Ramel Cary B. Jamen CSC124 - Design and Analysis of Algorithms
2019 - 2093

Tasks

- Implement the following sorting algorithms using swap:
    1. Bubble Sort
    2. Insertion Sort
    3. Selection Sort
    4. Partition Sort
- Use simplified and asymptotic models to analyze the runtime of each line of code.
- Create a driver program that will sort letters of your name, including spaces.
- Count the number of swap operations for each sorting algorithm. You may use automate the counting.

Runtime Analysis

Line #	Function Code	Simplified Model	Asymptotic Model
1	def swap(A, i, j):	0	O(1)
2	A[i], A[j] = A[j], A[i]	15	O(1)
Total		15	O(1)

Deliverable

Gray highlighted lines are made for counting the number of swaps, they are not included in analysis of runtime..

Line #	Bubble Sort	Simplified Model	Asymptotic Model
1	def bubbleSort(arr):	0	O(1)
2	length = len(arr)	6	O(1)
3	count = 0	-	-
4	i = 0	2	O(1)
5	while i < length - 1:	5 (n+2)	O(n)
6	j = 0	2 (n+1)	O(1)
7	while j < length - i - 1:	7 (n+1) (n+2)	O(n)
8	if arr[j] > arr[j + 1]:	11n (n+2)	O(1)
9	swap(arr, j, j + 1)	9n (n+1)	O(1)

10	count += 1	-	-
11	j += 1	4n (n+1)	O(1)
12	i += 1	4 (n+1)	O(1)
13	return arr, count - 1	2	O(1)
Total		15n^2 + 51n + 14.	O(n^2)

Line #	Insertion Sort	Simplified Model	Asymptotic Model
1	def insertionSort(arr):	0	O(1)
2	length = len(arr)	6	O(1)
3	count = 0	-	-
4	i = 1	2	O(1)
5	while i < length:	3 (n+2)	O(n)
6	key = arr[i]	5 (n+1)	O(1)
7	j = i - 1	4 (n+1)	O(1)
8	while j >= 0 and arr[j] > key:	9 (n+1) (n+2)	O(n)
9	swap(arr, j, j + 1)	9n (n+1)	O(1)
10	count += 1	-	-
11	j -= 1	4n (n+1)	O(1)
12	return arr, count - 1	2	O(1)
Total		11n^2 + 35n + 10	O(n^2)

Line #	Selection Sort	Simplified Model	Asymptotic Model
1	def selectionSort(arr):	0	O(1)
2	length = len(arr)	6	O(1)
3	count = 0	-	-
4	i = 0	2	O(1)
5	while i < length:	3 (n+2)	O(n)
6	min_index = i	2 (n+1)	O(1)
7	j = i + 1	4 (n+1)	O(1)
8	while j < length:	3 (n+1) [(n+2)]	O(n)
9	if arr[j] < arr[min_index]:	3n (n+2)	O(1)
10	min_index = j	9n (n+1)	O(1)
11	j += 1	4 (n+1)	O(1)
12	swap(arr, i, min_index)	7 (n+1)	O(1)

13	`        count += 1`	-	-
14	`        i += 1`	4 (n+1)	O(1)
15	`    return arr, count - 1`	2	O(1)
Total		6n^2 + 55n + 32	O(n^2)

