Gabrielle Vidal
Algorithms

11.5 Sorting

Bubble Sort Algorithm:

What is it?

- An algorithm runs through each element and switches them if necessary.

Best-case asymptotic notation: O(n)

Worst-case asymptotic notation: O(n^2)

PseudoCode:

```
function BubbleSort(arr: array of int):
   n = length of arr
   for i from 0 to n - 1:
      for j from 0 to n - i - 1:
         if arr[j] > arr[j + 1]:
            // Swap arr[j] and arr[j + 1]
            swap(arr[j], arr[j + 1])

function PrintNumbers(arr: array of int):
   for each number in arr:
      print(number + " ")
```

numbers = { 72, 69, 90, 47, 76, 71, 88, 40, 64, 38, 58, 40, 65, 78, 50, 69, 88, 18, 46, 54, 66, 65, 44, 69, 74, 73, 69, 67, 70, 62, 69, 63, 56, 40, 97, 81, 74, 50, 75, 57, 55, 58, 53, 59, 50, 65, 55, 66, 57, 82, 53, 77, 53, 88, 71, 33, 82, 52, 58, 0, 79, 39, 62, 69, 59, 67, 45, 60, 61, 39, 58, 63, 41, 61, 49, 44, 30, 80, 61, 62, 47, 49, 50, 72, 42, 73, 76, 71, 58, 73, 65, 27, 71, 43, 79, 78, 65, 63, 58, 65, 79, 68, 85, 60, 98, 58, 87, 66, 52, 70, 77, 62, 54, 51, 99, 84, 75, 78, 51, 55, 79, 91, 88, 63, 83, 87, 72, 65, 82, 51, 89, 53, 87, 75, 74, 58, 51, 70, 59, 71, 76, 59, 42, 57, 88, 22, 88, 73, 68, 100, 62, 77, 59, 54, 62, 70, 66, 60, 61, 66, 82, 75, 49, 52, 81, 96, 53, 58, 68, 67, 72, 94, 79, 63, 43, 81, 46, 71, 52, 97, 62, 46, 50, 65, 45, 65, 80, 62, 48, 77, 66, 76, 62, 77, 69, 61, 59, 55, 45, 78, 67, 65, 69, 57, 59, 74, 82, 81, 74, 58, 80, 35, 42, 60, 87, 84, 83, 34, 66, 61, 56, 87, 55, 86, 52, 45, 72, 57, 68, 88, 76, 46, 67, 92, 83, 80, 63, 64, 54, 84, 73, 80, 56, 59, 75, 85, 89, 58, 65, 68, 47, 71, 60, 80, 54, 62, 64, 78, 70, 65, 64, 79, 44, 99, 76, 59, 63, 69, 88, 71, 69, 58, 47, 65, 88, 83, 85, 59, 65, 73, 53, 45, 73, 70, 37, 81, 97, 67, 88, 77, 76, 86, 63, 65, 78, 67, 46, 71, 40, 90, 81, 56, 67, 80, 74, 69, 99, 51, 53, 49, 73, 66, 67, 68, 59, 71, 77, 83, 63, 56, 67, 75, 71, 43, 41, 82, 61, 28, 82, 41, 71, 47, 62, 90, 83, 61, 76, 49, 24, 35, 58, 61, 69, 67, 79, 72, 62, 77, 75, 87, 52, 66, 63, 46, 59, 61, 63, 42, 59, 80, 58, 85, 52, 27, 59, 49, 69, 61, 44, 73, 84, 45, 74, 82, 59, 46, 80, 85, 71, 66, 80, 87, 79, 38, 38, 67, 64, 57, 62, 73, 73, 77, 76, 57, 65, 48, 50, 85, 74, 60, 59, 53, 49, 88, 54, 63, 65, 82, 52, 87, 70, 84, 71, 63, 51, 84, 71, 74, 68, 57, 82, 57, 47, 59, 41, 62, 86, 69, 65, 68, 64, 61, 61, 47, 73, 50, 75, 75, 70, 89, 67, 78, 59, 73, 79, 67, 69, 86, 47, 81, 64, 100, 65, 65, 53, 37, 79, 53, 100, 72, 53, 54, 71, 77, 75, 84, 26, 72, 77, 91, 83, 63, 68, 59 };

```
print("Original Numbers:")
PrintNumbers(numbers)

BubbleSort(numbers)

print("\nSorted Numbers:")
PrintNumbers(numbers)
```

Sorting time: 0.15 seconds

Insertion Sort Algorithm:
What is it?

