Algorithm Analysis in OpenDSA – An Online, Open Source, Interactive Platform for Data Structures

Farbod Raubetean

Thesis submitted to the faculty of the
Åbo Akademi University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Science

Under the direction of:
Linda Mannila

May 2016
Åbo, Finland

ABSTRACT

New methods and opportunities for learning have appeared through the arrival of electronic learning frameworks. Over the past few years, several eLearning environments have been created by experts in the field to make collaborative teaching possible for both teachers and learners. Project OpenDSA is an online, open source, interactive data structures and algorithms course with the objective of developing a complete interactive book for courses in data structures and algorithms. Though designed with the purpose of enhancing efficiency in student learning in mind, some of the chapters have proved quite challenging for the students. The chapter under scrutiny in this thesis has to do with algorithm analysis. Students have struggled with this particular chapter more than with any other.

The purpose of this thesis project is to discover the roots of the aforementioned challenges and devise creative and student-friendly tools and interactions for overcoming the learning obstacles. The first step this thesis has taken towards tackling the learning challenges in this chapter, has been to create a visualization for a set of growth rate functions inside an online graphing calculator. The next implementation step has been to create novel exercise types for the chapter summary exercises. The following five interactive question types for this chapter have been implemented: Matching questions, Ordering questions, Fill-in-the-blank questions, Error-guessing questions, and multiple-choice questions on code snippets. At least two sample questions for each of these question types have been developed.

Keywords: Algorithm Analysis, Algorithm Visualization, Computer Science, Computer Science Education, Data Structures and Algorithms.

ACKNOWLEDGEMENTS

I would first like to express my gratitude to my supervisor, Dr. Linda Mannila, of the Faculty of Information Technologies at Åbo Akademi University, who kindly and continuously helped me through the uneven journey of finishing this thesis. She worked side by side throughout all stages of this thesis and steered me in the right direction. I would also like to thank Professor Ralph Back, who initially introduced me to the lab Linda works and also helped me by steering me in the direction of the right people to talk to.

In addition, I would like to thank Professor Clifford Shaffer from the Department of Computer Science at Virginia Tech, who also helped me in framing and untangling my initial implementation ideas during his visit to Åbo Akademi University. Professor Shaffer and the people in his lab are the main people behind project OpenDSA, and in my few correspondences with them, they were always willing to help in providing me with the information I needed.

I also want to thank my friends and family here in Finland, without whose support and unconditional love I would not have been able to complete this journey and accomplish this thesis. Ultimately, I want to thank my parents who have always been there to help and support me in all stages of my life.

Farbod Raubetean Turku, May 2016

PREFACE

Education and learning have long been areas of great interest to me. I remember that I used to constantly find myself in situations of tutoring different individuals and groups of people and instructing them in effective ways by which they could grasp the information in a systematic manner, after which they would encourage me through their positive feedback by saying how much I had been able to teach them. Having many such positive experiences in my early life, and having been equipped with the proper education, I chose to pursue a teaching career for several years after completing my bachelor's degree, during which time I became involved with the idea of developing an eLearning platform for the institute at which I was working and teaching. The fact that I had completed my bachelor's degree in the field of computer science, together with my teaching experience, made me a good candidate for participating in developing such a platform. We developed a complete online course from start to finish, including various course materials, exams, quizzes, and so forth.

Although many traditional classroom teachers, except for me of course, were a bit skeptical about the whole eLearning idea, since they imagined that the system might eventually replace them in virtual classrooms due to its high level of availability and scalability, we did a good job of gaining their support by emphasizing the real reasons behind the need for eLearning, which were as follows: the fast changing nature of technology, the high demand for skilled teachers like them throughout the country, the lack of experts in their field of expertise, and the inability of some students to attend traditional classrooms for various reasons. After we finished implementing the system and the system had been in place for a few months, we found that both students and teachers were quite content with and grateful for the online learning platform. Many teachers realized that their initial ideas about the eLearning project—that it would make them redundant—had been incorrect and that the students now had at their disposal a very intuitive and flexible online learning system, one in which they were not bound to certain classroom hours and physical locations.

Based on this experience, I went on to challenge myself with several courses on user experience and interaction while working on my master's degree here in Finland. I found the courses tremendously interesting. Moreover, after doing some research at the different labs in our department and after looking into the different projects and research fields that they were involved in, I was introduced to an eLearning project called eMath, which is a European Union project having to do with creating a new method for teaching high school mathematics. I did some research on this project to find out more about it and how I might be able to find a thesis topic related to it, but eventually it did not work out. After finding out about OpenDSA, I became quite interested in it and realized this would actually be the platform for my thesis project, as it was a perfect match with my field of interest and prior experience. Since OpenDSA is a large-scale project that had already been going on for a number of years before my involvement, it took me quite a while to dig in, find the right people and connections, and become familiar with the current platform technologies. My supervisor was always there to help me through this process, for which I am very grateful. OpenDSA is a joint project between Åbo Akademi University, which is my home university, and Virginia Tech in the United States.

After sitting down with the project team and leader during their visit to Åbo Akademi University, and after a number of different consulting sessions with my supervisor, I chose to mainly work on designing and implementing different types of interactive chapter summary exercises and visualizations in a way that would facilitate the learning of the complex algorithm analysis chapter.

CONTENTS

PART I

RESEARCH OVERVIEW

CHAPTER 1

INTRODUCTION

This chapter provides an overview of the thesis, together with the motivation behind it.

1.1 Context

The intricacy of living in the twenty-first century has led to the increasing need for students at different educational levels to be supported in constant learning [2]. Personal learning environments have been developed to address this need and people nowadays tend to sometimes even prefer such forms of support over conventional classroom environments for various reasons. Consider the case of a person living and working hundreds or maybe even thousands of kilometers away from where a course is actually being held. In a conventional classroom setting, this individual would need to travel all the way to where the class is being held for every single session of the course. Much time and money would have to be spent for this individual to be able to attend the class. Moreover, the classroom time is also another strict restriction that the attendant has to keep in mind. However, in a personal learning environment, not only does this individual not need to spend lots of money and time attending the class, but also he or she could use the course material at his or her own convenience and not be bound by the time aspect of a conventional classroom.

A personal learning environment could be as simple as an interactive eBook or as complex as a complete virtual learning environment comprised of a combination of interactive content, blogs, feeds, podcast widgets, video and text chat, and so forth [1]. On the other hand, each of these modules could have varying levels of complexity. For instance, one interactive eBook could have many in-depth features and be, in practice, far more complex than a virtual learning environment that is comprised of different tools and gadgets. One factor that specifically shapes and defines the level of complexity of these platforms is the intended audience. A complete virtual learning environment that is developed for small children needs to have far less complexity than an online eBook developed for college-level students. Moreover, the modules would need to utilize different presentation techniques in the process of developing the interactive course content.

One example of a personal learning environment is the OpenDSA project [9], which is short for Open Source Interactive Data Structures and Algorithms and is mainly aimed at college level students. As the title rightly suggests, its goal is to develop a comprehensive online and interactive textbook for courses in data structures and algorithms (DSA), which is a mandatory and crucial subject in the field of computer science (CS). The textbook is comprised of several chapters dealing with different issues pertinent to the topic, among which my work mostly focuses on the algorithm analysis (AA) chapter. The main reason behind this choice was the result of my communication with the project leader and team, who maintained that this chapter has been the most intricate and difficult to learn for students out of all the chapters. AA mainly deals with calculating the amount of resources necessary to execute certain algorithms. Due to the nature of its topic, AA is heavily dependent on mathematical formulas and content that in most cases could lead to difficulty in acquiring a deep understanding of the subject by students and also cause them to lose focus and become bored. On the other hand, it also requires an acceptable level of mathematical proficiency by the students, and since some undergraduate students have a tendency to evade mathematics courses, this on its own could pose yet another challenge in understanding.

However, there is a strong emphasis on interaction in general in OpenDSA, which makes it uniquely different from conventional paperbound textbooks. Since students can sometimes easily become bored and lose focus while studying a book, incorporating interactive activities in an eBook can help minimize this problem. Now these interactive activities can have varying forms and be implemented in different locations, such as directly inside the main content and text as illustrations or as simple and small games, or else inside chapter summary exercises. The main objective of this thesis project is twofold. First, it carries out theoretical research on algorithm analysis, visualizations, evaluation methods, and their history in CS and provides an overview of the OpenDSA platform. Second, the thesis implements new visualizations and interactive types of questions for the AA chapter in the OpenDSA platform to help learners in the challenging process of learning this chapter.

1.2 Motivation

My background of working as a teacher for several years after finishing my studies at the bachelor's degree level led me to see the world around me from a different perspective and to learn a great deal about different educational methods and mechanisms. It was as if my eyes were opened to a whole new realm of viewpoints in life after that experience. Furthermore, along with my teaching experience, I had been studying CS since high school. After graduating with a vocational degree in CS, I further studied CS for two more academic degrees, namely an associate's degree and a bachelor's degree. Each of these factors played a large role in motivating me to pursue this particular thesis project.

My job experience as an innovation expert, in which I took part in developing a complete distance-learning platform for the institute that I was working for at the time, was yet another contributing factor behind my desire to pursue this thesis project. My role in that project was to design and develop interactive teaching material and also to regulate online exams and act as a platform support teacher. Consequently, the main groundwork that contributed to the motivation behind this thesis project had to do with a multidisciplinary combination of my academic and work experience together with my tangible work experience in the field of electronic and distance learning.

The particular project I worked on as part of this thesis is OpenDSA. As part of the ongoing OpenDSA project, eTextbook content has already been published online. The developers have access to the content for the purpose of continued development. After a few discussion sessions with my supervisor and the OpenDSA team, it turned out that the chapter in need of the most development efforts is the one on AA due to the difficult nature of the topic and the students' consistent problems in learning this specific chapter. Therefore, I took on the responsibility of further developing this chapter. The focus from the start was to develop new types of interactive visualizations and exercises to help students in the process of learning and understanding the chapter better.

The main concepts that this chapter deals with in general are asymptotic AA, the growth rates of different algorithms, and calculating program running times. Concepts such as the upper bound and lower bound of an algorithm's time complexity are discussed as sub-sections to the larger notion of asymptotic AA, each of which is a mathematical function with a formal definition. The following definition is an accurate way of describing an upper bound, in which T (n) symbolizes the true running time of the algorithm and f (n) is an expression of the upper bound [16, 150]:

> For T (n) a non-negative function, T (n) is in set O (f (n)) if there exist two positive constants a and b such that T (n) $\leq af$ (n) for all $n > b$.

The chapter also discusses such ideas as classifying functions in asymptotic AA [16, 150]:

> Given functions f (n) and k (n) whose growth rates are indicated as algebraic equations, we would like to decide if one grows faster than the other. The best way to accomplish this is to take the limit of the functions as n grows towards infinity. If the limit goes to infinity, then f (n) is in Ω (k (n)), if the limit goes to some constant apart from zero, then f (n) = Θ (k (n)), and if the limit goes to zero, then f (n) is in O (k (n)).

The examples presented above clearly demonstrate the purely mathematical nature of the content of this chapter, which makes it difficult for students to grasp and learn. This signifies the importance of developing new visualizations and interactive types of questions in different sections of this chapter to help students learn these particular mathematical notions.

1.3 Structure of the Thesis

I began this thesis project by going through the course content of the AA chapter and taking notes on how to improve the presentation of the material. Though I was already familiar with the ideas and chapter content from my past studies, the review process helped me revisit my understanding of the topic and come up with a good estimate of its clarity of presentation and level of difficulty from a teacher's perspective. Eventually, I finished going through the whole chapter and wrote down every innovative idea that came to mind, which I deemed would be helpful in the chapter's presentation and in terms of the ability of students to understand it. The resulting report was sent to the OpenDSA team to start revising the chapter content based on these innovative ideas, which will be carried out as future work.

The thesis consists of two main parts: research overview and implementation. The research overview is mainly presented in chapters 2 and 3 of this thesis, whereas chapters 5 and 6 focus on the implementation process. The research overview part has five main subsections: algorithm analysis in CS education, visualizations in CS education, evaluation methods in CS education, from paper books to PDFs to interactive eBooks, and OpenDSA. The implementation part has two main subsections: material development and empirical evaluation. Chapter 4 presents the methodology, chapter 7 provides the discussion, and finally, chapter 8 provides some conclusions.

CHAPTER 2

BACKGROUND

This chapter provides the concise scientific background behind the algorithm analysis, visualizations, and evaluation methods in CS education. Moreover, interactive eBooks and eTextbooks are briefly introduced.

2.1 Algorithm Analysis in CS Education

2.1.1 Algorithms

An algorithm is any distinct computational procedure that takes a set of values as input and constructs a set of values as output. Thereby, an algorithm is a succession of computational steps that transforms the input into the output. An algorithm can also be considered as a measure to solve a specified computational problem. The problem declaration states the input/output relationship in conventional terms. The algorithm reports a certain computational procedure for attaining that input/output relationship. An algorithm is called correct if, for all input instances, it halts with the correct output. We maintain that a correct algorithm solves the specified computational problem. An incorrect algorithm, on the other hand, might halt at an incorrect answer. [153, 154, 164.]

If computer memory was free and computers were fast without limit, would one have any reason to study algorithms? The answer is yes, since one would still need to show that a solution procedure terminates and gives the correct answer. If computers were infinitely fast, any correct procedure for resolving a problem would do. However, one would most likely want her/his implementation to fall within the bounds of good software engineering practice, but she/he would most probably use whatever method was the simplest to implement. Computers may be fast, but they are not infinitely fast. Furthermore, memory may be cheap, but it is not free. Hence, computing time and memory space are limited resources. Algorithms that are efficient with regards to time and space will help one use these assets prudently. [153.]

2.1.2 Analysis of Algorithms

One reason why an algorithm is analyzed is to find its characteristics as a means to evaluate its suitability for different applications or to compare it with other algorithms for the same application. The attributes demanding attention are usually the main resources of time and space, in particular time. In other words, we are interested in finding out how long an implementation of a specific algorithm will run on a certain computer and how much space it will need. We normally try to keep the analysis separate from particular implementations. We focus rather on acquiring results for necessary characteristics of the algorithm that can be used to obtain accurate estimates of resource needs on difference machines. [151.]

The expression "analysis of algorithms" has been used to recount two divergent common approaches to settling the research of the performance of computer programs on a technological basis. The first, made popular by Aho et al. [152] and Cormen et al. [153] focuses on specifying the growth of the worst-case rendering of the algorithm. The second approach to the analysis of algorithms, made popular by Knuth [154, 155, 156, 157, 158], focuses on accurate characterizations of the worst-case, best-case, and average-case performance of algorithms utilizing a methodology that can be improved to manufacture accurate answers when desired.

Although the speed of hardware computation of basic operations has been considerably enhanced, efficiency still matters a great deal even in this day and age. This is because our hunger for computer applications has increased along with computer power. Many fields require a great increase in computation speed. Such areas include the simulation of continuous systems, medical applications, high-resolution graphics, information systems, and physical data interpretation. Furthermore, an algorithm may be so inefficient that, even with the speed of computation dramatically increased, it would not be viable to attain a result within a practical period of time. The time that the majority of algorithms take to execute is a non-linear function of the input size. This can lower their potential to benefit from the increase in speed when the size of input is large. [159.]

2.1.3 Learning in Analysis of Algorithms

Students usually struggle with learning different concepts in the analysis of algorithms. One reason behind this struggle is their poor math skills and problem solving abilities [8]. Mathematics is the main fundament behind the analysis of algorithms; therefore, students need to develop a good understanding of certain mathematical concepts, such as relations, summations, and logarithms, in order to successfully learn to analyze algorithms. Students who have a good background in math tend to perform better in learning how to analyze algorithms [161]. They are better able to understand data relationships, algorithm design, and scientific computations. This helps them to be better at solving problems and producing good designs based on the requirements. [160.]

The ability to solve problems is usually quite poor among computer science students. Beaubouef et al. [162] have explored ways to improve the problem-solving skills of computer science students. Students often have the most trouble solving basic word problems. The ability to solve a word problem is a math skill and it plays an important role in most computer science courses, including the analysis of algorithms. Students' syntactical knowledge is usually not the problem. The problem lies in their inability to form the algorithm in the first place. [160.]

2.2 Visualizations in CS Education

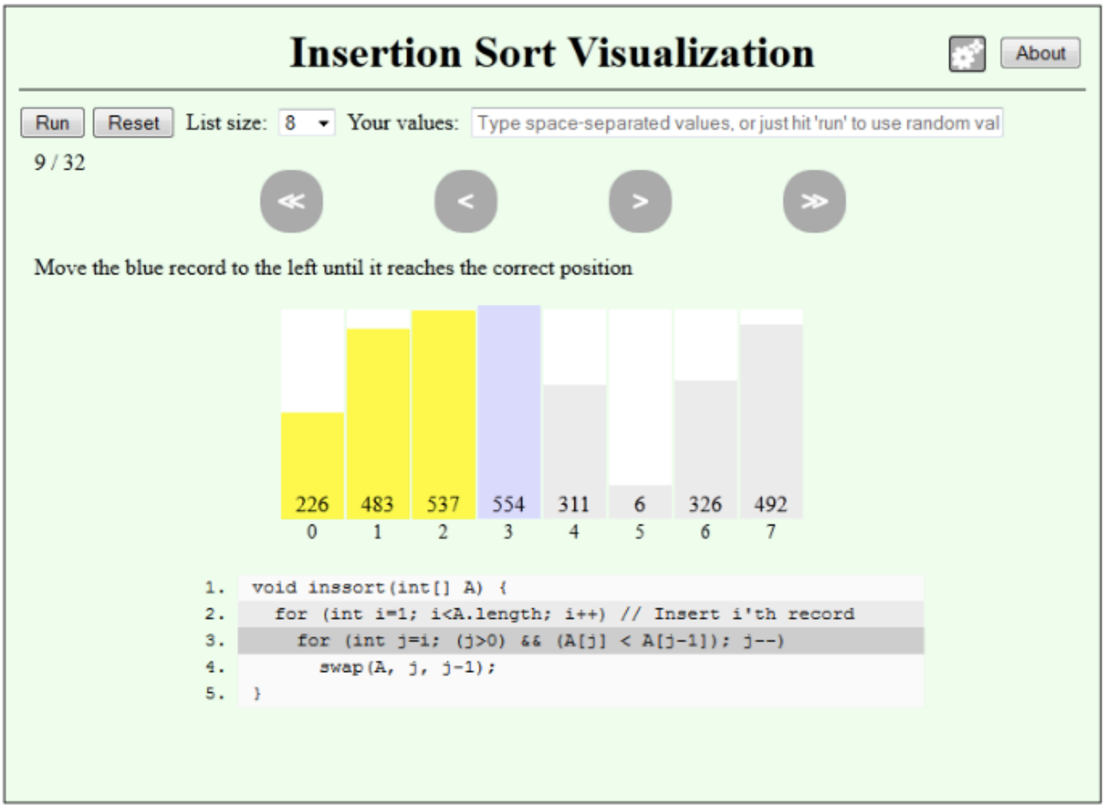
2.2.1 Incentives and History

Visualizations in computing have existed for several years, with one motivation for them having to do with the innate abstractness of the fundamental building blocks of this field [18]. Another important incentive behind the use of visualizations in CS education is to introduce changes to the already in-place teaching and presentation methods in the field [18, 19, 20]. New Internet-based technologies, Massively Open Online Courses together with new forms of interactive content delivery, such as Code Academy [163] and Khan Academy [11], are instances of drivers for this change. In most cases visualization, automated assessment and multimedia together have been able to dramatically change the idea of textbook and the approach to CS teaching [19]. However, previous studies in these areas have primarily focused on integrating interactive content with a learning management system as opposed to more recent studies [24, 25].

Visualization software first appeared in the late 1980s. Its main purpose was to help users interactively study and review graphical depictions of CS concepts [21, 22, 23]. When looking into the past, teachers used to cover blackboards and themselves with chalk when drawing complex figures depicting data structures and control flows, while still sometimes making mistakes in the process [20]. Today, computer-based visualization objects are capable of constructing physical depictions that would be difficult to create or make use of via other media [23]. Teachers and students of CS, together with programmers, make use of pictures and visualizations as aids to help convey and comprehend CS concepts nowadays [18, 20].

2.2.2 Effects of Visualizations on Learning

Visualization can be a strong tool for presenting program behavior in CS, since programs are intrinsically chronological and temporal and can be clearly expressed by animated objects that depict their processing routines and the changes in their core states over time [19, 20]. Moreover, numerous algorithms use recurrent computations that can be viewed effectively when expressed as a moving picture, thereby helping learners deduce what the program is up to by recognizing structure and the relationships of cause and effect [20]. Figure 1 below shows a visualization for the insertion sort algorithm. Animating programs for different purposes is not a small endeavor, however, since nearly all computer programs contain many complicated elements and features [19, 20]. It is important to decide what part of the program and what data to denote, and the most appropriate time to update their depictions while the program is running; likewise, it is important to select suitable execution and representation times and try to increase the applicable characteristics and decrease irrelevant details when formulating transparent, organized, and appealing graphic designs [20].



