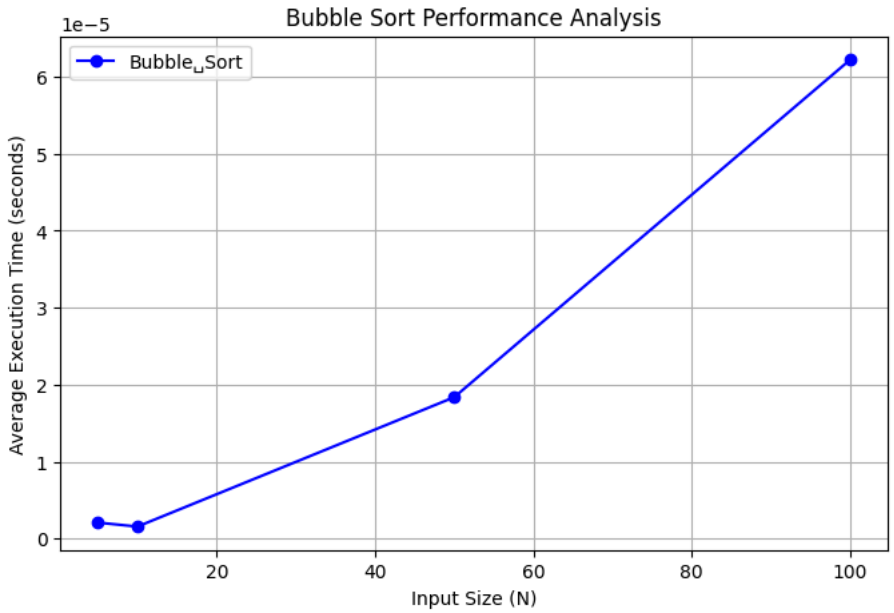
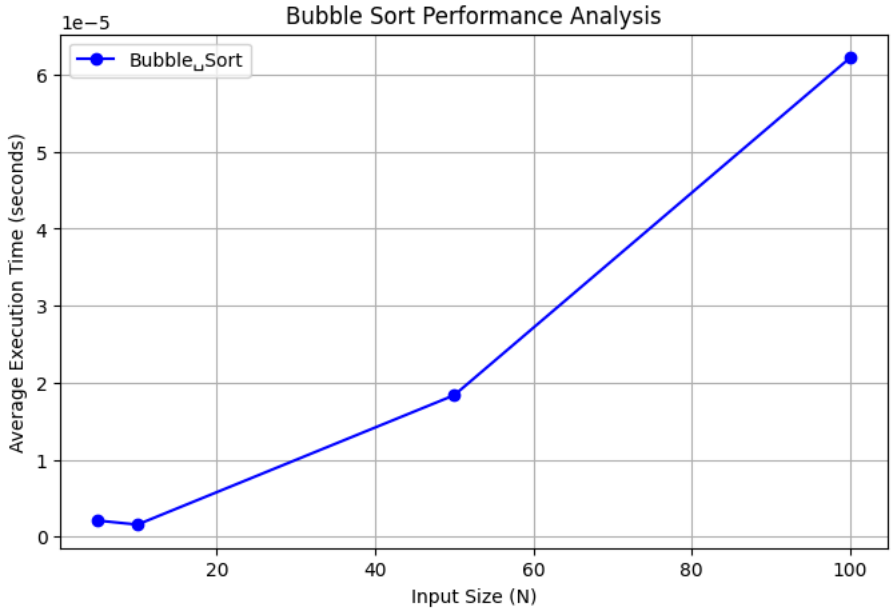
```
import time
import matplotlib.pyplot as plt

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]

Arr1 = list(range(1, 6))
Arr2 = list(range(1, 11))
Arr3 = list(range(1, 51))
Arr4 = list(range(1, 101))
arrays = [Arr1, Arr2, Arr3, Arr4]
input_sizes = [len(arr) for arr in arrays]

def measure_time(sort_function, arr):
  runs = 5
  total_time = 0
  for _ in range(runs):
   copy_arr = arr.copy()
   start = time.perf_counter()
   sort_function(copy_arr)
   end = time.perf_counter()
   total_time += (end - start)
   return total_time / runs


bubble_times = []
for arr in arrays:
 avg_time = measure_time(bubble_sort, arr)
 bubble_times.append(avg_time)
copy_arr = arr.copy()
bubble_sort(copy_arr)
print(f"Sorted array of size {len(arr)}: {copy_arr}")
print("\nAverage Execution Times:")
for size, time_taken in zip(input_sizes, bubble_times):
  print(f"Input size {size}: {time_taken:.8f} seconds")
# Plotting
  plt.figure(figsize=(8, 5))
  plt.plot(input_sizes, bubble_times, marker='o', color='blue', label="Bubble Sort")
  plt.title("Bubble Sort Performance Analysis")
  plt.xlabel("Input Size (N)")
  plt.ylabel("Average Execution Time (seconds)")
  plt.legend()
  plt.grid(True)
  plt.show()

