Journal of Multimedia Information System VOL. 8, NO. 3, September 2021 (pp. 191-196): ISSN 2383-7632 (Online)
http://doi.org/10.33851/JMIS.2021.8.3.191

Brief Paper:

Improving Computational Thinking Comprehension through Visualized Sorting App Development

Jongwan Kim[1]*, Taeseong Kim[2]

Abstract: Computational thinking refers to the process and method of solving everyday problems using computers. When teaching a computational thinking class for computer majors and non-majors at university, the easiest example to deliver the concept of computational thinking is sorting. Sorting is the concept of arranging given data in order. In this work, we have implemented four visualized sorting algorithms that anyone can easily use. In particular, it helps to understand the difference between the algorithms by showing the number of comparisons and exchanges between elements, which are the criteria for evaluating the performance of the sorting algorithm in real time. It was confirmed that the practice of using the sorting visualization app developed in this research contributed to the improvement of students' understanding of computational thinking.

Keywords: Computational Thinking, Sorting Algorithm, Visualized Sorting App, Comparison and Exchange Operation.

I. INTRODUCTION

Computational thinking refers to the process and method of solving everyday problems using computers [1]. Given a certain problem, the way humans solve this problem and the way an individual solves this problem using a computer may be different. There is also a study using App Inventor 2 developed at MIT as a method to increase the computational thinking learning ability of elementary school teachers [2]. This study suggests that it can stimulate the interest of elementary school students in that smartphone-linked education, a feature of App Inventor, is possible rather than Scratch. We will conduct a study on how to promote computational thinking in college students.

The principal researcher has been teaching computational thinking classes for computer majors and non-majors since 2016. In 2017, in the first class of the Computational Thinking course at UBC in Canada, he watched the practice of arranging 52 trump play cards along with the introduction of the class contents. Based on this experience, he has been conducting card sorting practice in the first hour of his computational thinking class. The process in which students place cards by hand is a kind of sorting process, and reports are being submitted that describe the process each team practiced step by step. By explaining that the content described in this report is an algorithm, it informs that the students can describe the algorithm even if they have never programmed before, thereby inducing the students' interest [3].

Sorting is the first subject to learn in data structure and algorithm classes in computer science and engineering majors. We describe and implement based on ascending sort with increasing numbers in this work. Representative sorting algorithms are divided into basic sorting group of bubble sorting, selection sorting, and insertion sorting as well as merge sorting, quick sorting, and heap sorting with good execution time efficiency. When the number of elements to be sorted is n, the average execution time of basic sort takes O(n2), and merge, quick, and heap sort takes O(nlog2n) on average [4].

Explaining basic sorting makes it easy for students to understand. However, even with the same basic sort, many students find it difficult to completely understand the difference between bubble sort, selection sort, and insertion sort. In particular, if the basic operation of sorting, the number of comparisons and exchanges of two elements, is calculated, the difference in understanding is evident. This work aims to develop an application program that easily shows the difference in sorting methods in the form of animation for anyone from children to computer majors, and to use it to experience the concept of computational thinking from the user's point of view.

Section 2 describes the related works and Section 3 explains the development of the visualized sorting

Manuscript received August 30, 2021; Revised September 13, 2021; Accepted September 15, 2021. (**ID No. JMIS-21M-08-028**)
Corresponding Author (*): Jongwan Kim, Gyeongsan, Daegu 38453 – S. Korea, +82-53-850-6575, jwkim@daegu.ac.kr
[1]Division of Computer and Information Engineering, Daegu University, Gyeongsan, S. Korea, jwkim@daegu.ac.kr
[2]Dept. of Computer Engineering, Daegu University, Gyeongsan, S. Korea, tay97kim@naver.com

program. Section 4 presents the results of a student satisfaction survey. Finally, conclusion is in Section 5.