List size: 8 Your values: Type space-separated values, or just hit 'run' to use random val

9 / 32

Move the blue record to the left until it reaches the correct position

```
void inssort(int[] A) {
  for (int i=1; i<A.length; i++) // Insert i'th record
    for (int j=i; (j>0) && (A[j] < A[j-1]); j--)
      swap(A, j, j-1);
}
```

Figure 1. A visualization for the insertion sort algorithm.

The work by Mayer et al. [33] on multimedia learning mostly indicates that providing learners with materials through multiple conduits is usually more effective than using a single means to convey information [3]. The advantages might be related to reductions in the cognitive load together with inherent distinctions in media for communicating specific types of information [3, 33, 34]. Mayer et al. [33] argued that it is not a valid comparison to compare two forms of media, such as text and pictures, and ask whether one is better than the other for conveying the same information due to the fact that the same information cannot always be conveyed through two different means [3]. Instead, a better question would be to ask whether multiple information channels help learners build a better representation of the material [3].

In spite of the fact that most reviews of CS instructors propose a widespread belief that visualization technology influences learning in a favorable way, certain investigative studies done to prove the effectiveness of such visualization technology do not prove this point [18, 24, 26]. These particular studies indicate that from a learner's point of view, visualization technology may not be beneficial; in contrast, from an educator's point of view the visualization technology may just bring about too much overhead to make it worth the effort [18, 26]. However, close examination of past experimental studies on visualization effectiveness reveals that learners who are actively engaged with visualizations have generally performed better than those who passively view visualization technology [18, 26]. For instance, visualization technology has been effectively used to engage learners in activities such as answering planned questions about visualizations, forming their own input datasets, programming target algorithms, building their own visualizations, and making predictions concerning future visualization conditions [27, 28, 29, 30, 31, 32].

2.2.3 Animations and their Effects on Learning

Animations could help learners be able to make and test better predictions about what is going to happen at each step of the algorithm [4, 35]. Learners can do a better job in their learning tasks when they are able to make clear predictions as compared to a scenario in which they only passively take in the information [3, 4, 35]. It is also possible for a learner to make predictions based on a static textual presentation of an algorithm, but the benefit of an animation is that it will naturally encourage a learner to make predictions without pause and delay and will equip the learner with quick feedback about the correctness of their predictions [3, 4, 35].

In order to investigate the effects that animations might have on helping individuals understand an algorithm, it is important to define what we mean by "understand" [4]. Let us turn to a cognitive explanation of the understanding process [35, p. 5]. Tversky et al. [36] describe it as a congruence principle, whereas Scaife and Roger [37] try to explain the process as resemblance fallacy [35]. However, defining this term is still not a trivial task since understanding can be measured in various ways and there might be different types of understanding [4].

The use of algorithm animation does not automatically enhance learning since there are numerous other issues that also need to be taken into account [4, 35]. On the other hand, the level of importance of the algorithm animation's design is so great that a poorly designed algorithm animation could even lead to a decrease in the efficiency of the learning process [3, 4, 35]. Simply putting an animation together and offering it to learners is not sufficient; thus, an accurate task analysis should be done to demonstrate precisely what knowledge the animation will be helping the learner acquire [3, 4, 35].

2.3 Evaluation Methods in CS Education

2.3.1 Definition and Computer-based Evaluation

Evaluation is defined as the process of choosing, collecting, analyzing, and reporting valid information on the obtaining of educational objectives in order to facilitate making accurate judgments about the usefulness of teaching methods or an educational curricula [131]. However, several other authors have defined evaluation in different ways [132, 133, 134, 135, 136]. No matter how good the teacher, no matter how talented the students and how effective the teaching methodology, if no provision is made for a method by which the whole teacher-student interaction and learning process can be evaluated, then the educational efforts may be severely diminished [5]. Computers and electronic technology nowadays provide a multitude of ways to improve educational evaluation both in the classroom and in extensive testing environments [116]. With dynamic sound, visuals, user interaction, adaptivity to different test-takers, and the almost instantaneous reporting of results, computer-based evaluation broadly extends testing potential beyond the restrictions of conventional paper tests [116].

One inherent restriction to achieving the benefits of computer-based evaluation has to do with designing the types of tasks and questions through which computers can efficiently communicate with students while still collecting relevant measurement data [116]. One of the main question types in computer-based testing and numerous eLearning evaluations is the conventional multiple-choice question, which typically consists of a prompt followed by a set of answers; students then choose from among those answers [116]. According to some research, multiple-choice testing sometimes urges "poor attitudes toward learning and wrong deductions about its purpose" [117]. Nevertheless, tendencies toward genuine evaluation, performance evaluation, portfolio systems, dynamic assessment, and other methods favoring richer evaluations are usually based on resulting validity arguments about the harmful effects of limited evaluations in the classroom on both teaching and learning [118].

2.3.2 Media Inclusion

Some reviewers claim that in practice, the multiple-choice question type relies heavily on well-structured problems with algorithmic answers, and that in theory it points to a view of learning whereby knowledge is treated as an addition rather than as an integral part of developing knowledge structures [119, 120, 121, 122, 123]. Scalise et al. [116] argue that by incorporating intermediate restriction types and altering the answers and forms of media inclusion, eLearning designers can develop a broad array of innovative evaluation approaches and match evaluation needs with proof for numerous instructional design purposes [116]. Examples of the various forms of media inclusion are numerous and include the multimedia-rich CRESST examples [124], simulations [125], and evaluations and tracking of problem-solving paths, such as those illustrated in Interactive Multimedia Exercises (IMMEX) [126].

However, scholars also admonish using caution when dealing with media inclusion: “Any new feature added to the test that is not crucial to the variable the test is meant to measure is a possible threat to validity” [127]. Others warn evaluation developers that items should not make the examinee spend time doing unrelated mental exercise [128]. The automatic development of items, are dealt with in further innovations [116]. Automatic item generation involves methods such as templates for items used to generate numerous other items from one particular item [129]; it also includes item modeling, in which explanations of task details are formulated and then developed to fit different contexts or outlines, such as those that can be represented with some psychometric methods [130].

2.3.3 Evaluation Task Design

The use of self-assessment questions and feedback is a method that can stimulate learning [137]. Therefore, these practices would also supposedly be beneficial in web-based learning [138, 139], but actual assessments are few [141]. One study found necessary proof that medical students favored web-based learning modules that included self-assessment questions [140]. Alternative research in medical education that contrasted questions on a self-assessment basis to a less interactive, computer-assisted learning format showed no distinction [142, 143] or even found that students preferred the less interactive format [144]; however, a study by college students found a notable advantage in the question format [145]. Consequently, while self-assessment questions and feedback possesses theoretical potential, the evidence remains unconfirmed [141].

Evaluation task design is a rich and intricate realm in the quickly changing field of computer-based evaluation, and it demands numerous considerations including interactivity, evaluation assembly statements, the course between items, learning interventions blended in the item flow, and considerations of item feedback. Many innovative question formats can be obtained from combinations or modifications of conventional types of question, and item formats across types can be varied depending on the area being measured and include novel media, such as audio, video, simulation, interactive graphics, and animation. [116.]

Mislevy [146] argues that if the information offered by an innovative question type is no better than that provided by standard multiple-choice questions, then the innovation appears to be futile [116]. Innovations must offer an alternative that is better than what is available through conventional formats [116]. Moreover, Bennett [117] claims that a high degree of limitation in the answer does not automatically prevent constructive responses, which might well be possible with many multiple-choice questions [116]. Yet, one criticism of multiple-choice questions has been that they are often too easily written to assess low-order skills that do not demand notable construction [116].

2.4 From Paper Books to PDFs to Interactive eBooks

2.4.1 eBooks

There is a long history behind the method of printing books. Printed transmission of ideas has ranged from prehistoric cave paintings to clay tablets to carved prints on woodblocks, followed by the invention of moveable type. Thus, books have existed for a very long time and only their form and structure have changed gradually over time. Though, it might not be possible for a typical twenty-first century individual to understand cave paintings dating back to the time of the Egyptian pharaohs or stone tablet carvings belonging to the Persians of nearly 2,500 years ago, once translated into any of the modern languages it becomes clear that people living thousands of years ago were actually doing the same thing as us when it comes to written forms of knowledge, only using different means. [6.]

Nowadays, however, electronic books (eBooks) have become universally accepted and quite standard [7]. Even though the first type of eBook, called Dynabooks, was envisioned by Alan Kay already in 1968 [39], the market and the public did not recognize the potential of eBooks until the turn of the last century [40]. The evolution of eBooks can be seen as a part of e-publishing, which dates back to the 1960s [41]. According to Ardito [42], the expression “electronic book” was coined by van Dam at Brown University in Providence, Rhode Island, during the 1960s when he was working on hypertext systems [41]. At that time, eBook readers entered the market and companies such as netLibrary, which were devoted exclusively to publishing and delivering computer-based eBooks, were established [40]. Moreover, various software for enhancing the display of eBooks were developed, and a growing number of eBook titles were made available to the public [40].

Nevertheless, the eBook boom lasted for only about two years, after which the financial market started to decline [40]. The situation became tough for eBooks when eBook divisions such as Random House were closed and the renowned eBook provider, netLibrary, also went out of business [40, 43]. Clifford Lynch declared in 2001 that "imprecise and erratic terminology has been a big source of confusion in the publicity of eBooks, and a barrier to untangling the issues involved" [41, 44].

Lynch continues by explaining how it is important to differentiate between an eBook and the physical device used for reading a digital book [41]. The definition of an eBook provided by Armstrong et al. [45] has been received favorably by many scholars:

> Any portion of electronic text regardless of size or structure, but excluding journal publications, made accessible electronically or optically for any device including handheld or desk-bound that includes a screen.

Another definition provided by Appleton [46] refers to an eBook as follows:

> Content that has been made available in digital form through an Internet connection and shown on a computer display and authorizes pages to be downloaded and printed locally.

2.4.2 eTextbooks

The dream of producing electronic textbooks has now been pursued for more than two decades, with goals such as improving exposition by means of a richer collection of technologies than are accessible through textbooks [18, 47]. As of April 2011, eBook sales have outpaced the sale of conventional books on Amazon.com [48], and they continue to increase as a share of the market [49, 50]. Nonetheless, eTextbook sales have yet to take off at the college level [51]. Reviews of student perceptions of eTextbooks constantly indicate a lack of interest, which seems not to be positively affected by familiarity and prior use [52, 53]. Evidently, eBooks and eTextbooks are not identical products [54]. Unlike eBooks, which are typically read for individual goals and pleasure, eTextbook readers have the added incentive of learning and sometimes even memorizing portions of the text [50]. Assuredly, textbooks are constructed with these objectives in mind [50]. Although looking much the same as a textbook page, most students do not view eTextbooks as useful alternatives for meeting these objectives [53].

Studies have found that students whose objectives often revolve around effectiveness more than learning impact [55] tend to prefer educational tactics that are not connected with learning [56, 57, 58, 59, 60]. This preference is not exclusive to university students, however, since humans in general are bad evaluators of how much they know [61]. Even though students may not make the best starting choices, there is proof that they do tend to make more effective decisions with suitable feedback [62]. When alerted to some of the advantages of using student-level technology in an academic environment, Clyde [63] notes that students do perceive a number of possible benefits that eTextbooks have over their paper-based equivalents [50]. One advantage is the greater level of flexibility and accessibility of eTextbooks over paper textbooks [64, 65, 66, 67].

If eTextbooks are to become a helpful option for college students, it is necessary to explore the possible learning influence of text delivered via this medium [50]. Some studies that have directly evaluated student performance and retention rates for material delivered through eTextbooks have found that learning may not be hindered as a result of this delivery method [52, 68]. On the other hand, questions remain concerning the differences between students' interactions with print and electronic media with respect to levels of tiredness [69], text scanning methods [70], and other aspects pertinent to learning [71]. With no variations in understanding, eTextbooks could be less effective than paper textbooks since students take notably longer to read eTextbooks than paper versions of the text [63]. This time difference could be because of the additional audio and visual parts the eTextbooks possess over paper textbooks or because of issues related to reading from a screen [50].

2.4.3 Effects of eTextbooks on Learning

Learning can be influenced by both the style of the text and the means by which the text is employed [72, 73, 74, 75]. Research in higher education settings has centered mainly on faculty and student choices regarding eTextbook formats [52, 53, 76, 77, 78, 79]. Only a few studies have investigated the impact of eTextbooks on college students' affective, cognitive, and psychomotor learning [80]. Moreover, many studies have not taken the medium used to obtain the eTextbook into account [53]. Hence, more research in this area is needed [81]. Most studies on eBooks define them as texts that are digital and accessed in the form of electronic displays [80]. eTextbooks are found in two formats: page-fidelity eTextbooks and reflowable digital eTextbooks [82, 83, 84, 85]. Page-fidelity eTextbooks are scanned versions of the paper text [80]. For instance, a PDF file with no active media, no dynamic web links, and no potential to change font or pictures could be an example of a page-fidelity eTextbook [80]. Reflowable eTextbooks, on the other hand, use an open format system that encompasses dynamic media and permits users to alter both the structure and the interactive attributes of the eTextbook to fit the display medium [85].

Current technologies for eReaders provide reflowable text to support the scholarly use of electronic textbooks on every electronic device, including handheld devices [86, 87]. Nevertheless, most of the research to date on eTextbooks has been carried out using static computer displays and page-fidelity text [52, 73, 82, 88, 89, 90, 91]. Although psychological theories about learning claim that there is a connection between context and cognition [92], little research has been done on reflowable text and the impact of the improved features of eTextbooks on mobile technologies [81]. Furthermore, some studies suggest that recall and retrieval is poorer when reading from an eText compared with a paper text [82, 88, 93, 94].

Several reasons for the contradictory results have been stated, most notably the effect of passage length on learning [80]. Studies involving shorter reading sessions found no difference with respect to reading comprehension [73, 95]. In contrast, studies involving longer texts showed poorer understanding due to eye fatigue and mental workload [72, 77]. Thus, Kang et al. [77] argued that the lower contrast resolution on computers and handheld screens compared to paper books might lead to eye fatigue when reading for longer periods of time [80]. This study suggests that college students who choose to read many pages of text on a screen may undergo more eye fatigue and mental workload than their counterparts who choose paper textbooks [80]. This mental weariness and eyestrain could eventually lead to poorer understanding and less effective learning [80].

CHAPTER 3

OPENDSA

This chapter provides an in-depth introduction to the OpenDSA platform.

3.1 Algorithm Visualizations

Courses in DSA play an important role in any CS study program [108], determining the turnaround time from learning programming to learning CS [8]. Although this change happens gradually over time, students often find this topic difficult to understand since most of the content has to do with the dynamic operations of algorithms, their effects on data structures, and their growth rates [8]. Moreover, it is exceedingly difficult to convey the concept behind dynamic processes using static presentation techniques, such as text and images from a textbook [8]. To enhance students' comprehension of DSA subjects, many Algorithm Visualization (AV) systems have been created that support a variety of algorithms and various engagement tactics [96].

AV has a long history of success in the classroom [97, 98, 99]. However, many of those who support AVs have been disappointed by how the absence of comprehensive classroom adoption fails to support the needs expressed by students and teachers for AVs in surveys [18, 100]. One of the main obstacles to adoption is the difficulty of fitting AVs in as supplements to current lecture material, which numerous initiatives have tackled by combining visualizations with learning material [24, 101, 102]. The main determinant of pedagogical worth for AVs, however, seems to be associated with the level of user engagement and the amount of interactivity [18, 26, 103]. Although numerous AVs have been produced, a large number of them are of low quality in terms of their general utility and usability [99]. Consequently, it is fair to believe that a poorly designed and implemented AV would lead to low educational productiveness [104].

3.2 OpenDSA Objectives

OpenDSA is short for Open Source Interactive Data Structures and Algorithms. This project works towards solving the problems mentioned in the section above by providing complete instruction units on all of the traditional topics in a DSA course [8]. OpenDSA was initiated in 2011 [9, 47, 105]; the collaborating institutions include Virginia Tech, the University of Wisconsin-Oshkosh, and Aalto University in Helsinki [104]. OpenDSA was developed in part as a means for overcoming some of the obstacles to AV adoption reported by teachers [18, 100]. The OpenDSA project tries to supply full instructional materials for a DSA course [104]. Furthermore, since all exercises in OpenDSA are evaluated automatically, students benefit from more practice than is customary for such a course by working on OpenDSA exercises [104]. OpenDSA was designed based on the AV community's experience after decades of research on developing educationally useful AVs and AV systems [104]. Some characteristics are intentionally modeled on other successful open source projects, with the hopes of establishing community involvement in the adoption, design, and implementation of the project [104].

The objective of OpenDSA is to create a completely open-source and online electronic textbook for DSA courses, one that combines traditional textbook text with different AVs and an abundant collection of interactive exercises as a set of tutorials implemented via HTML5 technology. This method of implementation guarantees availability for use on any modern browser with no need for any additional plugins or software, and it also provides the possibility of running on tablets and many mobile devices. The content material in OpenDSA modules are in the form of text, slideshow, simulations, and different types of evaluation questions. The assessment method for all the questions is based on an immediate feedback response to students as to whether or not the question was answered correctly. [8.]

OpenDSA can be freely reused, modified, and redistributed since it is distributed using the MIT license [104]. Moreover, OpenDSA supports the reuse of content by course teachers [104]. Since the authoring environment uses reStructuredText [106], instructors can easily modify the contents of the modules [104]. Instructors can also select from among existing materials using a configuration file that can be used to make a book instance with the particular materials that they want [104].

The following are the educational goals of OpenDSA [8]:

1. Material presentation in a way that makes learning easier using leverage technology. Surveys have revealed favorable attitudes towards AVs by both instructors and students [18, 100, 109]. Though it is not easy to evaluate the pedagogical effectiveness of AVs, a number of AV systems have led to the advancement of learners' performance [26, 103]. Therefore, it is important to deliver AVs to a broad audience.

2. Incorporate activities to engage and inspire students. AVs could result in higher student engagement levels. The taxonomy put forth by Naps et al. [18] defines an order for learner engagement. Higher levels of engagement provide a better interactive experience with the technology for learners. Therefore, delivering materials that provide meaningful engagement is of high priority.

3. Include assessments to estimate how well students learn. Each module should have methods for evaluating how well a student understands the concepts being presented. Self-evaluation can boost a learner's motivation, increase a student's ability to guide her or his own learning, and promote the learning process in general [110, 111].

3.3 JSAV

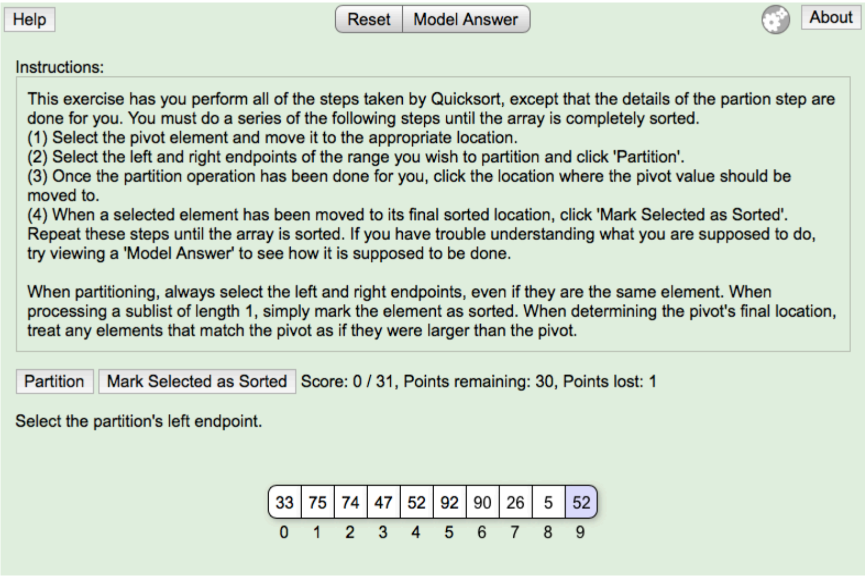
Support for AVs is essential to the eTextbook concept; therefore, it is also one of the main requirements for OpenDSA [104]. AVs are made in the JavaScript Algorithm Visualization (JSAV) library [96, 107] which is an open source project itself [104]. JSAV is written in JavaScript and supports the development of AVs for HTML5 [96]. A reader who knows the history of AVs might question the necessity for yet another AV support system, given that so many exist already [96]. The two main incentives for JSAV are as follows [96]:

1. With the unavoidable downfall of Java, an HTML5-based solution should be used. There is not much existing support for AV development in JavaScript. Currently, there are only a few other significant AV development efforts in HTML5, all of which lack some key characteristics required by effective AVs.

2. JSAV includes features designed to enable the development of AV-based tasks that require vigorous learning tactics, and visual themes that can be simply combined with online lessons. Specifically, the JSAV API maintains particular features for developing visual algorithm simulation tasks, together with support for integrating AVs with tutorials.

