ISSN 2278-3091
Volume 10, No.3, May - June 2021
International Journal of Advanced Trends in Computer Science and Engineering
Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse1061032021.pdf
https://doi.org/10.30534/ijatcse/2021/1071032021

Implementation of Selection Sort Algorithm in Various Programming Languages

Arisha Naz[1], Haque Nawaz[2], Abdullah Maitlo[3], Syed Muhammad Hassan[4]
[1]BSCS Student, Department of Computer Science, Sindh Madressatul Islam University, Karachi, Sindh, Pakistan, arisha.naz1234@gmail.com
[2]Department of Computer Science, Sindh Madressatul Islam University, Karachi, Sindh, Pakistan, hnlashari@smiu.edu.pk
[3]Department of Computer Science, Shah Abdul Latif University, Khairpur Mirs, Sindh, Pakistan, abdullah.maitlo@salu.edu.pk
[4]Department of Computer Science, Sindh Madressatul Islam University, Karachi, Sindh, Pakistan, m.hassan@smiu.edu.pk

ABSTRACT

Sorting algorithm deals with the arrangement of alphanumeric data in static order. It plays an important role in the field of data science. Selection sort is one of the simplest and efficient algorithms which can be applied for the huge number of elements it works like by giving list of unsorted information, the calculation which breaks into two partitions. One section has all the sorted information and another section has all the staying unsorted information. The calculation rehashes itself, by finding the smallest component inside the rundown of unsorted information and swapping it with the furthest left component, in the end setting everything straight information. This research presents the implementation of selection sort using C/C++, Python, and Rust and measured the time complexity. After experiment, we have collected the results in terms of running time, and analyzed the outcomes. It was observed that python language has very small amount of line of code, and it also consumes less storage and fast running time then other two languages.

Key words: Selection Sort, Time Complexity, Algorithm Implementation, C/C++, Python, Rust, Comparison.

1. INTRODUCTION

Selection sort is an algorithm used for sorting easily it is very simple, it divide the list into two parts and do sort the problem in a way that right part contain unsorted numbers and left part contain sorted number similarly it revolves over files in the cluster; for each file, selection sort calls minimum record and line of work. In the event that the length of the exhibit is n, there are n files in the cluster. Hence each outcome of the body of the program runs code of two lines. It may feel that two n, number lines of program are debugged by selection sort. However, a few lines of code are executed on recollected record in minimum [1].

At the point when given selection of unsorted information, the calculation was partition the rundown into two sections: one section that has all the sorted information and another part that has all the staying unsorted information. At the point when the calculation initial beginnings, there is no information in the initial segment neither in sorted order at this point and all the unsorted information is in the subsequent part. The calculation at that point begins to locate the smallest element in the unsorted information and trade it with the furthest is to be the left component of the run down. The part with the sorted information is then the furthest is the left component and the part with the unsorted information is to be the staying unsorted components. The calculation rehashes itself, by finding the smallest component inside the rundown of unsorted information and trading it with the furthest left component, in the end setting everything straight information [2].

1.1 Selection Sort Algorithm Working

- Find the lowest number.
- Move it with the initial value.
- Find the second lowest number.
- Move it with the second value.
- Find the third-lowest number
- Move it with the third value.
- Repeat finding the next lowest value, and moving it into the correct place until the array is sorted

1.2 Selection Sort Complexity

Selection sort isn't hard to investigate contrasted with other sorting techniques since none of the replacement relies upon the information in the cluster. Choosing the base requires checking n components and afterward exchanges it in to the primary position. Finding the following most reduced components requires examining the remaining n-1 is the components, etc. Hence, the absolute number of the correlations is regarding number of correlations. Every one of these steps requires one swap for n-1 is components [3].