II. RELATED WORKS

If you search the Internet for sorting algorithm implementation programs, you will find the VisuAlgo website (https://visualgo.net/en), a well-organized program that is easy to understand. In 2011, Dr. Steven Halim of the National University of Singapore (NUS) developed the VisuAlgo web site that can learn various data structures and algorithms through animation based on the JavaScript language [5]. Various computer algorithms such as sorting, search, trees and graphs, and hash tables are animated on the site. Announced in 2015, VisuAlgo features a web-based algorithmic visualization tool that requires no additional software to be installed, allows users to specify inputs, and the visualization works with those inputs. A large number of visualizable data structures and algorithms are included in VisuAlgo, detailed algorithm animation steps are provided, and an online quiz tool, which is an important learning element, is provided too [5].

However, despite the excellence of VisuAlgo itself, the sorting program is described in English with only some terms translated, which makes some students unfamiliar with English uncomfortable. In this research, we develop a visualized sorting program with Korean interface to make it more convenient to use than the VisuAlgo site. Figure 1 presents an execution screen of Computational Thinking for Everyone - Sorting. The program we developed can be used by anyone, and it is necessary to disclose the source when using it. CCL (Creative Commons License) is a positive method of indicating permission for use of works by which copyright holders can freely use their works by attaching certain conditions to them. We also allow free use under the condition of attribution, non-commercial, and prohibition of alteration (BY-NC-ND). Therefore, it can be freely used not only by students learning sorting algorithms, but also by instructors teaching computational thinking and/or algorithm courses.

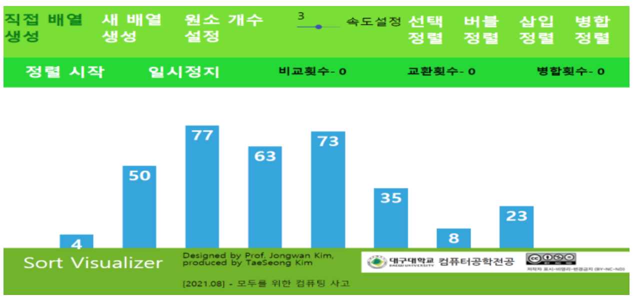


Fig. 1. Computational Thinking for Everyone – Sort Execution Screen.

Saranto Psycharis et al. developed a visualization algorithm for three basic sorting algorithms: selection, bubble, and insertion, and performed a study analyzing the effect of the visualization algorithm on self-efficacy, metacognition, and computational thinking concepts [6]. According to the experimental results of [6], it is said that creating a sorting simulation model in the Easy Java Simulator environment is as helpful to students as writing pseudo code, and confirming the hypothesis that students' self-efficacy for computational thinking is also improved.

III. VISUALIZED SORTING PROGRAM DEVELOPMENT

The sorting program developed in this work aims to make it easier for users to understand the process of sorting rather than simply presenting the result of the sorting algorithm. The app is written in JavaScript (JS), and as a library to use, React, which is useful for single page application (SPA) development, is used to display the sorting progress in real time [7]. The structure of the developed sorting program determines the size of the array, selects the sorting method to be sorted, and provides the start and pause functions of sorting. In particular, the app execution screen has a main feature that visually expresses the current state of the array and the entire process of sorting, a header that displays the number of comparisons, exchanges, and merges counted when sorting, which is a characteristic of the development app, and finally a footer with other information such as the program name, update date and developers. Figure 2 shows the structure of the developed application program.

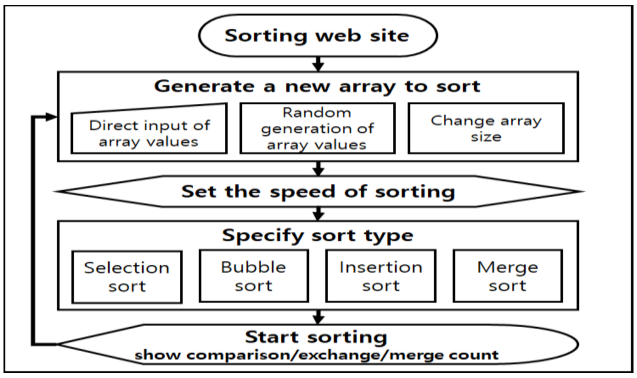


Fig. 2. Schematic Diagram of Sorting Visualizer App.