- A sorting algorithm that compares one item with others to determine its placement.

Best-case asymptotic notation: O(n)
Worst-case asymptotic notation: O(n2)
PseudoCode:

```
function InsertionSort(arr):
   n = length of arr
   for i from 1 to n - 1:
      key = arr[i]
      j = i - 1

      while j >= 0 and arr[j] > key:
         arr[j + 1] = arr[j]
         j = j - 1

      arr[j + 1] = key
```

numbers = { 72, 69, 90, 47, 76, 71, 88, 40, 64, 38, 58, 40, 65, 78, 50, 69, 88, 18, 46, 54, 66, 65, 44, 69, 74, 73, 69, 67, 70, 62, 69, 63, 56, 40, 97, 81, 74, 50, 75, 57, 55, 58, 53, 59, 50, 65, 55, 66, 57, 82, 53, 77, 53, 88, 71, 33, 82, 52, 58, 0, 79, 39, 62, 69, 59, 67, 45, 60, 61, 39, 58, 63, 41, 61, 49, 44, 30, 80, 61, 62, 47, 49, 50, 72, 42, 73, 76, 71, 58, 73, 65, 27, 71, 43, 79, 78, 65, 63, 58, 65, 79, 68, 85, 60, 98, 58, 87, 66, 52, 70, 77, 62, 54, 51, 99, 84, 75, 78, 51, 55, 79, 91, 88, 63, 83, 87, 72, 65, 82, 51, 89, 53, 87, 75, 74, 58, 51, 70, 59, 71, 76, 59, 42, 57, 88, 22, 88, 73, 68, 100, 62, 77, 59, 54, 62, 70, 66, 60, 61, 66, 82, 75, 49, 52, 81, 96, 53, 58, 68, 67, 72, 94, 79, 63, 43, 81, 46, 71, 52, 97, 62, 46, 50, 65, 45, 65, 80, 62, 48, 77, 66, 76, 62, 77, 69, 61, 59, 55, 45, 78, 67, 65, 69, 57, 59, 74, 82, 81, 74, 58, 80, 35, 42, 60, 87, 84, 83, 34, 66, 61, 56, 87, 55, 86, 52, 45, 72, 57, 68, 88, 76, 46, 67, 92, 83, 80, 63, 64, 54, 84, 73, 80, 56, 59, 75, 85, 89, 58, 65, 68, 47, 71, 60, 80, 54,

62, 64, 78, 70, 65, 64, 79, 44, 99, 76, 59, 63, 69, 88, 71, 69, 58, 47, 65, 88, 83, 85, 59, 65, 73, 53, 45, 73, 70, 37, 81, 97, 67, 88, 77, 76, 86, 63, 65, 78, 67, 46, 71, 40, 90, 81, 56, 67, 80, 74, 69, 99, 51, 53, 49, 73, 66, 67, 68, 59, 71, 77, 83, 63, 56, 67, 75, 71, 43, 41, 82, 61, 28, 82, 41, 71, 47, 62, 90, 83, 61, 76, 49, 24, 35, 58, 61, 69, 67, 79, 72, 62, 77, 75, 87, 52, 66, 63, 46, 59, 61, 63, 42, 59, 80, 58, 85, 52, 27, 59, 49, 69, 61, 44, 73, 84, 45, 74, 82, 59, 46, 80, 85, 71, 66, 80, 87, 79, 38, 38, 67, 64, 57, 62, 73, 73, 77, 76, 57, 65, 48, 50, 85, 74, 60, 59, 53, 49, 88, 54, 63, 65, 82, 52, 87, 70, 84, 71, 63, 51, 84, 71, 74, 68, 57, 82, 57, 47, 59, 41, 62, 86, 69, 65, 68, 64, 61, 61, 47, 73, 50, 75, 75, 70, 89, 67, 78, 59, 73, 79, 67, 69, 86, 47, 81, 64, 100, 65, 65, 53, 37, 79, 53, 100, 72, 53, 54, 71, 77, 75, 84, 26, 72, 77, 91, 83, 63, 68, 59 }

```
print("Original Numbers:")
PrintNumbers(numbers)

InsertionSort(numbers)

print("\nSorted Numbers:")
PrintNumbers(numbers)
```

Sorting time: 2.79 seconds

Selection Sort Algorithm:
What is it?

