Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 2, Desember 2025
e-ISSN 2685-998X
DOI 10.30865/json.v7i2.9306

Exchange Sort and Selection Sort Algorithms: Comparison and Theoretical Analysis

Tri Dharma Putra[1], Rakhmat Purnomo[2*]

Faculty of Computer Science ,Department of Informatics,Universitas Bhayangkara Jakarta Raya,Jakarta Selatan,Indonesia
Email: [1]tri.dharma.putra@dsn.ubharajaya.ac.id, [2,*]rakhmat.purnomo@dsn.ubharajaya.ac.id
Email Penulis Korespondensi: rakhmat.purnomo@dsn.ubharajaya.ac.id[*]
Submitted: **04/11/2025**; Accepted: **19/12/2025**; Published: **31/12/2025**

Abstract– The purpose of this study is to analyze the comparison of sorting algorithms. Searching algorithm is a main issue in computer science. Many algorithms have been presented by experts. The main issue is the efficiency of these algorithms. Sorting a large number of data will create time issues. Time needed to sort the data is very crucial in this context. Many sorting algorithms are in the field. Two of them are selection sort and exchange sort algorithms. These two are a popular algorithm in data structure. Exchange sort algorithm is very similar to Bubble Sort. Many say Bubble Sort is the same as Exchange Sort. The difference is in how it compares elements. Exchange sort compares one element to the other elements in the array and swaps elements if necessary. Selection sort algorithm is a combination of sorting and searching. For each process, comparison and theoretical analysis is given. The unsorted elements with the smallest or largest values are searched for and exchanged to their appropriate positions within the array. In the comparison and theoretical analysis 1, it was conducted by hand and we got the numbers to be descending from 11, 23, 30, 39, 41, 51 to be 51, 41, 39, 30, 23, 11. For the comparison and theoretical analysis 2, we got the numbers to be descending from 1, 22, 10, 9, 42, 26 to be 42, 26, 22, 10, 9, 1. The most swap happened in exchange sort of comparison and theoretical analysis 1 with 14 swaps, in the second exchange sort it is only 11 swaps. The research gap is about stages of processes algorithm with methodology systematic literature review. The stages of systematic literature review is presented which is planning, conducting literature search, and reporting. PRISMA flow chart is also presented. a visual representation of the article selection process. This diagram illustrates four main stages: Identification, Screening, Eligibility, Inclusion.

Keywords: algorithm; comparison; exchange sort; selection sort; swap; theoretical analysis.

1. INTRODUCTION

Searching algorithm is a main issue in computer science. Many algorithms have been presented by experts. The main issue is the efficiency of these algorithms [1]-[3]. Sorting a large number of data will create time issues. Time needed to sort the data is very crucial in this contexts [4]-[6]. Many sorting algorithms are in the field. Two of them are selection sort and exchange sort algorithms. These two are a known algorithms in data structure.

Exchange sort algorithm is very similar to Bubble Sort. Many say Bubble Sort is the same as Exchange Sort. The difference is in how it compares elements. Exchange sort compares one element to the other elements in the array and swaps elements if necessary [7] . This means that one element always serves as the centre (pivot). Bubble sort, on the other hand, compares the first/last element with the previous/following element, then that element becomes the centre (pivot) for comparison with the previous/following element, and so on [8],[9].

Selection sort algorithm is a combination of sorting and searching. For each process, the unsorted elements with the smallest or largest values are searched for and exchanged to their appropriate positions within the array [10],[11]. For example, in the first round, the data with the smallest value is searched for and placed at the smallest index (data[0]). In the second round, the second-smallest data is searched for and placed at the second index (data[1]). During the process, comparisons and changes are only made at the comparison index; the physical data exchange occurs at the end of the process.

Here are several related worked done by some experts. Sing et al, gave analysis about various sorting algorithms. The two of them were selection sort and exchange sort algorithms. It gave conclusions about efficiency of both algorithms [1]. R. Purnomo and Putra, gave analysis about theoretical analysis of standard selection sort algorithm. In this journal they only gave analysis about one sorting algorithm. The analysis is similar with this journal. Y. Chauhan and Duggal, gave analysis about the efficiency of these two algorithms. Their perspective was from the time point of view. Which one is the fastest[12] .

By analysing and comparing these two algorithms, we get a clear picture of how these two algorithms run theoretically. Two comparison and theoretical analyses are given with comparison between exchange sort and selection sort. Given the clear analysis of it. All of the numbers is sorted in descending order.

