FREE eBook

LEARNING sorting

Free unaffiliated eBook created from
Stack Overflow contributors.

#sorting

Table of Contents

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: sorting

It is an unofficial and free sorting ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official sorting.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapter 1: Getting started with sorting

Remarks

This section provides an overview of what sorting is, and why a developer might want to use it.

It should also mention any large subjects within sorting, and link out to the related topics. Since the Documentation for sorting is new, you may need to create initial versions of those related topics.

Examples

Installation or Setup

Detailed instructions on getting sorting set up or installed.

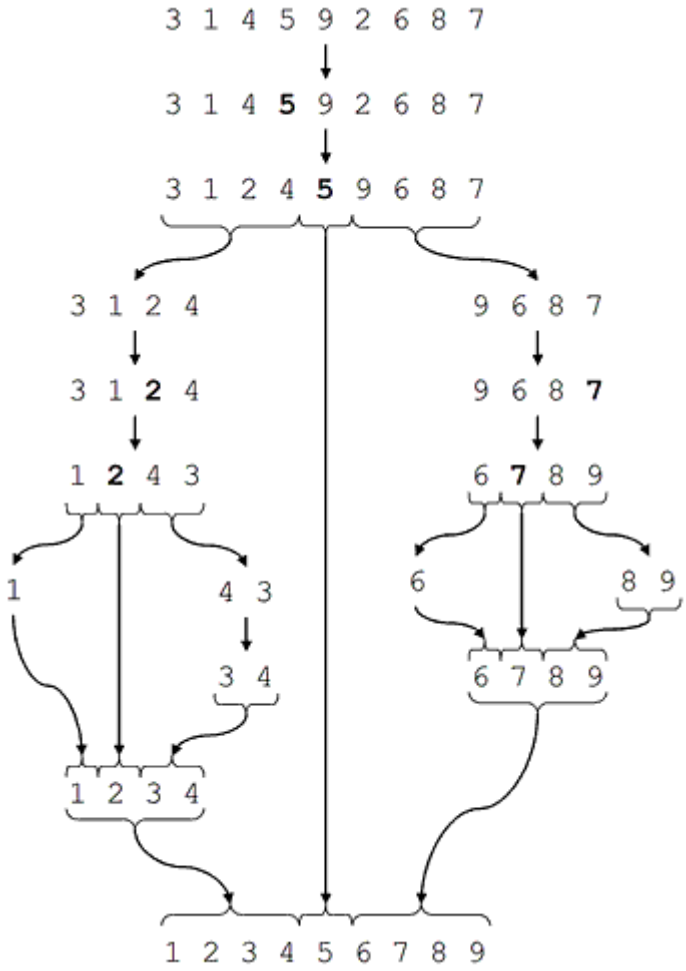
Read Getting started with sorting online: https://riptutorial.com/sorting/topic/5842/getting-started-with-sorting

Chapter 2: Quick Sort

Examples

Python

The below image shows the working of a quick sort.

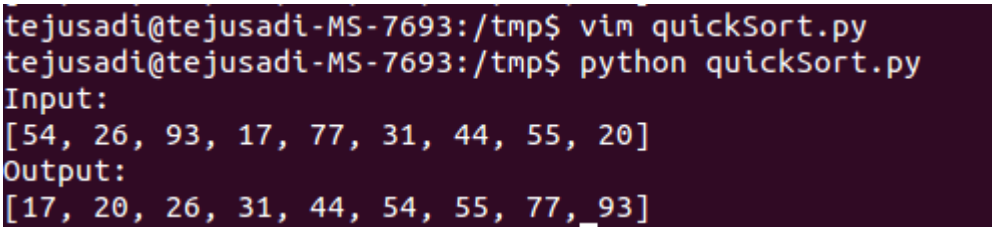


Below example shows the working program for quick sort in python:

```
def quickSort(alist):
   quickSortHelper(alist,0,len(alist)-1)

def quickSortHelper(alist,first,last):
   if first<last:

       splitpoint = partition(alist,first,last)

       quickSortHelper(alist,first,splitpoint-1)
       quickSortHelper(alist,splitpoint+1,last)


def partition(alist,first,last):
   pivotvalue = alist[first]
```

```
    leftmark = first+1
    rightmark = last

    done = False
    while not done:

        while leftmark <= rightmark and alist[leftmark] <= pivotvalue:
            leftmark = leftmark + 1

        while alist[rightmark] >= pivotvalue and rightmark >= leftmark:
            rightmark = rightmark -1

        if rightmark < leftmark:
            done = True
        else:
            temp = alist[leftmark]
            alist[leftmark] = alist[rightmark]
            alist[rightmark] = temp

    temp = alist[first]
    alist[first] = alist[rightmark]
    alist[rightmark] = temp


    return rightmark

alist = [54,26,93,17,77,31,44,55,20]
print("Input:")
print(alist)
quickSort(alist)
print("Output:")
print(alist)
```

Below is the output of the code:



Complexity of the above logic is : **O(n log n)**

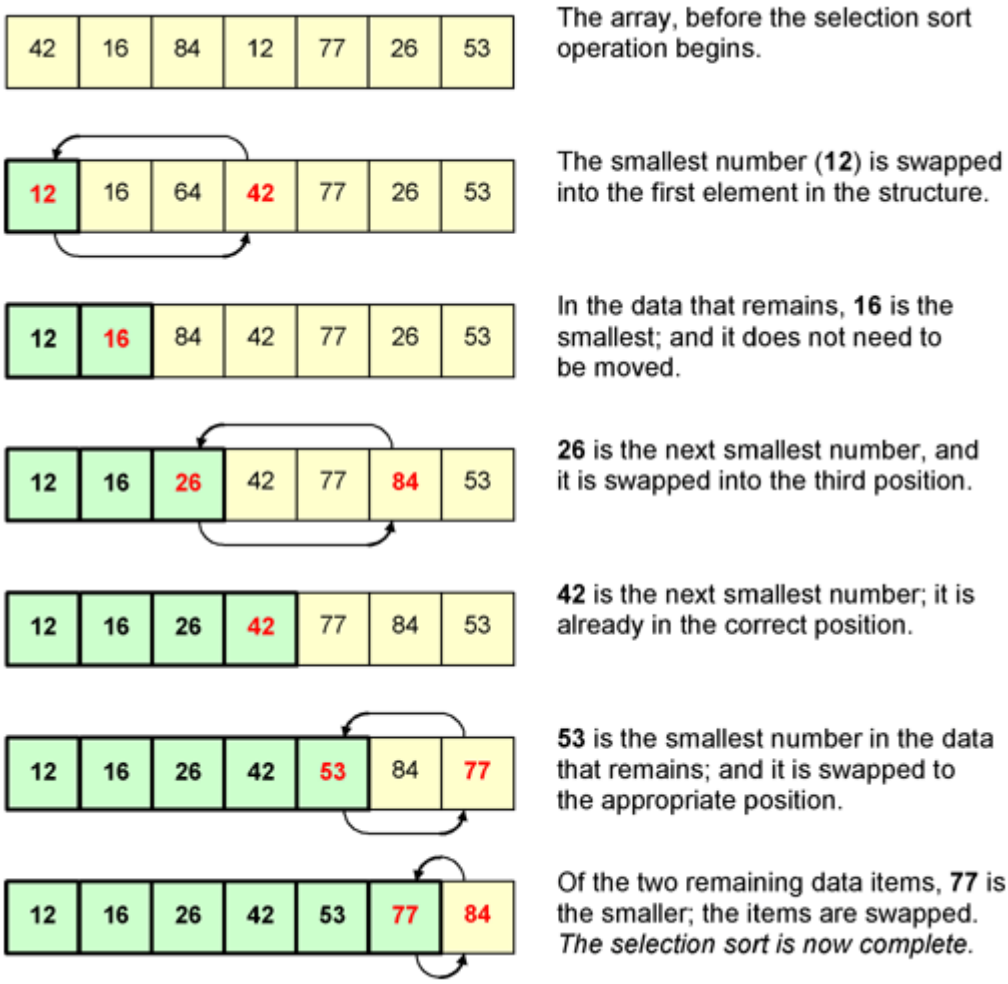
Read Quick Sort online: https://riptutorial.com/sorting/topic/7585/quick-sort

Chapter 3: Selection

Remarks

In computer science, a selection sort is a sorting algorithm, specifically an in-place comparison sort. It has O(n^2) time complexity, making it inefficient on large lists, and generally performs worse than the similar insertion sort. Selection sort is noted for its simplicity, and it has performance advantages over more complicated algorithms in certain situations, particularly where auxiliary memory is limited.

The below image shows how the selection sort works-



Below pseudo code helps in creating a program(in any language) or understanding selection sort.

```
procedure selection sort
list  : array of items
n     : size of list

for i = 1 to n - 1
/* set current element as minimum*/
  min = i

  /* check the element to be minimum */
```

```
  for j = i+1 to n
     if list[j] < list[min] then
        min = j;
     end if
  end for

  /* swap the minimum element with the current element*/
  if indexMin != i  then
     swap list[min] and list[i]
  end if

end for

end procedure
```

Advantages :

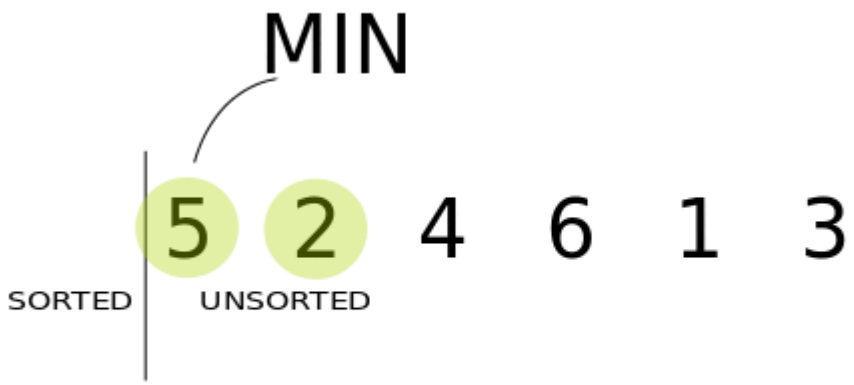
- it’s too simple to understand
- it performs well on a small list
- no additional temporary storage is required beyond what is needed to hold the original list