2. LITERATURE REVIEW

Initially a part of this moves surround the advantages and disadvantages of the language platforms which can be utilized to display program. It would complain on to the simplest language platform as long as, because at the level of once own choice. Starting an aspect of the moves to the focal points and drawbacks of the language stages which was be used to show program. Clearly, it would criticize on to the most straightforward language stage as long as it can imagine, because that is to all at the degree of once own choice of a self-want. In any case, in the event that it investigates the vast majority of the people groups – the novice software engineers – it was endeavor to discover a language that offers best help to the larger part of them and meets their requirements. So, in what capacity would it be a good idea for it to choose the learning platform for the initial programming course? Common answer main point is that restart and can be used to selected the common central and present language usable to access, even inside new developer's case, some other problem can be shown. It can be apply to the language stage that must be simple together all enough one that could become known with the needs of the language, similar to factors, access and rule over the techniques of stream or Input/output activities not even much of an initial that comes with the formal syntax and documentation [4].

Moreover, on the other side, simplicity shouldn't be a constraint once it advances to the more complicated aspects. Another significant factor ought to be accessibility. It would be best to use it with the point where circumstances useable to higher education, something be said about those people that need to or necessities to-learn at home. It's basic to be set up to give some common, publicly accessible list of options that are practical, and in best case potentially free from charges. Furthermore, in the event that it considers this from the perspective of the instructional exercise in organizations, it most likely wouldn't damage to have menu that don't accompany some sort of license expense. In the event that the menu is accessible for a few working frameworks, the stage wouldn't cause issues, which may even be being a strong component. End point that includes to the choices is that the takeover credit. Improvements like, utilization circumstances that stages to customized language which shows the essential to the scholars, yet these conditions urge to became old when individuals improve towards difficult levels. Credibility factor in addition reflects to the initial language stages, potential which can have for being complete aspects, at any rate tragically lack the assistance from the marketing perspective [5]. So all around the preliminary social issue of language even more precisely, the main criteria for inclusion was that the language expected to fulfill the above fundamentals and was a neither it more likely and commonly utilized language stage like Java, that has recently been proposed as an essential language stage inside the few prior examinations like Python. Which language stages did it choose other than it? The main common gathering of comparison, C, C++ and Java might be a for the most part great gathering of 6 examples of each of them because they have little special dedications to start with. For instance, that the C is conventional choice, C++ for those with that desire to start object with-direction. The Java is moreover selected because it's cutting edge and adaptable language, yet in addition because it's generally the language individuals see as a replacement standard. These three it was likely significant because they it were our most appreciated non-Python choices. C was utilized before in our initial courses when C++ is likewise normal and it is-built up language inside our courses. Java is at that of point being used on some advanced programming courses and along these lines expected choice. On the other side Python is a deciphered language, however the before referenced dialects are compiler-based, so direct comparison between them was off the mark. In this Rex and Perl [6]. It would be known inside the summary to deftly additional reference focuses inside the case of deciphered dialects against compiler-based dialects generally. Within the writing review it even that Matlab found there was a discussion on to the sum of earlier work to compare though it wasn't so much an childish language stage in customary sense [7]. Different algorithms studied and different platforms explored [9-15], and come up with idea to implement the selection sort algorithm in various programming languages.

3. RESEARCH METHODOLOGY

In this comparative analysis, selection sort algorithm is evaluated in C/C++, Python, Rust languages to distinguish the complexity of algorithm using software's such as: Dev C++, turbo C, Python and online Rust programming site to implement code of selection sort as all the language have different number of line of codes and compilation process, find the best language for this algorithm and highlighted the best platform from the different implemented code and outcomes:

3.1 Selection Sort – Algorithm Complexity

In Selection Sort Time Complexity for all the cases, worse case, best case or average case remain same as O(n^2) [8].

Table 1: Selection Sort Case Complexity

Selection	Comparison	Swap
Best Case	O(n^2)	O(n)
Average Case	O(n^2)	O(n)
Worst Case	O(n^2)	O(n)

