two main categories: programming (practical) courses and computer science theory courses. Since only a small fraction of the graduates will pursue an academic career, the goal of the theory courses is to provide them the tools that they may need in their professional life, such as the ability to choose between algorithms, implement known algorithms or develop algorithms based on similar algorithms that they have learned. It is worth mentioning that one of the goals of the theory CS courses, corresponds to those of the practical programming courses, is gaining good programming skills that are associated with in-depth algorithmic thinking [16]. Bruce et al. [8] presents several examples in which programmers in the industry need to evaluate and select algorithms, and how this task requires the kind of thinking fostered by the study of mathematics and CS theory courses.

The case study described in this paper follows "Data Structures" (DS) and "Algorithms" courses. We refer to the courses that belong to the knowledge area "AL-Algorithms and Complexity: Core-Tier1," as defined in the ACM and IEEE CS 2013 Curriculum [9]. The names and the exact syllabi of the courses vary from one institute to another, and the material may span multiple courses.

The material covered in our DS course includes basic data structures (such as list, queue, stack, search trees) and advanced data structures (such as hash tables and heap). The material covered in our "Algorithms" course includes classical algorithms, such as topological sort, searches in graphs (BFS, DFS), shortest paths, minimum spanning trees, flow in networks and string matching. In both courses, we analyze the complexity of algorithms and operations on data structures using asymptotic analysis, and prove their correctness rigorously.

Porter et al. [18] formed an expert panel in CS education and sketched learning goals for a DS course, aiming for students to be able to: (1) Analyze the 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.

The courses' learning goals can also be analyzed based on the Bloom's taxonomy. Bloom's taxonomy [6] is a model for cognitive perception. The classic model was revised by Anderson et al. [2], and later interpreted for CS assessment by Thompson et al. [21]. The levels of the taxonomy are: "Remember," "Understand," "Apply," "Analyze," "Evaluate" and "Create". It is commonly agreed that undergraduates are expected to experience the entire taxonomy, being able to perform action on all levels of the taxonomy to some extent. It almost goes without saying that it is not enough to aim only for the "Remember" level, in any undergraduate domain.

Momsen et al. analyzed assessments in science courses, showing that exam questions mostly assess the two lower order cognitive skills [17]. Simon et al. [19] specifically addressed DS course exams. They analyzed 76 exams from 14 institutes and found that most questions in the exams are "implementation" and "interface" questions (asking for the data structure's implementation and features) while the more sophisticated "application" questions (which require using the data structure to solve a problem) are the minority. The authors see that as a surprise, since the literature that they had reviewed clearly opted for the latter kind of higher level questions. They offer several reasons for this result, including the difficulty of presenting questions of reasonable length when the questions are more sophisticated or were not presented in class, and the difficulty teachers have in grading such questions.

One may argue that if students are merely required to repeat something they have seen, this necessarily relates to the "Remember" level of the Bloom taxonomy. Our work challenges this perception. We argue that in a closed book exam that requires students to recall solutions to problems, there is a need to "compress" the details which requires understanding. We will further establish this claim in this paper. On the other hand, we claim that many exams, which seemingly require "Analyze" and "Apply',' can actually be solved by merely memorizing pre-cooked solutions.

3 OUR PROPOSAL

We conduct our experiment in order to test the following claims: (a) it is good practice for tests to include questions that the students have seen before, and (b) these questions, once given, should be graded strictly. The questions can be classic algorithms presented in class, including their proofs of correctness and runtime analysis, class exercises, and homework exercises for which a solution was published. Hereafter we refer to these questions as **studied questions**. The students should be told in advance that some of the questions in the exam are studied questions. It is also important to notify the students that the grading is going to be strict, so they can set their expectations accordingly and be prepared.

We argue that applying the above:

(1) Encourages meaningful learning among the students.
(2) Reflects their understanding of the material.
(3) Enhances the students' proof understanding skills.
(4) Reduces the students' anxiety and promotes their self-efficacy.

We do not suggest that the entire exam should be based on studied questions, but we argue that the ratio should be towards 70-30 in favor of studied questions. Below, we will elaborate on our experience and what led us to this conclusion.

The justification for the suggestion relies on the following claims.

Claim 1 - Memorizing dozens of questions is impossible.

