JOURNAL OF MULTIDISCIPLINARY SCIENCES AND INNOVATIONS
VOLUME 5, ISSUE 03
MONTHLY JOURNALS
ISSN NUMBER: 2751-4390
IMPACT FACTOR: 9,08

# THE SELECTION SORT METHOD FOR ARRAY SORTING IN PYTHON PROGRAMMING

**Fathiddinov Saidma‘ruf Lazizxo‘ja o‘g‘li,**
Department of General Technical Sciences, Asia International University

**Abstract:**
This article provides an overview of the selection sort algorithm for sorting arrays in Python programming. Sorting algorithms are essential tools in computer science and play a significant role in data processing and organization. Selection sort is a simple comparison-based sorting method that works by repeatedly selecting the smallest element from an unsorted portion of the list and placing it at the correct position. Although it is not the most efficient algorithm for large datasets, selection sort is widely used for educational purposes because of its simple logic and easy implementation. The study highlights the importance of understanding basic sorting algorithms in order to develop algorithmic thinking and programming skills in Python.

**Keywords:** Python, selection sort, sorting algorithms, arrays, lists, data structures, programming.

## Introduction

In modern programming, efficient data organization is essential for developing reliable and optimized software systems. One of the most fundamental operations performed on collections of data is sorting. Sorting allows data to be arranged in a specific order, which simplifies searching, analyzing, and managing information.

Among the many sorting algorithms, selection sort is one of the simplest methods used to arrange elements in a list. The algorithm works by repeatedly finding the smallest element in the unsorted part of the list and swapping it with the first unsorted element. This process continues until the entire list becomes sorted.

Selection sort is often introduced in introductory programming courses because it clearly demonstrates fundamental programming concepts such as loops, comparisons, and element swapping. The main objective of this article is to analyze the selection sort method in Python programming and evaluate its effectiveness for educational purposes and small-scale applications.

## Research Methods

This research is based on theoretical analysis of sorting algorithms and their implementation in Python programming. At the initial stage, computer science textbooks, algorithm theory resources, and Python programming documentation were analyzed to understand the fundamental principles of the selection sort algorithm.

In the second stage, the selection sort algorithm was examined through simple experimental examples in Python. The algorithm was implemented and tested using small datasets to observe its behavior and performance.

Comparative analysis was also conducted between selection sort and other sorting methods such as bubble sort and Python’s built-in sorting functions. Logical reasoning and generalization methods were applied to identify the strengths and limitations of the algorithm.

## Results

The results of the study show that the selection sort algorithm is a simple and understandable method for sorting arrays (lists) in Python programming. The algorithm works by repeatedly scanning the list to identify the smallest element in the unsorted portion and placing it in its correct position in the sorted part of the list.

Through the analysis, it was observed that selection sort follows a systematic process in which each iteration selects the minimum value and moves it to the appropriate position. This step-by-step approach makes the algorithm easy to understand and suitable for explaining fundamental programming concepts such as iteration, comparison, and data manipulation.

The results also demonstrate that selection sort produces accurate sorting results for small datasets. However, as the size of the array increases, the number of comparisons required also increases significantly. This leads to slower execution compared to more advanced sorting algorithms.

Despite these limitations, the algorithm remains useful for educational purposes, as it helps beginners understand the internal logic of sorting processes and the basic principles of algorithm design.

**Discussion**

The findings indicate that selection sort is a useful algorithm for teaching fundamental programming concepts. Its simple structure allows students to understand how sorting works step by step.

Compared to more advanced algorithms such as quick sort or merge sort, selection sort is less efficient for large datasets because it requires a large number of comparisons. The algorithm has a time complexity of $O(n^2)$, which means the running time increases rapidly as the size of the dataset grows.

However, despite its limitations, selection sort remains an important educational tool. By studying this algorithm, students can develop algorithmic thinking and gain a better understanding of how data structures operate in programming.

**Conclusion**

In conclusion, the selection sort method is one of the simplest algorithms used for sorting arrays in Python programming. Its clear logic and straightforward implementation make it suitable for educational purposes and for understanding basic algorithmic concepts.

Although the algorithm is not efficient for large datasets due to its quadratic time complexity, it provides valuable insights into the principles of sorting and data manipulation. Learning selection sort helps programmers build a strong foundation for understanding more advanced sorting algorithms in the future.

**References**

1. Dubois, P. F. Python: batteries included. Comput. Sci. Eng. 9, 7–9 (2007).
2. Millman, K. J. & Aivazis, M. Python for scientists and engineers. Comput. Sci. Eng. 13, 9–12 (2011).
3. Oliphant, T. E. Python for scientific computing. Comput. Sci. Eng. 9, 10–20 (2007).
4. Oliphant, T. E. Guide to NumPy 1st edn (Trelgol Publishing, 2006).
5. Dubois, P. F., Hinsen, K. & Hugunin, J. Numerical Python. Comput. Phys. 10, 262–267 (1996).
6. Ascher, D., Dubois, P. F., Hinsen, K., Hugunin, J. & Oliphant, T. E. An Open Source Project: Numerical Python (Lawrence Livermore National Laboratory, 2001).
7. Pedregosa, F. et al. Scikit-learn: machine learning in Python. J. Mach. Learn. Res. 12, 2825–2830 (2011).