Name: Prerna Sunil Jadhav
Sap Id: 60004220127
Batch: C2-2
Course: Advance Algorithm

EXP 3

AIM: Implement Quick Sort using Randomized Algorithm and perform complexity analysis of the solution.

THEORY: Quicksort is a popular sorting algorithm that chooses a pivot element and sorts the i/p list around that pivot element.
Randomized quick sort is designed to decrease the chances of the algorithm being executed in the worst case time complexity of $O(n^2)$.
The worst case time complexity of quick sort arises when the i/p given is already sorted list, leading to $n(n-1)$ comparisons.
There are two ways to randomize the quicksort:-

1) Randomly shuffling the i/p: Randomization is done on the i/p list so that the sorted i/p is jumbled again which reduces time complexity However, this is not usually performed in the randomized quick sort.
2) Randomly choosing the pivot element: Making the pivot element a random variable is

commonly used method in randomized quick sort. Here, even if the ilp is sorted, the pivot is chosen randomly so the worst time complexity is avoided.

CONCLUSION: Hence we implement Randomized Quick sort. The time needed to execute with & without randomization is shown. And a difference is seen between them. The time needed for randomization algorithm to sort is lesser than traditional algorithm.

Name:	Prerna Sunil Jadhav
Sap Id:	60004220127
Class:	T. Y. B. Tech (Computer Engineering)
Course:	Advance Algorithm Laboratory
Course Code:	DJ19CEL602
Experiment No.:	03

AIM: **Implement Quick Sort using Randomized Algorithm and perform complexity analysis of the solution.**

CODE:

```
# Randomized quicksort
import random
import time
def randomized_quicksort(arr):
    global c1

    if len(arr) <= 1:
        return arr
    else:
        pivot = random.choice(arr)
        left = [x for x in arr if x < pivot]
        middle = [x for x in arr if x == pivot]
        right = [x for x in arr if x > pivot]
        c1+= len(left)+len(right)
        return randomized_quicksort(left) + middle +
randomized_quicksort(right)


def quicksort(arr):
    global c2

    if len(arr) <= 1:
        return arr
    else:
        pivot = arr[0]
        left = []
        right = []
        for i in range(1, len(arr)):
            if arr[i] < pivot:
                left.append(arr[i])
                c2+=1

            else:
```

```
                right.append(arr[i])
                c2+=1
        return quicksort(left) + [pivot] + quicksort(right)

arr = [1,2,3,4,5,6,7,8,9,10]
arr1 = arr.copy()
c1 ,c2 = 0,0
# print(arr)
st = time.time()
print("Sorted by randomized way:",randomized_quicksort(arr))
print("Time taken by randomized quicksort:",(time.time() - st) ,"Comparisons
:", c1)
st = time.time()
print("Sorted by normal way",quicksort(arr1))
print("Time taken by normal quicksort:" , (time.time()-st)  ,"Comparisons"
,c2)

print("------------------------------------------------------------------")

arr = [random.randint(0,100) for i in range(500)]
# arr = [1,2,3,4,5,6,7,8,9,10]
arr1 = arr.copy()
c1 ,c2 = 0,0
# print(arr)
st = time.time()
print("Sorted by randomized way:",randomized_quicksort(arr))
print("Time taken by randomized quicksort:",(time.time() - st) ,"Comparisons
:", c1)
st = time.time()
print("Sorted by normal way",quicksort(arr1))
print("Time taken by normal quicksort:" , (time.time()-st)  ,"Comparisons"
,c2)
```