The objective of this research is to compare two different algorithms by analysing step by step processes in each algorithm. In this step by step processes, theoretical analysis is given with clear explanations. The gap analysis is about comparison between these two algorithms. How many swaps, how many index changed. Since in the references, there was no comparisons before in this context. Analysis is given with clear step by step tables and flowcharts. In this research we are doing systematic literature review to explain the two algorithms in details, so that audience can understand it clearly. The research gap is about stages of processes algorithm with methodology systematic literature review.

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 2, Desember 2025
e-ISSN 2685-998X
DOI 10.30865/json.v7i2.9306

Analysis is given with two case studies, each with step by step processes. Random numbers are picked to explain the case studies thoroughly. With table to explain the details of processes. By these tables, the explanation hoped to be cleared. Each step with each tabel. The stages of systematic literature review is presented which is planning, conducting literature search, and reporting. PRISMA flow chart is also presented. a visual representation of the article selection process. This diagram illustrates four main stages: Identification, Screening, Eligibility, Inclusion.

2. RESEARCH METHODS

2.1 Research Stages

The stages of a Systematic Literature Review (SLR) include planning (formulating a research question and protocol), implementation (searching, title/abstract/full-text selection, data extraction and quality assessment), and reporting (results synthesis, discussion, and visualizations such as PRISMA diagrams). The aim of conducting an SLR is to systematically identify, evaluate, and synthesize the literature to answer a specific research question. The following are the detailed stages in conducting an SLR:

a. Planning

Formulate the Research Question: Define a specific and focused question (often using the PICO/PICOC format). Create a Review Protocol: A detailed plan including inclusion and exclusion criteria (year, language, study type), search strategy (keywords, databases: Scopus, WOS, Google Scholar), and data analysis methods.

b. Conducting Literature Search

Conduct a systematic search in academic databases using predetermined keywords. Study Selection (Screening): Stage 1 (Title & Abstract): Screen based on initial relevance. Stage 2 (Full Text): Review full articles to ensure compliance with the criteria. Data Extraction: Extract key information from selected studies (methods, results, etc.) using an extraction table. Quality Assessment: Evaluate the quality and potential bias of selected studies.

c. Reporting

Synthesis and Analysis: Combine findings from multiple studies, either qualitatively (narrative) or quantitatively (meta-analysis). Report Writing: Compile a report that includes an introduction, methodology, results (with PRISMA Flowchart visualization), discussion, and conclusion. Primary Goal: Produce a comprehensive evidence synthesis, identify research gaps, and establish future research directions [13].

2.2 PRISMA Flow Diagram

The most well-known part of PRISMA is the Flow Diagram, a visual representation of the article selection process. This diagram illustrates four main stages:

a. Identification: The total number of articles found from various databases (e.g., Scopus, IEEE Xplore, ScienceDirect).
b. Screening: Articles that have undergone a process of de-duplication and filtering based on title and abstract.
c. Eligibility: Articles that have been read in full and assessed for their suitability to the research criteria.
d. Inclusion: The final articles used in the analysis.

This diagram provides transparency into the literature selection process and allows readers to assess the validity of the research methodology. Please take a look at figure 1. below:

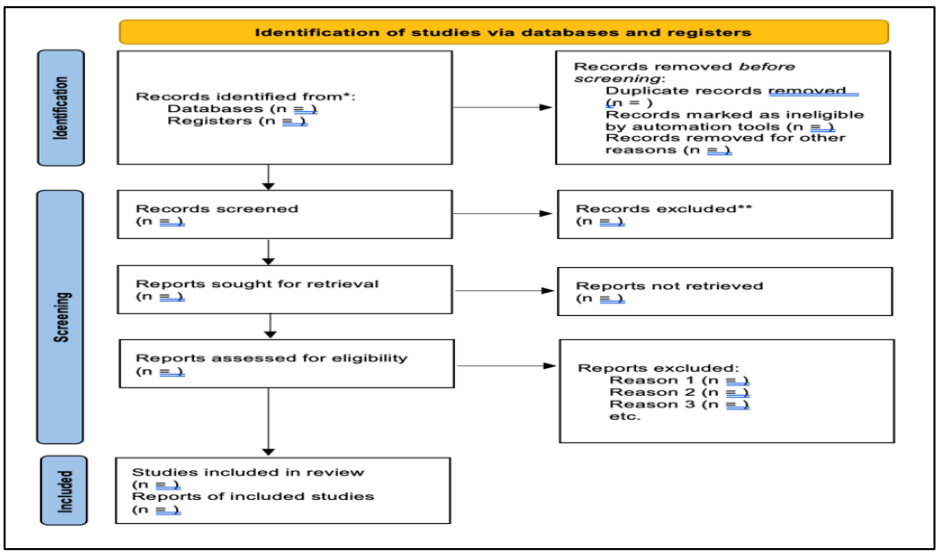

Figure 1. PRISMA Flowchart [14]