The first level in Bloom's taxonomy is "Remember." Perhaps an instant reaction to our suggestion is that studied questions fall into that category. This is far from the truth. It is impossible to memorize dozens of questions, and students must therefore **understand** the essence of a question in order to remember its solution. This means that such questions would indeed test the students' understanding of the material. If students understand a proof, they can "condense" it in their minds, remembering a smaller amount of specific data. Indeed if someone were to try and memorize the 18-digit number "123581321345589144," they would find it quite difficult, but if they

noticed that this is the beginning of a Fibonacci sequence, it would make the task of reproducing this number very easy. Thus, repeating a proof or an exercise that was shown in class would be an indicator of understanding.

An important component in this approach is strict grading. Studied questions do the job only if teachers do not give points for "going in the right direction." It is possible to remember the general solution of a problem. For example, students may remember that a specific proof calls for proof by contradiction, but without a deep understanding they cannot supply an accurate proof.

Claim 2 - Giving studied questions encourages meaningful learning and understanding.

This claim results from Claim 1. If students cannot memorize dozens of questions, they do not have any other choice but to understand their essence in order to succeed in the exam. Moreover, they know that grading will be strict, so they cannot try to memorize only the general spirit. Also, when students know that the exam is going to contain questions they have already seen, it motivates them to learn and understand the material rather than "learn for the exam". In many cases "new" questions can be partially solved by students without any real understanding. Following the first claim, only a deep understanding of a proof can make the student supply an accurate solution, and gain points for it. Teachers can grade studied questions more harshly, since the students have already seen their solutions.

Claim 3 - New questions are often graded generously and as a result do not accurately reflect students' understanding.

Some students pass exams by scoring partial grades on questions that they do not know how to solve only because the question is new and teachers feel they should "give something" for answers that contain some connection to the correct answer even when the connection is actually very loose. This is bad for two reasons: first, the exam does not fully reflect the students' capabilities, and second, students mistakenly feel that they have mastered the material. Ironically, we believe that it is giving "new" questions that tests the students' memory (the first level in Bloom's taxonomy). Students who go over many solutions to questions learn the 'trick' of the questions and the general direction for solving them, and do not go deeper to the understanding of the details. When they see a new question in their exam, they can relate it to some similar question they went over while preparing for the exam, and teachers tend to give partial grades for answers that are partially correct. Different teachers, of course, grade differently, however, as seen in [1], clearly answers that are not full and accurate still get points.

Claim 4 - Studied questions reduce anxiety and promote self-efficacy.

Questions that students recognize can reduce the natural stress students feel while solving exams. Even when the question is difficult, having seen its solution makes the students more confident. The anxiety is reduced not only during the exam itself but also during preparation for the exam, due to the knowledge that it will focus on questions that the students have encountered before.

Claim 5 - Studied Questions promote students' proving skills.

There have been many studies showing that many undergraduate students lack the ability to produce new proofs (e.g. [10, 15]). Asking a student to present a proof that was already discussed in class can, in this context, be seen as a reasonable compromise, as a way to test the students' proof understanding. Relying again on claim 1, that students cannot memorize all the material, this would still assess students' understanding of proofs and their ability to present an accurate proof. Moreover, using studied questions makes it possible to include more challenging questions that otherwise would be considered too hard for an exam.

Claim 6 - Studied questions are challenging hence assess well.

We argue that studied questions still pose a challenge, and that they can serve the purpose of assessment and grading well, as a major part of an exam. This claim again relies on claim 1, that students cannot just memorize the material and the challenge of answering studied questions correctly requires understanding.

In the following sections we describe our findings, both quantitative and qualitative, in support of our claims.

4 THE RESEARCH

The authors have been teaching "Data Structures" and "Algorithms" courses for over twenty years. We teach in one of the largest colleges in the country, and our students have high grades in the acceptance exam and in the matriculation exam in mathematics (significantly higher than the average). In the past we used to compose exams that contained mostly "new" questions, believing that this is the right way to test students' understanding. In the following sections, we describe what led us to believe the opposite.

4.1 Data Structures 2021B cohort

In the class of 2021*B*, one of the questions in the exam was a question we solved in class. Clearly we considered it to be a "gift" question. The question is presented below:

Question 1 - Implementing a Queue with Stacks:

*Given two stacks S*1 *and S*2 *(working in the LIFO method) as black boxes, with the regular methods: "Push", "Pop", and "isEmpty", you need to implement a Queue (specifically : Enqueue and Dequeue working in the FIFO method). Assume there are n Enqueue/ Dequeue operations on your queue. The time complexity of a single method Enqueue or Dequeue may be linear in n, however the total time complexity of the n operations should also be* $\Theta(n)$.

