ISSN 2278-3091
Volume 10, No.2, March - April 2021
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse311022021.pdf
https://doi.org/10.30534/ijatcse/2021/311022021

Analysis of Merge Sort and Bubble Sort in Python, PHP, JavaScript, and C language

Syed Muqeet Aqib[1], Haque Nawaz[1], Shah Muhammad Butt[1]
[1] BSCS Student, Sindh Madressatul Islam University, Karachi, Sindh, Pakistan, smuqeetaqib@gmail.com
[2]Sindh Madressatul Islam University, Karachi, Sindh, Pakistan, hnlashari@smiu.edu.pk
[3]Sindh Madressatul Islam University Karachi, Sindh, Pakistan, smbutt@smiu.edu.pk

ABSTRACT

Computer Science is all about the solving problems with the help of algorithms and sorting is one of the basic operations for any problem solving method. In sorting, the arrangement of data or objects in any particular order is done with the help of algorithms. There is more than one method available and also includes a wide range of choices in a programming language. These languages serve a different purpose in their field of the area but some can be used interchangeably for the same purpose especially for a server-side language like JavaScript can also implement for server-side tasks and right now it is being widely used all over the internet. Here this paper analyzed these languages with merge sort and bubble sort with the languages of the latest public stable versions for an idea of the performance of these languages because they are pretty much interchangeable for different uses in the market the only difference these server side languages have is their architecture. This paper compared these languages' capabilities with merge sort and bubble sort by executing them and observing them in terms of time by giving them different numbers of inputs. Analytics used an array of 2500, 5000, 7500, and 10000 lengths of an array that passes through these algorithms and noted the execution time to get a better idea of the capabilities of these languages. With this method, observes that in the latest public version of all languages python performs faster in merge sort while JavaScript performs better in bubble sort in executing 10000 inputs.

Key words : Merge Sort, Bubble Sort, Python, PHP, JavaScript, C Language, Sorting Comparison, Analysis of Algorithms, and Time Complexity.

1. INTRODUCTION

An algorithm is the set of steps that help us to solve any problem and implement it on computers to get results. Every algorithm takes some input and produces output concerning the algorithm by executing finite numbers of steps. To solve any problem, the process analyzes the problem and then comes up with steps to get the required result. Every algorithm for a solution may be different for every other person, one algorithm may take more time while one can get less and that's why the process needs an algorithm for the solution which takes less time and works efficiently with problem and which also depends on the used device. There are many algorithms built by other computer scientists which can be used for the solution to a problem because these are the most popular problems which may help to save time and work efficiently. During the process it might need to sort, searching, select, filter, create, update, and delete data to produce output and there are a lot of algorithms and the thing is there are even more than one algorithms to solve one problem which means that all algorithms dealing with the same problem might act different in term of speed and efficiency, so the question arrives that than why we need more than one algorithms for the solution why don't we drop one and always use the efficient solution. Like for sorting data, many algorithms deal with a sorting solution like merge sort, quick sort, bubble sort, insertion sort, selection sort, and many more just like that there are have many algorithms or methods for searching, selecting and other processes, it even uses these algorithms as a part of any other algorithm. To measure the efficiency of an algorithm the algorithms have tested with different numbers of inputs and analyze their time that is taken to execute successfully so that can analyze the efficiency of an algorithm[1].

For observation, languages have been used like JavaScript, PHP, Python, and C language to differentiate between the algorithms performance, these all languages have their pros and cons, and to analyze the performance of merge sort and bubble sort by running some tests to analyze their efficiency using their latest public versions of their languages[2].

1.1 Java Script

JavaScript is the most trending and popular language right now and there is a large community of developers using JavaScript. JavaScript is one of the all-rounder languages which can handle front-end tasks as well as Back-end tasks with its functionality. In this paper Node JS used to test out its

functionality with the public recommended version which is 12.18.3 because it is recommended by themselves as it is the most bug-free version this paper can use right now[3].

1.2 C Language

C Language is best for desktop applications and CPU-intensive applications. It is one of the oldest, still, the popular language that preferred by developers right now in 2020 and many developers choose C language as their tool for their solution because even it is one of the oldest languages, it is one of the fastest language even many JavaScript and python modules that developers use often are based on C language because of its efficiency. The latest version of the C language was published in 2018[4][5].