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 2, Desember 2025
e-ISSN 2685-998X
DOI 10.30865/json.v7i2.9306

The stages above, are implemented in this analysis. Clear explanation is presented. Each step of SLR methodology combined with table analysis are given. Analysis of systematic literature review is specific for both algorithms. Exchange sort and selection sort. Implementation of PRISMA combined with planning, conducting literature research, and reporting aligned with the steps of process's algorithms. Specific for this methodology for the two algorithms.

2.3 Exchange Sort

This diagram illustrates the conditional constructions, the loops and other flow control, including an algorithm of three elements. In this type of diagram, the flowchart symbolize the logical flow (control flow), the rectangles are process to be carried out, the parallel diagram indicating the entries/sorties (for example, a list of four entries for the user), the rectangles indicate the following specific procedures (instruction blocks) and the conditional constructions (branching points) [12][6]. Note that this algorithm implies two loops on the size of the list (blue and orange), which means that the size of the calculation is the same as the size of the entry (there is a list of N elements). This part can be divided into two parts to adjust the condition of the internal loop to « », and the largest elements must be maintained first in complete position in the final position. Please take a look at figure 2. below.

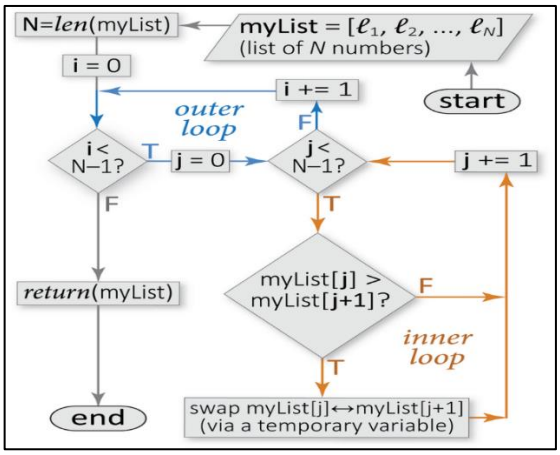

Figure 2. Flowchart of Exchange Sort [15]

2.3.1 Steps of Exchange Sort

The Exchange Sort algorithm works by comparing pairs of adjacent data elements and swapping them if they are in the wrong order. Then, repeating this process repeatedly until the entire data is sorted without further swapping, much like bubbles rising to the surface. The steps are: (1) Start with the first element, (2) Compare the current element with the next element, (3) Swap if the order is wrong, (4) Repeat this process for the entire list until no more swaps occur.

Here are the detailed steps:

a. Initialization: Assume the data is already sorted at the beginning (even if it isn't).
b. Iteration (Pass): Loop through the entire data (list/array).
c. Pair Comparison: Compare the first element with the second element. If the first element is greater than the second element (for ascending order), swap their positions. Continue comparing the second element with the third element, and so on until the last element is complete.
d. Iteration: Repeat steps 2 and 3. Each time the iteration completes, the largest element will "bubble" to the back of the list.
e. Termination: Continue the loop until no more swaps occur after a complete loop. This indicates that the data is completely sorted.

2.3.2 Pseudo Code of Exchange Sort

This is the pseudo code of exchange sort:

```
procedure exchange_sort
data ascending
i,j,N,temp : integer
Algorithm:
    for i <-- 1 to (N-1) do
      for j <-- N downto (i+1) do
        if (data(j) < data(j-1))
        then
          temp      <-- data(j)
          data(j)   <-- data(j-1)
          data(j-1) <-- temp
        endif
```

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 2, Desember 2025
e-ISSN 2685-998X
DOI 10.30865/json.v7i2.9306


```
    endfor
  endfor
endprocedure
```

