ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 3, January 2018

STOCKPILE ALGORITHM

NIRUPMA PATHAK
Assistant Professor
Computer Science and Engineering
S.R. Institute of Management & Technology, Lucknow, India
nirupmapathak@gmail.com

SHUBHAM TIWARI
B.Tech Student
Computer Science and Engineering
S.R. Institute of Management & Technology, Lucknow, India
shubham.tiwari540@outlook.com

ABSTRACT

In this paper, we present the work regarding the stockpile algorithm for sorting. This sorting algorithm is both theoretical and programmatically analysis show that the introduce stockpile algorithm which enhanced sorting algorithm. It is much faster than the other sorting algorithm because of its using the stack for sorting. Stockpile sort algorithm possibility of enhancing execution speed up to 40%. Code for this algorithm is written in Java programming Language. So easyto understand the concept of this sorting algorithm by everyone because Java is the popular language. Results and discussion show a higher level of performance for the sorting algorithm. It can theoretically prove that the algorithm can reduce steps with the stockpile sort and will improve sorts toward NlogN sort.

IndexTerms—Algorithm, Stockpile, Stack, Push, Pop, Top.

I. INTRODUCTION

The phenomenon of sorting has been known for a long time and has been used for thedatabase management system. Stockpile sorting is the most important method used to arrange data according to user requirement. In object-oriented software, represent the sorting by programming [1]. The practical work in sorting has so far, generally focused on ascending or descending order sequence. The important issue of sorting data has successfully addressed in this paper. The automated tool, such as JDK, has raised awareness of the importance of soring of data. Many sorting algorithms are available in theliterature. Sorting is a data structure operation, which is used for making searching and arranging of data item or record. The here arrangement of sorting involves either into ascending or descending order. Everything in this world has some advantage and disadvantage, some sorting algorithms are problem specific means they work well on some specific problem, not all the problem [2]. It saves time and helps searching data quickly. Sorting algorithm performance varies on which type of data being sorted, not easier to say that which one algorithm is better than another is. Here, theperformance of thedifferent algorithm is according to the data being sorted [3]. Some common sorting algorithms are the exchange or bubble sort, the selection sort, the insertion sort and the quicksort [4]. The Stockpile sort is a good one that uses for finding the greatest element and push in to the bottom of the sorted stack. This algorithm work as the algorithm of tower of hanoi [5]. Repeat until thestack is sorted. It is very intuitive and simple to program, offer quite good performance for particular strength being the small number of push and pop operations needed [6]. The stockpile sort always goes through a single comparisons in stack, for a given no of data items. However, sometimes question raise in front of us, is there any way through this sorting can be more effective and how to convert that algorithm into code [7, 8]. Then demonstrate a new algorithm, and finally to assign the coding implementing as a programming. This work introduces one new simple modification of sorting algorithm. The proposed algorithm of sorting algorithm is very simple and easy to implement in Javatool, a well known and easy to operate software for theobject oriented programmed implement. The theoretical and practical are successfully utilized for thesort algorithm to evaluate the caused for aparticular task and close agreement is revealed.

The organization of the paper is as follows: Section 1 covers the introduction to soring algorithm. The proposed stockpile algorithm is presented in Section 2. The proposed pseudo code of stockpile algorithm is presented in Section 3. Finally, the paper is concluded with the summarization of all the contents in Section 4.

II. OUR PROPOSED STOCKPILE ALGORITHM

In this section, one new stockpile algorithm has been introduced followed by the generalized pseudo code of proposed algorithm.

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 3, January 2018

Stockpile algorithm is an internal sorting algorithm. This sorting algorithm is an in-place comparison-based algorithm in stack where we uses two stack one is unsorted and other is sorted stack . It selects the element which is top of stack and compare to the element of sorted stack and then, if sorted stack is empty then we push element in to the sorted stack. If sorted stack is not empty then we compare element with element of sorted stack, if element is less than the top element of sorted stack then we push element into sorted stack otherwise we pop the element of sorted stack into variable temp and repeat the above process until all elements arrange in ascending order into the stack from top to bottom. Practical analysis of sort algorithm has been carried out Java which works on theobject orientedprogramming language. The results from the presented idea are in good agreement with the targeted results. Thus the developed stockpile algorithm is validated.