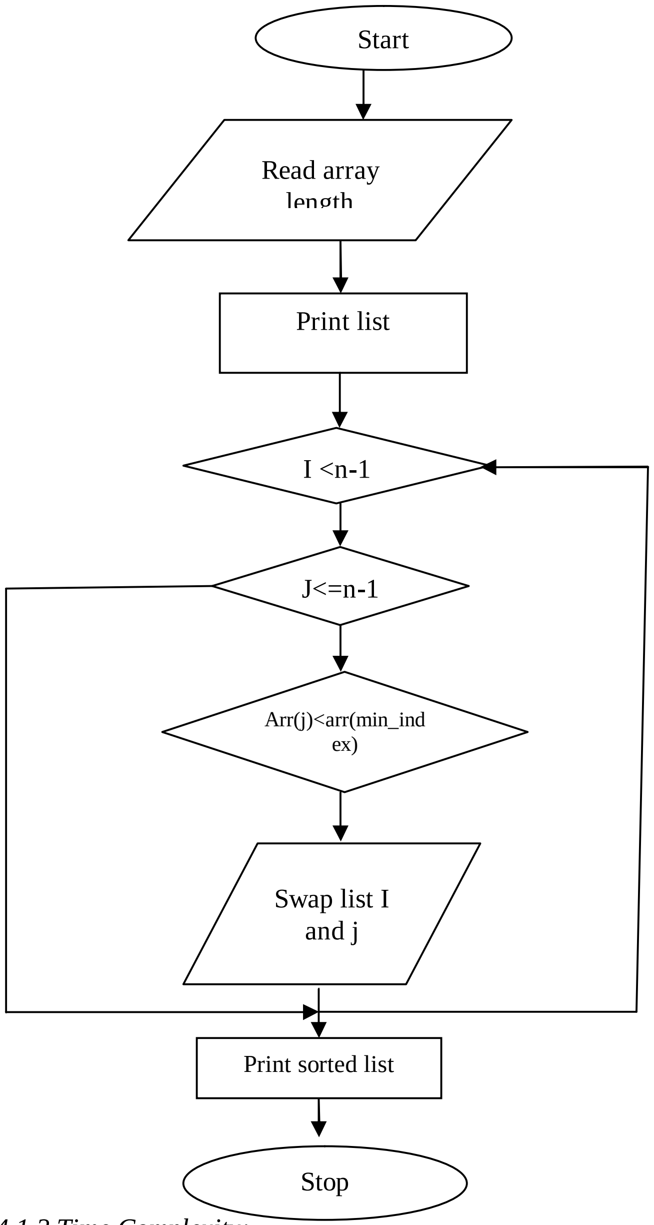
4. IMPLEMENTATION AND RESULT ANALYSIS

Selection sort move the numbers within the array. Each number change and swap position, it finds the index of the smallest number within the subset begin at that place. Number changes its position with the smallest number. Implement selection sort with different languages with multiple data size and analyze the results with respect to complexity.

4.1 C/C++Algorithm:

1. Discover the minimal array to its size
2. Get index of minimal data
3. Setting in accurate position
4. Create an array with great variety of element
5. Type the sorted array

4.1.1 Flowchart of selection sort in C/C++



4.1.2 Time Complexity:

Selection sort calls for repeated over n over changes mess with every start to finish itself. The element move through all of the numbers within place of the array till end of the string and it discover the smallest detail index the work of some other for move that's messing numbers in the outer for sorting.

So, let given a string of N to putting an array, the choice a type set of discipline has the subsequent time and complexity values. The time complexity of O (n^2) is in particular due to the usage for circular move. The best kind approach by no means takes more than O(n) swaps and is useful whilst the recollection write operation proves to be expensive[4].

4.1.3 Running time

For calculating running time we run code for three times by increasing the number of element in array and got different amount of time but not more than a minute it takes time in seconds to execute the output but time depend on the no of element if the element extend then running time also exceed.