1.3 Python

Python is also one of the most demanded languages in 2020. It can be used in almost every field of work, web development, data Science, desktop applications, artificial intelligence, and many more. It is one of the easiest and contains user-friendly syntax. Its latest version is 3.8.5 released in July 2020.

1.4 PHP

PHP is a backend server-side language used for web development. It is one of the most used languages in web development and has one of the largest communities to help you if you are a newbie at it. Its latest version is 7.4 which is a stable version for public use and the new version of PHP 8 is expected to release in November of 2020, so this work is carried out with the 7.4 version of PHP.

This paper structured into 6 sections: the first section contains the introduction, the second section discusses about studies related to sorting methods and their approach, the third section discusses about merge sort and bubble sort and their advantages and disadvantages, the fourth section represents the methodology, fifth section presents the implementation part of tests on JavaScript, PHP, C language and Python with merge sort and the bubble sort, the sixth section covers the results from tests, and the seventh section concludes the paper.

2. LITERATURE REVIEW

For computers, it is important not only to function but function effectively and efficiently for which using and creating different algorithms for optimizing algorithms or functions according to the situation can be beneficial. Searching and sorting is the algorithms which may help in making the algorithms efficient because it is the most used algorithm in any function which means about any function may end up using searching or sorting and both are related to each other, searching can be used in any function and so is sorting separately but efficient sorting of data can optimize out searching data efficiency even more, that can be taken as enter data based on the first come first serve base in database and when process need to fetch any data from database, then it have to read every other data until developer get required data and that seems pretty time taking task especially when problem have a large set of data, here comes binary search which is one of the most fastest data searching algorithms, algorithm can use but to execute that properly program must have the data in sorted form and so instead of feting every data from database and then sorting developers always prefer a sorted database to apply binary search on the go[6][7][8]. Bubble sort is a linear sort type of algorithm which is so basic that it is used to develop the understanding of beginners. It is not used in many projects because it is not very good for large sets of data but the developer cannot neglect it because it can be useful when programs are interacting with short data and want just a quick sort only [9].

Bubble sort and selection sort are very close in terms of implementation and also performance but the enhance bubble sort and enhance selection sort are there for those situations where you prefer bubble sort or selection sort but it is not a bad thing to just use the enhanced version of them which have O(n2) complexity but faster than selection sort or bubble sort[10]. The enhanced version of bubble sort is here but this paper observes data that can find the enhancement in algorithms according to the latest version of programming languages because every year programming languages update and enhance their performance[11].

In 2008, Song Qin took quick sort, bubble sort, and merge sort and compared them by their time complexity by about sixty thousand elements of an array as an input and found out that merge sort is more efficient than others with the time complexity of O (nlogn) [12].

Quick and merge sort uses Divide and Conquer strategy. Both have the average time complexity of O (nlogn). However, both algorithms are quite different. The merge sort is usually required while sorting a too large set to hold or handle in internal memory. It divides the set into a few subsets of one element and then repeatedly merges the subsets into increasingly larger subsets with the elements sorted correctly until one set is left[13].

In this paper an attempt is made to analyze the efficiency of algorithms because in the real world a program interacts with data and in every application, there are some situations where developers need to know how much data is needed to execute it successfully and also calculate how little or more data process needs according to the scenario and deal with it. As all algorithms having the same purpose can be efficient when dealing with more data and less efficient according to the other algorithm and other algorithms can be efficient with less data but when process make goes through big data it might not be as efficient as it was before. Like in the modern car system, a car keeps track of other cars within its surroundings so it might want to deal with at most hundred

cars more or less so the developer prefers to program it with an algorithm that is efficient for fewer data and choose an algorithm concerning the scenario. And for example, e-commerce applications may need an algorithm that is efficient for big data processing because it has to deal with every user with a large amount of data as quickly as possible. These applications are written in different programming languages and they have a tough competition with them. This paper uses the languages that are more preferable in daily working environment efficiently. For analysis of algorithms, JavaScript, PHP, Python, and C language is used.

3. SORTING

To solve any problem with a computer program need an algorithm that might contain a whole new different algorithm in it. The basic operation which computer use to solve its problem or solve its problem more efficiently than before is Searching, Sorting can help us in many solutions even developers use it in real-life games, own business like e-commerce data list, client list developers prefer to enter new data to make a sorted list in a database that way program can utilize data faster on the go[14].