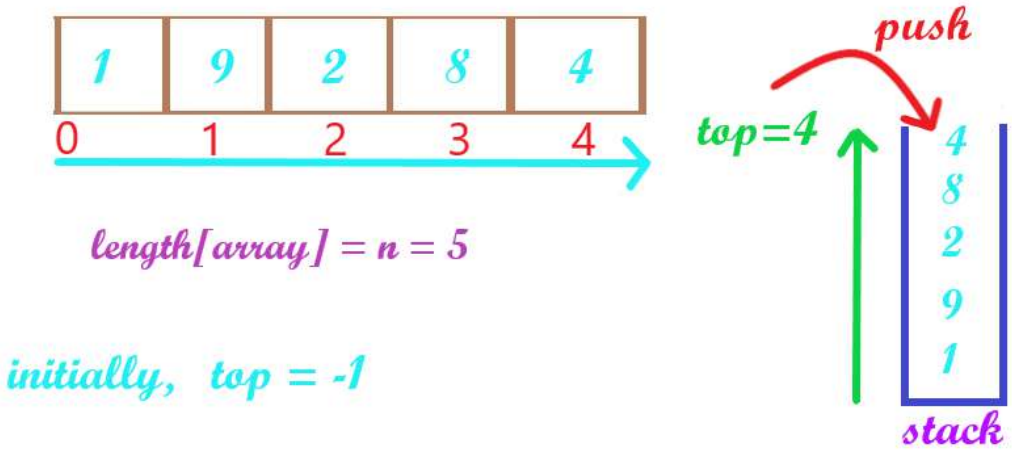
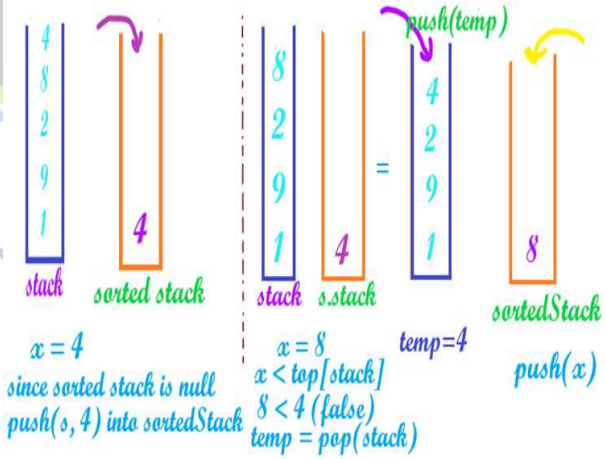


Figure 1. The methods of inserting data into stack.



ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 3, January 2018

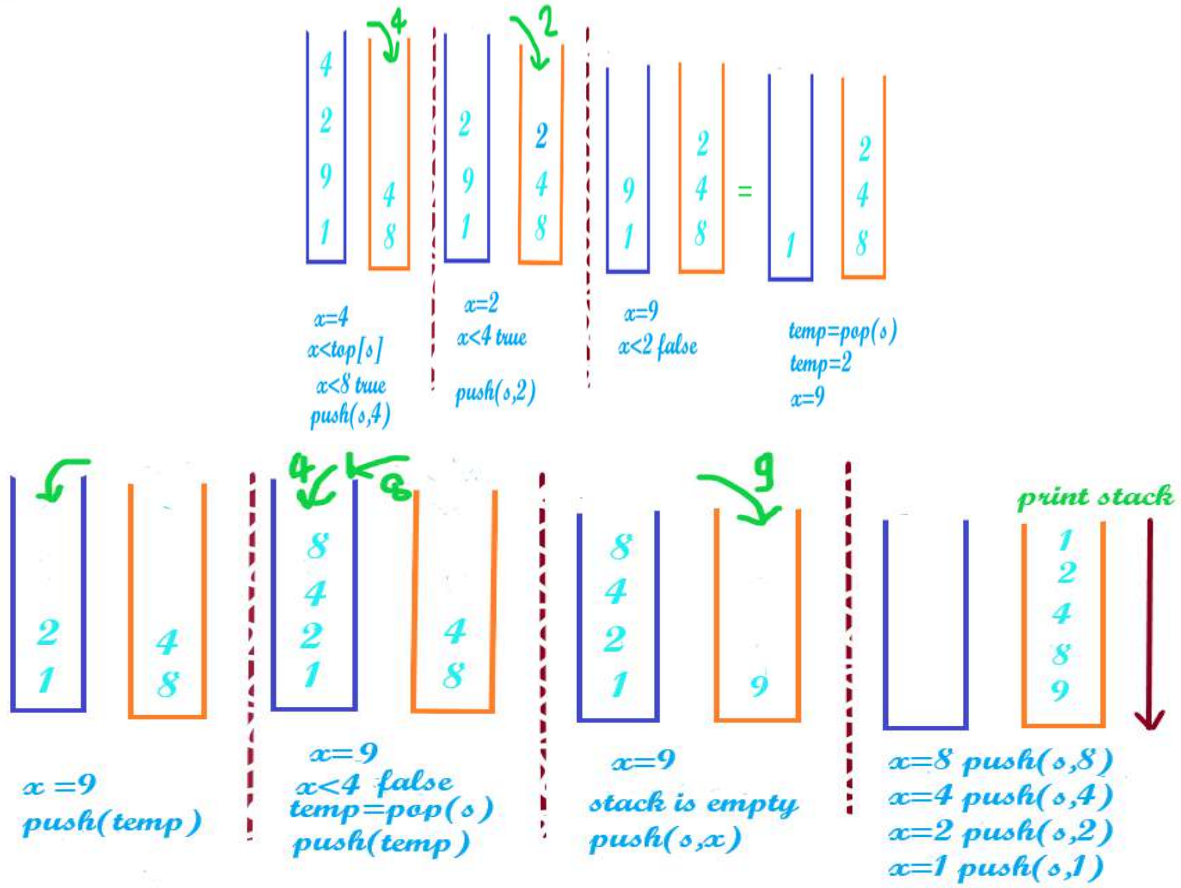


Figure 2. Sorting based on stack

At this point, we have developed anstockpile algorithm for the Database management system (DBMS) [5, 6, 7, 8]. Combining 8 steps the sort algorithm has been implemented. Algorithm 1 demonstrate the design procedure of the proposed idea of thesort algorithm. Line 1 of the algorithm assign array length. Line 6 push all elements into the stack. Line 5 call method SORT-STACK() and arrange elements into ascending order. Finally, Line 7 print result of algorithm.

Algorithm 1. Algorithm for the proposed idea of stockpile sorting.

1.n ← length[Array]
2.top ← -1
3.for i ← 0 to n-1
4.PUSH(s)
PUSH(s)
  i. top[stack] ← x
  ii. top ← top + 1
5. SORT-STACK(s)
SORT-STACK(s)
  i. if(stack ! Empty)
  ii. then x ← POP(s)
    POP(s)
      a) **x** ← stack[data]
      b) top ← top - 1
  iii. return SORT-STACK(s)
  iv. INSERT-INTO-SORTED-STACK(s, x)

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 3, January 2018

```
INSERT-INTO-SORTED-STACK(s, x)
    a) If(stack is empty || x < top[stack])
    b) then PUSH(s, x)
    c) else temp ← pop(s)
    d) INSERT-INTO-SORTED-STACK(s,
       x)
    e) PUSH(s, temp)
6.While (s)
7. print stack[data]
8. return stack[NextData]
```

III PROPOSED PSEUDO CODE OF STOCKPILE ALGORITHM

In this section, we have presented the pseudo code of stockpile algorithm. To present the stockpile sorting, we present the sorting methods in Section 2.