Table 2: Data Set Size and Time of Execution

S. no	No of Elements	Time in seconds
1	5	9.916
2	10	23.3
3	15	25.1

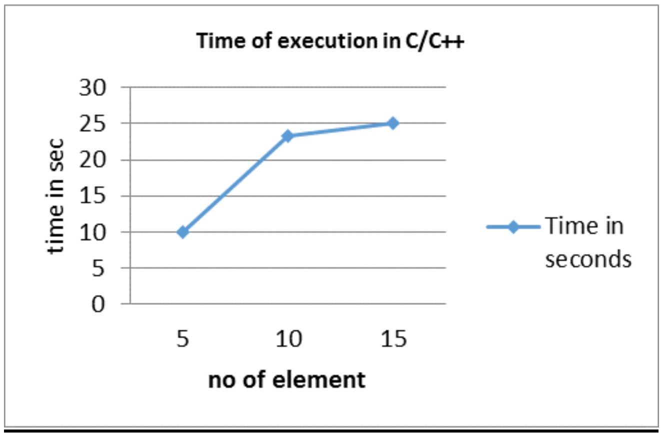


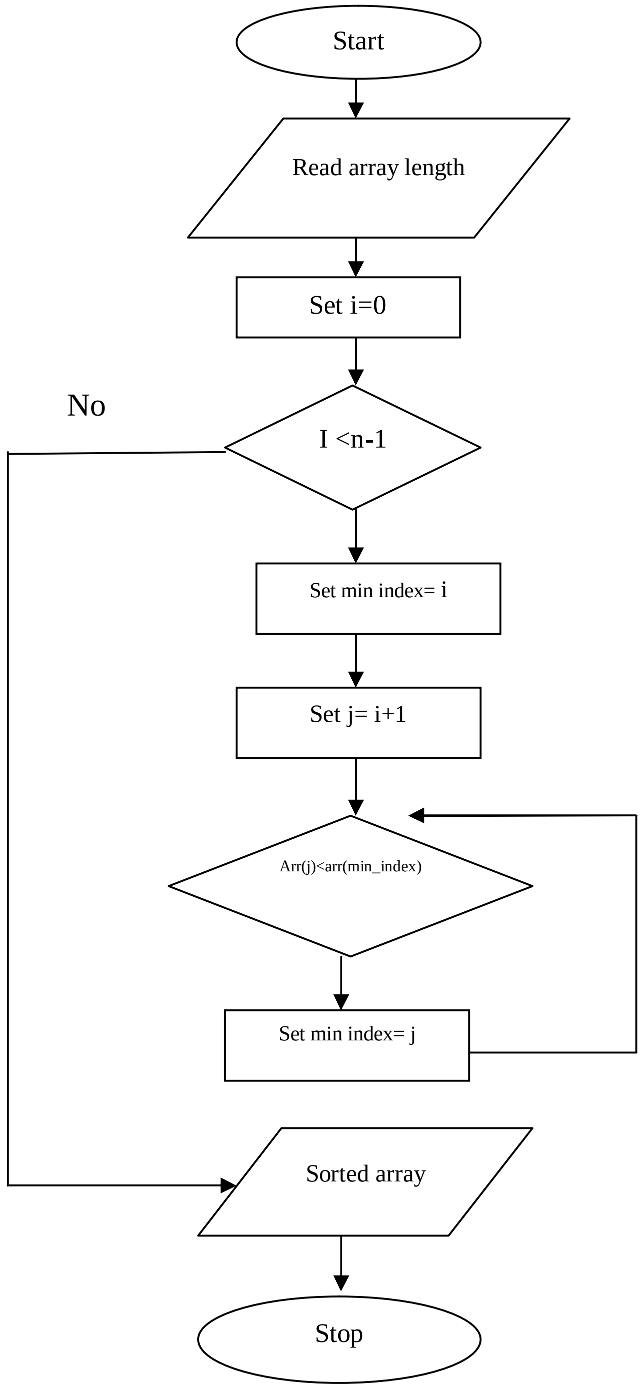
Figure 1: Time of execution in C/C++

4.2 Python Algorithm:

1. It shows what number of things it re arranged
2. To locate the base estimation of the unsorted portion
3. It initially accept that the main component is the most reduced
4. min_index = I
5. It at that point use j to round through the rest of the components

6. Update the min_index if the component at j is smaller than it.
7. In the list of finding the most reduced element of the unsorted array, exchange with the principal unsorted element.
8. END

4.2.1 Flowchart of selection sort in python



4.2.2 Time Complexity:

The sort complexity is utilized to communicate the quantity of execution times it takes to sort the rundown. The usage has two times.

The external recursion which picks the qualities individually from the rundown is executed n times where n is the all-out number of qualities in the rundown. The inward circle, which looks at the incentive from the external recursion with the remainder of the qualities, is likewise executed n times where n is the absolute number of components in the rundown. Along these lines, the extent of executions is (n * n), which can likewise be communicated as O(n^2).