New tech also needs sorting as this paper discuss that in web application data management is easier when it is sorted and programmers use it in robots like cars to always have a track of what cars are behind us and what are in front of us to simulate their movement and predict chances of possibilities that others can perform. This type of technology often helps us in making auto-driving cars, auto parking cars and helps drivers to keep track of other vehicles that users cannot even see. This technology uses different types of algorithms that seem to be more useful concerning the situation because the action any software or robot takes should not be too early or should not be late, users cannot get the benefit of the service.

2.1 Bubble Sort

Bubble Sort is an algorithm that works by comparing and swapping two elements if it is not sorted, and performing iterations to get the sorted result. It is one of the simplest and basic algorithms a person can learn and use to get an understanding of algorithms and also is efficient in terms of space complexity because it does all its work in temporary memory[15]. Its Time Complexity is O(n^2) which is not faster than other algorithms but as this paper discuss there are situations where developers need to implement and the best fit can be bubble sort but this paper is here to see if these algorithms can perform any better despite languages[16][17].

Advantages:

1. Simplest Algorithm.
2. Sometimes the efficient way to check if any list is already in order.
3. Don’t use too much memory.

Disadvantages:

1. Efficient for an only shortlist
2. Ordering large data can be extremely slow as compared to others.

2.2 Merge Sort

Merge Sort is a sorting algorithm that follows a divide and conquers technique to achieve its goal. Many algorithms use divide and conquer methods in their own way to sort or search the object or data from any entity. Developers cannot use any algorithm without any consultation with the prediction of the situation that the program can get the benefit from that algorithm. That's why developers cannot use merge sort instead of bubble sort everywhere nor can use bubble sort instead of merge sort or even any other sort. Its Time Complexity is O(n*log n)[11][18][19].

Advantages:

4. Having Time complexity of O (nlogn).
5. It used both internal and external sorting.
6. A Stable sort algorithm.

Disadvantages:

3. As a minimum double the memory necessities of the further sorts since it is recursive.
4. Merge sort required high space complexity[20].

4. METHODOLOGY

In this manuscript, JavaScript, Python, PHP, and C language used to test out the time complexity from the latest versions of the languages. The time complexity of an algorithm can be determined by testing and analyzing by applying different numbers of inputs to the algorithms and observe its behavior. In this paper, the different languages were used and implemented the algorithms and observed the performance, that which is the best algorithm and in what scenario user or developer should choose one from them. This paper, shows implementation in of algorithms in different languages and applied different inputs to analyze the time complexity.

5. IMPLEMENTATION OF BUBBLE SORT AND MERGE SORT ALGORITHM USING JAVASCRIPT, C, PYTHON, AND PHP

For analyzing the time complexity of Merge Sort and Bubble Sort, for observation, for analysis machine having Intel’s Core i5 of the First generation to be more precise 520M first gen of i5 and the RAM is 4GB DDR3 is used. During these executions, background tasks are not running at the time of execution for better and accurate results. Visual Studio Code is the choice for the editor and all the analyses taken in the same scenario.

For each algorithm in each language, different numbers of inputs were given to algorithms from 1000 to 10000 inputs. Inputs were given as an array of integers of random numbers from 0 to 1000. That integer passes on to the algorithm and then calculates the time taken to execute the code and sort all the arrays. Note that in calculating time for execution the time to execute and calculate the array of different numbers of random integers are not be included so the only time is just of algorithms performance and nothing else.

JavaScript is the language used in web development mainly as a frontend development but with the help of Node JS, can use it as a server-side back end language that is easier for developers to be a full-stack developer with just using one language. Besides its scope, it is one of the fastest solutions for server-side service which uses a single-threaded mechanism to handle requests to the server. Node JS is the choice to implement merge sort and bubble sort and extract the results for observation.

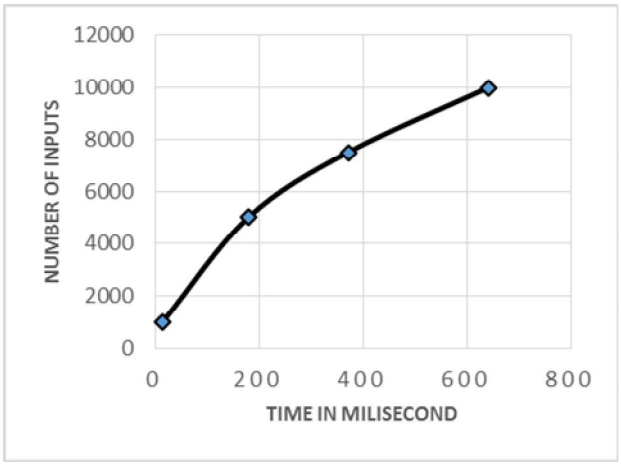


Figure. 1 Time complexity of bubble sort in JavaScript

Figure. 1 represents that the time at 1000 inputs is 13.273 millisecond which is pretty low but as in increase of inputs the time it takes is considerably high which is 639 millisecond.