```

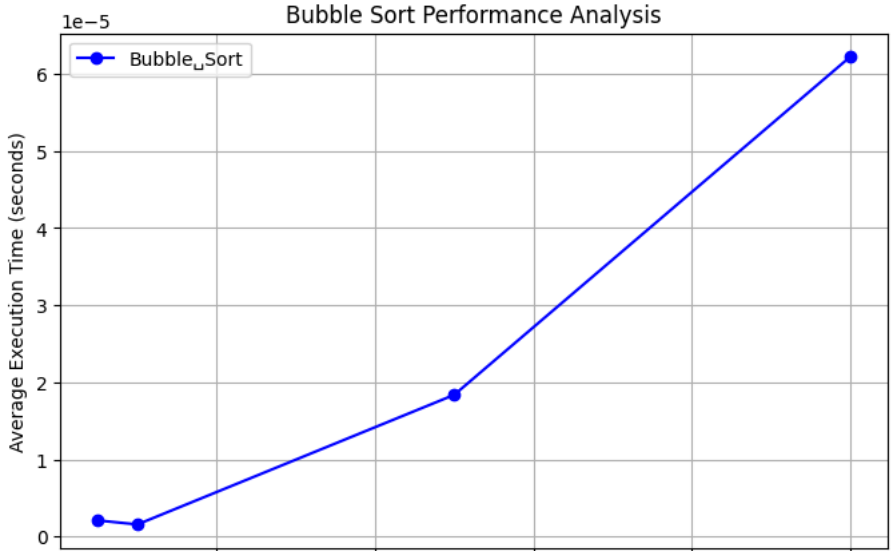
```
Sorted array of size 100: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29,

Average Execution Times:
Input size 5: 0.00000208 seconds
```

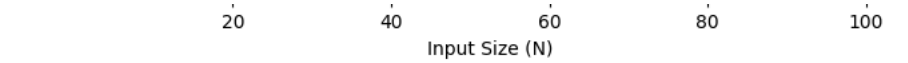



```
Input size 10: 0.00000157 seconds
```

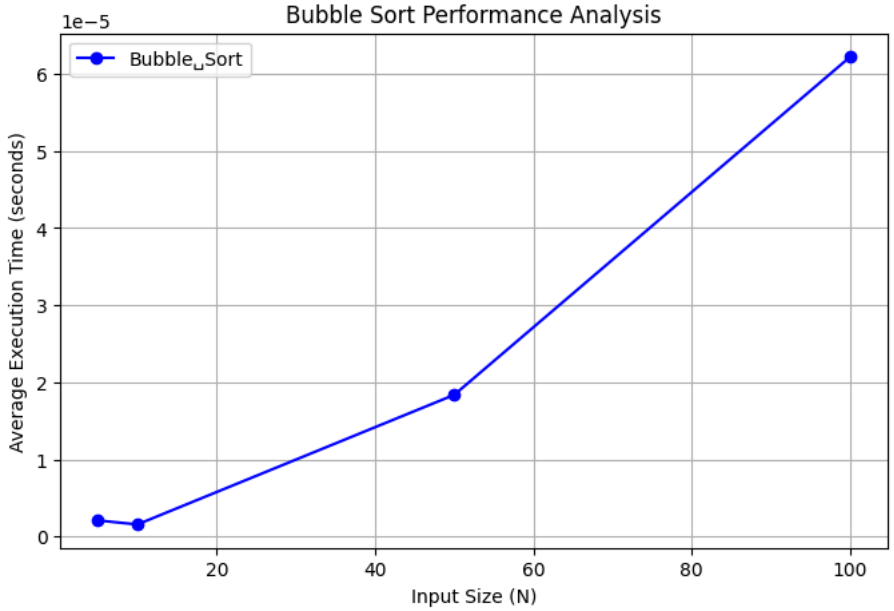



```
Input size 50: 0.00001838 seconds
```