The selection sort has three classes of complicated nature to be specific;

Worst case – this is the place the rundown gave is in dropping request. The calculation plays out the most extreme number of executions which is communicated as [Big-O] O (n^2)

Best case – this happens when the given list has arranged. The calculation plays out the base number of executions which is communicated as [Big-Omega] Ω (n^2)

Average case – this happens when the rundown is in arbitrary request. The normal complicated nature is communicated as [Big-theta] ⊖(n^2) [9]

4.2.3 Running time

For calculating running time run code for three times by increasing the number of element in array and got different amount of time but not more than a minute it takes seconds to execute the output but time depends on the no of element if the element extend then running time also exceed but it’s very faster to execute than c language.

Table 3: Data Set Size and Time of Execution

S. no	No of Elements	Time in seconds
1	5	1.39
2	10	2.5
3	15	3

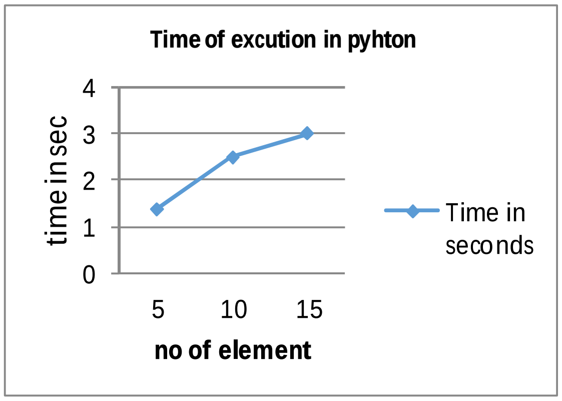
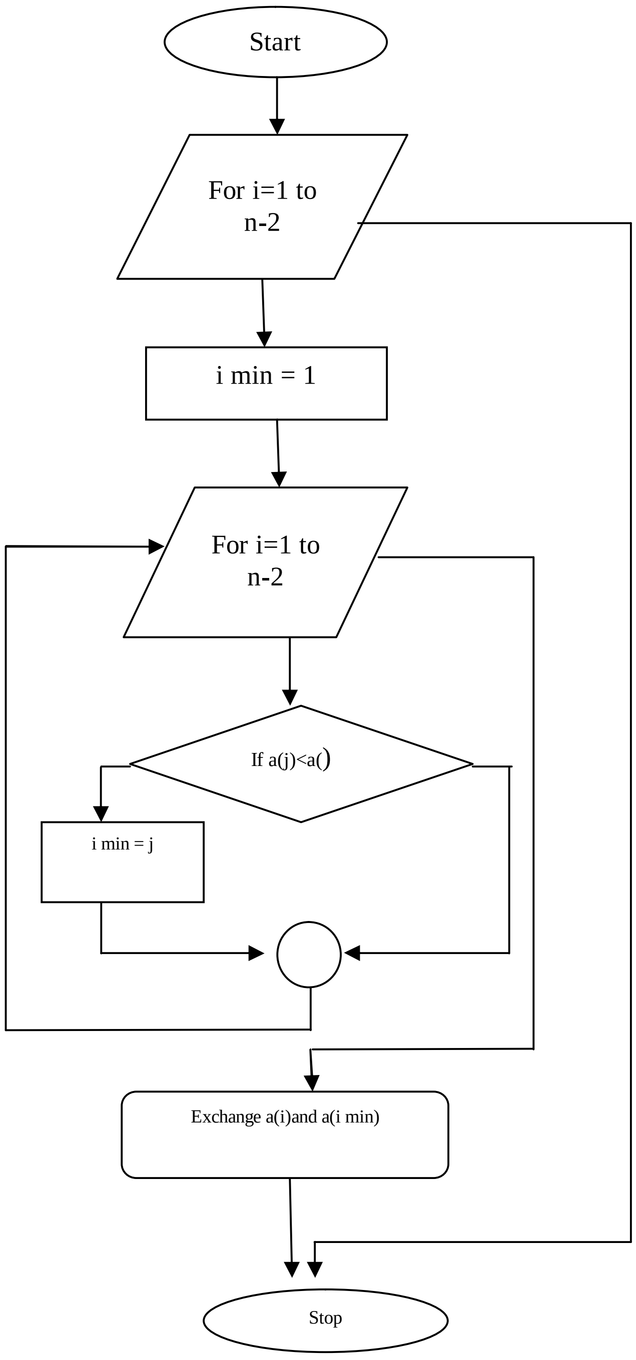


Figure 2: Time of execution in python

4.3 Rust Algorithm:

1. The set of rules perspectives the elements.
2. A looked after element, and an unsorted element.
3. At the beginning, it viewed after element is empty, and the whole listing was the unsorted element.
4. One at a time the smallest detail within array the unsorted element is moved to the last looked element.