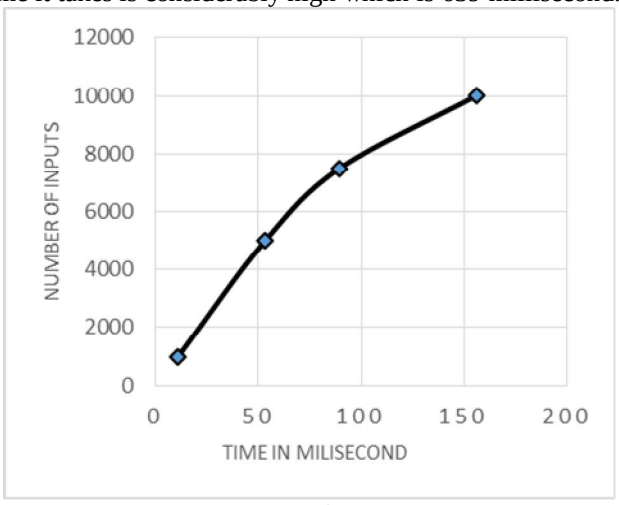
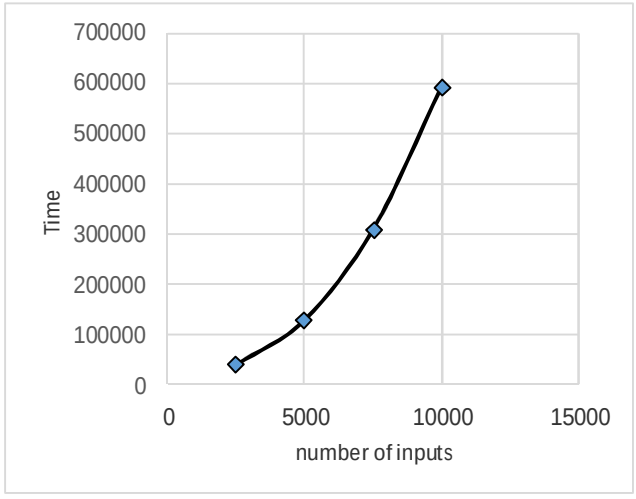


Figure. 2 Time complexity of merge sort in JavaScript

Figure. 2 represents that the graph show the result of merge sort. After comparing results for 1000 inputs, it take less time in bubble sort while in merge sort time is noticeably more, but it all changes as inputs increases merge sort execution and provide results way faster than bubble sort.

C language is best for applications close to the operating system and still retains the speed among newer languages out there. After executing inputs from merge sort and bubble sort



produces these results.

Figure. 3 Time Complexity of Bubble Sort in C Language

Figure. 3 represents that in this graph, time is getting higher and higher with the increase in inputs. That curve graph shows us that a large number of inputs can take a considerably long time to sort it all.

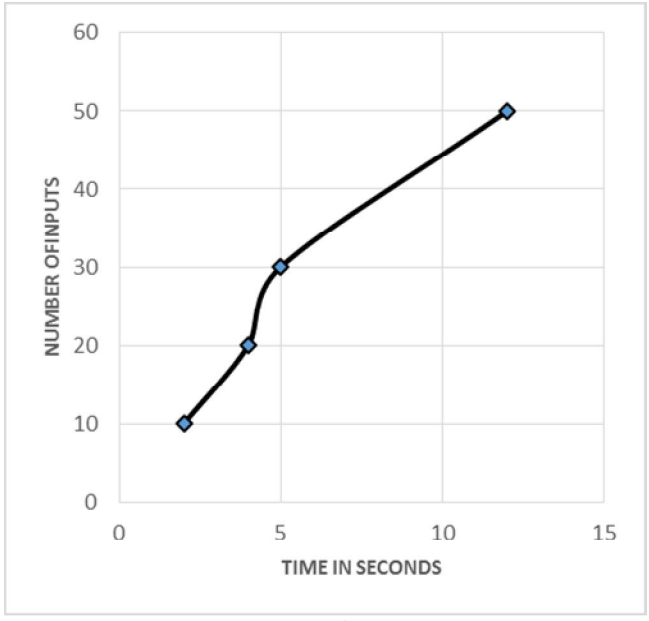


Figure. 4 Time complexity of Merge Sort in C language

Figure. 4 represents that after some inputs time complexity of the merge sort becomes close to linear and the other one does