2.4 Selection Sort

The selection sorting flowchart illustrates the steps involved in implementing selection sort algorithms. This classic flowchart allows you to iterate over the elements of an array, identify the minimum number of elements, and check the array elements for the various sort options[16] -[19]. The basic diagram began with an iteration from 0 to n-2, depending on the elements present in the array. In this case, the variable i(min) is initialized to i, which means that the index has a minimum element. The versions presented here are introduced from i+1 to n-1. This brings us closer to the element at index j with the current minimum element at index i(min). If a(j) is small so that a(i(min)), the order i(min) is created for j, which a new minimum element is created. When can see the flow diagram with the storage options for a(i) and a(i(min)). It is recommended to ensure that the minimum element is placed in the product generated by the array as defined [20]. The detailed diagram shows all the elements in your device, sorted. This flow diagram provides a visual overview of the open steps indicated by the use of a selection algorithm. Please take a look at figure 3. below.

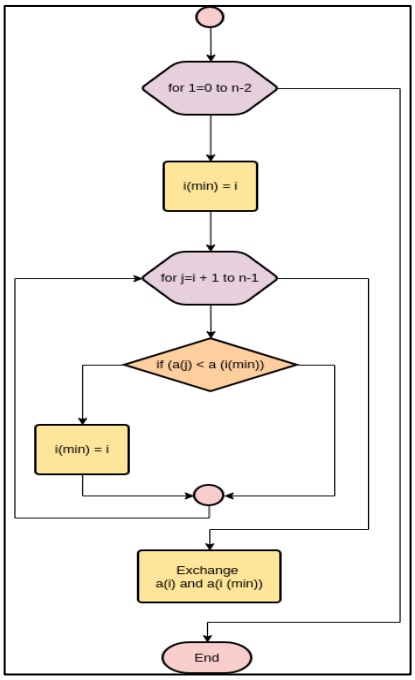

Figure 3. Selection Sort Flowchart [21]

2.4.1 Steps of Selection Sort

The steps of the Selection Sort algorithm are:

Find the smallest element in the entire unsorted array. Swap that smallest element with the first element of the unsorted portion, and then, repeat this process, shifting the boundary of the sorted portion one element to the right, until the entire array is sorted from smallest to largest.

Detailed Steps:

a. Initialization: Treat the entire array as an unsorted portion.
b. First Iteration (Finding the Smallest Element): Set the first element (index 0) as the temporary smallest element.
c. Scan the remaining elements in the array (from index 1 to the end) for a smaller element.If a smaller element is found, update the temporary smallest element.
d. Swap: After the entire unsorted portion has been scanned, swap the found smallest element with the first element of the unsorted portion.
e. Shift Boundary: Move the boundary between the sorted and unsorted portions one position to the right.
f. Iteration: Repeat steps 2, 3, and 4 for subsequent elements (starting with the second unsorted element, then the third, and so on) until all elements are in their correct positions and the array is sorted.

2.4.2 Pseudo Code of Selection Sort

This is the pseudo code of selection sort:

```
procedure select_sort
data ascending
i,j,N,min,temp : integer
Algorithm:
```

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 2, Desember 2025
e-ISSN 2685-998X
DOI 10.30865/json.v7i2.9306


```
    for i <-- 1 to (N-1) do
      min <-- i
       for j <-- (i+1) to N do
         if (data(j) < data(min))
         then
           min <-- j
         endif
       endfor
       temp     <-- data(i)
       data(i)  <-- data(min)
       data(min) <-- temp
    endfor
endprocedure
```

3. RESULTS AND DISCUSSION

3.1. Comparison and Theoretical Analysis

To analyse and compare the two algorithms, we conduct analysis as per below: Let's say for comparison and analysis 1, we have numbers like these:

11 23 30 39 41 51

These numbers will be sorted descending. We will conduct exchange sort and selection sort algorithm in descending order. Analysis will be given in Tables and boxes.

3.1.1. Comparison and Theoretical Analysis 1

11 23 30 39 41 51

a. Exchange Sort Analysis 1

This is the first process, please take a look at Table 1. below.

Table 1. 1st process

pivot					
index[0]	index[1]	index[2]	index[3]	index[4]	index[5]
11	23	30	39	41	51
11	23	30	39	41	51
30	23	11	39	41	51
39	23	11	30	41	51
41	23	11	30	39	51
51	23	11	30	39	41

This is the second process. Please take a look at Table 2. below.