Image Reference: RMIT University

Examples

Selection Sort (Python)

Animation to show how selection sort works



The below example shows selection sort in Python

```
def sort_selection(my_list):

for pos_upper in xrange( len(my_list)-1, 0, -1):
```

```
    max_pos = 0
    for i in xrange(1, pos_upper + 1):
        if(my_list[i] > my_list[max_pos]):
            max_pos = i
            print "resetting max_pos = " + str(max_pos)

    my_list[pos_upper], my_list[max_pos] = my_list[max_pos], my_list[pos_upper]
    print "pos_upper: " + str(pos_upper) + " max_pos: " + str(max_pos) + " my_list: " +
str(my_list)

return my_list


if __name__ == "__main__":

    my_list = [54,26,93,17,77,31,44,55,20]
    print "my_list: " + str(my_list)
    print sort_selection(my_list)
```

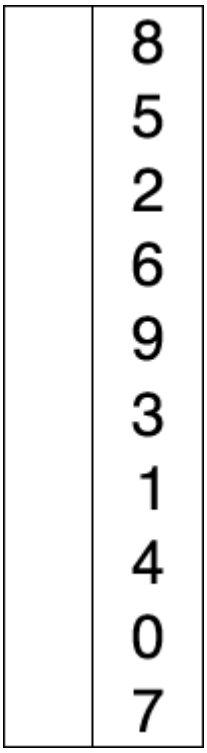
Output of the program:

```
my_list: [54, 26, 93, 17, 77, 31, 44, 55, 20]
resetting max_pos = 2
pos_upper: 8 max_pos: 2 my_list: [54, 26, 20, 17, 77, 31, 44, 55, 93]
resetting max_pos = 4
pos_upper: 7 max_pos: 4 my_list: [54, 26, 20, 17, 55, 31, 44, 77, 93]
resetting max_pos = 4
pos_upper: 6 max_pos: 4 my_list: [54, 26, 20, 17, 44, 31, 55, 77, 93]
pos_upper: 5 max_pos: 0 my_list: [31, 26, 20, 17, 44, 54, 55, 77, 93]
resetting max_pos = 4
pos_upper: 4 max_pos: 4 my_list: [31, 26, 20, 17, 44, 54, 55, 77, 93]
pos_upper: 3 max_pos: 0 my_list: [17, 26, 20, 31, 44, 54, 55, 77, 93]
resetting max_pos = 1
pos_upper: 2 max_pos: 1 my_list: [17, 20, 26, 31, 44, 54, 55, 77, 93]
resetting max_pos = 1
pos_upper: 1 max_pos: 1 my_list: [17, 20, 26, 31, 44, 54, 55, 77, 93]
[17, 20, 26, 31, 44, 54, 55, 77, 93]
```

Image Reference: Pirate Learner

Selection Sort (Java)

Animation to show how selection sort works



Below example shows selection sort in ascending order:

```
public class MySelectionSort {

    public static int[] doSelectionSort(int[] arr){

        for (int i = 0; i < arr.length - 1; i++)
        {
            int index = i;
            for (int j = i + 1; j < arr.length; j++)
                if (arr[j] < arr[index])
                    index = j;

            int smallerNumber = arr[index];
            arr[index] = arr[i];
            arr[i] = smallerNumber;
        }
        return arr;
    }
```

I've written a sample `main()` method to show the output of the selection sort:

```
    public static void main(String a[]){

        int[] arr1 = {10,34,2,56,7,67,88,42};
        int[] arr2 = doSelectionSort(arr1);
        for(int i:arr2){
            System.out.print(i);
            System.out.print(", ");
        }
    }
}
```

Output of the program :

```
2, 7, 10, 34, 42, 56, 67, 88
```

Below example shows selection sort in descending order:

```
public static void doDescendingSelectionSort ( int [ ] num )
{
     int i, j, first, temp;
     for ( i = num.length - 1; i > 0; i - - )
     {
          first = 0;   //initialize to subscript of first element
          for(j = 1; j <= i; j ++)   //locate smallest element between positions 1 and i.
          {
               if( num[ j ] < num[ first ] )
                 first = j;
          }
          temp = num[ first ];   //swap smallest found with element in position i.
          num[ first ] = num[ i ];
          num[ i ] = temp;
      }
}
```

Image Reference : Wikipedia

Read Selection online: https://riptutorial.com/sorting/topic/6170/selection

Credits

S. No	Chapters	Contributors
1	Getting started with sorting	Community
2	Quick Sort	Tejus Prasad
3	Selection	Tejus Prasad