not which means the merge sort performs better than the bubble sort. In this case number of inputs is not that much as compared to others but the analyzer can get a minor idea from its graph behavior.

In Python, this paper test out the merge sort and bubble sort, and the results are observed as.

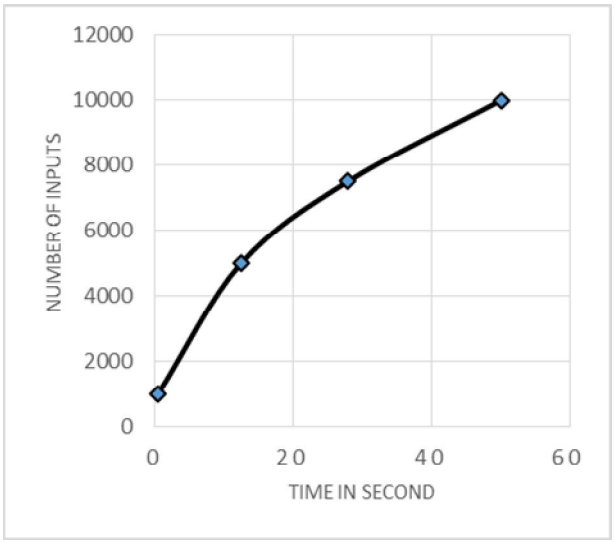


Figure. 5 Time complexity of bubble sort in Python

Figure. 5 represents that the python, when executing bubble sort for 1000 inputs, it takes 0.474 seconds which is pretty good but after increasing the input to 10000 inputs, it takes 50.0819 seconds.

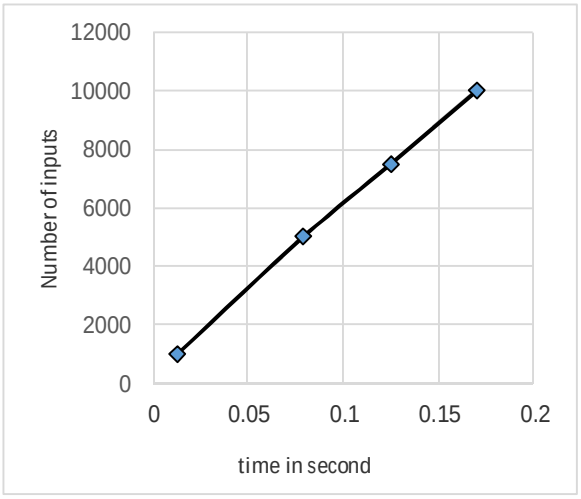


Figure. 6 Time complexity of merge sort in Python

Figure. 6 represent that, the time for 10000 inputs is low as 0.17 seconds while the bubble sorts struggle to sort 10000 elements and take almost 0 seconds which is way lower than others but the bubble sort takes almost no time for 1000 inputs while due to space complexity merge sort is not that good with fewer inputs. Python is great for testing because python can be the tool for almost every field in computer science and a lot of developers prefer python because of its scope. And in PHP also used to tests the algorithms and the results are pretty the same as other languages because of the nature of the algorithms.

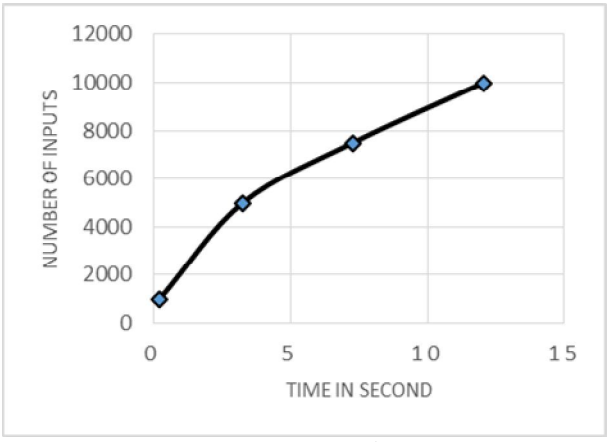


Figure. 7 Time complexity of bubble sort in PHP

Figure. 7 represents that the Bubble sort in PHP takes 0.2099 seconds for 1000 inputs where 10000 inputs, it takes the time of about 12.0506 seconds which increases much more than linear time, PHP did noticeably better than Python.