Figure 1 presents the trivial solution, which does not meet the complexity requirement, and the efficient solution, which does not return the elements back to S1 after a "Dequeue" operation. The proof of the linear complexity uses the amortized technique.

Complexity Analysis:

On every element that enters the structure there are at most four operations being made. It can be pushed once to S1, popped once out of S1, pushed once to S2, and popped once out of S2. Every operation costs $\Theta(1)$. Therefore the contribution of a single element to the total complexity is $\Theta(1)$. Since there can be at most n elements (all together there are n operations) the total cost is $O(n)$ as was requested.

```
Enqueue(x) :
  S1.push(x)

Dequeue() :
  while S1 is not empty
    S2.push(S1.pop())

  top = S2.pop()

  while S2 is not empty
    S1.push(S2.pop())

  return top
```

Inefficient Solution

```
Enqueue(x) :
  S1.push(x)

Dequeue() :

  if (S2 is not empty)
    return S2.pop()

  else
    while S1 is not empty
      S2.push(S1.pop())

    return S2.pop()
```

Efficient Solution

Figure 1: Implementing a Queue with Stacks - Solutions

While checking the exams, we noticed that though many students gave the correct answer, some other students remembered vaguely that there was some trick regarding not returning the elements to $S1$, however their solutions were wrong. Since the students saw the solution in class, we were surprised. We decided to pass out a questionnaire regarding the question and ask the following questions:

(1) While solving the question, did you recall solving it in class?
(2) If the answer is "yes" - Did you recall the solution?
(3) Do you think the question was "fair"?

There were 21 responses and in Figure 2 we present the answers in percentages. Although 76.2% recalled seeing the question and its solution in class, only 14.3% answered that they remembered the solution accurately, and the majority of students (61.9%) answered that they either remembered only the main trick or remembered the solution vaguely. This is not a desirable situation as it may imply that the studying was shallow.

We interviewed some students that admitted they have indeed remembered the question from class, but did not remember the solution exactly. Following are two samples of the interviews:

Ron:

Ron: "When I solved the exam I remembered seeing this question in class."

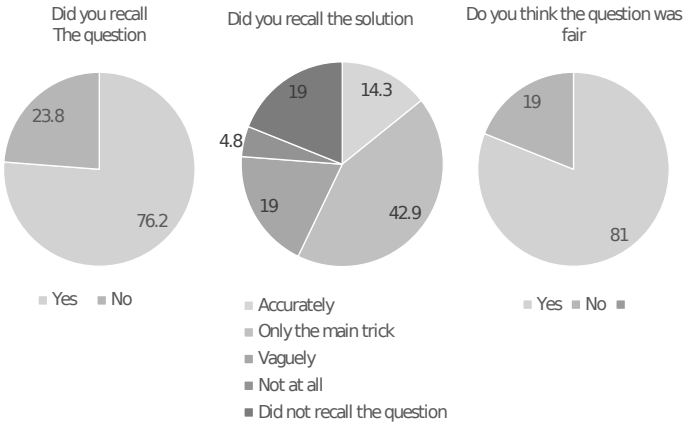

Figure 2: Questionnaire regarding Question on Stacks

Us: "Did you go over the solution while studying for the exam?"
Ron: "Not really. It didn't appear in past exams and the last three days before the exam I solved past exams."

Anna:

Us: "How did you prepare for the exam?"
Anna: "I went over my class notes and then solved past exams. When something was not clear to me I went back to my notebook"
Us: "And this question was familiar to you?"
Anna: "I remembered the solution from class."
Us: "But did you read it as part of the studying?"
Anna: "I don't think so.. Maybe."

Evidently many students did not pay enough attention to the exercise and spent most of their time preparing for the exam by solving past exams. We decided to examine the subject more deeply in the following course, "Algorithms".

4.2 Algorithms 2021S cohort

Intrigued by the results of Data Structures 2021B cohort, we made a little experiment in following course we taught, "Algorithms". Our final exam was composed of 7 questions, 5 of which were *studied questions*, meaning they were taken as-is from the course notebook or homework assignments. The students were informed two weeks before the exam about its format and were told to carefully go over the claims and exercises presented to them during the course.

When telling our colleagues about it, all of them suspected the grades would be very high. However, this was not the case at all. Only 74 students out of 124 (59.6%) passed the test and the average score was 62.18. The grades were actually lower than usual. A possible reason is the grading. We conjecture that many teachers, including ourselves, when asking "new" questions and receiving an answer that is even slightly in the right direction, tend to grade the answer with some grace. Sometimes we tend to think that students overall knew what they were doing but did not articulate well enough. However, when grading a question for which the answer was shown to the students, we are more harsh.