If a user click on the sorting app site https://sort-visualizer-duce.vercel.app/ shown in the schematic diagram in Figure 2, an array of size 8 consisting of random numbers ranging from 1 to 100 is created, and the user can start sorting with the array. Alternatively, you can directly create an array of elements of different sizes or numbers and start sorting. The purpose of this research is to allow people who want to understand sorting to compare the differences in sorting methods.

After the user has created an array suitable for sorting, he or she can select the desired sorting speed and sorting method. First, the user can specify from 1 to 5 as the sorting speed, and it was implemented smartly so that the sorting speed is increased in proportion to the size of the array. If the speed is set to 1, it is helpful to clearly understand the concept of the sorting algorithm to be compared by slowly checking the animation process of comparing and exchanging elements while sorting with a small number of elements. On the other hand, if the speed is set to 5, the sorting process of arrays with large element sizes can be visually confirmed very quickly, thus avoiding boredom. In the general case, you can run it using speed 2, 3, or 4 according to the purpose of the user's usage.

When the user selects the array size, sorting speed, and sorting method, the preparation stage is over. Then the user clicks the Sort Start button, the sorting starts as shown in Figure 3. In also, when the user clicks the pause button, the pause function works for use whenever you want to check the number of element comparisons and exchanges during sorting. Figure 3 shows the gray color of the bars of the elements that have already been sorted during the selected sorting operation to help users understand and check the process easily [8]. There are also red and purple bars during sorting. The red color indicates the target element that needs to be exchanged. For example, in selection sort, whenever the minimum value is found, the color of the bar corresponding to the minimum value is changed to red. In addition, purple indicates the element currently being inspected during the sorting process, and it is easy to check where the sorting is performed through the purple bar. Finally a light blue bar indicates an element that has not yet been sorted.

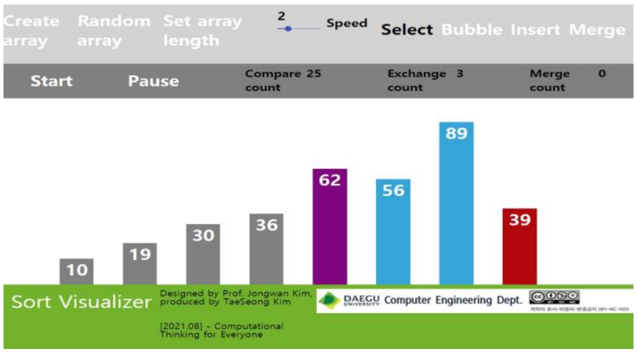


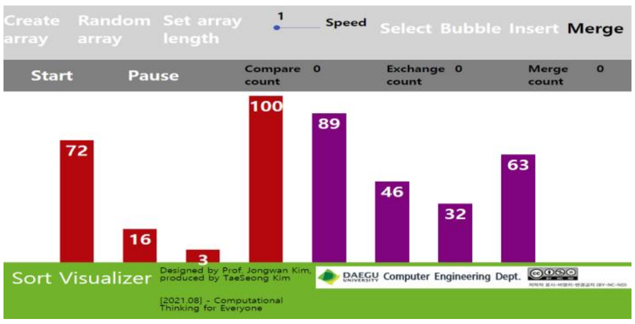
Fig. 3. Meaning of Bar Colors in the Sorting Process

As an example of sorting performance evaluation, when five elements 13, 47, 32, 4, and 25 are given, the number of comparisons, exchanges, and merges using the developed sorting app is presented in Table 1. As the data changes, the performance of the methods can also change.

