AA_LAB_01_Assignment

Aim : analysis between randomized quicksort and normal quicksort.

- Randomized Quick-Sort:

Code:

```
# -*- coding: utf-8 -*-
"""
Created on Fri Jul 17 16:35:12 2020

@author: DHRUV
"""



from random import randint
comparison = 0

def quicksort(arr, start, end):
    if (start < end):
        pivot_index = partition(arr, start, end)
        quicksort(arr, start, pivot_index - 1)
        quicksort(arr, pivot_index + 1, end)
    return arr

def partition(arr, start, end):
    global comparison
    pivot = randint(start, end)
    temp1 = arr[end]
    arr[end], arr[pivot] = arr[pivot], temp1
    pivot_index = start
    for i in range(start, end):
        comparison += 1
        if (arr[i] <= arr[end]):
            temp1 = arr[i]
            arr[i], arr[pivot_index] = arr[pivot_index], temp1
            pivot_index += 1
    temp2 = arr[end]
    arr[end], arr[pivot_index] = arr[pivot_index], temp2
    return pivot_index


if __name__ == "__main__":
    arr = []
    for i in range(1000, -1, -1):
        arr.append(i)
    print(quicksort(arr, 0, len(arr) - 1))
    print(comparison)
```

- Analysis between normal and randomized quicksort:

N = 1000 elements

Input type :	Normal Quick-sort : No. of comparison	Randomized Quick-sort : No. of comparison
Integers 0 to 1000 (asc)	499500	10533
Integers 0 to 1000 (dec)	500500	10242
Random 1000 Integers Between 0 and 100	13081	12048

N = 2500 elements

Integers 0 to 2500 (asc)	3123750	31111
Integers 0 to 2500 (dec)	3126250	31996
Random Intergers Between 1 and 100	53688	49771