4.3.1 Flow chart of selection sort in Rust



4.3.2 Time Complexity:

Selection sort calls for recursion over n over changes mess with every different to finish itself. One for movement move through all of the numbers within place of the array till end of the string and it discover the smallest detail index of next element the exchange the messing number in the array [9].

The selection sort has three classes of complicated nature to be specific;

Worst case – this is the place the rundown gave is in dropping request. The calculation plays out the most extreme number of executions which is communicated as [Big-O] O (n^2)

Best case – this happens when the given list is arranged. The calculation plays out the base number of executions which is communicated as [Big-Omega] Ω (n^2)

Average case – this happens when the rundown is in arbitrary request. The normal multifaceted nature is communicated as [Big-theta] ⊖(n^2) [10].

4.3.3 Running time

For calculating running time run code for three times by increasing the number of element in array and got different amount of time but not more than a minute it takes seconds to execute the output but time depend on the number of element if the element extend then running time also exceed.

Table 2: Data Set Size and Time of Execution

S. No	No of elements	Time in seconds
1	5	10
2	10	16.56
3	15	20.33

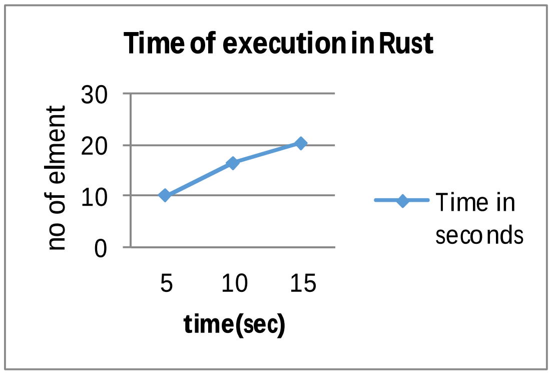


Figure 3: Time of execution in Rust

5. RESULTS AND DISCUSSION

As result, it gains the outcome that they have little bit of difference but not difficult selection is easiest if list given number of array is to be mention to be sorted and it take very small amount of time to be execute and according to this implementation on different language python consume less time than any other two languages and it can be shown in the below through the table and graph.

Table 3: Comparison of data size

S. no	No of Element	Time in second		
		C/C++	Python	Rust
1	5	9.9	1.39	10
2	10	23.3	2.5	16.56
3	15	25.1	3	20.33

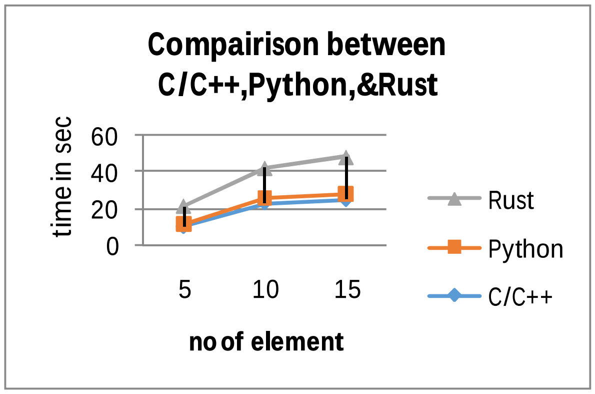


Figure 4: Comparison of time of execution between C/C++, Python and Rust

In terms of memory storage python consume less storage as compare to other two languages as shown in the table below the reason why it is take less time to execute and less memory hold its already define array in source file but other two define array size that hold more memory to execute the program.

6. CONCLUSION

In this study, we have explored the performance of selection sort algorithm in terms of complexity by using different platforms, such as C/C++, Python, and Rust language. We have collected the results in terms of running time, and analyzed the outcomes. It was observed that python language has very small amount of line of code, and it also consumes less storage and fast running time then other two languages. It has concluded that selection sort has less complexity using python platform as compared to c/c++ and rust platform.