Here is a case of one such studied question that appeared in the exam. The question was given to students in their homework and was later solved in class.

Following is the question and a correct proof for its claim.

Question 2 - Minimum Spanning Tree:
Let $G(V, E)$ be a non-directed simple connected weighted graph. Let $e = (u, v)$ be an edge that belongs to a cycle such that its weight is larger than all the weights of edges in the cycle. Prove or disprove: The edge e does not belong to any minimum spanning tree of G.

Proof: *Assume by contradiction that there is a minimum spanning tree T that contains e. If we remove e from T we get two connected components, one containing u and one containing v. Since e belongs to a cycle there is another path in G from u to v that does not contain e, namely the path $C \setminus e$. One of the edges in that path must connect between the two components. We name it f. So if we add f to $T \setminus e$ we receive a new spanning tree, $T' = T \cup f \setminus e$. Since the weight of e is larger than all the weights of edges in the cycle it holds that $w(f) < w(e)$ meaning T' weighs less than T in contradiction to T being a minimum spanning tree.*

A similar phenomenon to what happened in the Data Structures 2021B course occurred again. Many students recalled that some switch should be made between edges in T, however they did not necessarily understand the proof. A common inaccurate proof was the following:
Inaccurate Proof: *Assume by contradiction that there is a minimum spanning tree T that contains e. Since e belongs to a cycle, we can choose another edge f from C such that adding f to T creates a cycle in $T \cup f$. Let $T' == T \cup f \setminus e$. Since the weight of e is larger than all the weights of edges in the cycle, it holds that $w(f) > w(e)$ meaning T' weighs less than T in contradiction to T being a minimum spanning tree.*

The proof implies that any edge $f \in C$ that we add to T creates a cycle, which is true, however if we are not careful in choosing f we can choose f that does not create a cycle with e and the proof collapses.
Out of 124 students, only 26 (21%) received all points. 86 students (69%) made that mistake.

After holding the exam and before publishing the grades, we distributed a questionnaire to the students regarding the exam. Two of the questions referred to the above studied question.

We asked:

(1) What grade do you expect to receive on it?
(2) How did you feel when you saw this question?

We received 73 responses out of 124 examinees.

Regarding Question (1), out of the 73 students, 35 of them thought they received all the points while clearly not all of them did. We conclude these students settled for a shallow study without a deep comprehension of the details.

Also, the average of the expected grades was 10.9 out of 15 points for a full answer, where the actual average was 4.6.

This supports claim 3, as students could check their answer against the correct one as written in their notes (the question was solved in class), but did not think their mistakes or inaccuracies were serious or expected a more generous grading based on a partial or somewhat flawed answer.

Table 1: How did you feel seeing the spanning trees question

Answer	Num Students	Percentage
Great	53	72.6%
Ok	18	24.7%
Mediocre	2	2.7%
Bad	0	0%

Regarding Question (2), almost all students (97.3%) answered "Great" or "Ok," as can be seen in Table 1.

We also interviewed a few students after they got back their grades. Here are some of the answers we received from students regarding their reactions to a studied question:
Eva:
"I was really calm and relaxed when I saw this question."
Sean:
"I chose to begin with this question. It was familiar from class and it gave me a boost of energy."
Lia:
"It was very nice to see a question I was sure I recognized."
This supports claim 4.

4.3 Data Structure 2022A cohort

The following semester, 2022A, we decided to take a different approach. In contrast to the Algorithms 2021S cohort, we told the students in this one, throughout the course, that their final exams will contain 70% studied questions. We repeated this multiple times and even presented on the course's website the example of *Question 1 - Implementing a Queue with Stacks*, which was given in Data Structures 2021B's final exam.
Here, for example is one of the studied questions presented in their exam:
Question 3 - Disjoint Sets
The abstract data type "disjoint sets" ("union find") was implemented on the set {1,2,...,n} using trees saved in an array, controlled using the "union by size" and "path compression" methods.

- *Part 1 - Given an array of size n, after some union and find operations were performed, prove that a single find operation costs at most $O(log(n))$.*
- *Part 2 - Given an array of size n, after performing $MakeSet(i)$ for each $1 \leq i \leq n$, describe a sequence of operations after which a single find operation costs $log(n)$.*

Both parts of the question are studied questions which were solved in class. However, the first part is more difficult and requires induction, whereas the second part is easier and requires only presenting an example.
Since the grading is affected by the grader (see [1]), in Table 2 we present the success rates of the two parts of the questions divided into three categories: how many did perfectly, how many did so-so and how many received zero.