JSAV's user interface is HTML, with the functionality embedded in JavaScript and the design defined using CSS. Therefore, its platform currently has the broadest potential audience. JSAV also makes use of some high-quality JavaScript libraries, and it utilizes jQuery to help in settling browser differences when working with the DOM. jQueryUI applies animation effects and element locations. Raphael simplifies the use of SVG to animate changes to the visual primitives. Furthermore, MathJax is used to process mathematics defined in the LaTeX format. JSAV formally supports all current versions of Safari, Chrome, and Firefox browsers, since it has been completely tested by students for each of these platforms. JSAV is an open source platform, is distributed under the MIT license, and its source is available from GitHub. [112.]

One of JSAV's most novel features is its support for proficiency exercises, demanding that students simulate the functions of an algorithm by, for instance, clicking on tree nodes or array indices to exchange the items. Students are basically building their own AV, albeit within boundaries that the exercise developer has offered to authorize student control. This student-centered AV can be evaluated automatically by comparing it to a model solution that is available for students to observe as a JSAV slideshow. Figure 2 shows an example of the Quicksort proficiency exercise. [112.]



Instructions:

This exercise has you perform all of the steps taken by Quicksort, except that the details of the partion step are done for you. You must do a series of the following steps until the array is completely sorted.
(1) Select the pivot element and move it to the appropriate location.
(2) Select the left and right endpoints of the range you wish to partition and click 'Partition'.
(3) Once the partition operation has been done for you, click the location where the pivot value should be moved to.
(4) When a selected element has been moved to its final sorted location, click 'Mark Selected as Sorted'.
Repeat these steps until the array is sorted. If you have trouble understanding what you are supposed to do, try viewing a 'Model Answer' to see how it is supposed to be done.

When partitioning, always select the left and right endpoints, even if they are the same element. When processing a sublist of length 1, simply mark the element as sorted. When determining the pivot's final location, treat any elements that match the pivot as if they were larger than the pivot.

Score: 0 / 31, Points remaining: 30, Points lost: 1

Select the partition's left endpoint.

33	75	74	47	52	92	90	26	5	52
0	1	2	3	4	5	6	7	8	9

Figure 2. A proficiency exercise for Quicksort. The main components include guidelines, feedback for grading, and the interactive user interface.

3.4 OpenDSA Architecture

The eTextbooks generated in OpenDSA work by means of a client-server implementation, as shown in figure 3. This figure provides an overview of the OpenDSA architecture. The content server delivers the HTML documents along with embedded visualizations and exercises that constitute the eTextbook. The data collection server is a web application that is built using the Django framework with a MySQL database for the sake of data persistence. Since OpenDSA aims to combine text, dynamic content, and automatic assessment, the authoring tool needs to produce content in a web-accessible form while also providing a way to glue the dynamic elements, such as the AVs and exercises that would not be normally produced using the authoring system, together. [8.]

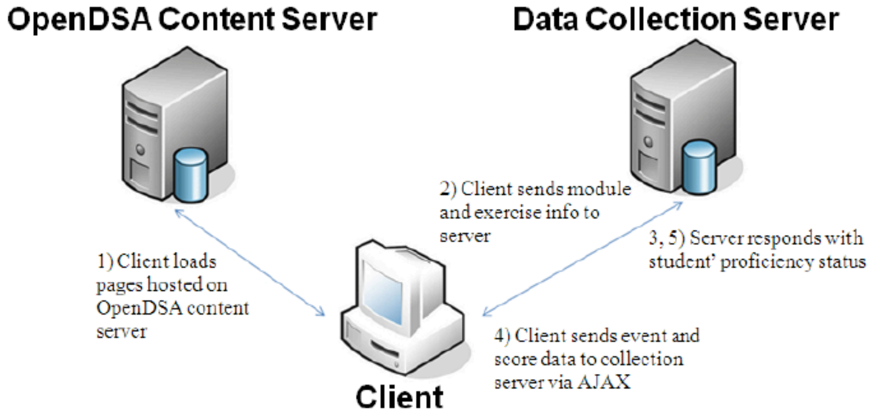

OpenDSA Content Server
Data Collection Server
2) Client sends module and exercise info to server
1) Client loads pages hosted on OpenDSA content server
3, 5) Server responds with student' proficiency status
4) Client sends event and score data to collection server via AJAX
Client

Figure 3. Overview of the OpenDSA architecture [8].

The architecture of OpenDSA is driven by the following factors [8]:

1. The instructor for a specific course needs to have control over the content to make use of existing materials, to select from among existing

materials, or to make changes to or add to existing materials. This specifies the fact that the nature of the licensing structure should be open source and the authoring system user friendly, it should be easy to modify content, and it should also be easy to select and choose different dynamic components, such as visualizations and exercises.

2. Since one of the main features of the system will be the inclusion of AVs, the system must have the capacity to support the presentation of dynamic content, such as visualizations and interactive exercises. Though the ease of authoring dynamic content is important, it is not considered a core requirement for the system. However, the ease of reusing AV modules within different formations of the content is a crucial requirement.

3. Another principal requirement of the system is support for a broad and flexible range of interactive exercises with automated evaluation. The system should be able to support all kinds of exercises, from multiple-choice questions to random problem instances for interactive exercises with modern user interfaces. However, as mentioned above, the reusability of the exercises is of greater concern than the ease of their authoring.

4. Keeping track of student progress, such as which exercises are attempted and which are completed, is yet another core requirement of the system. It is important for both learners and instructors alike to see how one is progressing through the course, and since looking at the accomplished exercise tasks is one method of doing so, it is important to have that as a support feature.

5. Various reasons would necessitate the existence of support for tracking student interactions with the content, which might well include support for evaluating the pedagogical effectiveness of different methods together with support for gathering data about the usability of system components that would enable future improvement.

6. One additional requirement that could place several constraints on the content deployment technology is that the system needs to be easy to access and portable across a broad range of computing platforms. In this way, it would be easy to ensure that almost anyone would have the possibility to access the course content on any platform.

3.5 Khan Academy

With respect to the interactive content and exercises, automated exercise assessment is a major feature of OpenDSA. It provides students with the opportunity to practice quite often all aspects of the content they are studying in a DSA course. Various exercises exist in different forms and locations throughout a typical OpenDSA chapter, such as slideshows, visualizations, proficiency exercises, programming exercises, and chapter summary questions. The chapter summary questions make use of a range of standard question types, such as true/false, multiple-choice and fill-in-the-blank questions, which generally do not take a long time to complete individually. While this type of exercise is considered to be less attractive than interactive question types, such as proficiency or programming exercises, they do play a significant role in OpenDSA, since well-crafted questions of these kinds can be quite useful pedagogically. Therefore, a rich infrastructure that can support these types of questions is crucial. [8.]

The Khan Academy Exercise Infrastructure (KAEI) is used to store and present chapter summary questions in OpenDSA [8]. Khan Academy defines itself as “a non-for-profit with the objective of improving education by offering a free world class teaching for everyone everywhere” [113]. Khan Academy commenced in 2006 as a series of short YouTube videos developed by a person with a laptop and an Internet connection [114]. Since then, the Khan Academy has grown into a series of thousands of video lessons that cover K–12 topics [114]. The site provides practice tests to gain skills; it also provides resources for teachers to track their students' progress and help in case of need [114]. This free resource allows educators to obtain new information outside the classroom, and it allows teachers to use classroom time to strengthen learning and answer students' questions [114].

The philosophy behind the Khan Academy is to offer content for nearly all subjects as a means of developing "the world's first free, global, virtual school where everyone can learn anything". This is in keeping with the suggestion by Bloom [115] regarding the success of one-to-one tutoring. Khan Academy offers individual tutoring for students free of charge and permits the solution to scale through the Internet. Khan Academy has lessons in science, the humanities, economics, math, and computer programming. The larger part of the Khan Academy learning environment involves watching a video description of the topic in question and engaging in self-evaluation exercises. Nonetheless, the learning environment for computer programming is different in the sense that it has a built-in code editor and execution pane. [113.]

Apart from the exercise engine that OpenDSA makes use of, the Khan Academy offers an unmatched set of educational material for science and math education in the form of concise, free, and publicly available video clips. The clips are in 10–20 minute chunks, allowing the learner to fill in almost any of their knowledge gaps and providing a conversational style with a fresh and new perspective on math and science instruction. Using some large funding sources, such as Google and the Gates foundation, the Khan Academy aims to cover all topics that appear in normal high school or college level math and science courses, together with translating the videos into the major languages of the world. [11.]

CHAPTER 4

METHODOLOGY

The thesis work consisted of two main phases: material development and empirical evaluation. The material development phase consisted of going through the chapter on AA and taking notes while paying special attention to how the content is set forth and how effectively it is conveyed to the reader. Based on this review, suggestions for improvement were made. More important, however, was the design and implementation of new summary exercise types and sample questions based on these suggestions.

The following materials were development during the material development phase: a graphing visualization, matching questions, ordering questions, fill-in-the-blank questions, error-guessing questions, and multiple-choice questions. At least two sample questions were designed and developed for each of the aforementioned question types. The software platforms used during the material development phase are as follows: OpenDSA, KAEI, JavaScript, jQuery, jQueryUI, HTML5, and CSS.

Empirical evaluation of the developed material was conducted with five CS students during their last year of study. All of the students had already taken courses on DSA, and they were thus familiar with AA. The study was conducted using a deep one-on-one evaluation approach that included a think-aloud protocol [10]. The evolution of this protocol is often ascribed to Ericsson and Simon [147], and the method has been used broadly in cognitive psychology research, usually to explore problem solving [10]. This method has been used to study human-computer interactions and evaluate new software [148, 149], often under the cover of "usability-testing" [10]. In this context, the study involved observing individual students while they used the newly implemented material.

The participants were given the freedom to return to and/or refer to any part of the testing material if they so desired and were asked to verbalize all thoughts that entered their heads at the time. There was no specific timetable for the prompts and they were posed in a manner that fit the needs of each individual student. The individuals present at each testing session were one student and myself, and each session lasted for less than 45 minutes. The testing sessions were carried out at different locations inside the ICT building. The results were arranged in the form of written reports. The learning results, consequently, were obtained by analyzing these written reports together with the information procured during the participants' user interactions.

PART II

IMPLEMENTATION

CHAPTER 5

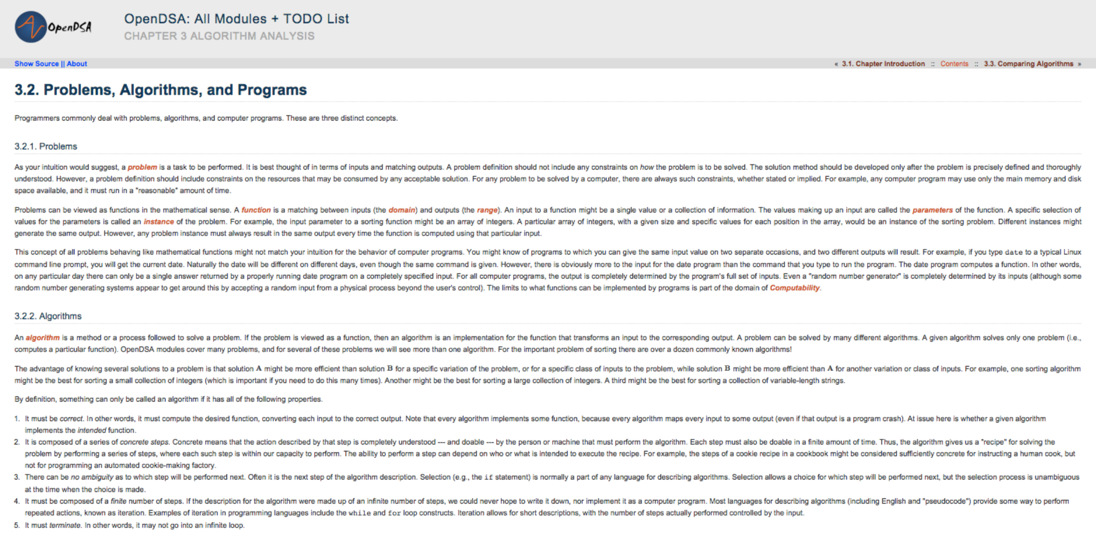
UPDATING OPENDSA

An overview of the different stages of the implementation process together with the implementation work itself is provided in this chapter.

5.1 Personal Experience

5.1.1 Initial Impressions

The first step after the conception of the first eBook around forty years ago [39] was to convert printed books into electronic format through only typing and scanning. Extra features, such as interactive figures, case examples, and hyperlinks to related topics, could assist readers in learning the content [172]. Today, however, new features such as animation, video, audio, and interactive simulations are being added so that readers can interact with the content of eBooks more dynamically than when only reading them [83, 91]. The focus in this section is to review and analyze the content of the AA chapter in OpenDSA.



OpenDSA
OpenDSA: All Modules + TODO List
CHAPTER 3 ALGORITHM ANALYSIS
Show Source || About
« 3.1. Chapter Introduction :: Contents :: 3.3. Comparing Algorithms »

3.2. Problems, Algorithms, and Programs

Programmers commonly deal with problems, algorithms, and computer programs. These are three distinct concepts.

3.2.1. Problems

As your intuition would suggest, a ***problem*** is a task to be performed. It is best thought of in terms of inputs and matching outputs. A problem definition should not include any constraints on *how* the problem is to be solved. The solution method should be developed only after the problem is precisely defined and thoroughly understood. However, a problem definition should include constraints on the resources that may be consumed by any acceptable solution. For any problem to be solved by a computer, there are always such constraints, whether stated or implied. For example, any computer program may use only the main memory and disk space available, and it must run in a "reasonable" amount of time.

Problems can be viewed as functions in the mathematical sense. A ***function*** is a matching between inputs (the ***domain***) and outputs (the ***range***). An input to a function might be a single value or a collection of information. The values making up an input are called the ***parameters*** of the function. A specific selection of values for the parameters is called an ***instance*** of the problem. For example, the input parameter to a sorting function might be an array of integers. A particular array of integers, with a given size and specific values for each position in the array, would be an instance of the sorting problem. Different instances might generate the same output. However, any problem instance must always result in the same output every time the function is computed using that particular input.

This concept of all problems behaving like mathematical functions might not match your intuition for the behavior of computer programs. You might know of programs to which you can give the same input value on two separate occasions, and two different outputs will result. For example, if you type `date` to a typical Linux command line prompt, you will get the current date. Naturally the date will be different on different days, even though the same command is given. However, there is obviously more to the input for the date program than the command that you type to run the program. The date program computes a function. In other words, on any particular day there can only be a single answer returned by a properly running date program on a completely specified input. For all computer programs, the output is completely determined by the program's full set of inputs. Even a "random number generator" is completely determined by its inputs (although some random number generating systems appear to get around this by accepting a random input from a physical process beyond the user's control). The limits to what functions can be implemented by programs is part of the domain of ***Computability***.

3.2.2. Algorithms

An ***algorithm*** is a method or a process followed to solve a problem. If the problem is viewed as a function, then an algorithm is an implementation for the function that transforms an input to the corresponding output. A problem can be solved by many different algorithms. A given algorithm solves only one problem (i.e., computes a particular function). OpenDSA modules cover many problems, and for several of these problems we will see more than one algorithm. For the important problem of sorting there are over a dozen commonly known algorithms!

The advantage of knowing several solutions to a problem is that solution $\mathbf{A}$ might be more efficient than solution $\mathbf{B}$ for a specific variation of the problem, or for a specific class of inputs to the problem, while solution $\mathbf{B}$ might be more efficient than $\mathbf{A}$ for another variation or class of inputs. For example, one sorting algorithm might be the best for sorting a small collection of integers (which is important if you need to do this many times). Another might be the best for sorting a large collection of integers. A third might be the best for sorting a collection of variable-length strings.

By definition, something can only be called an algorithm if it has all of the following properties.

1. It must be *correct*. In other words, it must compute the desired function, converting each input to the correct output. Note that every algorithm implements some function, because every algorithm maps every input to some output (even if that output is a program crash). At issue here is whether a given algorithm implements the *intended* function.
2. It is composed of a series of *concrete steps*. Concrete means that the action described by that step is completely understood --- and doable --- by the person or machine that must perform the algorithm. Each step must also be doable in a finite amount of time. Thus, the algorithm gives us a "recipe" for solving the problem by performing a series of steps, where each such step is within our capacity to perform. The ability to perform a step can depend on who or what is intended to execute the recipe. For example, the steps of a cookie recipe in a cookbook might be considered sufficiently concrete for instructing a human cook, but not for programming an automated cookie-making factory.
3. There can be *no ambiguity* as to which step will be performed next. Often it is the next step of the algorithm description. Selection (e.g., the `if` statement) is normally a part of any language for describing algorithms. Selection allows a choice for which step will be performed next, but the selection process is unambiguous at the time when the choice is made.
4. It must be composed of a *finite* number of steps. If the description for the algorithm were made up of an infinite number of steps, we could never hope to write it down, nor implement it as a computer program. Most languages for describing algorithms (including English and "pseudocode") provide some way to perform repeated actions, known as iteration. Examples of iteration in programming languages include the `while` and `for` loop constructs. Iteration allows for short descriptions, with the number of steps actually performed controlled by the input.
5. It must *terminate*. In other words, it may not go into an infinite loop.

Figure 4. Contents of a page from the AA chapter in OpenDSA.

Figure 4 above shows a typical page from the AA chapter in OpenDSA. However, it is worth mentioning that other pages in the chapter make use of diagrams and tables to assist with the learning process, and it is important to note the effective role of these instruments in smoothly conveying information to the learner. Even still, no interaction takes place between the learner and the visual objects. On the other hand, OpenDSA is an open-source project that is under constant development; therefore, it is quite possible that new visualizations and interactions will soon be included in the chapter.

One challenging issue that can be improved with respect to a better learning outcome is the heavy mathematical content of the chapter. This also leads us to consider the level of math knowledge of the course participants and what could be done in that respect. The mathematical content of the chapter requires a certain advanced level of mathematics in order to grasp the related topics introduced in this chapter. Hence, it could be advantageous to add some references to the purely mathematical parts of the material for students to refer to while studying the chapter. Some extra references to mathematical concepts, such as logarithms and probability theory, would no doubt prove helpful.

5.1.2 Ideas for Creative Change

It would be a good idea to include some pre-recorded lectures by a teacher [173]. The lectures should include their voice and an animated whiteboard in the platform to spice up the platform a bit. These additions could be helpful for challenging chapters, such as the AA chapter. Another idea for creative change would be to have the lecturer on one pane and have another presentation pane that is updated in real-time next to the lecturer for the lecture notes. Having such a combination would make it possible to include a specific link through which the students could access the lecture notes on a separate web page, thereby providing the functionality of learners accessing the lecture notes in their own time and at their own convenience. Furthermore, it would be a good idea to have a certain standard for grading the assignments [174, 175], and then publishing them on the course web pages. Learners might find this added clarity motivating with respect to grades, in that sense they might work harder to pursue even better grades after completing a specific exercise module.

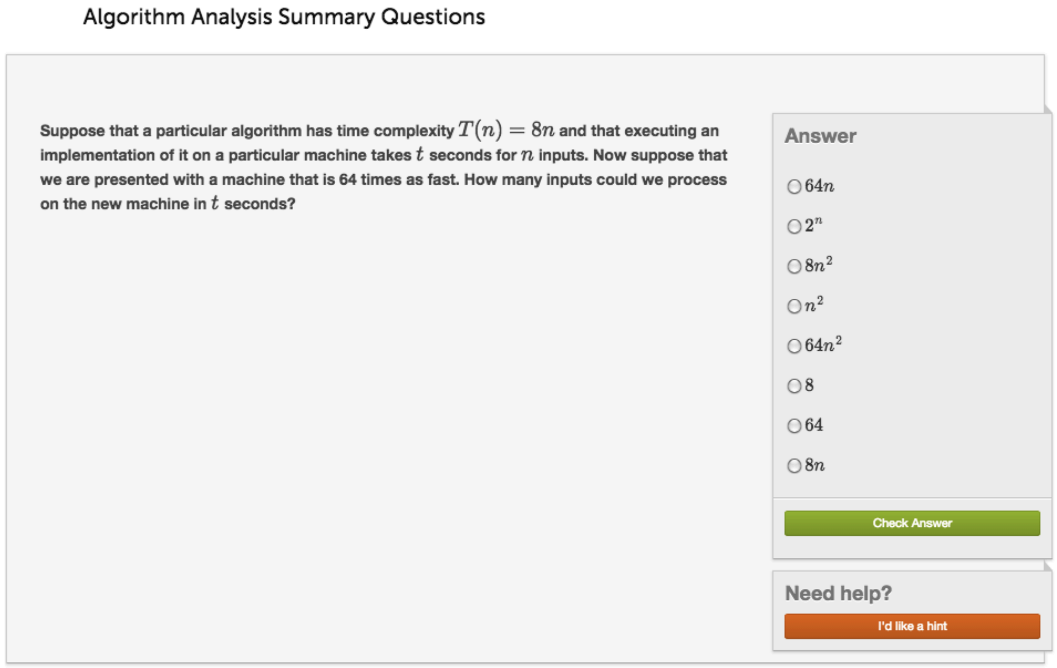
Considering the fact that not all students are good at planning their own online studies, it would be beneficial to include a course calendar inside the OpenDSA to help learners with planning their chapter studies and the assignment due dates. Some recommended course plans and assignment submission times could be provided to the students on the calendar provided inside the OpenDSA, which would help them better plan their study and exercise times. Since the study times would only be provided as recommendations, learners would by no means be forced to follow them; they would still have the option of accessing the course material and completing the assignments on their own time and at their own convenience in their own rooms.