REFERENCES

[1] J. B. Hayfron-acquah and P. D, "**Improved Selection Sort Algorithm**," *International Journal of Computer Applications*, vol. 110, no. 5, pp. 29–33, 2015.

[2] R. N. Vilchez, "**Bidirectional Enhanced Selection Sort Algorithm Technique**," *International Journal of Applied and Physical Sciences*, vol. 5, no. 1, pp. 28–35, 2019.

[3] T. M. Fagbola and S. C. Thakur, "**Investigating the Effect of Implementation Languages and Large Problem Sizes on the Tractability and Efficiency of Sorting Algorithms**,"*International Journal of Engineering Research and Technology*, vol. 12, no. 2, p. 196-203, 2019.

[4] H. Fangohr, "**A Comparison of C, MATLAB, and Python as Teaching Languages in Engineering**," in *Computational Science - ICCS 2004*, Berlin, Heidelberg, 2004, pp. 1210–1217. doi: 10.1007/978-3-540-25944-2_157.

[5] D. E. Knuth, **The art of computer programming**, *volume 3*. *Addison Wesley Longman* Publishing Co., Inc.350 Bridge Pkwy suite 208 Redwood City, CAUnited States, 1998. Accessed: Jun. 15, 2021. [Online]. Available: https://dl.acm.org/doi/book/10.5555/280635

[6] M. Stein, and A. Geyer-Schulz), "**A Comparison of Five Programming Languages in a Graph Clustering Scenario**," *Journal of Universal Computer Science*, vol. 19, no. 3, pp. 428–456, 2013.

[7] D. K. Kavitha, "**Comparative Study of Various Enhanced Selection Sort Algorithm**," *International Journal of Engineering Research*, vol. 3, no. 30, pp. 1–3, 2015.

[8] F. G. Furat, "**A Comparative Study of Selection Sort and Insertion Sort Algorithms**," *International Research Journal of Engineering and Technology (IRJET)*, vol. 03, no. 12, pp. 326–330, 2016.

[9] D. Rajagopal and K. Thilakavalli, "**Different Sorting Algorithm's Comparison based Upon the Time Complexity**," *IJUNESST*, vol. 9, no. 8, pp. 287–296, Aug. 2016, doi: 10.14257/ijunesst.2016.9.8.24.

[10] M. Abdulla, "**Selection Sort with Improved Asymptotic Time Bounds**," *The International Journal Of Engineering And Science (IJES)*, vol. 5, no. 5, pp. 125–130, 2016.

[11] I. Ali, H. Nawaz, I. Khan, A. Maitlo, M. Ameen, and M. Malook, "**Performance Comparison between Merge and Quick Sort Algorithms in Data Structure**," *International Journal of Advanced Computer Science and Applications*, vol. 9, no. 11, 2018. doi: 10.14569/IJACSA.2018.091127.

[12] S. M. Aqib, H. Nawaz, and S. M. Butt, "**Analysis of Merge Sort and Bubble Sort in Python, PHP, JavaScript, and C language**," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 10, no. 2, pp. 680–686, Apr. 2021. doi: 10.30534/ijatcse/2021/311022021.

[13] M. A. Hingoro and H. Nawaz, “**A Comparative Analysis of Search Engine Ranking Algorithms**,” *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 10, no. 2, pp. 1247–1252, Apr. 2021, doi: 10.30534/ijatcse/2021/1081022021.

[14] F. A. Agha and H. Nawaz, “**Comparison of Bubble and Insertion Sort in Rust and Python Language**,” *International Journal of Advanced Trends in Computer Science and Engineerin*, vol. 10, no. 2, pp. 1020–1025, Apr. 2021, doi: 10.30534/ijatcse/2021/761022021.

[15] H. Ali and H. Nawaz, “**Performance Analysis of Heap Sort and Insertion Sort Algorithm**,” *International Journal of Emerging Technologies in Engineering Research* (*IJETER*), vol. 9, no. 5, pp. 580–586, May 2021, doi10.30534/ijeter/2021/08952021.