There were 222 examinees, and as can be seen in the first part, only 20.2% received all points, whereas in the second part, which was easier, 56% of the students received all points. We compared the two questions using chi square and got a significant result, concluding that the distributions for the results of these two questions

Table 2: Distributions of Question 3 - disjoint sets

Answer	Num Students	Percentage
First part:		
Received all points	45	20.2%
Mediocre	27	12.1%
No points	150	67.5%
Second part:		
Received all points	125	56%
Mediocre	67	30.2%
No points	30	13.5%

are two different distributions (with a chi-square of 134.67 and p-value < 0.00001). The fact that two studied questions get such different success results means that not all studied questions are the same, which supports the claim that not all studied questions can be just memorized. We would emphasize that the length of the correct answer for each of these questions was about the same (half a page) and both these studied questions were taught in class in the same lesson. The only difference that we see is that indeed one of them is more complicated, a difference that should not affect simple memorizing, but most probably affects understanding.

The difference in the success rate of the two parts, which correlates with their difficulty level, raises two points. First, the students most probably did not memorize the solutions without understanding them. The solutions of the question cannot be attributed to the first cognitive level, since the students could have memorized both solutions and there would have been no difference in the success rates between them. This supports claim 1.

Second, the solutions differed in their phrasing, compared to the solution presented in class, which clearly indicates that the students who solved a question correctly did so based on their understanding and wrote it down in their own words. This supports claim 2.

Third, the first part is a difficult question. Probably we would not have given such a question had it not been a studied question. This supports claims 2 and 5.

Finally, even after seeing the solution in class, only 20% could give the exact right solution, getting all points for this question, and a very large percentage (67.5%) were completely off. This means that studied questions still serve as a meaningful challenge and assessment. This supports claim 6.

5 DISCUSSION

Based on our experience, we suggest that teachers integrate a large percentage of "Studied Questions" in final exams, and let the students know about this in advance. We claim that this will achieve the desired outcome of forcing the students to learn the material, read the notebook and go over the lectures, rather than trying to solve many past exams and hoping to encounter in their exam questions that resemble something they saw.

Students nowadays have access to a vast amount of questions from past exams, so it is likely they will encounter many classical questions. Moreover, the "new" questions we give are often variations of such problems that appeared in the past. When students present a solution to a complicated new problem, which is only partially correct, it is difficult for teachers to determine whether the student recalled the question after having seen a variation of it without really understanding it. Therefore, many teachers may give partial grading for inaccurate solutions, which encourages students to learn techniques of how to pass exams without understanding the material, by memorizing answers to questions and using them even when they do not entirely suit.

We showed that having "studied questions" in the exam does not impair the assessment process. Solving studied questions is not trivial, and, in our case study, still produced variance within the grades, so these questions can still differentiate between students who understood the material and those who did not.

Studied questions are also perceived as fair. Students do not consider something they saw as tricky or difficult. It also promotes their self-efficacy and reduces their anxiety, as presented in the questionnaire that we distributed.

Another pertinent point worth addressing is that there is usually a big difference between the grading of different graders. Albluwi [1] found that the disagreement between graders in CS1 exams is significant, and that there is no consensus on how code writing questions should be graded, due to the subjective judgment of each grader. Grading studied questions strictly may reduce this variance, which also serves well the fairness of the exam.

We mentioned the problem of graduates who lack basic knowledge in their domain. We believe that if the majority of the exam requires showing actual knowledge, as opposed to presenting the skill of what we argue is "passing an exam," students will inevitably have better knowledge in their domain. We also believe that they deserve to pass an exam if they at least show the knowledge gained in the course.

Another aspect of studied questions is their contribution to the legitimacy of giving "proof" questions in exams. The question of whether or not students can and should be required to write rigorous correctness proofs for algorithms in exams is in dispute. Some teachers believe an algorithm with no correctness proof is incomplete, while others doubt its applicability [3]. Sowder and Hazel [20], in the field of mathematics, claim that students often demonstrate an immature level of PUPA (Proof Understanding, Production and Appreciation), which is expressed by unsatisfactory ability to produce proofs, but also by severe problems in appreciating proofs. Studied questions are something of a compromise between the perhaps unrealistic expectation that students be able to supply a proof of their own, to the need to find a way to test their understanding of one.