Including one big project for the whole course or several smaller projects at different points of the course, either of which would cover the provided topics, could be a good addition to the OpenDSA platform. These projects should differ from all the other question types throughout the OpenDSA, such as the chapter summary questions or the exercises provided in each chapter. Rather, they should act as a fun assessment method for students to now put all the theory they have learned throughout the course to the test and see how well they are able to connect their theoretical knowledge to a practical application.

One could argue, however, that the current OpenDSA platform mainly focuses on providing an eTextbook version of the course to the course participants rather than a complete eLearning platform. Even still, the inclusion of such features would only be beneficial to the whole OpenDSA experience. On the other hand, a good and comprehensive online learning platform could start from an extensive eTextbook. Consequently, it would be a good idea to enhance the features and capabilities of OpenDSA with the design plans mentioned above and expand its educational borders using these innovative concepts.

5.1.3 Summary Exercises

Current exercises in OpenDSA can either be proficiency exercises, which demand that the student simulate an operation from an algorithm, or summary exercises, which ask questions about the various notions explored in the chapters [179]. There are different types of summary exercises at the end of the AA chapter, all of which are implemented using the same Khan Academy framework. Figure 5 shows a multiple-choice question. The question is set within a large and simple rectangular frame, in which there is a question title, a frame for the answer, several radio choices inside the answer frame, and a frame for hints in case the student needs a hint. The number of radio choices for this question type varies depending on the question, ranging from two choices for a true/false question to sometimes even eight choices [176].



Algorithm Analysis Summary Questions

Suppose that a particular algorithm has time complexity $T(n) = 8n$ and that executing an implementation of it on a particular machine takes t seconds for n inputs. Now suppose that we are presented with a machine that is 64 times as fast. How many inputs could we process on the new machine in t seconds?

Answer

○ $64n$

○ 2^n

○ $8n^2$

○ n^2

○ $64n^2$

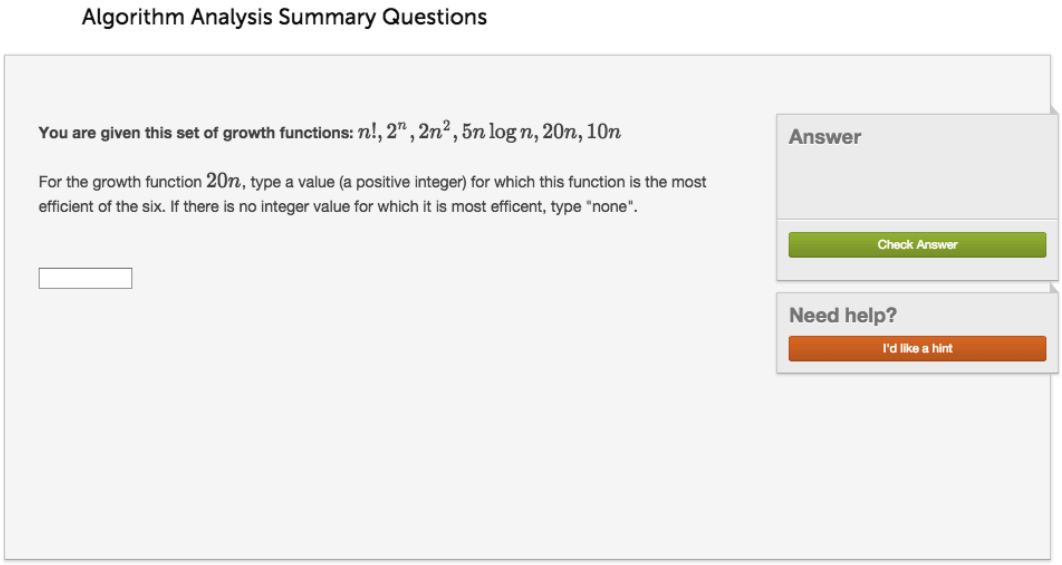
○ 8

○ 64

○ $8n$

Figure 5. A typical multiple-choice chapter summary question type in OpenDSA.

Figure 6 below shows the fill-in-the-answer question type, in which a small textbox is provided for students to type in their answer and then check whether it is correct by pressing the associated button. The nature of this question type requires that the individual have a good understanding of the topic in question and actually know the correct answer before proceeding to the next question [176]. Nevertheless, this raises the issue that in cases where students do not know the answer to this question type at all, but would still like to continue the test, they would not have the possibility to do so!



Algorithm Analysis Summary Questions

You are given this set of growth functions: $n!, 2^n, 2n^2, 5n \log n, 20n, 10n$

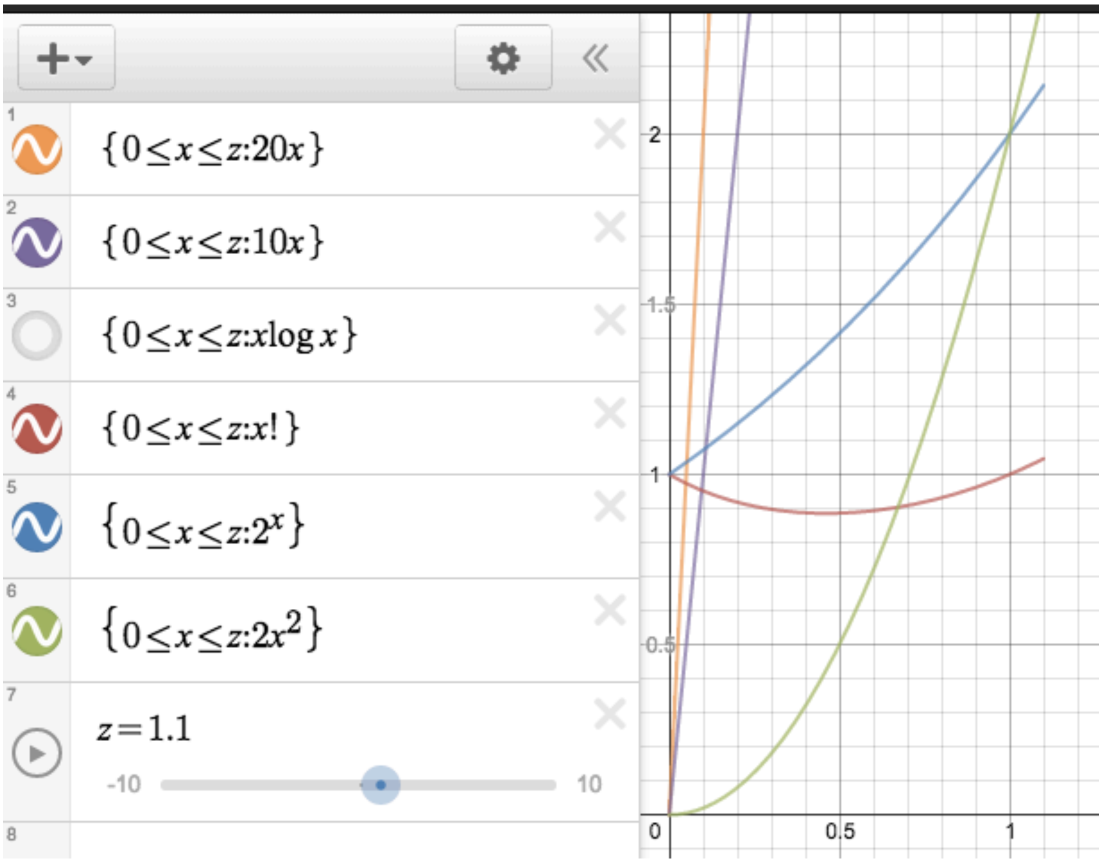
For the growth function $20n$, type a value (a positive integer) for which this function is the most efficient of the six. If there is no integer value for which it is most efficent, type "none".

Figure 6. A typical fill-in-the-answer chapter summary question type in OpenDSA.

5.2 Implementation Work

5.2.1 Graphing Visualization

The implementation work was started by creating a visualization for math functions so that the learner gains a better understanding of the functions mentioned in the AA chapter. The online infrastructure “Desmos” [165] was chosen for this purpose. Desmos is a graphing calculator that can plot different equations and that has the aim of changing the negative attitudes of students towards mathematics [166, 167]. The finished work is an interactive visualization that includes various graphs for different growth rate functions to assist students in learning about the growth rates. Figure 7 provides a snapshot of this visualization.



1 $\{0 \le x \le z : 20x\}$

2 $\{0 \le x \le z : 10x\}$

3 $\{0 \le x \le z : x \log x\}$

4 $\{0 \le x \le z : x!\}$

5 $\{0 \le x \le z : 2^x\}$

6 $\{0 \le x \le z : 2x^2\}$

7 $z = 1.1$

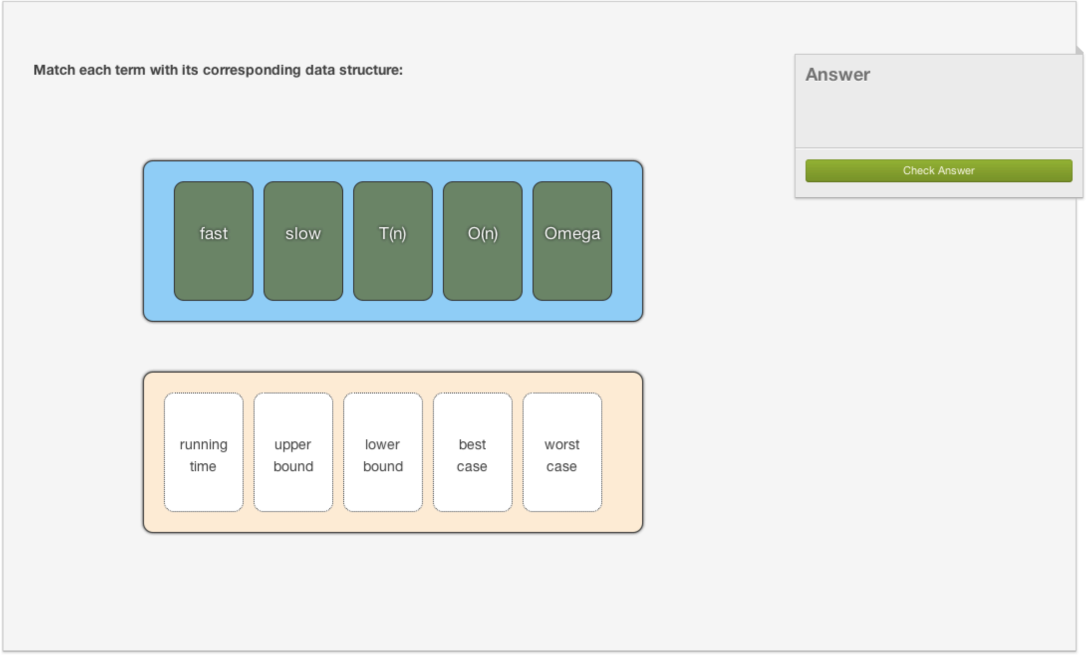
Figure 7. Graphing visualization for certain growth rate functions.

Each function is defined on the left pane and assigned a graph line color. For instance, it is easy to see that the $2x^2$ growth rate function has a green line color on the graph. The z slider at the bottom left corner makes the growth rate functions interactive and gives users the ability to change the function variable by dragging the slider either left or right, while at the same time portraying the effect of that particular change on the visual graph [165]. The slider also offers the functionality of a play button; pressing on it would initiate a slideshow presentation of different values together with their graph. Using this interactive graphing visualization, students would have the opportunity to visually and tangibly see the differences between various growth rate functions and be able to play with different function values on the fly.

5.2.2 Matching Questions

More than ten classification systems for question types have been proposed by researchers [184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195]. The use of matching questions has been recommended to overcome some of the flaws of utilizing multiple-choice questions to assess factual knowledge [180]. Figure 8 shows an implemented matching question. It is a mini-game in which students have the possibility to have a fun, game-like experience [177]. Students are provided with two rows of rectangular shapes, the first of which contains the question objects and the other the targets. Students are supposed to first read and find the connection between each item in the first row and the item in the second row, which requires that they have good command of the AA chapter material. After finding the connections, they are supposed to hold and drag each item from the first row to its associated target slot in the row below.

If the student correctly identifies the connection, then the target slot will accept the question item after releasing it on top of the slot; otherwise, the question item will not stick and will revert to its original location. The learner would be granted permission to move on to the next question after putting all the provided question items in their corresponding answer slots. Research shows that matching questions with eight options or less make more efficient use of testing time and result in better score accuracy, in contrast with questions with larger numbers of options [181, 182, 183]. Therefore, five options were chosen when designing this matching question type.



Match each term with its corresponding data structure:

Figure 8. A typical matching question type implemented for the AA chapter in OpenDSA.

The figure above is the result of integrating one of the questions already designed for the AA chapter with the matching question type that was developed in the KAEI using JavaScript and HTML5. To be more specific, jQuey together with jQueryUI widgets and themes were used to be able to implement the user interface interactions. jQueryUI is a set of plugins built on top of the jQuery JavaScript library, which add new functionalities to the jQuery core library [12]. These functionalities include, but are not limited to, accordion menus, an auto-complete mechanism for input fields, a tabs mechanism to facilitate the display on the page, and dialog boxes that are superimposed on top of the page [12].

The question and the answer items are created inside the two arrays of random and inserts respectively using the following JavaScript lines:

```
// the array of question items
var random = [ 'top', 'fifo', 'binary', 'index', 'edge' ];
    // randomize the question items each time the exercise is loaded
    random.sort( function() { return Math.random() - .5 } );
        // iterate through the array and create each draggable question item
        for ( var i=0; i<5; i++ ) {
            $('<div>' + random[i] + '</div>').data( 'number', random[i] )
            .attr( 'id', 'element'+random[i] ).appendTo( '#stack' ).draggable( {
                containment: '#content',
                stack: '#stack div',
                cursor: 'move',
                revert: true
            } );
        }

// the array of answer items
var inserts = [ 'stack', 'queue', 'tree', 'array', 'graph' ];
    // iterate through the array and create each droppable answer item
    for ( var i=1; i<=5; i++ ) {
        $('<div>' + inserts[i-1] + '</div>').data( 'number', inserts[i-1] )
        .appendTo( '#inserts' ).droppable( {
            accept: '#stack div',
            hoverClass: 'hovered',
            drop: handleDrop
        } );
    }
```

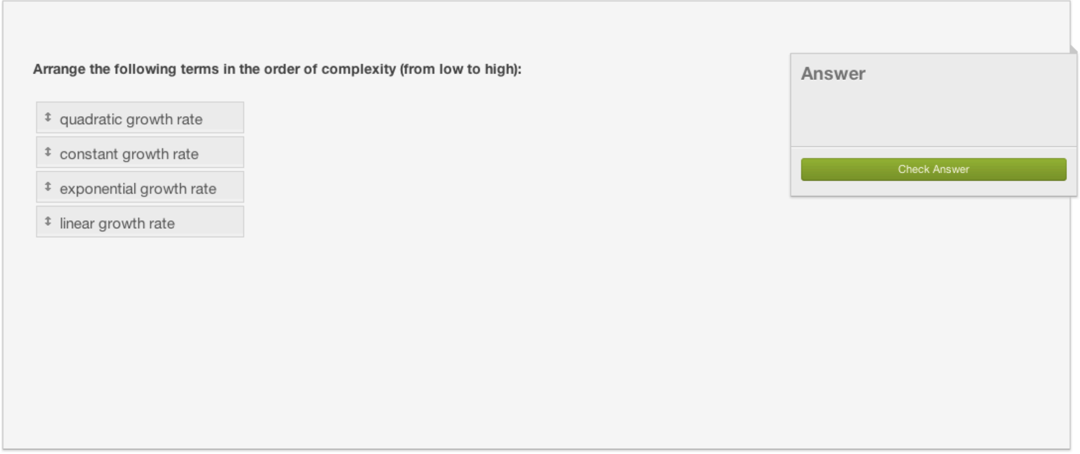
A function called handleDrop checks to see if the drop was the correct answer. This function is triggered after every drop of a question item and is defined as follows:

```
    function handleDrop( event, ui ) {
        var inserts = $(this).data( 'number' );
        var stack = ui.draggable.data( 'number' );
        //The reaction after the drop of a draggable question item
        if   (( inserts == 'stack' && stack == 'top') || ( inserts == 'queue' && stack == 'fifo')
             || ( inserts == 'tree' && stack == 'binary') || ( inserts == 'array' && stack == 'index')
             || ( inserts == 'graph' && stack == 'edge')) {
                ui.draggable.draggable( 'disable' );
                $(this).droppable( 'disable' );
                ui.draggable.position( { of: $(this), my: 'left top', at: 'left top' } );
                ui.draggable.draggable( 'option', 'revert', false );
                correctAnswers++;
            }
    }
```

The variables correctAnswers together with check keep an eye on how well students complete the mini-game.

5.2.3 Ordering Questions

The next question type developed is the ordering question type, which can be seen in figure 9. This question type provides students with a set of rectangular boxes, each of which contains different answer elements that can be dragged around and put under or above one another in the order desired by the student [178]. The student is responsible for figuring out the correct order for the elements based on the question being asked and then arranging them in the correct order. Even though these elements can be dragged around anywhere on the page, they will simply snap back to their original location after being released if there is no interaction with the other elements on the list [178]. After arranging all of the items in the desired order, students can check their answer by pressing the corresponding button and move on to the next question, provided that all the elements are arranged in the correct order.


Arrange the following terms in the order of complexity (from low to high):

- quadratic growth rate
- constant growth rate
- exponential growth rate
- linear growth rate

Figure 9. A typical ordering question type implemented for the AA chapter in OpenDSA.

The sample question shown in the figure above assesses a student’s ability to comprehend the “time complexity” concept [150] introduced in the AA chapter by asking the individual to arrange the growth rates provided in an ascending order based on their time complexity. An unordered list (ul) object has been used to create the ordering items as follows:

```
<ul style="cursor: pointer" id="sortable">
    <! - - first ordering item - - >
    <li id="3" class="ui-state-default">
        <span class="ui-icon ui-icon-arrowthick-2-n-s">
        </span> O ( N&#178 )
   </li>
   <! - - second ordering item - - >
   <li id="4" class="ui-state-default">
       <span class="ui-icon ui-icon-arrowthick-2-n-s">
       </span> O ( N&#179 )
   </li>
   <! - - third ordering item - - >
   <li id="1" class="ui-state-default">
       <span class="ui-icon ui-icon-arrowthick-2-n-s">
       </span> O ( 1 )
   </li>
   <! - - fourth ordering item - - >
   <li id="2" class="ui-state-default">
       <span class="ui-icon ui-icon-arrowthick-2-n-s">
       </span> O ( N )
   </li>
</ul>
```

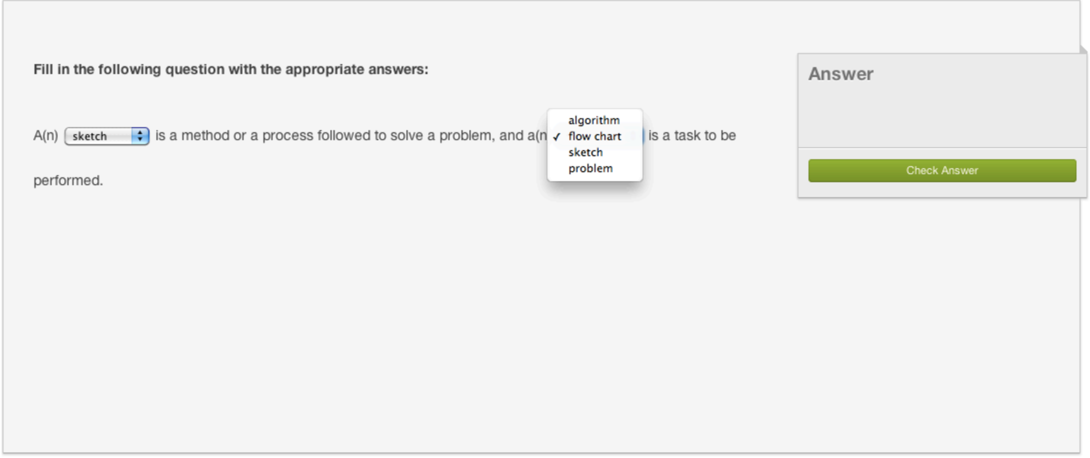
The following checkAnswer function determines whether the exercise has been correctly completed or not:

```
function checkAnswer() {
    $("#sortable").children().each(function (i) {
        var li = $(this);
        // fill an array with the current item ids
        items[i] = li.attr("id");
    });
  // compare the current item ids with their predefined numbers
  if ( items[0] == 1 && items[1] == 2 && items[2] == 3 && items[3] == 4 )
      return true;
```

```
        else
            return false;
    }
```

5.2.4 Fill-in-the-blank Questions

The third question type is the fill-in-the-blank question type, as shown in figure 10. This question type enables students to fill in the blank using a selectmenu widget [168] inside jQueryUI. The student is provided with a sentence; one or more words inside the sentence need to be filled in using the supplied selectmenu. The student would have to first select the desired options and then check her or his choices using the answer button. On the condition that she or he has filled in all the blanks with the correct answer, then she or he will be granted the possibility to move on to the next question. Otherwise, the "check answer" button will shake to let the user know that one or more of the selections are incorrect. Fill-in-the-blank questions are one of the most favored question types in testing [196].