Table 2. 2nd process

	pivot				
index[0]	index[1]	index[2]	index[3]	index[4]	index[5]
51	**23**	11	30	39	41
51	**23**	11	30	39	41
51	**30**	11	23	39	41
51	**39**	11	23	30	41
51	**41**	11	23	30	39

This is the third process. Please take a look at Table 3. below.

Table 3. 3rd process

		pivot			
index[0]	index[1]	index[2]	index[3]	index[4]	index[5]
51	**41**	**11**	23	30	39
51	**41**	**23**	11	30	39
51	**41**	**30**	11	23	39
51	**41**	**39**	11	23	30

This is the fourth process. Please take a look at Table 4. below.

Table 4. 4th process

			pivot		
index[0]	index[1]	index[2]	index[3]	index[4]	index[5]

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 2, Desember 2025
e-ISSN 2685-998X
DOI 10.30865/json.v7i2.9306

51	**41**	**39**	**11**	23	30
51	**41**	**39**	**23**	11	30
51	**41**	**39**	**30**	11	23

This is the fifth process. Please take a look at Table 5. below.

Table 5. 5th process

				pivot	
index[0]	index[1]	index[2]	index[3]	index[4]	index[5]
51	**41**	**39**	**30**	**11**	23
51	**41**	**39**	**30**	**23**	11

The result is a sorted descending number like this: 51, 41, 39, 30, 23, 11.

b. Selection Sort Analysis 1

The same as exchange sort, the selection sort will be sorted descending.

11 23 30 39 41 51

This is the first process. Please take a look at table 6. below. There are five index changes here.

Table 6. 1st Process

Index:	**0**	**1**	**2**	**3**	**4**	**5**
Number:	**11**	23	30	39	41	51
Comparison		Position				
11<23 (change index)		1				
23<30 (change index)		2				
30<39 (change index)		3				
39<41 (change index)		4				
41<51 (change index)		5				
Comment: Swap data on index 0 with index 5						
Index:	0	1	2	3	4	5
Number:	51	23	30	39	41	11

This is the second process. Please take a look at table 7. below. Three index change happens here.

Table 7. 2nd Process

Index:	**0**	**1**	**2**	**3**	**4**	**5**
Number:	51	**23**	30	39	41	11
Comparison		Position				
23<30 (change index)		2				
30<39 (change index)		3				
39<41 (change index)		4				
41>11		4				
Comment: Swap data on index 1 with index 4						
Index:	0	1	2	3	4	5
Number:	51	41	30	39	23	11

This is the third process. Please take a look at table 8. below. Only one index change happens here.

Table 8. 3rd Process

Index:	**0**	**1**	**2**	**3**	**4**	**5**
Number:	51	41	**30**	39	23	11
Comparison		Position				
30<39 (change index)		3				
39<23		3				
39<11		3				
Comment: Swap data on index 2 with index 3						
Index:	0	1	2	3	4	5
Number:	51	41	39	30	23	11

In this fourth process, the numbers are already sorted out. Please take a look at table 9. below. There is no index change.

Table 9. 4th Process

Index:	**0**	**1**	**2**	**3**	**4**	**5**
Number:	51	41	39	**30**	23	11
Comparison		Posisition				
30<23		3				

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 2, Desember 2025
e-ISSN 2685-998X
DOI 10.30865/json.v7i2.9306

30<11		3				
Comment: There is no swap						
Index:	0	1	2	3	4	5
Number:	51	41	39	30	23	11

This is the fifth process. Please take a look at table 10. below. Again, there is no index change in this process. This fifth process is the last process.

Table 10. 5th Process

Index:	**0**	**1**	**2**	**3**	**4**	**5**
Number:	51	41	39	30	**23**	11
Comparison		Position				
23<11		4				
Comment: There is no swap s						
Index:	0	1	2	3	4	5
Number:	51	41	39	30	23	11

3.1.2. Comparison and Theoretical Analysis 2

Let's say we have these numbers to sorted out descending.
1 22 10 9 42 26

a. Exchange Sort Analysis 2

This is the first process. Please take a look at Table 11. below. These numbers will be sorted descending.

Table 11. 1st process

pivot					
index[0]	index[1]	index[2]	index[3]	index[4]	index[5]
1	22	10	9	42	26
22	1	10	9	42	26
42	1	10	9	22	26

This is the second process. Please take a look at Table 12. below.