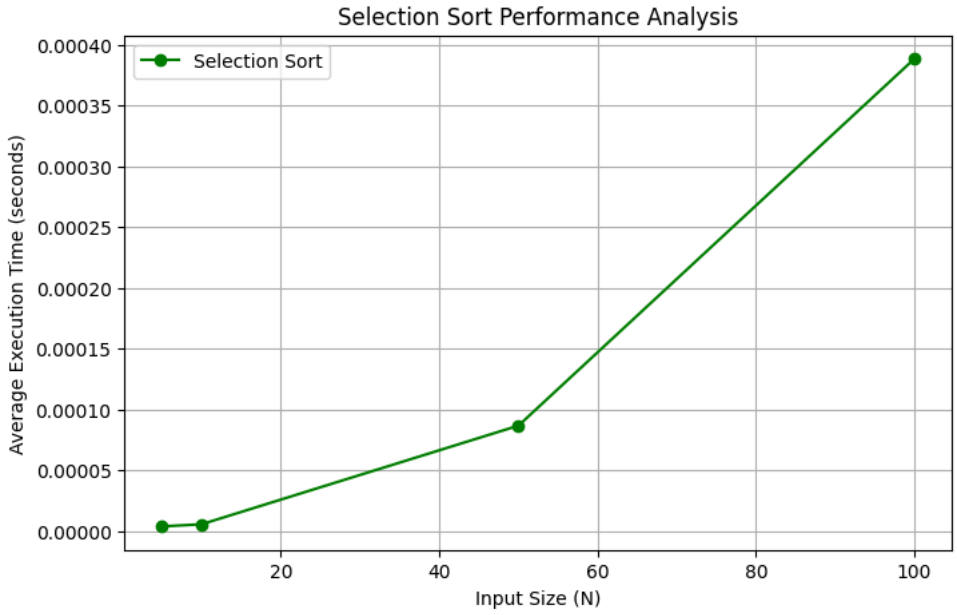

```
Input size 100: 0.00006223 seconds
```




```
import time
import matplotlib.pyplot as plt

def selection_sort(arr):
 n = len(arr)
 for i in range(n):
  min_idx = i
  for j in range(i+1, n):
   if arr[j] < arr[min_idx]:
    min_idx = j
 arr[i], arr[min_idx] = arr[min_idx], arr[i]

Arr1 = list(range(1, 6))
Arr2 = list(range(1, 11))
Arr3 = list(range(1, 51))
Arr4 = list(range(1, 101))
arrays = [Arr1, Arr2, Arr3, Arr4]
input_sizes = [len(arr) for arr in arrays]

def measure_time(sort_function, arr):
 runs = 5
 total_time = 0
 for _ in range(runs):
  copy_arr = arr.copy()
  start = time.perf_counter()
  sort_function(copy_arr)
  end = time.perf_counter()
  total_time += (end - start)
 return total_time / runs


selection_times = []
for arr in arrays:
 avg_time = measure_time(selection_sort, arr)
 selection_times.append(avg_time)
 copy_arr = arr.copy()
 selection_sort(copy_arr)
 print(f"Sorted array of size {len(arr)}: {copy_arr}")
 print("\nAverage Execution Times for Selection Sort:")
for size, time_taken in zip(input_sizes, selection_times):
 print(f"Input size {size}: {time_taken:.8f} seconds")
# Plotting
plt.figure(figsize=(8, 5))
plt.plot(input_sizes, selection_times, marker='o', color='green',label="Selection Sort")
plt.title("Selection Sort Performance Analysis")
plt.xlabel("Input Size (N)")
```

```
plt.ylabel("Average Execution Time (seconds)")
plt.legend()
plt.grid(True)
plt.show()

```

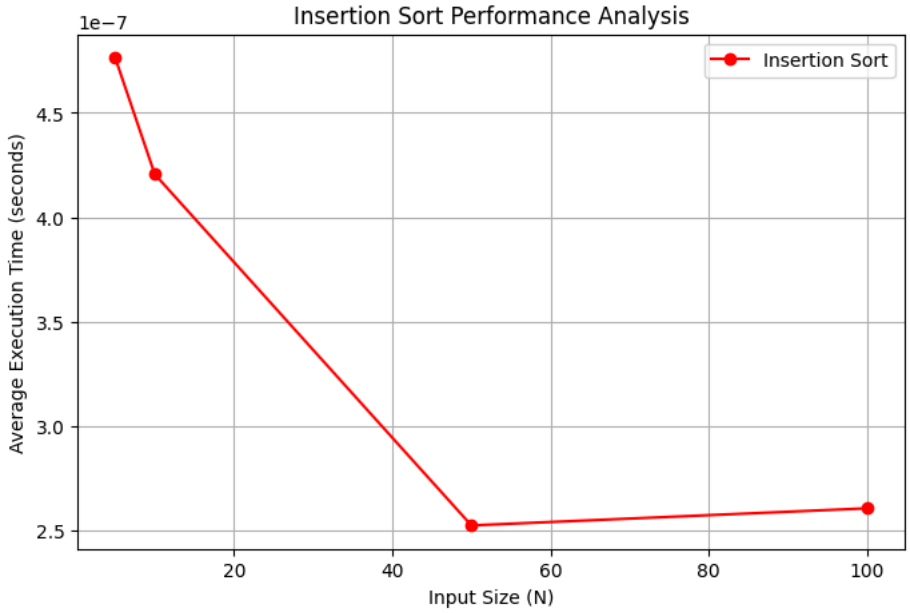
```
Sorted array of size 5: [1, 2, 3, 4, 5]

Average Execution Times for Selection Sort:
Sorted array of size 10: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

Average Execution Times for Selection Sort:
Sorted array of size 50: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30,

Average Execution Times for Selection Sort:
Sorted array of size 100: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30

Average Execution Times for Selection Sort:
Input size 5: 0.00000391 seconds
Input size 10: 0.00000572 seconds
Input size 50: 0.00008679 seconds
Input size 100: 0.00038847 seconds
```