- It's a sorting algorithm that runs through all numbers and switches only when it returns to the original number.

Best-case asymptotic notation: O(n2)
Worst-case asymptotic notation: O(n2)
PseudoCode:

```
function SelectionSort(arr):
   n = length of arr
   for i from 0 to n - 2:
      minIndex = i
      for j from i + 1 to n - 1:
         if arr[j] < arr[minIndex]:
            minIndex = j

      swap(arr[minIndex], arr[i])
```

```
numbers = { 72, 69, 90, 47, 76, 71, 88, 40, 64, 38, 58, 40, 65, 78, 50, 69, 88, 18, 46, 54, 66, 65, 44, 69, 74, 73, 69, 67, 70, 62, 69, 63, 56, 40, 97, 81, 74, 50, 75, 57, 55, 58, 53, 59, 50, 65, 55, 66,
```

57, 82, 53, 77, 53, 88, 71, 33, 82, 52, 58, 0, 79, 39, 62, 69, 59, 67, 45, 60, 61, 39, 58, 63, 41, 61, 49, 44, 30, 80, 61, 62, 47, 49, 50, 72, 42, 73, 76, 71, 58, 73, 65, 27, 71, 43, 79, 78, 65, 63, 58, 65, 79, 68, 85, 60, 98, 58, 87, 66, 52, 70, 77, 62, 54, 51, 99, 84, 75, 78, 51, 55, 79, 91, 88, 63, 83, 87, 72, 65, 82, 51, 89, 53, 87, 75, 74, 58, 51, 70, 59, 71, 76, 59, 42, 57, 88, 22, 88, 73, 68, 100, 62, 77, 59, 54, 62, 70, 66, 60, 61, 66, 82, 75, 49, 52, 81, 96, 53, 58, 68, 67, 72, 94, 79, 63, 43, 81, 46, 71, 52, 97, 62, 46, 50, 65, 45, 65, 80, 62, 48, 77, 66, 76, 62, 77, 69, 61, 59, 55, 45, 78, 67, 65, 69, 57, 59, 74, 82, 81, 74, 58, 80, 35, 42, 60, 87, 84, 83, 34, 66, 61, 56, 87, 55, 86, 52, 45, 72, 57, 68, 88, 76, 46, 67, 92, 83, 80, 63, 64, 54, 84, 73, 80, 56, 59, 75, 85, 89, 58, 65, 68, 47, 71, 60, 80, 54, 62, 64, 78, 70, 65, 64, 79, 44, 99, 76, 59, 63, 69, 88, 71, 69, 58, 47, 65, 88, 83, 85, 59, 65, 73, 53, 45, 73, 70, 37, 81, 97, 67, 88, 77, 76, 86, 63, 65, 78, 67, 46, 71, 40, 90, 81, 56, 67, 80, 74, 69, 99, 51, 53, 49, 73, 66, 67, 68, 59, 71, 77, 83, 63, 56, 67, 75, 71, 43, 41, 82, 61, 28, 82, 41, 71, 47, 62, 90, 83, 61, 76, 49, 24, 35, 58, 61, 69, 67, 79, 72, 62, 77, 75, 87, 52, 66, 63, 46, 59, 61, 63, 42, 59, 80, 58, 85, 52, 27, 59, 49, 69, 61, 44, 73, 84, 45, 74, 82, 59, 46, 80, 85, 71, 66, 80, 87, 79, 38, 38, 67, 64, 57, 62, 73, 73, 77, 76, 57, 65, 48, 50, 85, 74, 60, 59, 53, 49, 88, 54, 63, 65, 82, 52, 87, 70, 84, 71, 63, 51, 84, 71, 74, 68, 57, 82, 57, 47, 59, 41, 62, 86, 69, 65, 68, 64, 61, 61, 47, 73, 50, 75, 75, 70, 89, 67, 78, 59, 73, 79, 67, 69, 86, 47, 81, 64, 100, 65, 65, 53, 37, 79, 53, 100, 72, 53, 54, 71, 77, 75, 84, 26, 72, 77, 91, 83, 63, 68, 59 }

```
print("Original Numbers:")
PrintNumbers(numbers)

SelectionSort(numbers)

print("\nSorted Numbers:")
PrintNumbers(numbers)
```

Sorting time: 3.46 seconds

Heap Sort Algorithm:
What is it?