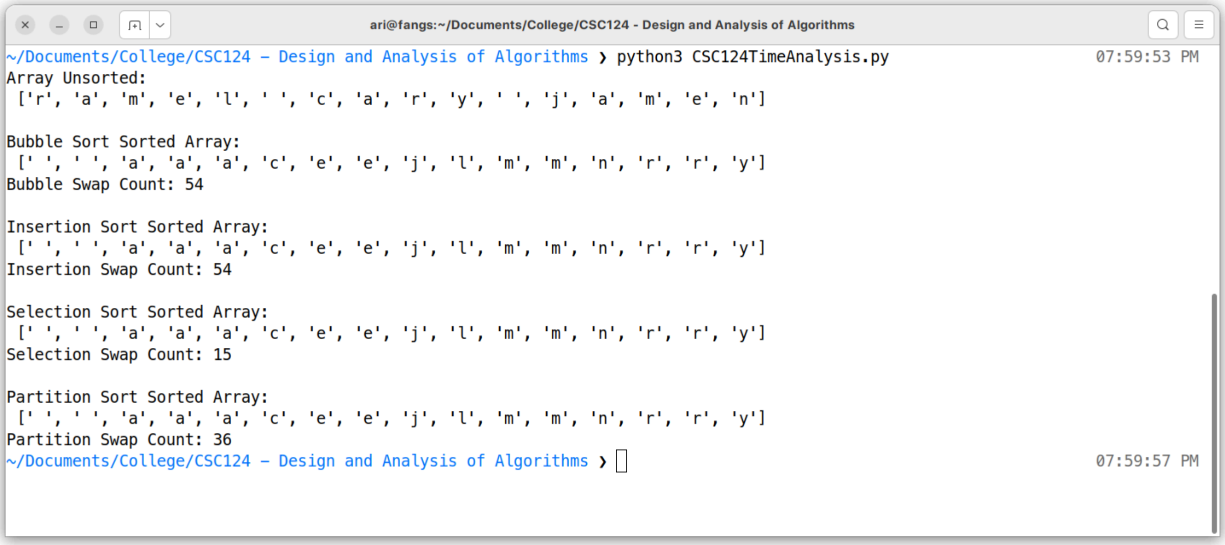
Line #	Partition Sort	Simplified Model	Asymptotic Model
1	`def partitionSort(arr):`	0	O(1)
2	`    length = len(arr)`	6	O(1)
3	`    count = 0`	-	-
4	`    def quicksort(arr, low, high):`	0	O(1)
5	`        if low < high:`	3	O(1)
6	`            pi = partition(arr, low, high)`	8	O(1)
7	`            quicksort(arr, low, pi - 1)`	9	O(log n)
8	`            quicksort(arr, pi + 1, high)`	9	O(log n)
9			-
10	`    def partition(arr, low, high):`	0	O(1)
11	`        nonlocal count`	-	O(1)
12	`        pivot = arr[high]`	5	O(1)
13	`        i = low - 1`	4	O(1)
14	`        j = low`	2	O(1)
15	`        while j < high:`	3 (n+2)	O(n)
16	`            if arr[j] < pivot:`	6 (n+1) (n+2)	O(1)
17	`                i += 1`	4 (n+1)	O(1)
18	`                swap(arr, i, j)`	7 (n+1)	O(1)
19	`                count += 1`	-	-
20	`            j += 1`	4	O(1)
21	`        swap(arr, i + 1, high)`	9	O(1)
22	`        count += 1`	-	-
23	`        return i + 1`	2	O(1)
24			
25	`    quicksort(arr, 0, length - 1)`	9	O(1)
26	`    return arr, count`	2	O(1)
total		6n^2 + 40n + 72	O(n log n)

Sample Driver Program for **Counting Swaps**:

```
def driver(A):
        print('Array Unsorted:\n',A)
        bSort, bCount = bubbleSort(A.copy())
        iSort, iCount = insertionSort(A.copy())
        sSort, sCount = selectionSort(A.copy())
        pSort, pCount = partitionSort(A.copy())
        print('\nBubble Sort Sorted Array:\n',bSort,'\nBubble Swap Count:', bCount)
        print('\nInsertion Sort Sorted Array:\n',iSort,'\nInsertion Swap Count:', iCount)
        print('\nSelection Sort Sorted Array:\n',sSort,'\nSelection Swap Count:', sCount)
        print('\nPartition Sort Sorted Array:\n',pSort,'\nPartition Swap Count:', pCount)
if __name__ == "__main__":
        # Array
        A = ['r','a','m','e','l',' ','c','a','r','y',' ','j','a','m','e','n']
        # Driver Code for Sorting
        driver(A)
```

Sorter	Number of Swaps
Bubble Sort	54
Insertion Sort	54
Selection Sort	15
Partition Sort	37

Sample Output:


ari@fangs:~/Documents/College/CSC124 - Design and Analysis of Algorithms

```
~/Documents/College/CSC124 - Design and Analysis of Algorithms ❯ python3 CSC124TimeAnalysis.py          07:59:53 PM
Array Unsorted:
 ['r', 'a', 'm', 'e', 'l', ' ', 'c', 'a', 'r', 'y', ' ', 'j', 'a', 'm', 'e', 'n']

Bubble Sort Sorted Array:
 [' ', ' ', 'a', 'a', 'a', 'c', 'e', 'e', 'j', 'l', 'm', 'm', 'n', 'r', 'r', 'y']
Bubble Swap Count: 54

Insertion Sort Sorted Array:
 [' ', ' ', 'a', 'a', 'a', 'c', 'e', 'e', 'j', 'l', 'm', 'm', 'n', 'r', 'r', 'y']
Insertion Swap Count: 54

Selection Sort Sorted Array:
 [' ', ' ', 'a', 'a', 'a', 'c', 'e', 'e', 'j', 'l', 'm', 'm', 'n', 'r', 'r', 'y']
Selection Swap Count: 15

Partition Sort Sorted Array:
 [' ', ' ', 'a', 'a', 'a', 'c', 'e', 'e', 'j', 'l', 'm', 'm', 'n', 'r', 'r', 'y']
Partition Swap Count: 36
~/Documents/College/CSC124 - Design and Analysis of Algorithms ❯                                          07:59:57 PM
```