Table 12. 2nd process

	pivot				
index[0]	index[1]	index[2]	index[3]	index[4]	index[5]
42	**1**	10	9	22	26
42	**10**	1	9	22	26
42	**22**	1	9	10	26
42	**26**	1	9	10	22

This is the third process. Please take a look at Table 13. below.

Table 13. 3rd process

		Pivot			
index[0]	index[1]	index[2]	index[3]	index[4]	index[5]
42	**26**	**1**	9	10	22
42	**26**	**9**	1	10	22
42	**26**	**10**	1	9	22
42	**26**	**22**	1	9	10

This is the fourth process. Please take a look at Table 14. below.

Table 14. 4th process

			pivot		
index[0]	index[1]	index[2]	index[3]	index[4]	index[5]
42	**26**	**22**	**1**	9	10
42	**26**	**22**	**9**	1	10
42	**26**	**22**	**10**	1	9

This is the fifth process. Please take a look at Table 15. below.

Table 15. 5th process

				pivot	
index[0]	index[1]	index[2]	index[3]	index[4]	index[5]

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 2, Desember 2025
e-ISSN 2685-998X
DOI 10.30865/json.v7i2.9306

42	**26**	**22**	**10**	**1**	9
42	**26**	**22**	**10**	**9**	1

The end result is sorted descending.
42 26 22 10 9 1

b. Selection Sort Analysis 2

1 22 10 9 42 26

Selection sort analysis: These numbers will be sorted descending. This is the first process. Please take a look at table 16. below. Two index changes happen.

Table 16. 1st Process

Index	**0**	**1**	**2**	**3**	**4**	**5**
Number	**1**	22	10	9	42	26
Comparison		Position				
1<22 (change index)		1				
22>10		1				
22>9		1				
22<42 (change index)		4				
42<26		4				
Comment: Swap data on index 0 with index 4						
Index:	0	1	2	3	4	5
Number:	42	22	10	9	1	26

This is the second process. Please take a look at table 17. below. There is one index change here.

Table 17. 2nd Process

Index:	**0**	**1**	**2**	**3**	**4**	**5**
Number:	42	**22**	10	9	1	26
Comparison		Position				
22>10		1				
22>9		1				
22>1		1				
22<26 (change index)		5				
Comment: Swap data on index 1 with index 5						
Index:						0
Index:	0	1	2	3	4	5
Number:	42	26	**10**	9	1	22

This is the third process. Please take a look at table 18. below. One index change happens here.

Table 18. 3rd Process

Index:	**0**	**1**	**2**	**3**	**4**	**5**
Number:	42	26	**10**	9	1	22
Comparison		Position				
10>9		2				
10>1		2				
10<22 (change index)		5				
Comment: Swap data on index 2 with index 5						
Index:	0	1	2	3	4	5
Number:	42	26	22	**9**	1	10

This is the fourth process. Please take a look at table 19. below. Again, here, one index change happens.

Table 19. 4th Process

Index:	**0**	**1**	**2**	**3**	**4**	**5**
Number:	42	26	22	**9**	1	10
Comparison		Position				
9>1		3				
9<10 (change index)		5				
Comment: Swap data on index 3 with index 5						

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 2, Desember 2025
e-ISSN 2685-998X
DOI 10.30865/json.v7i2.9306

Index:	0	1	2	3	4	5
Number:	42	26	22	10	1	9

This is the fifth process. Please take a look at table 20. below. One index change happens.

Table 20. 5th Process

Index:	**0**	**1**	**2**	**3**	**4**	**5**
Number:	42	26	22	10	**1**	9
Comparison		Position				
1<9 (change index)		5				
Comment: Swap data on index 4 with index 5						
Index:	0	1	2	3	4	5
Number:	42	26	22	10	9	1