- It's a sorting algorithm that makes Selection sort more efficient using a "tree" rather than a linear structure.

Best-case asymptotic notation: O(n log n)
Worst-case asymptotic notation: O(n log n)
PseudoCode:

```
function HeapSort(arr):
   n = length of arr

   for i from n/2 - 1 to 0 step -1:
```

```
        Heapify(arr, n, i)

    for i from n-1 to 1 step -1:
        swap(arr[0], arr[i])

        Heapify(arr, i, 0)

function Heapify(arr, n, i):
    largest = i
    left = 2*i + 1
    right = 2*i + 2

    if left < n and arr[left] > arr[largest]:
        largest = left

    if right < n and arr[right] > arr[largest]:
        largest = right

    if largest ≠ i:
        // Swap arr[i] and arr[largest]
        swap(arr[i], arr[largest])

        Heapify(arr, n, largest)

numbers = { 72, 69, 90, 47, 76, 71, 88, 40, 64, 38, 58, 40, 65, 78, 50, 69, 88, 18, 46, 54, 66, 65,
44, 69, 74, 73, 69, 67, 70, 62, 69, 63, 56, 40, 97, 81, 74, 50, 75, 57, 55, 58, 53, 59, 50, 65, 55, 66,
57, 82, 53, 77, 53, 88, 71, 33, 82, 52, 58, 0, 79, 39, 62, 69, 59, 67, 45, 60, 61, 39, 58, 63, 41, 61,
49, 44, 30, 80, 61, 62, 47, 49, 50, 72, 42, 73, 76, 71, 58, 73, 65, 27, 71, 43, 79, 78, 65, 63, 58, 65,
79, 68, 85, 60, 98, 58, 87, 66, 52, 70, 77, 62, 54, 51, 99, 84, 75, 78, 51, 55, 79, 91, 88, 63, 83, 87,
72, 65, 82, 51, 89, 53, 87, 75, 74, 58, 51, 70, 59, 71, 76, 59, 42, 57, 88, 22, 88, 73, 68, 100, 62,
77, 59, 54, 62, 70, 66, 60, 61, 66, 82, 75, 49, 52, 81, 96, 53, 58, 68, 67, 72, 94, 79, 63, 43, 81, 46,
71, 52, 97, 62, 46, 50, 65, 45, 65, 80, 62, 48, 77, 66, 76, 62, 77, 69, 61, 59, 55, 45, 78, 67, 65, 69,
57, 59, 74, 82, 81, 74, 58, 80, 35, 42, 60, 87, 84, 83, 34, 66, 61, 56, 87, 55, 86, 52, 45, 72, 57, 68,
88, 76, 46, 67, 92, 83, 80, 63, 64, 54, 84, 73, 80, 56, 59, 75, 85, 89, 58, 65, 68, 47, 71, 60, 80, 54,
62, 64, 78, 70, 65, 64, 79, 44, 99, 76, 59, 63, 69, 88, 71, 69, 58, 47, 65, 88, 83, 85, 59, 65, 73, 53,
45, 73, 70, 37, 81, 97, 67, 88, 77, 76, 86, 63, 65, 78, 67, 46, 71, 40, 90, 81, 56, 67, 80, 74, 69, 99,
51, 53, 49, 73, 66, 67, 68, 59, 71, 77, 83, 63, 56, 67, 75, 71, 43, 41, 82, 61, 28, 82, 41, 71, 47, 62,
90, 83, 61, 76, 49, 24, 35, 58, 61, 69, 67, 79, 72, 62, 77, 75, 87, 52, 66, 63, 46, 59, 61, 63, 42, 59,
80, 58, 85, 52, 27, 59, 49, 69, 61, 44, 73, 84, 45, 74, 82, 59, 46, 80, 85, 71, 66, 80, 87, 79, 38, 38,
67, 64, 57, 62, 73, 73, 77, 76, 57, 65, 48, 50, 85, 74, 60, 59, 53, 49, 88, 54, 63, 65, 82, 52, 87, 70,
```

84, 71, 63, 51, 84, 71, 74, 68, 57, 82, 57, 47, 59, 41, 62, 86, 69, 65, 68, 64, 61, 61, 47, 73, 50, 75, 75, 70, 89, 67, 78, 59, 73, 79, 67, 69, 86, 47, 81, 64, 100, 65, 65, 53, 37, 79, 53, 100, 72, 53, 54, 71, 77, 75, 84, 26, 72, 77, 91, 83, 63, 68, 59 }

```
print("Original Numbers:")
PrintNumbers(numbers)

HeapSort(numbers)

print("\nSorted Numbers:")
PrintNumbers(numbers)
```

Sorting time: 0.47 seconds

Quick Sort Algorithm:
What is it?