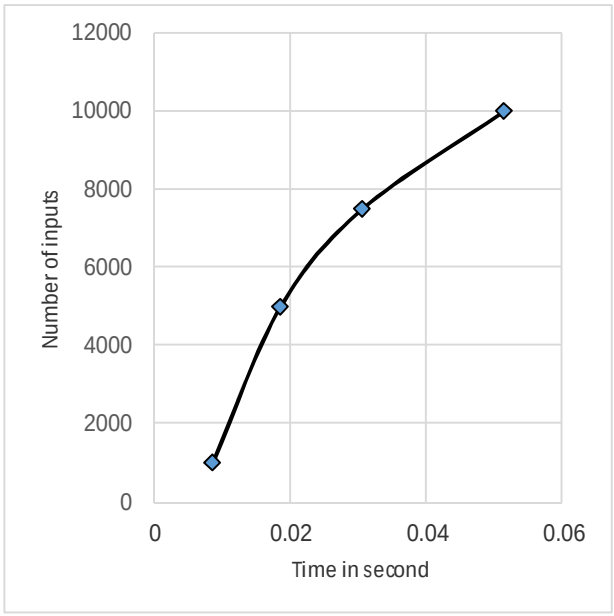


Fig. 8 Time complexity of merge sort in PHP

Fig. 8 represents that the merge sort works better with large inputs, on 1000 inputs it takes 0.00872 seconds and in 10000 inputs it takes 0.051447 seconds which is also better than python. It is one of the oldest server-side languages and this result give the positive image of PHP that can compete with modern languages and worth using for web application.

6. RESULT AND DISCUSSION

After performing these tests, in results, there are some time differences in all languages which means there are some speed differences that are interacting with these results but the idea that python is best with merge sort and JavaScript take the least time with bubble sort. Python is used by many developers and scope and speed in one the reasons for this and merge sort is one the fastest sorting algorithms and with the combination of python and merge sort these results can assure that if you are developing an application then python is a great tool for managing large scale data for the application while bubble sort cannot be neglected because of simple design and quick function for small data. JavaScript can be used for web applications both front end and back end but also used for game development where a developer might need not apply sorting algorithm where small data needs to be sort and for that merge sort is not a better choice so with the combination of JavaScript and bubble sort developer might get the goal with the power of JavaScript and simplicity of bubble sort.

7. CONCLUSION

Concerning languages, the fastest language in this analysis is python. It is best with merge sort with 10000 inputs while JavaScript performs better with bubble sort. Both tools are wide in scope and use in modern applications for computers and robotics as well as web development and game development also that may use some serious lengths of inputs and results may require sooner the better for a better experience of players. But developers always select tools first and then select algorithms according to that so it really depends on the scenario when you select sorting algorithms. In many applications functions, developers might know the expected data to pass on algorithms and some of them are low based on function like sorting student's objects according to name in class or may deal with larger inputs like sort the student's objects within the province so it really depends on that. Concerning Algorithms, the developer should select bubble sort for low expected inputs that may sort faster than others and if you are dealing with a larger number of inputs then consider merge sort for better efficiency and results but you have to deal with its space complexity.