Fill in the following question with the appropriate answers:

A(n) sketch is a method or a process followed to solve a problem, and a(n ✓ flow chart is a task to be performed.

Figure 10. A typical fill-in-the-blank question type implemented for the AA chapter in OpenDSA.

This question type requires a higher level of accuracy by students compared to the previous question types mentioned so far due to the fact that learners need to have mastery over a specific piece of textbook with a high level of detail. In other words, the learner needs to know the exact words used in the textbook since the question zooms in on specific lines of text. However, since the chapter summary questions are designed with the goal of learning in mind, the questions in this category are still asking about different important concepts. For instance, the question in the figure above is asking about the two important notions regarding "problem" [16] and "algorithm" [150] that are introduced at the beginning of the AA chapter.

The following lines implement the fill-in-the-blank question mentioned above:

```
$(function() {
    // first selectmenu widget
    $( "#choice1" ).selectmenu();
    // second selectmenu widget
    $( "#choice2" ).selectmenu();
});
<font size="3">
<fieldset>
    // the first part of the sentence
    <label for="choice1">
        A(n)
    </label>
    // the first blank
    <select name="choice1" id="choice1">
```

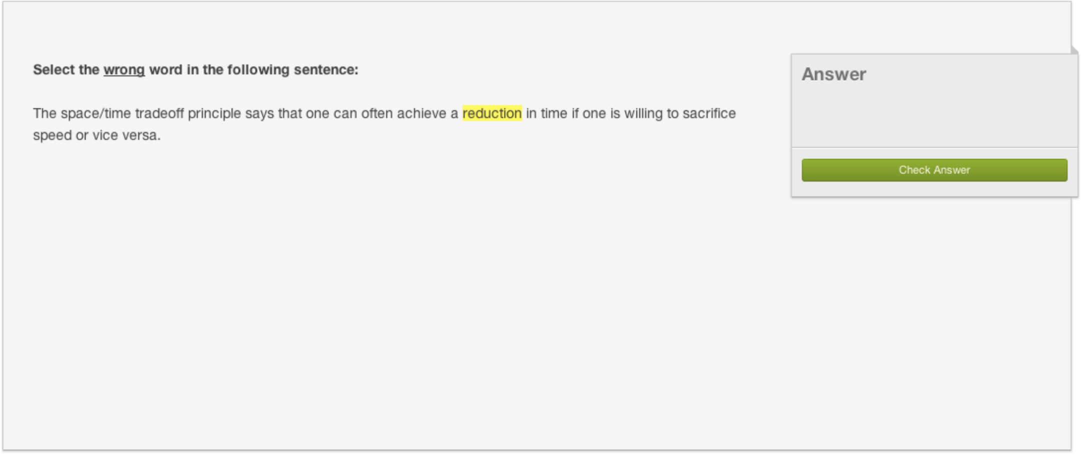
```
            <option>
                algorithm
            </option>
            <option>
                flow chart
            </option>
            <option selected="selected">
                sketch
            </option>
            <option>
                program
            </option>
        </select>
        // the second part of the sentence
        <label for="choice2">
            is a method or a process followed to solve a problem, and a(n)
        </label>
        // the second blank
        <select name="choice2" id="choice2">
            <option>
                algorithm
            </option>
            <option selected="selected">
                flow chart
            </option>
            <option>
                sketch
            </option>
            <option>
                problem
            </option>
        </select>
        // the last part of the sentence
        <p\n>
            is a task to be performed.
        </p>
    </fieldset>
```

The following checkAnswer function determines whether or not the exercise has been completed correctly:

```
    function checkAnswer() {
        // the value of the first filled blank
        var check1 = $( "#choice1" ).val();
        // the value of the second filled blank
        var check2 = $( "#choice2" ).val();
        // compare the values of the filled blanks with the correct answers
        if (check1 == "algorithm" && check2 == "problem")
            return true;
        else
            return false;
    }
```

5.2.5 Error-guessing Questions

Error-guessing questions [169], as illustrated in figure 11, have also been implemented. This question type as well as the fill-in-the-blank question type both work on a textual basis. However, this question type focuses mainly on spotting the incorrect word or phrase inside a sentence or a collection of sentences. Therefore, the individual needs to be a keen reader in order to notice the incorrect word and select it by clicking on it. After clicking on any word, the selected word will become highlighted, after which the individual can check the correctness of her or his choice by pressing the "check answer" button. Once again, if a student's selection is correct, she or he will automatically be directed to the next question; otherwise the answer button will shake, notifying the student of her or his incorrect selection and requiring that she or he make another attempt at finding the correct answer [176].



Select the wrong word in the following sentence:

The space/time tradeoff principle says that one can often achieve a reduction in time if one is willing to sacrifice speed or vice versa.

Figure 11. A typical error-guessing question type implemented for the AA chapter in OpenDSA.

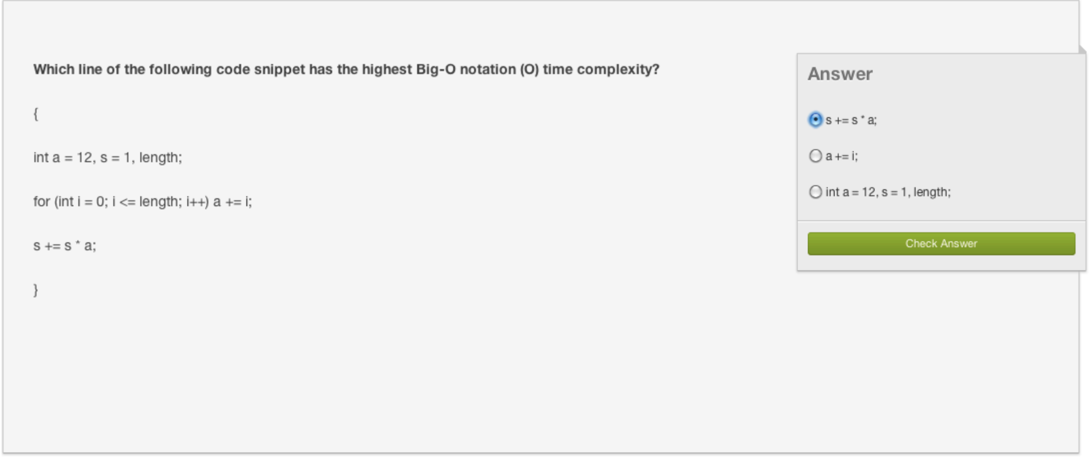
The questions in this category of question type are designed to mainly attend to important concepts in the chapter. Although this question type might look simple, from an implementation standpoint it was not trivial to implement the highlight feature for all the words in the sentence due to the fact that a highlighting component did not already exist inside the available libraries of jQueryUI and therefore a custom function in JavaScript had to be built to provide this functionality. However, the end result was quite worth the effort since this question type now has a custom-built highlight function for all questions. The following lines implement this highlight functionality:

```
$( document ).ready( function() {
    // put the whole sentence in a variable
    var p = $( "p", "#sentence" );
    // split the sentence into word chunk objects
    $.each( p, function( i, obj ){
        $( obj ).attr( "class", "highlightable" );
        obj.innerHTML = "<per>" +
        obj.innerHTML.split(" ").join( "</per> <per>" ) +
        "</per>";
    });
    // highlight the selected word object by changing its background color
    $( ".highlightable per", "#sentence" ).on( "click", function(){
        $( "per", $( this ).parent() ).css( "background-color", "");
        $( this ).css( "background-color", "#ffff45" );
        a = this.innerText;
    });
});
```

5.2.6 Multiple-choice Questions

Finally, last but not least, a multiple-choice question type for code snippets has also been designed [169]. Even though this question type was already in use in the AA chapter of OpenDSA, the inclusion of well-designed multiple-choice questions for code snippets as part of the implementation work was necessary. Multiple-choice questions dominate large-scale computer-based testing and eLearning assessments [116]. Well-designed multiple-choice questions can assess higher-order cognitive processing [197]. Nevertheless, even though multiple-choice questions may not be any more valid, their realism makes them more satisfactory [198, 199, 200].

Figure 12 shows a typical multiple-choice question designed for this chapter. After selecting an answer, students will have to press the check answer button in order to move on to the next question, at which point they will also be informed about the status of their selected answer. When choosing an incorrect option, they will not be granted access to the following question, until they are able to locate the correct answer.



Which line of the following code snippet has the highest Big-O notation (O) time complexity?

```
{

int a = 12, s = 1, length;

for (int i = 0; i <= length; i++) a += i;

s += s * a;

}
```

Answer

● s += s * a;

○ a += i;

○ int a = 12, s = 1, length;

Figure 12. A typical multiple-choice question type implemented for the AA chapter in OpenDSA.

The multiple-choice question shown in the figure above asks the learner to locate the part of the code snippet with the largest Big-O time complexity [150]. In other words, it asks students which part of the provided code snippet takes the most time to execute on a computer compared to the rest of the code snippet. In order to answer a question like this, the student needs to have a clear understanding of Big-O notation time complexity and the ability to at least come up with a close estimate of this time complexity for the different parts of the code provided. Therefore, well-designed multiple-choice questions can be beneficial to the learning process [200].

CHAPTER 6

USER TESTING

This chapter presents the results from the user testing phase of the study together with the resulting implementation changes.

6.1 Testing Experience

6.1.1 Scenario

Scenarios have found their part in testing, mainly in staging or locating test examples, usually in a very detached manner where the test samples focus on abstract tasks. In contrast to this detached manner of testing, some researchers see usability as an essential part of the design process and have examined the application of scenarios more extensively in designs [201, 202]. It is necessary to create use situations as part of the testing process to be able to stage users' experiences with respect to the future [204]. It is equally imperative to explore ways in which users can feed back thoughts to usability people in ways that are anchored in the specifics of certain use scenarios [203].

The students chosen for the user testing had taken the DSA course in the past. The testing was carried out on an individual basis rather than in groups to be able to do a deeper level of testing. A testing supervisor was present to look closely at each and every one of the student's interactions with the material at hand while at the same time taking notes and encouraging the participants to verbalize their thoughts and feelings as they navigated their way through the material. The method provided a rich data set, the benefits of which exceeded the time-consuming nature of the exercise. A static scenario could have saved time with every individual and a great deal of time overall in each testing session, but based on the aforementioned reasons pertaining to testing experience, the think-aloud method [205] was chosen instead for richer and more effective results. User encouragement played a key role in ensuring the effectiveness of the data collection and the production of useful results [206].

In some cases, the participants looked a bit bewildered or asked questions more than usual during the testing. . In these cases, they were given hints to assist the decision-making process. The aim was to simulate a testing experience that was as close to reality as possible. Therefore, the hints did not provide any directions on how to use the resource [207]. Normally, during the first few minutes, when the participants had just received the material, they were more confused and thus more in need of interactions with the researcher. A few effective prompts at the beginning of their work in most cases led to a smooth and focused experience for the participants, one in which they were able to engage with and effectively think aloud their thought processes [208].

6.1.2 Techniques

The majority of the testing prompts merely acted as a reminder for the participants to think aloud after intervals of silent reading. Since reading and working with new content is a task that is usually done silently [209], without the need for readers to think aloud about their thought processes, the participants seemed to be quite prone to carry out the task quietly, out of habit, even though they already knew that they were supposed to think out loud. Therefore, helping the participants remain in a continual state of thinking out loud from beginning to end appeared to be one of the most fundamental methodologies used during the testing session [210].

One research study suggests that interrupting users who are engaged in problem-solving exercises does not have any notable effect on the reactivity of users compared to a standard think aloud protocol [10]. For situations in which the interruption was longer than a few seconds, the participant had to do some extra thinking to return to her or his previous thinking process and in some cases needed to repeat or rewind to an earlier thought process in order to accomplish the testing task successfully. Consequently, the aim was to provide the prompts only during periods of inactivity or silence or at stages when the participants verbally requested help [208].

Another technique that was used to avoid interrupting participants was to make use of a technique called "stimulated recall" [211]. This method allows participants to retrieve their interactive decision-making processes soon after the event. The basic concept behind the stimulated recall method is to enable participants to construct an original condition with clarity and precision if they are presented with a substantial amount of clues or stimuli that occurred during the original situation [10]. In case of an interruption, the participant was provided with enough clues to enable her or him to reconstruct the original situation with respect to her or his current activity.

6.1.3 Settings

Usability inspection is the common name for a set of methods for assessing user interfaces and discovering usability problems. Normally, usability inspections are mostly concerned with finding usability problems in a design; however, some techniques also address such issues as the severity of the usability problems and the general usability of a complete design [13]. Certain factors have to be taken into account when carrying out a usability inspection, such as individuals versus teams, the expertise of the evaluator, prescribed tasks versus self-guided exploration, and the usage and usefulness of guidelines [14].

The testing session was carried out based on a self-guided exploration method with only a few guidelines given to the participants at the beginning of the testing session. The participants were not given a static scenario with individual constant tasks to complete, since this method could lead to similar results for most participants [212]. Instead, they were asked to use the material as they would normally. The researcher did not interact with the participants much after giving the initial directions on how to complete the task, except for providing a few questions to stimulate the feedback loop [213]. It was a priority to make sure that the presence of the researcher had as little effect on the participant as possible so that the researcher could reproduce a testing experience that closely mirrored real-world situations [214].

A concise interview was added at the end of the monitoring period [215]. This interview included only a few questions, most of which dealt with the participants' overall feelings and thoughts about the resource they had worked with. The interview questions served as a good means for confirming and in some cases reminding participants of the issues of concern. Moreover, the questions were designed in such a way so as not to require any further reference to the resource to be answered and to enable the participant to respond using the knowledge and understanding already acquired throughout the user experience [216].

6.2 Testing Results and Implementation Changes Made

6.2.1 Matching Questions

It turned out that the row responsible for accepting the question elements did not possess the necessary color contrast [218] for the students to be sure that a question element had successfully been accepted, as shown in figure 13. This problem became evident when participants tried to look for and find the question element after it had been accepted by a specific answer slot. The underlying cause of this problem was the fact that after the acceptance of a question element by a particular answer slot, the colors of these two objects would normally blend together and result in a faint and shaded color that was difficult for the participants to notice [217].

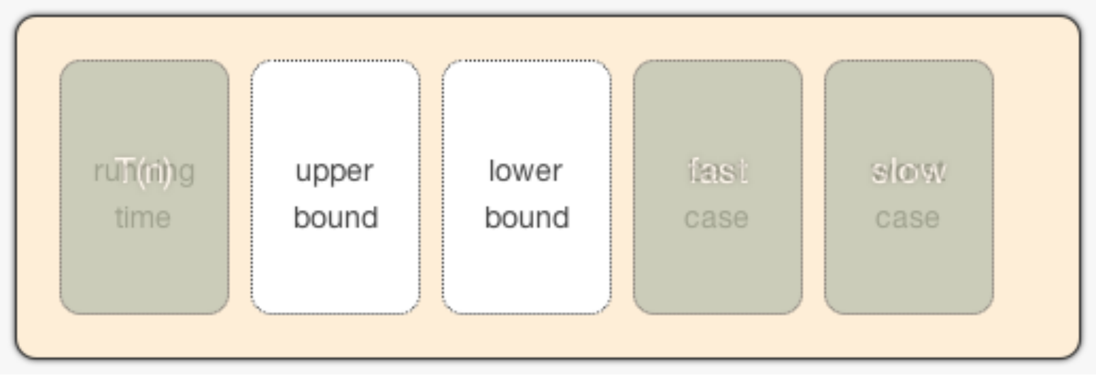


Figure 13. The vague color problem in the matching question type after the question elements had been accepted.

The problem shown in the figure above was overcome by changing the entire background color of the answer row. The answer row has an acceptable level of contrast now that is easy for users to notice [219]. Figure 14 below shows the result of this change. Users will now immediately notice the acceptance of a question element by a particular answer slot, even when using the question type for the first time.

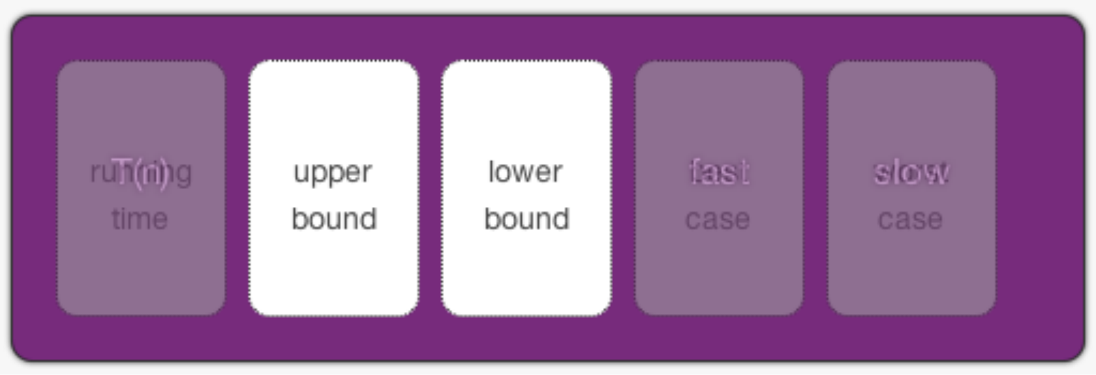
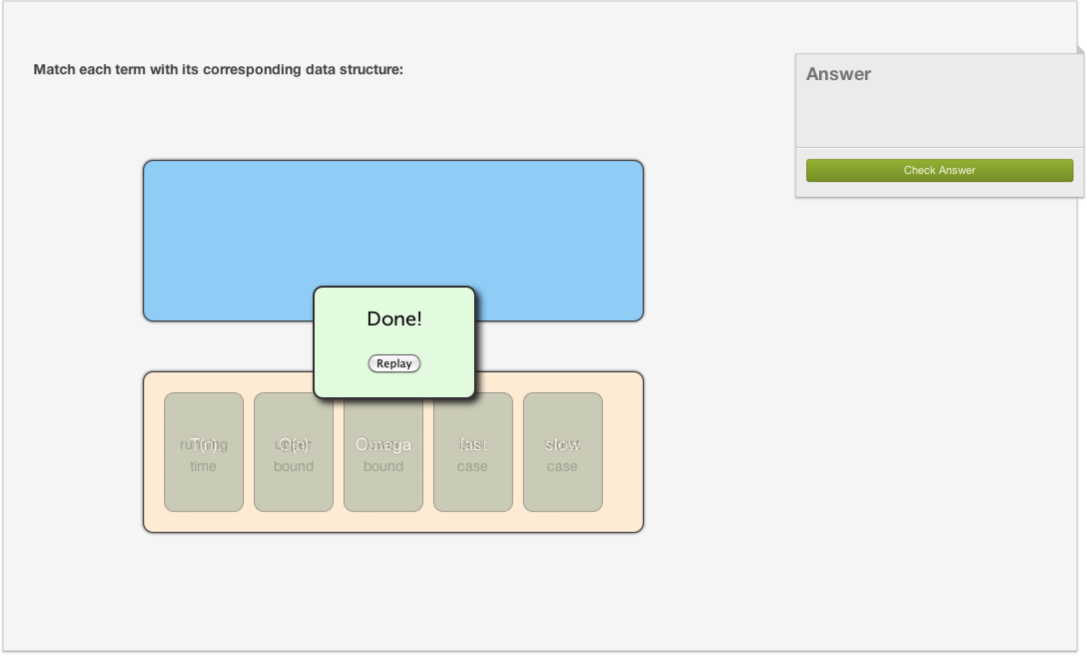


Figure 14. The answer row for a typical matching question after fixing the vague color issue.