The end result is sorted descending
42 26 22 10 9 1

3.2. Comparative Discussion and Analysis

Based on the two analyses of selection sort and exchange sort above, if we compare it with some previous study then, it is concluded that the efficiency based on time executed programming, can not be analysed here. Since some previous study did research in terms of time efficiency. And in this research, we only discussed the step by step processes. In the comparison and theoretical analysis 1, we had two 'there is no changing position' of index, with selection sort algorithm. It was in the first process and the third process. In the third process of selection sort, in the comparison and theoretical analysis 1, the numbers are already sorted out, albeit we still have to carry on until the last process. It was not happened in the selection sort of comparison and theoretical analysis 2. Based on results we get from the comparison and theoretical analysis above, we have this Table 21., to be discussed. Each comparison and Analysis has two algorithms with total process or iteration done are 5. But the total swaps are different. For comparison and theoretical analysis 1, we got total swaps 14 with exchange sort, and total swaps 5 with selection sort. For comparison and theoretical analysis 2, we got 11 total swaps for exchange sort and 5 swaps for selection sort. There is no index change in exchange sort, but for comparison and theoretical analysis 1, in selection sort, total index change is 9 and for selection sort in comparison and theoretical analysis 2 the total index change is 6. This table, namely, table 21, illustrates the differences of total process, total swapped and total index change of the two algorithms.

Table 21. Discussion of the results

Comparison and Analysis	**Algorithm**	**Total Process**	**Total Swapped**	**Total Index Changed**
1	Exchange Sort	5	14	-
	Selection Sort	5	5	9
2	Exchange Sort	5	11	-
	Selection Sort	5	5	6

4. CONCLUSION

In the comparison and theoretical analysis 1, we did the procedure step by step and we concluded that, it was conducted by hand and we got the numbers to be descending from 11, 23, 30, 39, 41, 51 to be 51, 41, 39, 30, 23, 11. For the comparison and theoretical analysis 2, also conducted by hand, and we got the numbers to be descending from 1, 22, 10, 9, 42, 26 to be 42, 26, 22, 10, 9, 1. The main issue in this research is the more swaps then the longer the time needed to execute the processes. Also, the more the index changed, the longer the time needed to execute it. The most swap happened in exchange sort of comparison and theoretical analysis 1 with 14 swaps, in the second exchange sort it is only 11 swaps. Major known sorting algorithms in terms of speed are available and the comparison in terms of the time needed to execute these algorithms are very important. The contribution of these comparisons are to explain thoroughly the two algorithms based on step by step theoretical analysis. It is also concluded that the theoretical analysis of these two algorithms have been achieved with clear explanation by describing it step by step. This is a systematic literature review analysis to analyse both algorithms. The method of PRISMA flowchart is also presented. PRISMA diagram provides transparency into the literature selection process and allows readers to assess the validity of the research methodology. The research gap is about stages of processes algorithm with methodology systematic literature review. The objective of the stages above, are implemented in this analysis. Clear explanation is presented. Each step of SLR methodology combined with table analysis are given. Analysis of systematic literature review is specific for both algorithms. Exchange sort and selection sort. Implementation of PRISMA combined with planning, conducting literature research, and reporting aligned with the steps of process's algorithms. Specific for this methodology for the two algorithms.

Jurnal Sistem Komputer dan Informatika (JSON)
Volume 7, Nomor 2, Desember 2025
e-ISSN 2685-998X
DOI 10.30865/json.v7i2.9306

The limitation of this SLR analysis is that we use only six numbers to be analysed. However, here there are some limitations, since in terms of dataset in the real world, it can be millions or thousands dataset, not only 6 numbers as we discussed here. So that this journal only put emphasized on theoretical analysis

For future works, thorough experimental analysis must be done, to reveal the exact efficiency of these two algorithms, with suggested millions of datasets. Implementation with known programming language like Java, C++, Phyton, or other programming languages also suggested to be done. So that the speed of each comparison and theoretical analysis can be discussed.

REFERENCES

[1] Y. Singh, M. Verma, I. Pandey, A. Saini, P. Chawla, and V. Niranjan, "Analysis and Comparison of Various Sorting Algorithms," *Tuijin Jishu/Journal Propuls. Technol.*, vol. 45, no. 2, pp. 5885–5890, 2024.

[2] G. T. Heineman, G. Pollice, and S. Selkow, *Algorithms in a Nutshell*. 2025.

[3] J. Erickson, *Algorithms*, 1st ed. Urbana: University of Illinois, 2019.

[4] M. A. S. Ekowati, Z. P. Nindyatama, W. Widianto, and K. Dananti, "Comparative Analysis of the Speed of the Sorting Method on Google Translate Indonesian-English Using Binary Search," *Int. J. Glob. Oper. Res.*, vol. 3, no. 3, pp. 108–115, 2022, doi: 10.47194/ijgor.v3i3.167.

[5] D. J. Mankowitz *et al.*, "Faster sorting algorithms discovered using deep reinforcement learning," *Nature*, vol. 618, no. 7964, pp. 257–263, 2023, doi: 10.1038/s41586-023-06004-9.