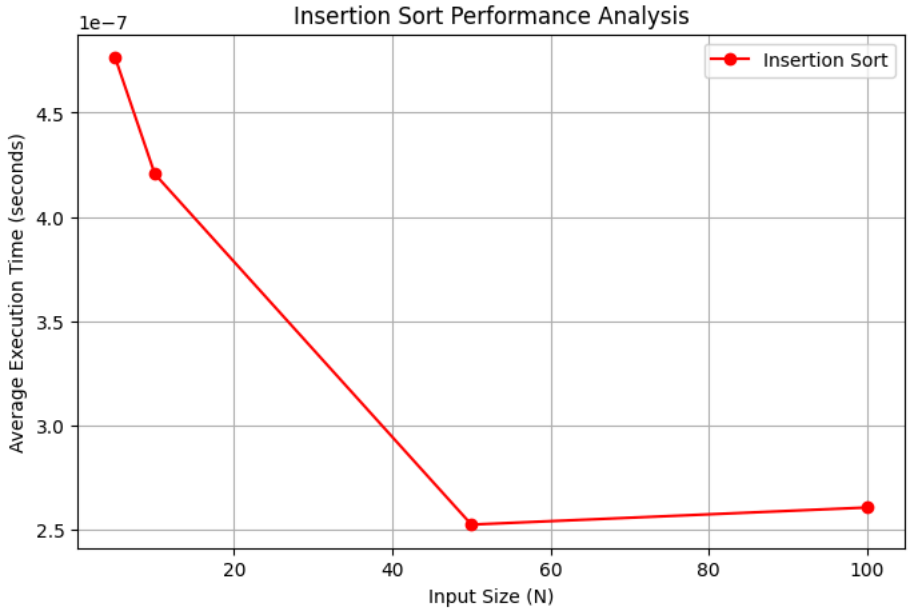

```
import time
import matplotlib.pyplot as plt

def insertion_sort(arr):6
for i in range(1, len(arr)):
 key = arr[i]
 j = i - 1
 while j >= 0 and arr[j] > key:
   arr[j + 1] = arr[j]
   j -= 1
   arr[j + 1] = key

Arr1 = list(range(1, 6))
Arr2 = list(range(1, 11))
Arr3 = list(range(1, 51))
Arr4 = list(range(1, 101))
arrays = [Arr1, Arr2, Arr3, Arr4]
input_sizes = [len(arr) for arr in arrays]


def measure_time(sort_function, arr):
 runs = 5
 total_time = 0
 for _ in range(runs):
  copy_arr = arr.copy()
  start = time.perf_counter()
  sort_function(copy_arr)
  end = time.perf_counter()
```

```
    total_time += (end - start)
  return total_time / runs

insertion_times = []
for arr in arrays:
  avg_time = measure_time(insertion_sort, arr)
  insertion_times.append(avg_time)

  copy_arr = arr.copy()
  insertion_sort(copy_arr)
print(f"Sorted array of size {len(arr)}: {copy_arr}")
print("\nAverage Execution Times for Insertion Sort:")
for size, time_taken in zip(input_sizes, insertion_times):
  print(f"Input size {size}: {time_taken:.8f} seconds")
  plt.figure(figsize=(8, 5))
  plt.plot(input_sizes, insertion_times, marker='o', color='red',label="Insertion Sort")
  plt.title("Insertion Sort Performance Analysis")
  plt.xlabel("Input Size (N)")
  plt.ylabel("Average Execution Time (seconds)")
  plt.legend()
  plt.grid(True)
  plt.show()

```

```
Sorted array of size 100: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29,

Average Execution Times for Insertion Sort:
Input size 5: 0.00000048 seconds
```




```
Input size 10: 0.00000042 seconds
```




```
Input size 50: 0.00000025 seconds
```