Algorithm 2 presents a formal representation of the proposed idea of stockpile algorithm based on Java programming language. In algorithm 2, these outputs are taken from line 33. The time complexity of Algorithm 2 depends on the loops, data storage array, and temporary storage. Thus the complexity is O(log n) where n is the size of thealgorithm. This is the optimum number of size of sorting algorithm for the proposed idea as proved in the Algorithm 2. The details pseudo code of stockpile algorithm is presented below.

Algorithm 2.

* **Pseudo code for Stockpile Algorithm ***

```
1)  import java.util.Scanner;
2)  import java.util.ListIterator;
3)  import java.util.Stack;

4)  class Stockpile
5)        {
      // Recursive Method to insert an item x in
sorted way
6)  static                                  void
    sortedStackInsert(Stack<Integer> s, int x)
7)  {
     // Base case: Either stack is empty or newly
inserted
     // item is greater than top (more than all
existing)
8)  if (s.isEmpty() || x < s.peek())
9)   {
10)       s.push(x);
11)      return;
12)    }

     // If top is greater, remove the top item and
recur
13)   int temp = s.pop();
14)    sortedStackInsert(s, x);

     // Put back the top item removed earlier
15)    s.push(temp);
16) }

       // Method to sort stack
```

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 3, January 2018

```
17)   static void sortStack(Stack<Integer> s)
18)   {
        // If stack is not empty
19)     if (!s.isEmpty())
20)     {
          // Remove the top item
21)       int x = s.pop();

          // Sort remaining stack
22)       sortStack(s);

        // Push the top item back in sorted stack
23)       sortedStackInsert(s, x);
24)     }
25)   }

   // Utility Method to print contents of stack
26)  static void printStack(Stack<Integer> s)
27)   {
28)    ListIterator<Integer> lt = s.listIterator();


    // forwarding
29)     while(lt.hasNext())
30)       lt.next();
31)
  // printing from top to bottom
32)     while(lt.hasPrevious())
33)       System.out.print( lt.previous()+" ");
34)  }

35)   public static void main(String[] args)
36)   {
37)    Stack<Integer> s = new Stack<>();

38)     int n;

39)     Scanner sc = new Scanner(System.in);

40)     System.out.print("Enter no. of elements
you want in array:");

41)     n = sc.nextInt();

 42)    int a[] = new int[n];

43)     System.out.println("Enter      all      the
elements:");

44)    for(int i = 0; i < n; i++)

 45)   {

46)       a[i] = sc.nextInt();
 47)                s.push(a[i]);

48)   }
```

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com

International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 5, Special Issue 3, January 2018

```
49)         System.out.println("Stack   elements
before sorting: ");
   50)   printStack(s);

51)    sortStack(s);

52)     System.out.println(" \n\nStack  elements
after sorting:");
53)   printStack(s);

54)    }
   55)       }
```

IV CONCLUSION AND FUTURE WORK

This paper presented the stockpile algorithm. In the customized software development, sorting of array provide less time and low-cost development. The programmability of sorting algorithm is based on the algorithms. This saves theroutine time of array sorting. In stockpile, sort makes up the small number of sorts that is easy to understand and faster than other sorting. In stockpile sort, the number of comparisons is minimum.As a result, we have designed minimal steps based sorting methodology to make the algorithm faster and less time to compute. An optimal step is maintained to reduce the runtime and finally, performance is enhanced. Finally, this study reveals that it is anoptimal algorithm and it applied to any system where the shorting is required. A possible future work of stockpile algorithm is the object-oriented based algorithm with fewer steps.

REFERENCES

[1] Thomas H. Coreman,Charles E. Leiserson and Ronald L. Rivest, “Introduction to Algorithms”, Printice Hall of India.
[2] “Data Structure using C” book by Udit Agrawal.
[3] Nirupma Pathak
Assistant Professor
Computer Science and Engineering
S.R. Institute of Management & Technology, Lucknow, India
nirupmapathak@gmail.com.