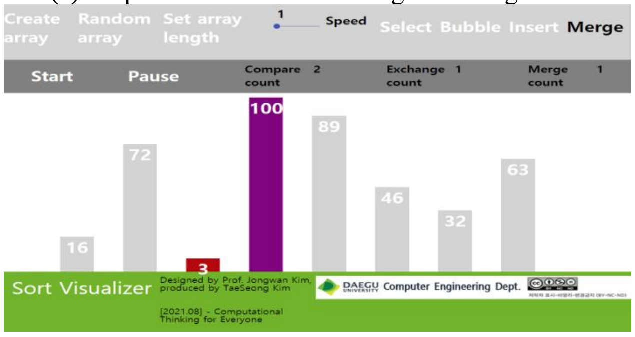
Table 1. Performance Evaluation of Sorting Methods for five elements 13, 47, 32, 4, and 25.

Sorting Methods	# of Comparisons	# of Exchanges	# of Merges
Selection	10	4	-
Bubble	10	6	-
Insertion	9	6	-
Merge	14	6	4

Unlike the three basic sorts, each merge sort recursively calls merge sort twice on left and right elements inside, so you cannot directly specify the sorted elements when the merge sort is executed once. Instead, light gray is used for the bar color of elements that are not in the range of merge sort currently in progress. Figure 4(a) shows how the merge sort is divided into two ranges, and Figure 4(b) shows that the rest of the elements that are not being sorted are set to light gray to distinguish them. With this color classification, the user can more easily understand which part the merge sort is currently in and it helps to understand the sorting process.



(a) Separate Two Division Ranges in Merge Sort.



(b) Displays the Color of the Element Currently Being Inspected (except for Gray).

Fig. 4. Visually Check the Progress of Merge Sort.

IV. STUDENT SATISFACTION SURVEY RESULTS

The usage of the developed sorting program was lectured, and students were given a task to use selection, bubble, and insertion sort for seven random elements, and to understand the difference in algorithms focusing on comparison/exchange operations of each sort. The first author was in charge of two lectures for the first-year students majoring in computer engineering in the first semester of 2021, and 27 out of 29 students in Class A submitted assignments for convenience, and 21 out of 28 students in Class B. A survey was conducted with the students at the end of the class. In Class A, 25 people responded to the questionnaire, and in Class B, 24 people responded.

Table 2. Survey Questions for Students.

Q1: Do you think this subject is easy?
Q2: Do you think Task 1 is an appropriate task for understanding algorithms?
Q3: Did Task 1 help you understand computational thinking?
Q4: Do you think Task 2 is an appropriate task for understanding algorithms?
Q5: Did Task 2 help you understand computational thinking?
Q6: Compared to VisuAlgo, is the Daegu University sorting app developed by the authors easier to use?

The main content of the survey related to this study is to find out whether Task 1 (Practice of sorting using Trump cards) and Task 2 (Practice of sorting algorithm app and writing an algorithm comparison report) are helpful in understanding the sorting method. In addition, by receiving students' opinions on the difficulty of the subject, we tried to figure out whether the proposed method is helpful for class understanding, and the Q6 question was to measure whether the developed app is easier to use compared to the existing VisuAlgo web. Table 2 presents the survey questions. Students responded on a 5-point Likert scale of 1 (very negative), 2 (negative), 3 (moderate), 4 (positive), and 5 (very positive).

Table 3 shows the results of understanding the sorting algorithm through Task 1 and Task 2 for 49 respondent students. Although the difference is slight, the satisfaction of the method using the trump card (Q2) and the sorting comprehension method (Q4) experienced in the sorting app practice showed different results, with class A having higher Q4 and class B having higher Q2. Naturally, similar results were obtained for Q3 and Q5, which are questions to determine whether Tasks 1 and 2 were helpful in understanding computational thinking. As a result of integrating and comparing the results of the two groups (A+B), the satisfaction level of Task 1 was slightly higher, but the level of help in understanding computational thinking was high in Task 2, although it was subtle. It can be seen from this that Task 1 performs the sorting action directly without using a computer, thus it presents a higher result in terms of student satisfaction than Task 2, which requires more time and effort to practice the app. However, it can be observed that students judge that the Task 2 practice is slightly helpful in understanding computational thinking. In addition, it can be confirmed from the Q6 response that students who use the app developed in this work rather than the VisuAlgo app get high scores in terms of task purpose and ease of use. In particular, it can be seen that the satisfaction responses of the two classes to this question are not different from the other questions.