OUTPUT:

```
PS C:\Users\Jadhav\Documents\BTech\Docs\6th Sem\AA\Code> & C:/msys64/mingw64/bin/python.exe "c:/Users/Jadhav/Documents
/BTech/Docs/6th Sem/AA/Code/Randomized_QuickSort.py"
Sorted by randomized way: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Time taken by randomized quicksort: 0.0010294914245605469 Comparisons : 30
Sorted by normal way [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Time taken by normal quicksort: 0.0 Comparisons 45
------------------------------------------------------------
Sorted by randomized way: [0, 0, 0, 0, 0, 0, 0, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 6, 6
, 6, 6, 7, 7, 7, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 9, 9, 9, 9, 9, 9, 9, 10, 10, 10, 11, 11, 11, 11, 12, 12, 12, 12, 13, 14
, 14, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 17, 17, 17, 17, 17, 18, 18, 18, 18, 18, 18, 19, 19, 19, 19, 19, 20,
20, 20, 20, 21, 21, 21, 21, 21, 21, 22, 22, 22, 22, 22, 22, 23, 23, 24, 24, 24, 24, 25, 25, 25, 25, 26, 26, 26, 27, 27, 28
, 28, 28, 28, 28, 28, 28, 29, 29, 29, 29, 29, 29, 29, 29, 29, 29, 30, 31, 31, 32, 32, 32, 32, 32, 32, 32, 33, 33, 33,
33, 33, 33, 33, 34, 34, 34, 34, 34, 35, 35, 35, 35, 35, 35, 36, 36, 37, 37, 38, 38, 38, 38, 39, 39, 39, 39, 39, 40, 40
, 40, 41, 41, 41, 41, 41, 41, 42, 42, 42, 42, 42, 42, 42, 43, 43, 43, 43, 43, 43, 43, 43, 43, 44, 44, 44, 45, 45, 45,
46, 46, 46, 46, 46, 47, 47, 47, 47, 47, 48, 48, 48, 48, 48, 49, 49, 49, 49, 49, 50, 50, 51, 51, 51, 51, 51, 52, 52, 52
, 52, 52, 52, 53, 53, 53, 53, 53, 54, 54, 54, 54, 54, 55, 55, 55, 55, 55, 55, 56, 56, 56, 56, 56, 56, 56, 57, 57, 57,
57, 57, 57, 57, 58, 58, 58, 58, 58, 58, 58, 59, 59, 59, 59, 59, 59, 59, 59, 59, 59, 60, 60, 60, 60, 60, 61, 61, 61, 61
, 61, 61, 62, 62, 62, 62, 62, 63, 63, 63, 63, 64, 64, 64, 64, 64, 64, 65, 65, 65, 66, 66, 66, 66, 66, 66, 67, 67, 67,
67, 67, 67, 67, 67, 67, 67, 68, 68, 68, 68, 69, 69, 69, 69, 69, 69, 69, 70, 70, 71, 71, 72, 72, 72, 72, 72, 72, 72, 73
, 73, 73, 73, 73, 74, 74, 74, 74, 74, 74, 74, 75, 75, 76, 76, 76, 76, 76, 76, 77, 77, 77, 77, 77, 77, 77, 78, 78, 78,
78, 78, 79, 79, 79, 80, 80, 80, 80, 80, 80, 81, 81, 81, 81, 81, 81, 81, 81, 82, 82, 82, 82, 82, 83, 83, 84, 84, 84, 84
, 85, 85, 85, 85, 85, 85, 85, 85, 86, 87, 87, 87, 87, 87, 87, 87, 87, 88, 88, 88, 88, 89, 89, 89, 90, 91, 91, 91, 91,
91, 91, 92, 92, 92, 92, 93, 93, 93, 94, 94, 94, 94, 95, 95, 95, 95, 95, 95, 95, 96, 96, 97, 97, 97, 97, 97, 98, 98, 98
, 98, 98, 98, 98, 98, 98, 98, 99, 99, 99, 99, 99, 99, 100, 100, 100, 100, 100, 100, 100, 100]
Time taken by randomized quicksort: 0.0 Comparisons : 3319
```

```
Sorted by normal way [0, 0, 0, 0, 0, 0, 0, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 5, 5, 6, 6, 6,
6, 7, 7, 7, 8, 8, 8, 8, 8, 8, 8, 8, 8, 9, 9, 9, 9, 9, 9, 9, 10, 10, 10, 11, 11, 11, 12, 12, 12, 12, 13, 14, 14,
 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 17, 17, 17, 17, 17, 18, 18, 18, 18, 18, 18, 19, 19, 19, 19, 19, 20, 20, 2
0, 20, 21, 21, 21, 21, 21, 21, 22, 22, 22, 22, 22, 23, 23, 24, 24, 24, 24, 25, 25, 25, 25, 26, 26, 26, 27, 27, 28, 28,
 28, 28, 28, 28, 28, 29, 29, 29, 29, 29, 29, 29, 29, 29, 29, 30, 31, 31, 32, 32, 32, 32, 32, 32, 32, 33, 33, 33, 33, 3
3, 33, 33, 34, 34, 34, 34, 34, 35, 35, 35, 35, 35, 35, 36, 36, 37, 37, 38, 38, 38, 38, 39, 39, 39, 39, 39, 40, 40, 40,
 41, 41, 41, 41, 41, 41, 42, 42, 42, 42, 42, 42, 42, 43, 43, 43, 43, 43, 43, 43, 43, 43, 44, 44, 44, 45, 45, 45, 46, 4
6, 46, 46, 46, 47, 47, 47, 47, 47, 48, 48, 48, 48, 48, 49, 49, 49, 49, 49, 50, 50, 51, 51, 51, 51, 51, 52, 52, 52, 52,
 52, 52, 53, 53, 53, 53, 53, 54, 54, 54, 54, 54, 55, 55, 55, 55, 55, 55, 56, 56, 56, 56, 56, 56, 56, 57, 57, 57, 57, 5
7, 57, 57, 58, 58, 58, 58, 58, 58, 58, 59, 59, 59, 59, 59, 59, 59, 59, 59, 59, 60, 60, 60, 60, 60, 61, 61, 61, 61, 61,
 61, 62, 62, 62, 62, 62, 63, 63, 63, 63, 64, 64, 64, 64, 64, 64, 65, 65, 65, 66, 66, 66, 66, 66, 66, 67, 67, 67, 67, 6
7, 67, 67, 67, 67, 67, 68, 68, 68, 68, 69, 69, 69, 69, 69, 69, 69, 70, 70, 71, 71, 72, 72, 72, 72, 72, 72, 72, 73, 73,
 73, 73, 73, 74, 74, 74, 74, 74, 74, 74, 75, 75, 76, 76, 76, 76, 76, 76, 77, 77, 77, 77, 77, 77, 77, 78, 78, 78, 78, 7
8, 79, 79, 79, 80, 80, 80, 80, 80, 80, 81, 81, 81, 81, 81, 81, 81, 81, 82, 82, 82, 82, 82, 83, 83, 84, 84, 84, 84, 85,
 85, 85, 85, 85, 85, 85, 85, 86, 87, 87, 87, 87, 87, 87, 87, 87, 88, 88, 88, 88, 89, 89, 89, 90, 91, 91, 91, 91, 91, 9
1, 92, 92, 92, 92, 93, 93, 93, 94, 94, 94, 94, 95, 95, 95, 95, 95, 95, 95, 96, 96, 97, 97, 97, 97, 97, 98, 98, 98, 98,
 98, 98, 98, 98, 98, 98, 99, 99, 99, 99, 99, 99, 100, 100, 100, 100, 100, 100, 100, 100, 100]
Time taken by normal quicksort: 0.008697986602783203 Comparisons 4865
PS C:\Users\Jadhav\Documents\BTech\Docs\6th Sem\AA\Code>
```