REFERENCES

1. Y. Yang, P. Yu, and Y. Gan. **Experimental study on the five sort algorithms**. in *2011 Second International Conference on Mechanic Automation and Control Engineering*, Inner Mongolia, China, Jul. 2011, pp. 1314–1317. doi: 10.1109/MACE.2011.5987184.
2. J. Hugunin. **Python and Java: The Best of Both Worlds**. *undefined*, 1997. /paper/Python-and-Java%3A-The-Best-of-Both-Worlds-Hugunin/db6651ccd7d876f42e9427989704cc79872f2608 (accessed Feb. 28, 2021).
3. S. Tilkov and S. Vinoski. Node.js: **Using JavaScript to Build High-Performance Network Programs**. *IEEE Internet Comput.*, vol. 14, no. 6, pp. 80–83, Nov. 2010, doi: 10.1109/MIC.2010.145.
4. D. M. Ritchie, B. Labs, and M. Hill. **The Development of the C Language** .*Bell LabsLucent Technol. Murray Hill NJ 07974 USA*, p. 16.
5. C. Xiaofeng. **The use of Data Sorting Cases in C Language Teaching.** *Int. J. Trend Res. Dev.*, vol. 8, p. 2, 2021.
6. S. Jadoon, S. F. Solehria, P. Dr, S. Rehman, and P. H. Jan. **Design and Analysis of Optimized Selection Sort Algorithm**. *International Journal of Electric & Computer Sciences*, vol. 11, no. 1, pp. 16–21, 2011.
7. J. Mundra and B. L. Pal. **Minimizing Execution Time of Bubble Sort Algorithm.** *Int. J. Comput. Sci. Mob. Comput.*, vol. 4, no. 9, pp. 173–181, 2015.
8. M. Woźniak, Z. Marszałek, M. Gabryel, and R. K. Nowicki. **Modified Merge Sort Algorithm for Large Scale Data Sets.** in *Artificial Intelligence and Soft Computing*, Berlin, Heidelberg, 2013, pp. 612–622. doi: 10.1007/978-3-642-38610-7_56.
9. O. Astrachan, "Bubble Sort: **An Archaeological Algorithmic Analysis.** *SIGCSE Bull. Assoc. Comput. Mach. Spec. Interest Group Comput. Sci. Educ.*, Mar. 2003. doi: 10.1145/611892.611918.
10. J. Alnihoud and R. Mansi. **An Enhancement of Major Sorting Algorithms**. *The International Arab Journal of Information Technology*, vol. 7, no. 1, p. 8, 2010.
11. W. Min. **Analysis on Bubble Sort Algorithm Optimization**. in *2010 International Forum on Information Technology and Applications*, Kunming, China, Jul. 2010, pp. 208–211. doi: 10.1109/IFITA.2010.9.
12. S. Qin. **Merge Sort Algorithm.** Dept. of Computer Sciences Florida Institute of Technology Melbourne, FL 3290, pp. 1-4.
13. J. Lobo and S. Kuwelkar. **Performance Analysis of Merge Sort Algorithms.** in *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*, Coimbatore, India, Jul. 2020, pp. 110–115, doi: 10.1109/ICESC48915.2020.9155623.
14. "min2010.pdf." Accessed: Feb. 28, 2021. [Online]. Available: https://dacemirror.sci-hub.do/proceedings-article/0e8c910eb671a76208071bd39ea4a46a/min2010.pdf#view=FitH.
15. S. M. Cheema, N. Sarwar, and F. Yousaf. **Contrastive analysis of bubble & merge sort proposing hybrid approach.** in *2016 Sixth International Conference on Innovative Computing Technology (INTECH)*, Dublin, Ireland, Aug. 2016, pp. 371–375.

doi: 10.1109/INTECH.2016.7845075.

16. N. Faujdar and S. P. Ghrera. **A practical approach of GPU bubble sort with CUDA hardware.** in *2017 7th International Conference on Cloud Computing, Data Science & Engineering - Confluence*, Noida, India, Jan. 2017, pp. 7–12. doi: 10.1109/CONFLUENCE.2017.7943115.
17. W. Min. Analysis on Bubble Sort Algorithm Optimization. in *2010 International Forum on Information Technology and Applications*, Jul. 2010, vol. 1, pp. 208–211. doi: 10.1109/IFITA.2010.9.
18. H. S. Garg, V. Jain, and G. Chaudhary. **The Curious Case of Modified Merge Sort.** in *Proceedings of International Conference on Artificial Intelligence and Applications*, Singapore, 2021, pp. 481–487. doi: 10.1007/978-981-15-4992-2_45.
19. S. Paira, S. Chandra, and S. S.K. Enhanced Merge Sort. **A New Approach to the Merging Process.** *Int. Conf. Adv. Comput. Commun.*, vol. 93, pp. 982–987, 2016.
20. I. Ali, H. Nawaz, I. Khan, A. Maitlo, M. Ameen, and M. Malook. Performance Comparison between Merge and Quick Sort Algorithms in Data Structure. *Int. J. Adv. Comput. Sci. Appl.*, vol. 9, no. 11, 2018. doi: 10.14569/IJACSA.2018.091127.