IDs: 9091 | 9205 | 8735

Time Comparison Report

Report on Time complexity between **Bubble**, **Insertion** and **Selection** Sorting Algorithms

1- **Bubble Sort:** Time Comp. is **O(n²)**

```python
import random
import time
def bubble_sort(arr):
    n = len(arr)
    for i in range(n):
        for j in range(0, n-i-1):
            if arr[j] > arr[j+1]:
                arr[j], arr[j+1] = arr[j+1], arr[j]

def generate_random_array(size):
    return [random.randint(1, 100000) for _ in range(size)]
def measure_time(sort_function, arr):
    start_time = time.time()
    sort_function(arr)
    end_time = time.time()
    return (end_time - start_time) * 1000
array_sizes = [50, 100, 500, 1000, 5000, 10000, 50000, 100000, 150000, 250000]
results = {}
for size in array_sizes:
    arr = generate_random_array(size)
    time_taken = measure_time(bubble_sort, arr.copy())
    results[size] = time_taken
    print(f"Running time for Bubble Sort with array size {size} is {time_taken:.2f} ms")
```

Array size	Bubble Sort
50	0.98
100	0.97
500	9.33
1000	114.49
5000	1949.9
10000	4590.22
50000	1168.42
100000	491495.13
150000	923456.78
250000	2345678.9

2- **Insertion Sort:** Time Comp. is **O(n²)**

```python
import random
import time

def insertion_sort(arr):
    for i in range(1, len(arr)):
        j = i
        while arr[j-1] > arr[j] and j>0:
            arr[j-1] , arr[j] = arr[j] , arr[j-1]
            j -=1

arr = [random.randint(0,50000) for _ in range(50000)]

startTime = time.time()

insertion_sort(arr)

endTime = time.time()
elapsedTime = endTime - startTime

print(f"Execution time: {elapsedTime:.3f} seconds")
```

Array size	Insertion Sort
50	0
100	0
500	7
1000	34
5000	962
10000	3963
50000	95674
100000	425713
150000	893356
250000	1798465

3- **Selection Sort:** Time Comp. is **O(n²)**

```python
import random
import time

def selection_sort(array):
    ln = len(array)
    for i in range (ln-1):
        minIndx = i
        for j in range (i+1, ln):
            if array[j] < array[minIndx]:
                minIndx = j
        array[j], array[minIndx] = array[minIndx], array[j]
    return array

# Generating Random array ---------------------------
# arr Sizes: [50, 100, 500, 1000, 5000, 10000, 50000, 100000, 150000, 250000]
siz = 250000
array = [random.randint(0,siz) for _ in range(siz)]

startTime = time.time()
selection_sort(array)
endTime = time.time()
elTime = endTime - startTime

print(f"The execution time is: {elTime:.4f} seconds, size:{siz}")
```

Array size	Selection Sort
50	0
100	0
500	3.1
1000	9.1
5000	310
10000	1277.1
50000	31124.4
100000	134643.3
150000	340005.4
250000	1266866.3

Time Comparison Graph