There are several ways to present "Studied Questions" in an exam. One of the ways is to create a pool of questions that would be shared with the students, then have the exam draw questions from that pool. Our experience is that when a large percentage of the exam (between 50% and 70%) is based on studied questions, it enhances meaningful learning, improves the fairness of the exam and reduces students' anxiety.

REFERENCES

[1] Ibrahim Albluwi. A closer look at the differences between graders in introductory computer science exams. *IEEE Transactions on Education*, 61(3):253–260, 2018.

[2] Lorin W Anderson, David R Krathwohl, et al. A taxonomy for learning, teaching, and assessing: A revision of bloom's taxonomy of educational objectives. *Allyn & Bacon. Boston, MA (Pearson Education Group)*, 2001.

[3] Michal Armoni. On the role of proofs in a course on design and analysis of algorithms. *ACM SIGCSE Bulletin*, 38(4):39–42, 2006.
[4] Albert Bandura. Perceived self-efficacy in cognitive development and functioning. *Educational psychologist*, 28(2):117–148, 1993.
[5] Jennifer Barrows, Samantha Dunn, and Carrie A Lloyd. Anxiety, self-efficacy, and college exam grades. *Universal Journal of Educational Research*, 1(3):204–208, 2013.
[6] Benjamin S Bloom, Max D Engelhart, EJ Furst, Walker H Hill, and David R Krathwohl. Handbook i: cognitive domain. *New York: David McKay*, 1956.
[7] George M Bodner. I have found you an argument: The conceptual knowledge of beginning chemistry graduate students. *Journal of Chemical Education*, 68(5):385, 1991.
[8] Kim B Bruce, Robert L Scot Drysdale, Charles Kelemen, and Allen Tucker. Why math? *Communications of the ACM*, 46(9):40–44, 2003.
[9] Strawman Draft. Computer science curricula 2013. *ACM and IEEE Computer Society, Incorporated: New York, NY, USA*, 2013.
[10] Tommy Dreyfus. Why johnny can't prove. *Forms of mathematical knowledge: Learning and teaching with understanding*, pages 85–109, 1999.
[11] Abigail Entwistle and Noel Entwistle. Experiences of understanding in revising for degree examinations. *Learning and instruction*, 2(1):1–22, 1992.
[12] Richard M Felder. Designing tests to maximize learning. *Journal of Professional Issues in Engineering Education and Practice*, 128(1):1–3, 2002.
[13] Kristi L Hall, Jessica E Watkins, Janet E Coffey, Todd J Cooke, and Edward F Redish. Examining the impact of student expectations on undergraduate biology education reform. *arXiv preprint arXiv:1105.6349*, 2011.
[14] David Hammer, Andrew Elby, Rachel E Scherr, and Edward F Redish. Resources, framing, and transfer. *Transfer of learning from a modern multidisciplinary perspective*, 89, 2005.
[15] Gila Hanna. Proof, explanation and exploration: An overview. *Educational studies in mathematics*, 44:5–23, 2000.
[16] Katalin Harangus and Zoltán Kátai. Algorithmic thinking vs. text comprehension. *Procedia Manufacturing*, 22:1031–1037, 2018.
[17] Jennifer Momsen, Erika Offerdahl, Mila Kryjevskaia, Lisa Montplaisir, Elizabeth Anderson, and Nate Grosz. Using assessments to investigate and compare the nature of learning in undergraduate science courses. *CBE Life Sciences Education*, 12(2):239–249, 2013.
[18] Leo Porter, Daniel Zingaro, Cynthia Lee, Cynthia Taylor, Kevin C Webb, and Michael 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.
[19] Beth Simon, Mike Clancy, Robert McCartney, Briana Morrison, Brad Richards, and Kate Sanders. Making sense of data structures exams. In *Proceedings of the Sixth international workshop on Computing education research*, pages 97–106, 2010.
[20] Larry Sowder and Guershon Harel. Case studies of mathematics majors' proof understanding, production, and appreciation. *Canadian Journal of Math, Science & Technology Education*, 3(2):251–267, 2003.
[21] Errol Thompson, Andrew Luxton-Reilly, Jacqueline L Whalley, Minjie Hu, and Phil Robbins. Bloom's taxonomy for cs assessment. In *Proceedings of the tenth conference on Australasian computing education-Volume 78*, pages 155–161, 2008.
[22] Norman L Webb. Criteria for alignment of expectations and assessments in mathematics and science education. research monograph no. 6. *ERIC*, 1997.