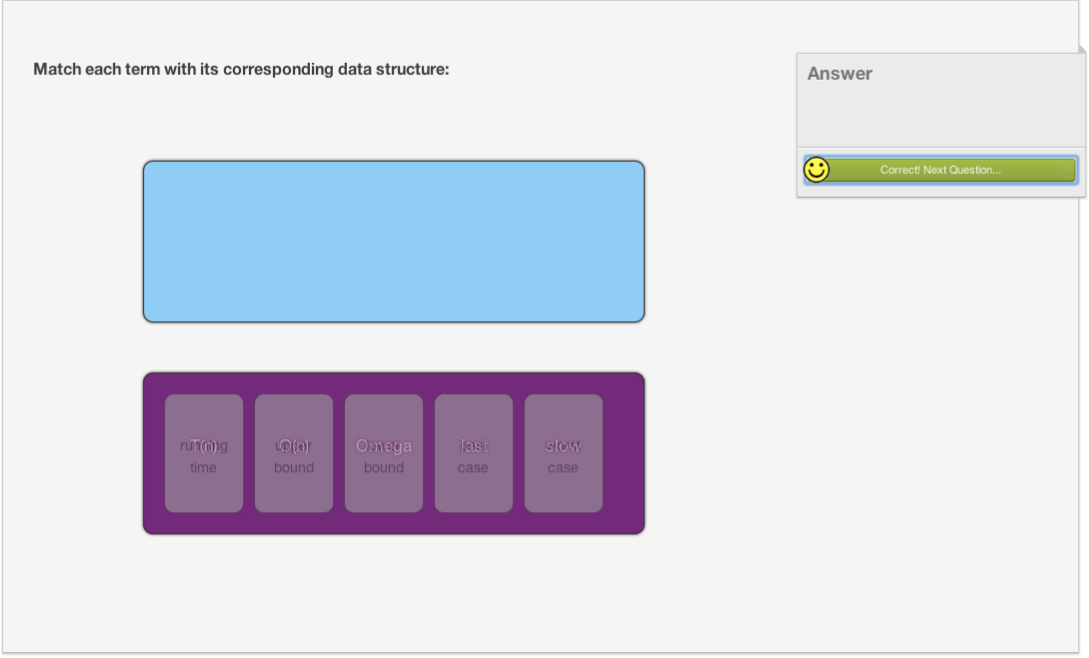
Another potential issue in the matching question type revealed during the testing sessions was the fact that a pop-up message box containing a “Replay” button would appear right after the user had placed the last question element in its corresponding answer box, as seen in figure 15. The button congratulated the learner on successfully completing the matching question mini-game. The problem was that the appearance of the pop-up box, together with the answer button already present on the same page, confused the learner about which button they needed to press. This also served to confuse the user interaction pattern of the learner with respect to the other question types, since the learner was just about to uncover the interaction patterns. Therefore, users expected the same pop-up mechanism to appear in all the other question types as well, which was an evident inconsistency [220].



Match each term with its corresponding data structure:

Figure 15. The pop-up message that appeared for the matching question type after participants had completed the exercise.

Since the answer button was already a fixed part of the KAEI used in the OpenDSA, this particular problem was fixed by completely removing the congratulatory pop-up message at the end of the mini-game and equipping the answer button with the functionality to provide the required feedback to the learner regarding whether or not they had succeeded in answering the matching question. The result of this fix is shown in figure 16 below.



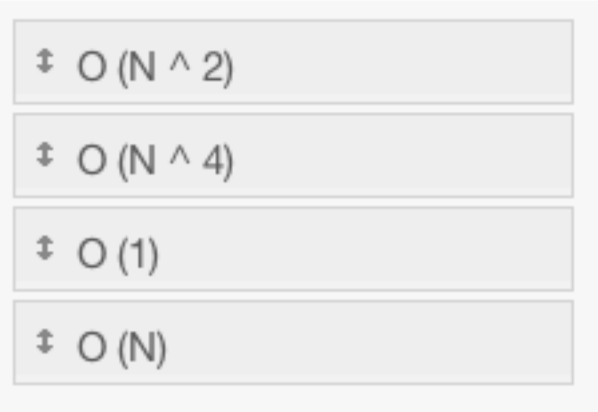
Match each term with its corresponding data structure:

Figure 16. The matching question type after removing the problematic pop-up message.

The figure above shows the final result of the two corrections that were made to the matching question type based on problems that had emerged during the user testing sessions. The resulting matching question type now has the potential to provide users with a better and more effective user experience and usability than the one that was initially designed and implemented. Moreover, the check answer mechanism used in this question type is now completely consistent with all the other chapter summary question types throughout OpenDSA [221].

6.2.2 Ordering Questions

Different individuals encountered problems with the ordering question type as well. The first issue that arose had to do with the way the exponent element was portrayed in a specific question, which is for the most part common in computing platforms, as shown in figure 17 [222]. Feedback was received on changing the way the exponent elements were portrayed in conventional mathematical representations together with the fact that the font was not legible enough due to its small size.



↕ O (N ^ 2)

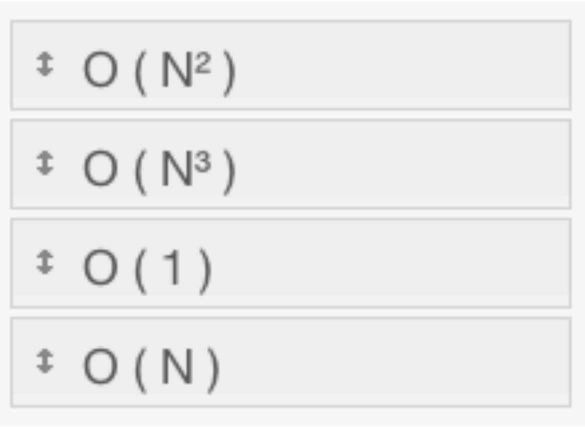
↕ O (N ^ 4)

↕ O (1)

↕ O (N)

Figure 17. Element representation of a particular ordering question before changes.

The aforementioned issue was fixed by changing the representation style of the exponents to a more conventional style using html codes for special characters [170], together with increasing the font size [223] of all the elements, as seen in figure 18 below. Using the new layout with this particular question leads to less time being wasted focusing on the details of how the components are displayed and allows for more time to be focused on the main objective, which is understanding the concept behind each component. It can also lead to a more effective user experience.



↕ O (N^2)

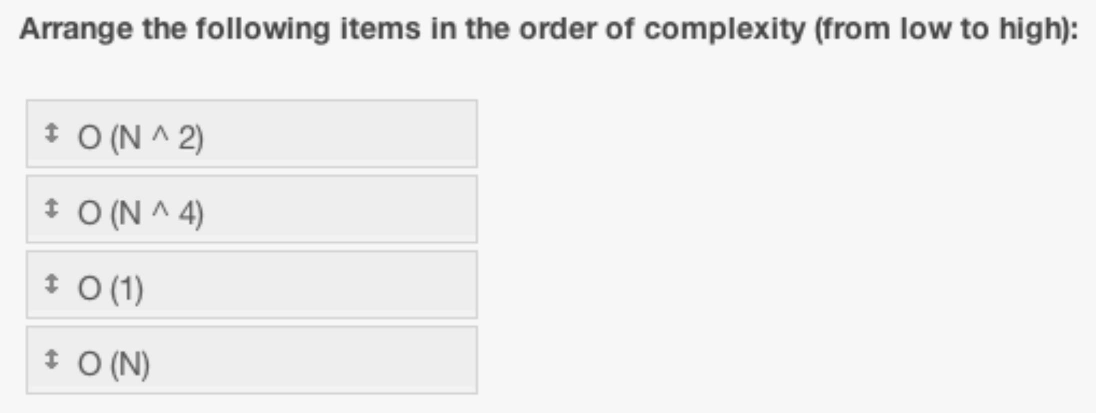
↕ O (N^3)

↕ O (1)

↕ O (N)

Figure 18. Element representation of a particular ordering question after changes.

The next problem that emerged when using the same question shown in the figure above was the fact that some individuals found the question title confusing when reading and thinking about the title for the first time [224]. As shown in figure 19, the question asks the learner to arrange the items provided in the order of complexity on an ascending basis. However, the participants had a hard time grasping the meaning of the title after reading it for the first time.



Arrange the following items in the order of complexity (from low to high):

↕ O (N ^ 2)

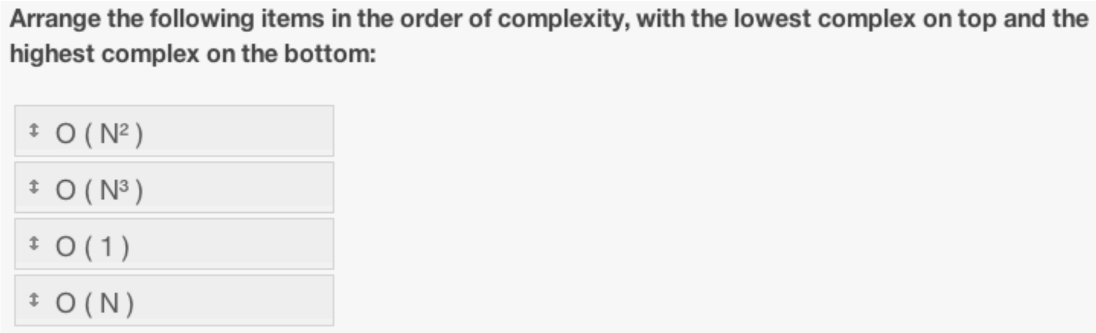
↕ O (N ^ 4)

↕ O (1)

↕ O (N)

Figure 19. A sample question title for an ordering question that participants found confusing.

The aforementioned problem was resolved by rephrasing the question title and completely changing the part of the question that led to the confusion. Figure 20 below shows that instead of mentioning *from low to high* inside parentheses, the eventual location of the element with the lowest level of complexity together with the eventual location of the one with the highest level of complexity are specifically mentioned [226]. It might be worth mentioning that the font size of the original question title was also increased for better legibility [225]. Therefore, the figure below is the result of the changes both to the question title and the answer elements.



Arrange the following items in the order of complexity, with the lowest complex on top and the highest complex on the bottom:

↕ O (N^2)

↕ O (N^3)

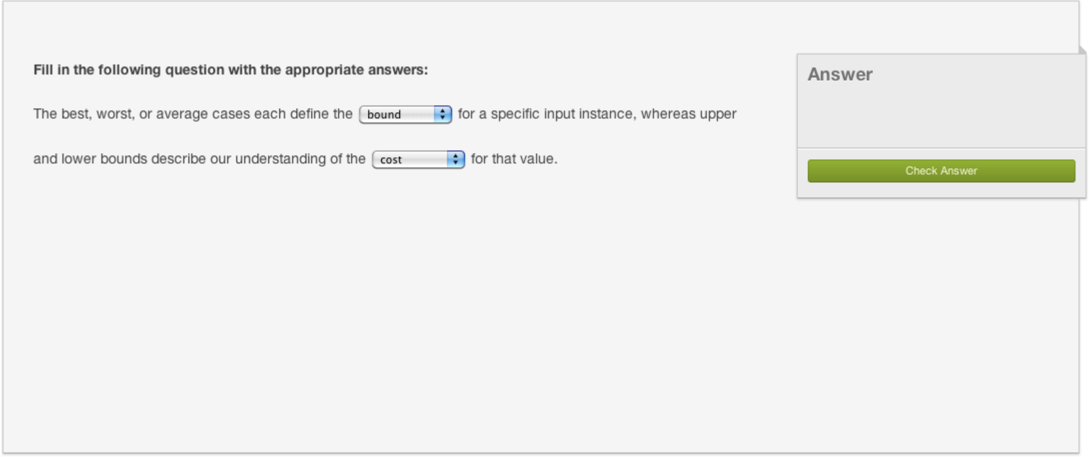
↕ O (1)

↕ O (N)

Figure 20. The result of a specific ordering question after the testing sessions and corrections.

6.2.3 Fill-in-the-blank Questions

The only problem that emerged during the testing of the fill-in-the-blank question type had to do with the low legibility level of the questions in this category due to the small font size used for them [223, 225]. Figure 21 shows the original appearance of a typical fill-in-the-blank question before any changes.

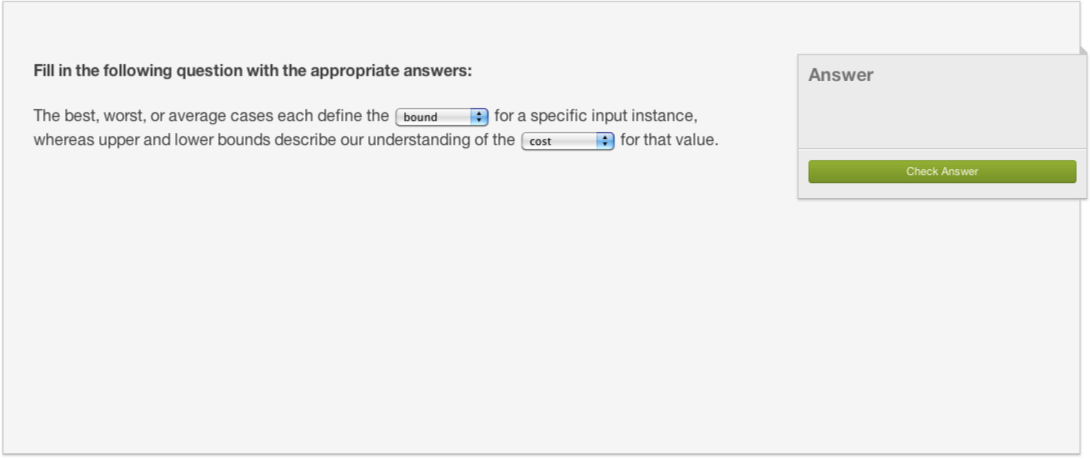

Fill in the following question with the appropriate answers:

The best, worst, or average cases each define the bound for a specific input instance, whereas upper

and lower bounds describe our understanding of the cost for that value.

Figure 21. The original state of a typical fill-in-the-blank question before user testing.

Since the only problem here had to do with the small font size, the issue was fixed by increasing the font size of both the question title and the answer. In addition, the line break between the two parts of the answer sentence was also removed. Figure 22 below illustrates these changes.

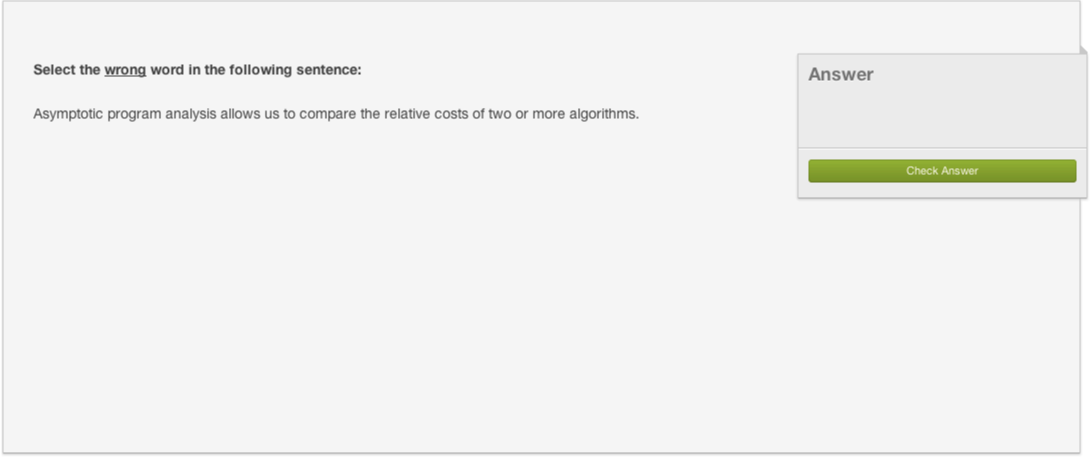

Fill in the following question with the appropriate answers:

The best, worst, or average cases each define the bound for a specific input instance,
whereas upper and lower bounds describe our understanding of the cost for that value.

Figure 22. A particular fill-in-the-blank question after making necessary corrections following the user testing sessions.

6.2.4 Error-guessing Questions

Participants found no significant problems with the error-guessing question type apart from the fact that, once again, the font size was not quite legible, as shown in figure 23 [223, 225].

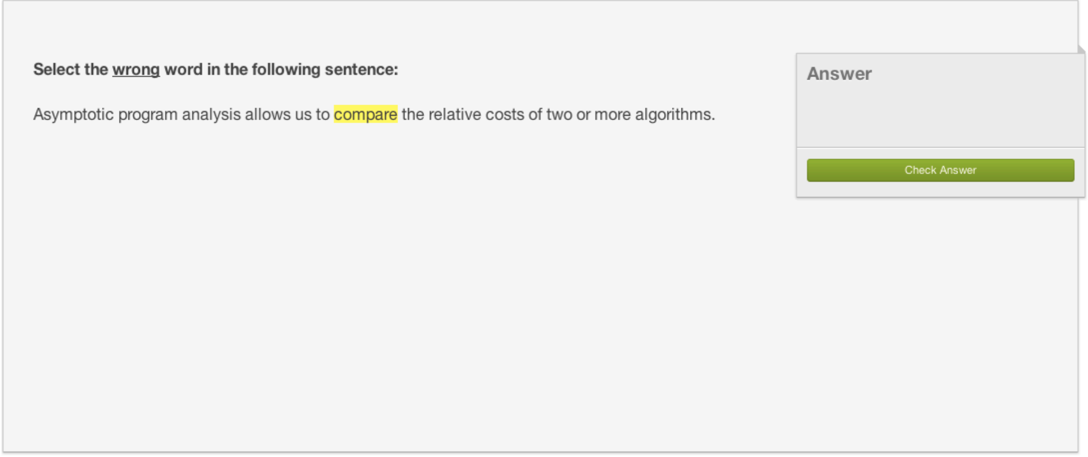


Select the wrong word in the following sentence:

Asymptotic program analysis allows us to compare the relative costs of two or more algorithms.

Figure 23. The original state of a typical error-guessing question before making corrections.

Once again, the aforementioned issue was addressed by increasing the font size of both the question title and the answer sentence, as shown in figure 24 below. The new font size provides better overall legibility for different audiences in varying age groups. It is important to note, however, that considering an average font size that would be able to satisfy a wide range of users would be the wisest decision [171].



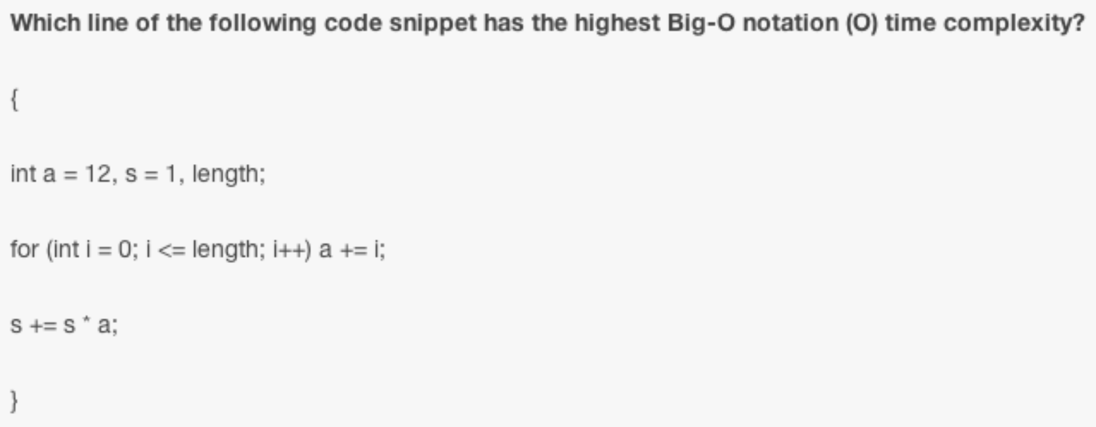
Select the wrong word in the following sentence:

Asymptotic program analysis allows us to compare the relative costs of two or more algorithms.

Figure 24. A typical error-guessing question after making corrections following user testing.

6.2.5 Multiple-choice Questions

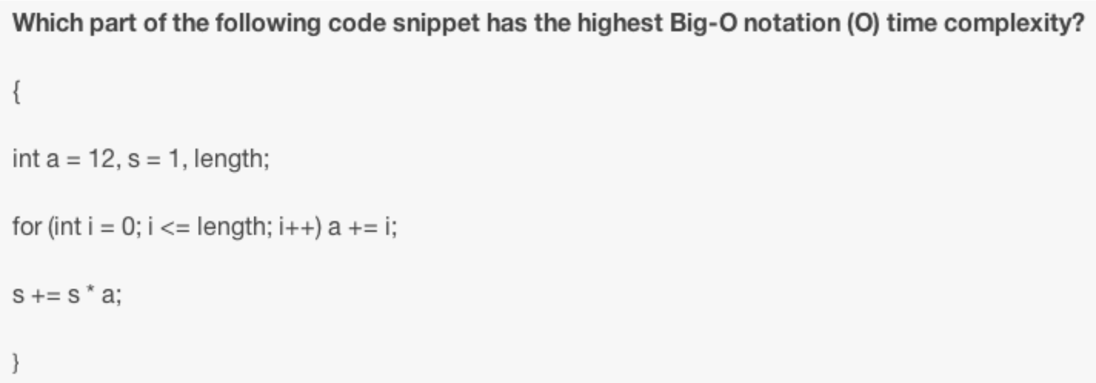
The collected feedback on the multiple-choice question type was positive. It was pointed out, however, that the use of the word "line" to refer to different segments of a code snippet is not accurate [224, 226]. Therefore, this word was replaced with the word "part" in the question title. Moreover, the font size in all parts of the question, including the question title, code snippet, and the answer choices was increased [223, 225]. Figures 25 and 26 show the original state of this multiple-choice question and how it looked after making the corrections.