The average response to the Q1 question, the difficulty of this subject, is 3.29, which is 19.2% ((4.07-3.29)/4.07) difference from the Q2-Q6 question average of 4.07, thus many students think that the computational thinking subject is not easy. However, it can be seen that students' understanding of computational thinking is improved by performing two tasks. The proposed research through these survey responses is judged to have contributed to the achievement of students' learning objectives.

Table 3. Survey Response Results (Students A&B in 2021, Students C in 2018).

Questions	Avg(A)	Avg(B)	Avg(A+B)	Avg(C)
Q1	3.2	3.38	3.29	-
Q2	4.08	4.42	4.24	4.19
Q3	3.88	4	3.94	4.0
Q4	4.28	3.96	4.12	4.06
Q5	4	3.92	3.96	4.09
Q6	4.09	4.08	4.09	-

Class C in the right column of Table 3 was additionally presented as a result of conducting Q2-Q5 questionnaires to students in 2018. As with the results conducted for students in 2021, the satisfaction of Task 1 was high, but the level of help in understanding computational thinking was slightly higher in Task 2. The sorting comprehension method (Q4) experienced in the sorting app practice in class C is the result of using the VisuAlgo app since it was before the development of the sorting app in this work.

V. CONCLUSION

In this paper, we implemented four sorting algorithms that anyone could use easily. In particular, it was confirmed by the students' survey that it was helpful to understand the difference between the sorting algorithms by showing the number of comparisons and exchanges between elements as the criteria for performance evaluation in real-time animation. Developing a sorting program that is easy to explain to anyone as a visualized algorithm helps to understand the concept of sorting in detail in terms of computational thinking represented by abstraction and automation. The app developed by this research team responded better to university students participating in the survey than VisuAlgo developed at the NUS. The survey confirmed that the educational effect of the proposed method was improved by 19.2% when comparing the development of the proposed sorting app and the concept understanding level 4.07 using it compared to the average difficulty of 3.29 in the computational thinking class. The sorting app program developed in this work can be used by other instructors and students in accordance with the purpose of open software that allows free use under the conditions of attribution, non-profit, and prohibition of alteration.

As a future research direction, it is expanding to linear search and binary search app development as a search method that is good for beginners to understand.

REFERENCES

[1] J. Wing, "Computational Thinking," *Communications of the ACM*, vol. 49, no. 3, pp. 33-35, 2006.
[2] B. Lim, "App Inventor 2 As a Tool for Enhancement of Computational Thinking," *Journal of The Korean Association of Information Education*, vol. 20, no. 5, pp. 519-526, 2016.
[3] J. Kim and J. Chae, *Computational Thinking for Everyone*. Seoul, Korea: Jungiksa, 2020.
[4] B. Moon, *Easy-to-Learn Algorithms 2nd Edition*. Seoul, Korea: Hanbit Academy, 2018.
[5] S. Halim, "VisuAlgo – Visualising Data Structures and Algorithms Through Animation," *Olympiads in Informatics*, vol. 9, pp. 243-245, 2015.
[6] S. Psycharis, D. Mastorodimos, K. Kalovrektis, P. Papazoglou, L. Stergioulas, and M. Abbasi, "Algorithm Visualization and its Impact on Self-efficacy, Metacognition and Computational Thinking Concepts Using the Computational Pedagogy Model in STEM Content Epistemology," in *Proceedings of International Journal of Physics and Chemistry Education*, vol. no. 4, pp. 71-84, 2018.
[7] React: A JavaScript library for building user interfaces, https://ko.reactjs.org/.
[8] T. Kim and J. Kim, "Developing an Easy-to-Use Sorting Visualization Application," in *Proceedings of the 2021 Spring Conference of the Korea Multimedia Society*, vol. 24, no. 1, pp. 462-463, April 2021.