

Assignment Number A04

Name: Tejas Dattatray Mote

Branch: Computer Engineering

Roll number: BCA-09

Subject: Design Analysis of Algorithms

```
#!/usr/bin/env python3

# Python implementation QuickSort using
# Lomuto's partition Scheme.
import random

'''
The function which implements QuickSort.
arr :- array to be sorted.
start :- starting index of the array.
stop :- ending index of the array.
'''
def quicksort(arr, start , stop):
    if(start < stop):

        # pivotindex is the index where
        # the pivot lies in the array
        pivotindex = partitionrand(arr, start, stop)

        # At this stage the array is
        # partially sorted around the pivot.
        # Separately sorting the
        # left half of the array and the
        # right half of the array.

        quicksort(arr , start , pivotindex-1)
        quicksort(arr, pivotindex + 1, stop)

# This function generates random pivot,
# swaps the first element with the pivot
# and calls the partition function.
def partitionrand(arr , start, stop):

    # Generating a random number between the
    # starting index of the array and the
    # ending index of the array.
    randpivot = random.randrange(start, stop)

    # Swapping the starting element of
    # the array and the pivot
    arr[start], arr[randpivot] = arr[randpivot], arr[start]
```

```

    return partition(arr, start, stop)
'''
This function takes the first element as pivot, places the pivot element at the
correct position in the sorted array. All the elements are re-arranged
according to the pivot, the elements smaller than the pivot is places on the
left and the elements greater than the pivot is placed to the right of
pivot.
'''
def partition(arr,start,stop):
    pivot = start # pivot

    # a variable to memorize where the
    i = start + 1

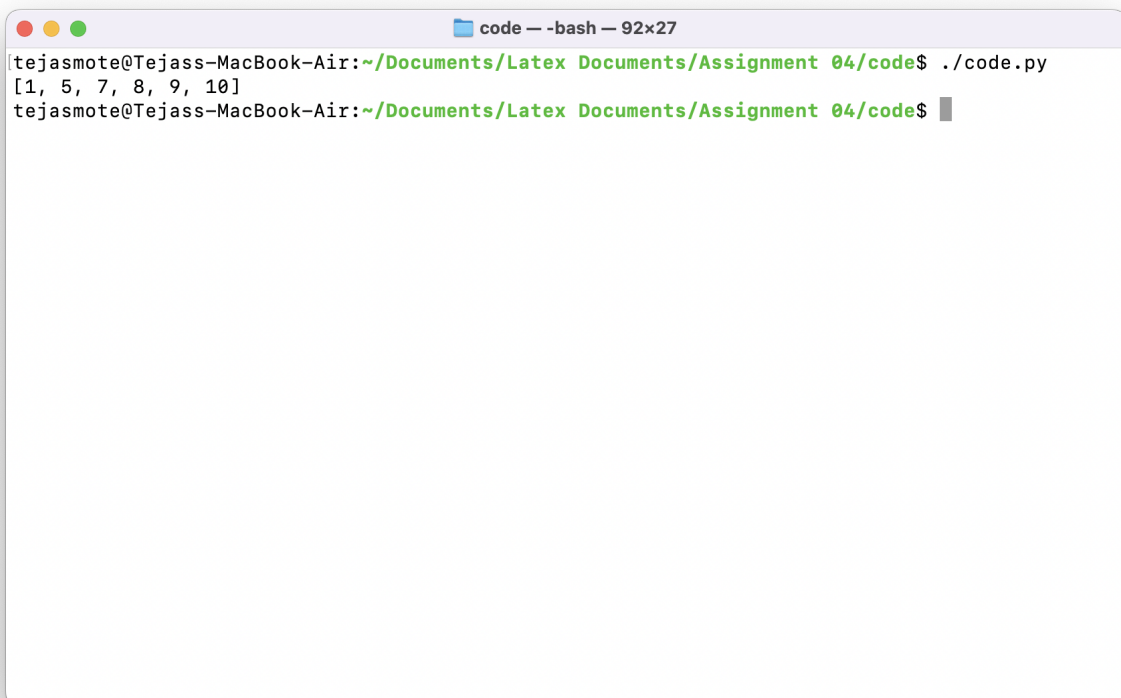
    # partition in the array starts from.
    for j in range(start + 1, stop + 1):

        # if the current element is smaller
        # or equal to pivot, shift it to the
        # left side of the partition.
        if arr[j] <= arr[pivot]:
            arr[i], arr[j] = arr[j] , arr[i]
            i = i + 1
    arr[pivot] , arr[i - 1] = arr[i - 1] , arr[pivot]
    pivot = i - 1
    return(pivot)

# Driver Code
if __name__ == "__main__":
    array = [10, 7, 8, 9, 1, 5]
    quicksort(array, 0, len(array) - 1)
    print(array)

```

Program 1: Program program for analysis of quick sort by using deterministic and randomized variant



A terminal window titled "code — bash — 92x27" is shown. The prompt is "tejasmote@Tejass-MacBook-Air:~/Documents/Latex Documents/Assignment 04/code\$". The user has entered the command `./code.py`, and the output is `[1, 5, 7, 8, 9, 10]`. The prompt is now ready for the next command.

```
tejasmote@Tejass-MacBook-Air:~/Documents/Latex Documents/Assignment 04/code$ ./code.py
[1, 5, 7, 8, 9, 10]
tejasmote@Tejass-MacBook-Air:~/Documents/Latex Documents/Assignment 04/code$
```

Figure 1: Output of The Program