Which line of the following code snippet has the highest Big-O notation (O) time complexity?

```
{
int a = 12, s = 1, length;
for (int i = 0; i <= length; i++) a += i;
s += s * a;
}
```

Figure 25. The original state of a typical multiple-choice question on code snippets.



Which part of the following code snippet has the highest Big-O notation (O) time complexity?

```
{
int a = 12, s = 1, length;
for (int i = 0; i <= length; i++) a += i;
s += s * a;
}
```

Figure 26. The same multiple-choice question as in the previous figure after the corrections.

CHAPTER 7

DISCUSSION

In an era in which educational technologies are being developed quite quickly, the Internet has become a strong means to supply learners with a unique learning environment worldwide. Distance education on the Internet has remarkably affected the ways in which we communicate and learn and has fostered different learning and teaching opportunities. One of the several advantages of distance education is that it provides both teachers and learners with a flexible learning environment in terms of time and location, since learners are not required to be physically present at the same place and time as the instructor. [15.]

Distance education can be implemented in a variety of different forms and for different purposes, such as supported learning, blended learning, and fully online learning, all of which pursue the same concept in learning development, that is, one in which learning is developed through sharing ideas and thoughts and through distinctive interactions between participants. Several factors could affect the online learning experience: infrastructure, the quality of support systems, content, and assessment as well as peer-support networks, to name a few. Therefore, planning and implementing distance education courses is an intricate task that includes numerous factors that need to be taken into account to provide learners with effective learning environments. [15.]

People are being bombarded with new computing and IT developments and the release of each new type of technology eventually leads to a change in everyday living processes and alters the way in which things are done. This could point to the significant role of how individuals are being provided with an appropriate education and learning, and specifically to the critical role of personal learning environments as one form of education and learning provision. However, it is of significance to bear in mind that learning is an exceptionally complex process and is mainly located in rich contexts due to the fact that usable and solid knowledge can only be attained by participating in tasks and environments that are genuine [17].

This thesis work explored the concept of personal learning environments and presented the fact that they are rapidly expanding everywhere and becoming increasingly popular. Information technologies have increasingly expanded and have found their way into every aspect of people's lives, from talking wirelessly on a personal cellphone to controlling different devices and gadgets via eye movements while wearing special glasses. Everything in people's lives seems to be happening at an incredibly fast pace, with the result being that people's thought processes are also occurring at a faster pace. However, when considering the ever-increasing pace of IT expansion throughout the world, it sometimes seems difficult to keep up with every technological advancement that takes place in different areas of CS. Therefore, it is important that each computer scientist has a specific area of expertise in which he or she could carry out original research and develop academically. In this modern digital age, in which almost everything is done through digital processes, education and the methods by which it is provided are also becoming increasingly digitalized and modern.

Thanks to online personal learning environments, both educators and learners now have the opportunity to decrease some of the pressure of their demanding daily lives by choosing to take part in virtual online learning classroom sessions for different course topics and actually study any part of their chosen online courses at another location in the world as long as they possess an Internet connection and a compatible device. While exploring such personal learning environments, it was mentioned that they could be as simple as an online eBook or as intricate as a completely virtual distance-learning environment, one that includes various educational gadgets together with the course material.

Following the exploration of the AA chapter in OpenDSA for the first time, it became evident that the chapter content is boring, makes use of too much text, and includes little or no opportunity for interaction. The implementation part of this thesis provided a solution to tackle the difficulty of CS students in learning different topics inside this chapter. This was made possible by designing and developing novel interactive summary exercise types together with a graphing visualization for the AA chapter. Furthermore, a few sample questions for each summary question type were developed.

CHAPTER 8

CONCLUSION

The background chapter to this study first investigated algorithm analysis in CS education, in which algorithms, analysis of algorithms, and learning in analysis of algorithms were discussed. Visualizations in CS education were introduced next, in which different concepts such as software visualization, media, and multimedia learning, together with animations and their effects on learning, were explored. It was mentioned that software visualization is founded upon the use of graphical representations to make the process of describing the operations of a computer easier and more tangible. Visualizations in CS education is a field that has been brought to the attention of educationalists on a broad scale, and achievements have been reached on a large scale in this area of study. Instructors are looking for better methods for teaching and presentation than those provided by the conventional textbook approach and established methodologies, for which visualizations can be an innovative and useful answer.

Evaluation methods in CS education were explored in the same background chapter on visualizations, in which it was stated that only with the help of well-designed and effective evaluation and assessment methods can one obtain a correct estimate of student learning and be provided with constructive feedback on the status of their progress. Without an effective assessment mechanism in place, the learning process will experience major loss and will not be as efficient. This section discussed topics such as computer-based evaluation and evaluation task design.

Interactive eBooks, eTextbooks, as well as the effects of eTextbooks on learning were discussed next. Each of the concepts was explored in detail. It was mentioned, for instance, that eBooks have become universally accepted and standard and that eTextbooks are not equivalent to eBooks. Furthermore, some forms of content presentation together with certain factors that influence learning were discussed in the same section.

The OpenDSA platform was introduced in detail in the next chapter. This chapter started by giving an introduction to algorithm visualizations. At this point, the objective of the OpenDSA project together with the architecture and the technology behind it were explored. JSAV and Khan Academy were also presented since they belonged to the technology behind the OpenDSA. The chapter that came next investigated the methodology behind this thesis work.

The implementation contribution of this thesis was documented and discussed in the following chapter, which started by reviewing the AA chapter content and then moved on to summary exercises and eventually to the actual implementation work, which was presented using screenshots. Each implemented question type was presented together with the challenges behind each type of question. The programmed codes for each question type were also presented together with code documentations in red.

The user testing chapter came next. It consisted of two parts: testing experience, and testing results and implementation changes made. The testing scenario, techniques, and settings were presented in the first part. The results of the user testing were discussed in the last part. The last part also discussed the decisions that were made after analyzing the testing results as well as the actual implementation changes that were made. Before/after screenshots were provided at this point to show the exact changes that were made to the implemented project.

BIBLIOGRAPHY

[1] G. Attwell. Personal Learning Environments for creating, consuming, remixing and sharing. Proceedings of the TENComptence Open Workshop (2007), 36–41.

[2] W. Harlen, and R. D. Crick. Testing and Motivation for Learning, Graduate School of Education. Assessment in Education (2003), 10(2).

[3] R. Catrambone, A. Fleming Seay. Using Animations to Help Students Learn Computer Algorithms. Human Factors (2002), 44(3).

[4] M. D. Byrne, R. Catrambone, and J. T. Stasko. Do Algorithm Animations Aid Learning? GVU Technical Report (1996).

[5] A. Ochoche. A Proposed Evaluation Technique for Computer Science Studies in Primary and Secondary Schools. Leonardo Electronic Journal of Practices and Technologies (2008), 139–150.

[6] X. Lin, and J. Hubbard. Books of the Future. Drexel University INFO 653: Digital Libraries, Final Paper (2000).

[7] J. B. Fenwick Jr., B. L. Kurtz, P. Meznar, R. Phillips, and A. Weidner. Developing a Highly Interactive eBook for CS Instruction. SIGCSE (2013).

[8] E. Fouh, V. Karavirta, D. A. Breakiron, S. Hamouda, S. Hall, T. L. Naps, and C. A. Shaffer. Design and architecture of an interactive eTextbook – The OpenDSA system. Science of Computer Programming (2014), 88, 22–40.

[9] C. A. Shaffer, V. Karavirta, A. Korhonen, and T. L. Naps. OpenDSA: Beginning a Community Active-eBook Project. Koli (2011), 112–117.

[10] D. Cotton, and K. Gresty. Reflecting on the think-aloud method for evaluating e-learning. British Journal of Educational Technology (2006), 37(1), 45–54.

[11] J. A. Dijksman, and S. Khan. Khan Academy: the world's free virtual school. American Physical Society (2011).

[12] E. Sarrion. *jQuery UI*. O'Reilly Media Inc. (2012).

[13] J. Nielsen. Usability Inspection Methods. Conference Companion (1994).

[14] C. M. Karat, R. Campbell, and T. Fiegel. Comparison of Empirical Testing and Walkthrough methods in User Interface Evaluation. SIGCHI (1992), 397–404.

[15] I. Sahin. Predicting student satisfaction in distance education and learning environments. Turkish Online Journal of Distance Education (2007), 8(2).

[16] C. A. Shaffer. *Data Structures and Algorithm Analysis (Java Version)*. Department of Computer Science, Virginia Tech (2013).

[17] S. S. Liaw, W. C. Chang, W. H. Hung, and H. M. Huang. Attitudes toward search engines as a learning assisted tool: approach of Liaw and Huang's research model. Computers in Human Behavior (2006), 177–190.

[18] T. Naps et al. Exploring the Role of Visualization and Engagement in Computer Science Education. Report of the Working Group on Improving the Educational Impact of Algorithm Visualization, ITiCSE -WGR '02 Working Group Reports from ITiCSE on Innovation and technology in computer science education (2002), 131–152.

[19] A. Korhonen et al. Requirements and Design Strategies for Open Source Interactive Computer Science eBooks. ITICSE -WGR '13 Working Group Reports (2013), 53–72.

[20] R. Baecker. Sorting out sorting: A Case Study of Software Visualization for Teaching Computer Science. Software Visualization: Programming as a Multimedia Experience (1998).

[21] M. H. Brown. *Algorithm Animation*. MIT Press (1998).

[22] J. Stasko. TANGO: A Framework and System for Algorithm Animation. IEEE Computer (1990), 23, 27–39.

[23] L. M. Phillips, and S Norris. *Visualization in Mathematics, Reading and Science Education*. Springer (2010), 77–80

[24] G. Ro ̈ßling, T. Naps, M. S. Hall, V. Karavirta, A. Kerren, C. Leska, A. Moreno, R. Oechsle, S. H. Rodger, J. Urquiza-Fuentes, and J. A. Vela ́zquez-Iturbide. Merging interactive visualizations with hypertextbooks and course management. SIGCSE Bull. (2006), 38(4), 166–181.

[25] G. Ro ̈ßling, and T. Vellaramkalayil. A visualization-based computer science hypertextbook prototype. Trans. Comput. Educ. (2009), 9(2), 11:1–11:13.

[26] C. D. Hundhausen, S. A. Douglas, and J. T. Stasko. A Meta-Study of Algorithm Visualization Effectiveness. *Journal of Visual Languages and Computing* (2002).

[27] A. W. Lawrence. *Empirical Studies of the Value of Algorithm Animation in Algorithm Understanding.* PhD thesis (1993).

[28] M. D. Byrne, R. Catrambone, and J. Stasko. Evaluating Animations as Student Aids in Learning Computer Algorithms. *Computers & Education* (1996), 33, 253–278.

[29] C. Kann, R. W. Lindeman, and R. Heller. Integrating Algorithm Animation into a Learning Environment. *Computers & Education* (1997), 28, 223–228.

[30] S. R. Hansen, N. H. Narayanan, and D. Schrimpsher. Helping Learners Visualize and Comprehend Algorithms. *Interactive Multimedia Electronic Journal of Computer-Enhanced Learning* (2000), 2(1).

[31] T. Naps, J. Eagan, and L. Norton. JHAVE ́: An Environment to Actively Engage Students in Web-based Algorithm Visualizations. *31st ACM SIGCSE Technical Symposium on Computer Science Education* (2000), 109–113.

[32] C. D. Hundhausen, and S. Douglas. Using Visualizations to Learn Algorithms: Should Students Construct Their Own, or View an Expert's? *IEEE Symposium on Visual Languages* (2000), 21–28.

[33] R. E. Mayer. *Multimedia learning*. Cambridge University Press (2001).

[34] S. Y. Mousavi, R. Low, and J. Sweller. Reducing cognitive load by mixing auditory and visual presentation models. *Journal of Educational Psychology* (1995), 87, 319–334.

[35] S. Ainsworth. How do animations influence learning? *Recent Innovations in Educational Technology that Facilitate Student Learning* (2008), 37–67.

[36] B. Tversky, J. B. Morrison, and M. Bétrancourt. Animation: can it facilitate? International Journal of Human-Computer Studies (2002), 57(4), 247–262.

[37] M. Scaife, and Y. Rogers. External cognition: How do graphical representations work? International Journal of Human-Computer Studies (1996), 45(2), 185–213.

[38] J. J. Zhang, and D. A. Norman. Representations in distributed cognitive tasks. Cognitive Science (1994), 18(1), 87–122.

[39] A. Kay. Dynabooks: past, present, and future. LibraryQuarterly (2000), Vol.70 No.3, 95–385.

[40] H. Chu. Electronic books: viewpoints from users and potential users. Library Hi Tech (2003), Vol. 21 Iss 3, 340–346.

[41] L. A. Tedd. E-Books in academic libraries: an international overview. New Review of Academic Librarianship (2005), 11(1), 57–79.

[42] S. Ardito. Electronic books: to 'E' or not to 'E' that is the question. Searcher (2000), 8(4).

[43] D. T. Hawkins. Electronic books: reports of their death have been exaggerated. ETATS-UNIS (2002), Vol. 26 No. 4, 42–48.

[44] C. Lynch. The battle to define the future of the book in the digital world. First Monday (2001), 6(6).

[45] C. J. Armstrong, L. Edwards, and R. Lonsdale. Virtually there? E-books in UK academic libraries. Program: electronic library and information systems (2002), 36(4), 27–216.

[46] L. Appleton. The use of eBooks in midwifery education: the student perspective. Health and Information Libraries Journal (2004), 21(4), 52–245.

[47] C. A. Shaffer, T. L. Naps, and E. Fouh. Truly Interactive Textbooks for Computer Science Education. *Proceedings of the Sixth Program Visualization Workshop* (2011), 97–103.

[48] C. C. Miller, and J. Bowman. E-books outsell print books at Amazon. The New York Times (2011).

[49] AAP estimates: E-book sales rose 117% in 2011 as print fell. Publisher's Weekly (2012).

[50] D. B. Daniel, and W. D. Woody. E-textbooks at what cost? Performance and use of electronic v. print texts. Department of Psychology, James Madison University (2013), 62, 18–23.

[51] UPDATE: Electronic book and eReader device report March 2011. National Association of College Bookstores (2011).

[52] J. A. Shepperd, J. L. Grace, and E. J. Koch. Evaluating the electronic textbook: is it time to dispense with the paper text? Teaching of Psychology (2008), 35, 2–5.

[53] W. D. Woody, D. B. Daniel, and C. A. Baker. E-books or textbooks: students prefer textbooks. Computers & Education (2010), 55, 945–948.

[54] D. B. Daniel, and D. Willingham. Electronic textbooks: what's the rush? Science (2012), 335, 1570–1571.

[55] R. B. Kvavik. Convenience, communications, and control: how students use technology. In D. G. Oblinger, and J. L. Oblinger (Eds.), Educating the net generation. EDUCAUSE Center for Applied Research (2005).

[56] R. A. R. Gurung. Pedagogical aids and student performance. Teaching of Psychology (2003), 30, 92–95.

[57] R. A. R. Gurung. Pedagogical aids: learning enhancers or dangerous detours. Teaching of Psychology (2004), 31, 164–166.

[58] R. A. R. Gurung, and D. Daniel. Evidence-based pedagogy: do pedagogical features enhance student learning? In D. S. Dunn, and S. L. Chew (Eds.), Best practices for teaching, introduction to psychology (2005), 41–55.

[59] R. A. R. Gurung, J. Weidert, and A. Jeske. Focusing on how students study. Journal of the Scholarship of Teaching and Learning (2010), 10(1), 28–35.

[60] R. Wesp, and J. Miele. Student opinion of the quality of teaching activities poorly predicts pedagogical effectiveness. Teaching of Psychology (2008), 35, 360–362.

[61] D. Dunning, K. Johnson, J. Ehrlinger, and J. Kruger. Why people fail to recognize their own incompetence. Current Directions in Psychological Science (2003), 12, 83–87.

[62] D. B. Daniel, and W. D. Woody. They hear, but do not listen: retention for podcasted material in a classroom context. Teaching of Psychology (2010), 37, 199–203.

[63] L. A. Clyde. Electronic books. Teacher Librarian (2005), 32(5), 45–47.

[64] G. Coleman. E-books and academics: an ongoing experiment. Canadian Library Association Feliciter (2004), 4, 124–125.

[65] T. Cavanaugh. E-books and accommodations: is this the future of print accommodation? Teaching Exceptional Children (2002), 35(2), 56–61.

[66] S. A. Long. The case for e-books: an introduction. New World Library (2003), 104(1/2), 29–32.

[67] D. G. Myers. Using new interactive media to enhance the teaching of psychology (and other disciplines) in developing countries. Perspectives on Psychological Science (2009), 4(1), 99–100.

[68] A. K. Taylor. Students learn equally well from digital as from paperbound texts. Teaching of Psychology (2011), 38, 278–281.

[69] A. Dillon. Reading from paper versus screens: a critical review of the empirical literature. Ergonomics (1992), 35(10), 1297–1326.

[70] J. Nielson, and K. Pernice. *Eyetracking web usability*. Berkeley (2010).

[71] M. Kropman, H. P. Schoch, and H. Y. Yeoh. An experience in e-learning: using an electronic textbook. In R. Atkinson, C. McBeath, D. Jonas-Dwyer, and R. Phillips (Eds.), Beyond the comfort zone: Proceedings of the 21st ASCILITE Conference (2004), 512–515.

[72] R. E. Mayer, J. Heiser, and S. Lonn. Cognitive constraints on multimedia learning: when presenting more material results in less understanding. Journal of Educational Psychology (2001), 93, 187–198.

[73] T. Morineau, C. Blanche, L. Tobin, and N. Guéguen. The emergence of the contextual role of the e-book in cognitive processes through an ecological and functional analysis. International Journal of Human-Computer Studies (2005), 62, 329–348.

[74] L. Nelson, and F. O'Neil. Electronic monographs in the academic library: an implementation story. LASIE (2001), 32, 13–20.

[75] L. S. Vygotski. *Mind in society: The development of higher psychological processes*. Cambridge (1978).

[76] D. T. Clark, S. P. Goodwin, T. Samuelson, and C. Coker. A qualitative assessment of the Kindle e-Book reader: results from initial focus groups. Performance Measurement and Metrics (2008), 9(2), 118–129.

[77] Y. Kang, M. Wang, and R. Lin. Usability evaluation of e-books. Displays (2009), 30(2), 49–52.

[78] H. R. Jamali, D. Nicholas, and I. Rowlands. Scholarly e-books: the views of 16,000 academics. New Information Perspectives (2010), 61(1), 33–47.

[79] S. Robinson. Student use of a free online textbook. Academy of Educational Leadership Journal (2011), 15(3), 1–10.

[80] J. Amanda, R. Szapkiw, J. Courduff, K. Carter, and D. Bennett. Electronic versus traditional print textbooks: A comparison study on the influence of university students' learning. *Computers & Education* (2013), 63, 259–266.

[81] C. Connell, L. Baliss, and W. Farmer. Effects of e-book readers and tablet computers on reading comprehension. International Journal of Instructional Media (2012), 39(2), 131–140.

[82] H. Jeong. A comparison of the influence of electronic books and paper books on reading comprehension, eye fatigue, and perception. The Electronic Library (2012), 30(3), 390–408.

[83] M. R. Nelson. E-books in higher education. EDUCAUSE Review (2008), 43(2), 40–56.

[84] M. Vassiliou, and J. Rowley. Progressing the definition of e-book. Library Hi Tech (2008), 26(3), 355–368.

[85] W. D. Chesser. The e-textbook revolution. In Library Technology Reports (2011), 47(8), 28–40.

[86] A. Hoseth, and M. McLure. Perspectives on e-books from instructors and students in the social sciences. Reference & User Services Quarterly (2012), 51(3), 278–288.

[87] R. Reynolds. Trends influencing the growth of digital textbooks in US higher education. Publishing Research Quarterly (2011), 27(2), 178–187.

[88] S. A. Berg, K. Hoffmann, and D. Dawson. Not on the same page: undergraduates' information retrieval in electronic and print books. The Journal of Academic Librarianship (2010), 36(6), 518–525.

[89] M. Murray, and J. Pérez. E-textbooks are coming: are we ready? Issues in Informing Science & Information Technology (2011), 8, 49–60.

[90] A. Shamir, and I. Shlafer. E-books effectiveness in promoting phonological awareness and concept about print: a comparison between children at risk for learning disabilities and typically developing kindergarteners. Computers & Education (2011), 57(3), 1989–1997.