[6] Q. Mehdi Rizvi, H. Rai, and R. Jaiswal, "Sorting Algorithms in Focus: A Critical Examination of Sorting Algorithm Performance," *Emerg. Trends IoT Comput. Technol.*, no. March, pp. 103–106, 2024, doi: 10.1201/9781003535423-19.

[7] R. Purnomo and T. D. Putra, "Research and Analysis of Exchange Sort Algorithm in Data Structure," *Sink. J. dan Penelit. Tek. Inform.*, vol. 9, no. 4, pp. 1935–1941, 2025, doi: https://doi.org/10.33395/sinkron.v9i4.15005.

[8] A. Zutshi and D. Goswami, "Systematic review and exploration of new avenues for sorting algorithm," *Int. J. Inf. Manag. Data Insights*, vol. 1, no. 2, p. 100042, 2021, doi: 10.1016/j.jjimei.2021.100042.

[9] Z. G. Zhu, "Analysis and Research of Sorting Algorithm in Data Structure Based on C Language," *J. Phys. Conf. Ser.*, vol. 1544, no. 1, 2020, doi: 10.1088/1742-6596/1544/1/012002.

[10] R. Purnomo and T. D. Putra, "Theoretical Analysis of Standard Selection Sort Algorithm," *SinkrOn*, vol. 8, no. 2, pp. 666–673, 2023, doi: 10.33395/sinkron.v8i2.12153.

[11] A. M. Rabiu, E. J. Garba, B. Y. Baha, and M. I. Mukhtar, "Comparative Analysis between Selection Sort and Merge Sort Algorithms," *Niger. J. Basic Appl. Sci.*, vol. 29, no. 1, pp. 43–48, 2022, doi: 10.4314/njbas.v29i1.5.

[12] Y. Chauhan and A. Duggal, "Different Sorting Algorithms comparison based upon the Time Complexity," *Int. J. Res. Anal. Rev.*, vol. 7, no. 3, pp. 114–121, 2020, [Online]. Available: www.ijrar.org

[13] Pujiati, "Tahapan Systematic Literature Review & Contohnya," deepublish. Accessed: Nov. 21, 2024. [Online]. Available: https://penerbitdeepublish.com/systematic-literature-review/

[14] B. University, "Memahami Metode PRISMA dalam Systematic Literature Review (SLR)," School of Computer Science.

[15] B. Ekmekci, C. E. McAnany, and C. Mura, "An Introduction to Programming for Bioscientists: A Python-Based Primer," *PLoS Comput. Biol.*, vol. 12, no. 6, 2016, doi: 10.1371/journal.pcbi.1004867.

[16] K. Priambodo and J. Sasongko Wibowo, "Implementasi Algoritma Selection Sort Untuk Perangkingan Poin Pada E-Sports Tournament Garuda League," no. 2020, pp. 978–979, 2021, [Online]. Available: www.garudaleague.com

[17] M. R. Hanafi, M. A. Faadhilah, M. T. Dwi Putra, and D. Pradeka, "Comparison Analysis of Bubble Sort Algorithm with Tim Sort Algorithm Sorting Against the Amount of Data," *J. Comput. Eng. Electron. Inf. Technol.*, vol. 1, no. 1, pp. 29–38, 2022, doi: 10.17509/coelite.v1i1.43794.

[18] A. Naz, H. Nawaz, A. Maitlo, and S. M. Hassan, "Implementation of Selection Sort Algorithm in Various Programming Languages," *Int. J. Adv. Trends Comput. Sci. Eng.*, vol. 10, no. 3, pp. 2371–2377, 2021, doi: 10.30534/ijatcse/2021/1231032021.

[19] R. N. Vilchez, "Modified Selection Sort Algorithm Employing Boolean and Distinct Function in a Bidirectional Enhanced Selection Technique," *Int. J. Mach. Learn. Comput.*, vol. 10, no. 1, pp. 93–98, 2020, doi: 10.18178/ijmlc.2020.10.1.904.

[20] S. Selvi, M. A. C. Evert, and B. Case, "Implementation of Effecient Sorting Algorithm in C/C++," vol. 514, no. 3, pp. 34–40, 2021.

[21] V. Paradigm, "Visual Paradigm Online," Web Page. Accessed: Nov. 30, 2025. [Online]. Available: https://www.visual-paradigm.com/features/