[91] J. Sun, J. Flores, and J. Tanguma. E-textbooks and students' learning experiences. Decision Sciences Journal of Innovative Education (2012), 10(1), 63–77.

[92] E. Thelen, G. Schoner, C. Scheier, and L. B. Smith. The dynamics of embodiment: a dynamic field theory of infant preservative reaching errors. Behavioral and Brain Sciences (2001), 24, 1–86.

[93] D. K. Mayes, V. K. Sims, and J. M. Koonce. Comprehension and workload differences for VDT and paper-based reading. International Journal of Industrial Ergonomics (2001), 28, 367–378.

[94] J. M. Noyes, and K. J. Garland. VDT versus paper-based text: reply to Mayes, Sims and Koonce. International Journal of Industrial Ergonomics (2003), 31, 411–423.

[95] R. McFall. Electronic textbooks that transform how textbooks are used. The Electronic Library (2005), 23(1), 72–81.

[96] V. Karavirta, and C. A. Shaffer. JSAV: The JavaScript Algorithm Visualization Library. *Proceedings of the 18th ACM conference on Innovation and technology in computer science education* (2013), 159–164.

[97] S. Diehl. Software Visualization. Number 2269 in Lecture Notes in Computer Science (2002).

[98] E. Fouh, M. Akbar, and C.A. Shaffer. The role of visualization in computer science education. Computers in the Schools (2012), 29, 95–117.

[99] C. A. Shaffer, M.L. Cooper, A.J.D. Alon, M. Akbar, M. Stewart, S. Ponce, and S. H. Edwards. Algorithm visualization: The state of the field. ACM Transactions on Computing Education (2010), 10, 1–22.

[100] C. A. Shaffer, M. Akbar, A. J. D. Alon, M. Stewart, and S. H. Edwards. Getting algorithm visualizations into the classroom. In Proceedings of the 42nd ACM Technical Symposium on Computer Science Education (SIGCSE) (2011), 129–134.

[101] R. J. Ross, and M. T. Grinder. Hypertextbooks: Animated, active learning, comprehensive teaching and learning resources for the web. In S. Diehl, editor, Software Visualization, number 2269 in Lecture Notes in Computer Science (2002), 269–284.

[102] G. Ro ̈ßling, and T. Vellaramkalayil. First steps towards a visualization-based computer science hypertextbook as a Moodle module. In Proceedings of the 5th Program Visualization Workshop, volume 224 of Electronic Notes in Theoretical Computer Science (2009), 47–56.

[103] J. Urquiza-Fuentes, and J. A. Velázquez-Iturbide. A survey of successful evaluations of program visualization and algorithm animation systems. ACM Trans (2009), 9, 1–21.

[104] M. L. Cooper, C. A. Shaffer, S. H. Edwards, and S. P. Ponce. Open source software and the algorithm visualization community. Science of Computer Programming (2014), 88, 82–91.

[105] S. Hall, E. Fouh, D. Breakiron, M. Elshehaly, and C. A. Shaffer. Education innovation for data structures and algorithms courses. Number 5951 in Proceedings of ASEE Annual Conference (2014).

[106] reStructuredText. reStructuredText: Markup syntax and parser component of docutils. reStructuredText (2013).

[107] JSAV. JSAV project repository. JSAV (2013).

[108] CS2008 Review Taskforce. Computer science curriculum 2008: An interim revision of CS 2001. Review Taskforce (2008).

[109] C. Kehoe, J. Stasko, and A. Taylor. Rethinking the evaluation of algorithm animations as learning aids: an observational study. Int. J. Hum.-Comput. Stud. (2001), 54, 265–284.

[110] J. H. McMillan, and J. Hearn. Student self-assessment: The key to stronger student motivation and higher achievement. Educ. Horizons (2008), 87, 40–49.

[111] H. Andrade, and A. Valtcheva. Promoting learning and achievement through self-assessment. Theory Into Practice (2009), 48, 12–19.

[112] V. Karavirta, and C. A. Shaffer. Creating Engaging Online Learning Material with the JSAV JavaScript Algorithm Visualization Library. IEEE transactions on learning technologies (2015).

[113] B. B. Morrison, and B. DiSalvo. Khan Academy Gamifies Computer Science. *Proceedings of the 45th ACM technical symposium on Computer science education* (2014), 39–44.

[114] N. B. Mehta, A. L. Hull, J. B. Young, and J. K. Stoller. Just Imagine: New Paradigms for Medical Education. Educating Physicians: A Call for Reform of Medical (2010).

[115] B. S. Bloom. The 2 Stigma Problem: The Search for Methods of Group Instruction as Effective as One-to-One Tutoring. Educational Researcher (1984), Vol. 13 No. 6.

[116] K. Scalise, and B. Gifford. Computer-Based Assessment in E-Learning: A Framework for Constructing 'Intermediate Constraint' Questions and Tasks for Technology Platforms. The Journal of Technology, Learning, and Assessment (2006), Vol. 4 No. 6.

[117] R. E. Bennett. On the Meaning of Constructed Response. In R. E. Bennett, Ward, W. C. (Ed.), Construction versus Choice in Cognitive Measurement: Issues in Constructed Response, Performance Testing, and Portfolio Assessment (1993), 1–27.

[118] S. J. Osterlind. Constructing Test Items: Multiple-Choice, Constructed-Response, Performance, and Other Formats. Norwell (1998).

[119] R. Glaser. Cognitive and environmental perspectives on assessing achievement. Paper presented at the Assessment in the Service of Learning ETS Invitational Conference (1988).

[120] R. Glaser. Expertise and Assessment. In M. C. Wittrock & E. L. Baker (Eds.), Testing and Cognition. Englewood Cliffs (1991).

[121] L. B. Resnick, and D. P. Resnick. Assessing the inking Curriculum: New Tools for Educational Reform. In B. R. Gifford & M. C. O'Connor (Eds.), Changing Assessments: Alternative Views of Aptitude, Achievement and Instruction (1992), 37–76.

[122] L. Shepard. Interview on assessment issues with Lorrie Shepard. Educational Researcher (1991), 20(2), 21–23, 27.

[123] L. Shepard. Psychometricians' beliefs about learning. Educational Researcher (1991), 20(6), 2–16.

[124] G. Chung, and E. Baker. Year 1 Technology Studies: Implications for Technology in Assessment. CSE Technical Report (1997).

[125] C. G. Parshall, J. Spray, J. Kalohn, and T. Davey. Issues in Innovative Item Types. In Practical Considerations in Computer-Based Testing (2002), 70–91.

[126] J. Pellegrino, N. Chudowsky, and R. Glaser. Knowing What Students Know: The Science and Design of Educational Assessment. In N. R. C. Center for Education (2001).

[127] W. Van der Linden. On Complexity in CBT. In C. Mills, M. Potenza, J. Fremer, and W. Ward, Computer-Based Testing (2002), 89–102.

[128] W. Stout. Test Models for Traditional and Complex CBTs. In C. Mills, M. Potenza, J. Fremer, and W. Ward, Computer-Based Testing (2002), 103–118.

[129] T. M. Haladyna. Item Shells and Conclusions. In Developing and Validating Multiple-Choice Test Items (1994), 109–161.

[130] F. Rijmen, and P. DeBoeck. The random weights linear logistic test model. Applied Psychological Measurement (2002), 26, 217–285.

[131] U. C. Ben. Evaluation techniques in higher institutions in Nigeria. Education-trust Fund Capacity Building Workshop for Lecturers of Universities in Nigeria (2006).

[132] J. Nagy. Adapting to market conditions: plagiarism, cheating and strategies for cohort customization. Studies in Learning, Evaluation, Innovation, and Development (2006), 3(2), 37–47.

[133] C. Kissock. Curriculum planning for social studies. John Wiley and Sons Ltd. (1981), 94–113.

[134] M. C. Alkin. Product for improving educational evaluation. Evaluation Comment (2002), Vol. 2 No. 3.

[135] N. E. Gronlund. Measurement and evaluation in teaching. Macmillan Publishing Coy (2002).

[136] M. B. Ogunniyi. Educational measurement and evaluation. Longman Group (2004).

[137] G. Brown, and M. Manogue. Refreshing lecturing: a guide for lecturers. AMEE Medical Education Guide (2001), 23, 231–44.

[138] D. A. Cook, D. M. Dupras. A practical guide to developing effective Web-based learning. J Gen Intern Med. (2004), 19, 698–707.

[139] G. E. Kennedy. Promoting cognition in multimedia interactivity research. J Interact Learn Res. (2004), 15, 43–61.

[140] D. Swagerty Jr., S. Studenski, R. Laird, S. Rigler. A case-oriented web-based curriculum in geriatrics for third-year medical students. J Am Geriatr Soc. (2000), 48, 12–1507.

[141] D. A. Cook, W. G. Thompson, K. G. Thomas, M. R. Thomas, and V. S. Pankratz. Impact of Self-Assessment Questions and Learning Styles in Web-Based Learning: A Randomized, Controlled, Crossover Trial. IT in Medical Education (2006), Vol. 81 No. 3.

[142] M. Maleck, M. R. Fischer, B. Kammer et al. Do computers teach better? A media comparison study for case-based teaching in radiology. Radiographics (2001), 21, 32–1025.

[143] J. N. Hudson. Computer-aided learning in the real world of medical education: does the quality of interaction with the computer affect student learning? Med Educ. (2004), 38, 95–887.

[144] P. Devitt, and E. Palmer. Computer-aided learning: an overvalued educational resource? Med Educ. (1999), 33, 39–136.

[145] T. Gao, and J. D. Lehman. The effects of different levels of interaction on the achievement and motivational perceptions of college students in a Web-based learning environment. Journal of Interactive Learning Research (2003), 14, 86–367.

[146] R. J. Mislevy. Test Theory Reconceived. Journal of Educational Measurement (1996), 33, 379–417.

[147] K. A. Ericsson, and H. A. Simon. Protocol analysis: verbal reports as data (Revised ed.). *Psychological review* (1980), 87(3), 215.

[148] M. S. Crowther, C. C. Keller, and G. L. Waddoups. Improving the quality and effectiveness of computer-mediated instruction through usability evaluations. British Journal of Educational Technology (2004), 35(3), 289–304.

[149] M. Oliver. J. McBean, G. Conole, and J. Harvey. Using a toolkit to support the evaluation of e-learning. Journal of Computer Assisted Learning (2002), 18(2), 199–208.

[150] C. A. Shaffer. *Data Structures and Algorithm Analysis (C++ Version)*. Department of Computer Science, Virginia Tech (2013).

[151] R. Sedgewick, and P. Flajolet. *An Introduction to the Analysis of Algorithms*. Addison-Wesley (2013).

[152] A. Aho, J. E. Hopcroft, and J. D. Ullman. *The Design and Analysis of Algorithms*. Addison-Wesley (1975).

[153] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press (2009).

[154] D. E. Knuth. *The Art of Computer Programming. Volume 1: Fundamental Algorithms*. Addison-Wesley (1968).

[155] D. E. Knuth. *The Art of Computer Programming. Volume 2: Seminumerical Algorithms*. Addison-Wesley (1969).

[156] D. E. Knuth. *The Art of Computer Programming. Volume 3: Sorting and Searching*. Addison-Wesley (1973).

[157] D. E. Knuth. *The Art of Computer Programming. Volume 4A: Combinatorial Algorithms, Part 1*. Addison-Wesley (2011).

[158] D. E. Knuth. Mathematical analysis of algorithms. Information Processing 71, Proceedings of the IFIP Congress, Ljubljana (1971), 19–27.

[159] I. Pu. Software engineering, algorithm design and analysis. Volume 2. Goldsmiths, University of London (2006).

[160] T. Beaubouef, J. Mason. Why the High Attrition Rate for Computer Science Students: Some Thoughts and Observations. Inroads, The SIGCSE Bulletin (2005), Vol. 37 No. 2.

[161] T. Beaubouef. Why Computer Science Students Need Math. Inroads, SIGCSE Bulletin (2002), Vol. 34 No. 4.

[162] T. Beaubouef, R. Lucas, and J. Howatt. The Unlock System: Enhancing Problem Solving Skills in CS1 Students. Inroads, SIGCSE Bulletin (2001), Vol. 33 No. 2, 43–46.

[163] Codeacademy. Codeacademy - About the Company. Codeacademy (2016).

[164] N. S. Yanofsky. Towards a definition of an algorithm. *Journal of Logic and Computation* (2010).

[165] Desmos. Desmos - About. Desmos (2016).

[166] R. Thomas. A graphing approach to algebra using Desmos. ICTCM (2015).

[167] D. Ebert. Graphing Projects with Desmos. *Mathematics Teacher* (2015), 108(5), 388–391.

[168] jQuery UI. jQuery user interface – Selectmenu. jQuery UI (2016).

[169] R. Harper. Correcting computer-based assessments for guessing. *Journal of Computer Assisted Learning* (2003), 19(1), 2–8.

[170] HTML Codes Table. Characters and symbols. ASCII (2016).

[171] I. Darroch et al. The effect of age and font size on reading text on handheld computers. Springer Berlin Heidelberg (2005), 253–266.

[172] M. K. McGowan, P. R. Stephen, and C. W. Bradley. Student perceptions of electronic textbooks. *Issues in Information Systems* (2009), 10(2), 459–465.

[173] A. Z. Krohn, and M. Kugel. Remotely operable teaching system and method therefor. U.S. Patent No. 5,458,494 (1995).

[174] P. L. Scriffiny. Seven reasons for standards-based grading. *Educational Leadership* (2008), 66(2), 70–74.

[175] D. R. Sadler. Interpretations of criteria-based assessment and grading in higher education. *Assessment & Evaluation in Higher Education* (2005), 30(2), 175–194.

[176] OpenDSA. OpenDSA Project Homepage. AlgoViz (2016).

[177] C. Aldrich. *Learning by doing: A comprehensive guide to simulations, computer games, and pedagogy in e-learning and other educational experiences*. John Wiley & Sons (2005).

[178] jQuery UI. jQuery user interface – Sortable. jQuery UI (2016).

[179] D. A. Breakiron. Evaluating the integration of online, interactive tutorials into a data structures and algorithms course. Diss., Virginia Tech (2013).

[180] B. Bhakta et al. Using item response theory to explore the psychometric properties of extended matching questions examination in undergraduate medical education. *BMC Medical Education* (2005), 5(1).

[181] D. B. Swanson, K. Z. Holtzman, and K. Allbee. Measurement characteristics of content-parallel single-best-answer and extended-matching questions in relation to number and source of options. *Academic Medicine* (2008), 83(10), 21–24.

[182] D. B. Swanson et al. Psychometric characteristics and response times for content-parallel extended-matching and one-best-answer items in relation to number of options. *Academic Medicine* (2006), 81(10), 52–55.

[183] S. M. Case, D. B. Swanson, and D. R. Ripkey. Comparison of items in five-option and extended-matching formats for assessment of diagnostic skills. *Academic Medicine* (1994), 69(10), 1–3.

[184] T. H. Adams. The development of a method for analysis of questions asked by teachers in classroom discussion. Diss., Rutgers University (1964).

[185] M. J. Aschner. Asking questions to trigger thinking. NEA Journal (1961), 50, 44–46.

[186] B. S. Bloom. Taxonomy of educational objectives: Handbook I: Cognitive domain. David McKay (1956).

[187] R. L. Carner. Levels of questioning. Education (1963), 83, 546–550.

[188] R. D. Clements. Art student-teacher questioning. Studies in Art Education (1964), 6, 14–19.

[189] J. J. Gallagher. Expressive thought by gifted children in the classroom. Elementary English (1965), 42, 559–568.

[190] F. J. Guszak. Teacher questioning and reading. The Reading Teacher (1967), 21, 227–234.

[191] J. R. Moyer. An exploratory study of questioning in the instructional processes in selected elementary schools. Diss., Columbia University (1966).

[192] R. T. Pate, and N. H. Bremer. Guiding learning through skillful questioning. Elementary School Journal (1967), 67, 417–422.

[193] N. M. Sanders. Classroom questions: What kinds? Harper & Row (1966).

[194] J. E. Schreiber. Teachers' question-asking techniques in social studies. Diss., University of Iowa (1967).

[195] M. D. Gall. The use of questions in teaching. *Review of educational research* (1970), 40(5) 707–721.

[196] E. Sumita, F. Sugaya, and S. Yamamoto. Measuring non-native speakers' proficiency of English by using a test with automatically-generated fill-in-the-blank questions. *Proceedings of the second workshop on Building Educational Applications Using NLP* (2005).

[197] S. M. Case, and D. B. Swanson. Constructing Written Test Questions for the Basic and Clinical Sciences. 3rd edition. National Board of Medical Examiners (2001).

[198] S. J. Peitzman et al. Comparison of fact recall with high order questions in multiple choice examinations as predictors of clinical performance of medical students. Academic Medicine (1990), 65, 59–60.

[199] S. M. Case, D. B. Swanson, and D. F. Becker. Verbosity, window dressing and red herrings: do they make a better test item? Academic Medicine (1996), 71, 28–30.

[200] P. McCoubrie. Improving the fairness of multiple-choice questions: a literature review. *Medical Teacher* (2004), 26(8), 709–712.

[201] J. M. Carroll. Scenario-based Design. Envisioning Work and Technology in System Development Wiley (1995).

[202] J. Buur, S. Bødker. Creative common grounds–reframing usability work. Department of Computer Science, University of Aarhus (2000).

[203] S. Bødker. Scenarios in user-centred design–setting the stage for reflection and action. *Interacting with computers* (2000), 13(1), 61–75.

[204] S. Bødker. Through the Interface–a Human Activity Approach to User Interface Design. Lawrence Erlbaum (1991).

[205] P. C. Wright, and A. F. Monk. The use of think-aloud evaluation methods in design. *ACM SIGCHI Bulletin* (1991), 23(1), 55–57.

[206] J. D. Gould, and C. Lewis. Designing for usability: key principles and what designers think. *Communications of the ACM* (1985), 28(3), 300–311.

[207] M. Kuniavsky. *Observing the user experience: a practitioner's guide to user research*. Morgan Kaufmann (2003).

[208] M. Hassenzahl, S. Diefenbach, and A. Göritz. Needs, affect, and interactive products–Facets of user experience. *Interacting with computers* (2010), 22(5), 353–362.

[209] P. Saenger. *Space between words: The origins of silent reading*. Stanford University Press (2000).

[210] M. Nørgaard, and K. Hornbæk. What do usability evaluators do in practice? An explorative study of think-aloud testing. *Proceedings of the 6th conference on Designing Interactive systems* (2006).

[211] S. M. Gass, and A. Mackey. *Stimulated recall methodology in second language research*. Routledge (2000).

[212] R. Atterer, M. Wnuk, and A Schmidt. Knowing the user's every move: user activity tracking for website usability evaluation and implicit interaction. *Proceedings of the 15th international conference on World Wide Web* (2006).

[213] J. S. Dumas, and J. Redish. *A practical guide to usability testing*. Intellect Books (1999).

[214] D. Wildman. Getting the most from paired-user testing. *Interactions* (1995), 2(3), 21–27.

[215] C. Abras, D. Maloney-Krichmar, and J. Preece. User-centered design. *Bainbridge, W. Encyclopedia of Human-Computer Interaction. Thousand Oaks: Sage Publications* (2004), 37(4), 445–456.

[216] L. E. Wood. Semi-structured interviewing for user-centered design. *Interactions* (1997), 4(2), 48–61.

[217] P. Baudisch, and C. Gutwin. Multiblending: displaying overlapping windows simultaneously without the drawbacks of alpha blending. *Proceedings of the SIGCHI conference on Human factors in computing systems* (2004).

[218] D. Jameson, and L. M. Hurvich. Theory of brightness and color contrast in human vision. *Vision research* (1964), 4(1), 135–154.

[219] Y. F. Ma, and H. J. Zhang. Contrast-based image attention analysis by using fuzzy growing. *Proceedings of the eleventh ACM international conference on Multimedia* (2003).

[220] W. O. Galitz. *The essential guide to user interface design: an introduction to GUI design principles and techniques*. John Wiley & Sons (2007).

[221] B. Laurel, and S. J. Mountford. *The art of human-computer interface design*. Addison-Wesley Longman Publishing Co. (1990).

[222] W. A. Martin. Computer input/output of mathematical expressions. *Proceedings of the second ACM symposium on Symbolic and algebraic manipulation* (1971).

[223] M. Bernard, C. H. Liao, and M. Mills. The effects of font type and size on the legibility and reading time of online text by older adults. *Extended abstracts on Human factors in computing systems* (2001).

[224] J. Crossley, et al. Good questions, good answers: construct alignment improves the performance of workplace-based assessment scales. *Medical education* (2011), 45(6), 560–569.

[225] D. Hawthorn. How universal is good design for older users? *Computers and the Physically Handicapped* (2003), 73–74.

[226] J. Groenendijk, and M. Stokhof. On the semantics of questions and the pragmatics of answers. *Semantics: Critical Concepts in Linguistics* (1985), 288.