- It's a sorting technique that divides large amounts of data into smaller ones to sort it more efficiently.

Best-case asymptotic notation: O(n log n)
Worst-case asymptotic notation: O(n2)
PseudoCode:

```
function QuickSort(arr, low, high):
   if low < high:
      pivotIndex = Partition(arr, low, high)

      QuickSort(arr, low, pivotIndex - 1)
      QuickSort(arr, pivotIndex + 1, high)

function Partition(arr, low, high):
   pivot = arr[high]
   i = low - 1

   for j from low to high - 1:
      if arr[j] < pivot:
         i = i + 1
         // Swap arr[i] and arr[j]
         swap(arr[i], arr[j])

   swap(arr[i + 1], arr[high])
```

```
    return i + 1

numbers = { 72, 69, 90, 47, 76, 71, 88, 40, 64, 38, 58, 40, 65, 78, 50, 69, 88, 18, 46, 54, 66, 65,
44, 69, 74, 73, 69, 67, 70, 62, 69, 63, 56, 40, 97, 81, 74, 50, 75, 57, 55, 58, 53, 59, 50, 65, 55, 66,
57, 82, 53, 77, 53, 88, 71, 33, 82, 52, 58, 0, 79, 39, 62, 69, 59, 67, 45, 60, 61, 39, 58, 63, 41, 61,
49, 44, 30, 80, 61, 62, 47, 49, 50, 72, 42, 73, 76, 71, 58, 73, 65, 27, 71, 43, 79, 78, 65, 63, 58, 65,
79, 68, 85, 60, 98, 58, 87, 66, 52, 70, 77, 62, 54, 51, 99, 84, 75, 78, 51, 55, 79, 91, 88, 63, 83, 87,
72, 65, 82, 51, 89, 53, 87, 75, 74, 58, 51, 70, 59, 71, 76, 59, 42, 57, 88, 22, 88, 73, 68, 100, 62,
77, 59, 54, 62, 70, 66, 60, 61, 66, 82, 75, 49, 52, 81, 96, 53, 58, 68, 67, 72, 94, 79, 63, 43, 81, 46,
71, 52, 97, 62, 46, 50, 65, 45, 65, 80, 62, 48, 77, 66, 76, 62, 77, 69, 61, 59, 55, 45, 78, 67, 65, 69,
57, 59, 74, 82, 81, 74, 58, 80, 35, 42, 60, 87, 84, 83, 34, 66, 61, 56, 87, 55, 86, 52, 45, 72, 57, 68,
88, 76, 46, 67, 92, 83, 80, 63, 64, 54, 84, 73, 80, 56, 59, 75, 85, 89, 58, 65, 68, 47, 71, 60, 80, 54,
62, 64, 78, 70, 65, 64, 79, 44, 99, 76, 59, 63, 69, 88, 71, 69, 58, 47, 65, 88, 83, 85, 59, 65, 73, 53,
45, 73, 70, 37, 81, 97, 67, 88, 77, 76, 86, 63, 65, 78, 67, 46, 71, 40, 90, 81, 56, 67, 80, 74, 69, 99,
51, 53, 49, 73, 66, 67, 68, 59, 71, 77, 83, 63, 56, 67, 75, 71, 43, 41, 82, 61, 28, 82, 41, 71, 47, 62,
90, 83, 61, 76, 49, 24, 35, 58, 61, 69, 67, 79, 72, 62, 77, 75, 87, 52, 66, 63, 46, 59, 61, 63, 42, 59,
80, 58, 85, 52, 27, 59, 49, 69, 61, 44, 73, 84, 45, 74, 82, 59, 46, 80, 85, 71, 66, 80, 87, 79, 38, 38,
67, 64, 57, 62, 73, 73, 77, 76, 57, 65, 48, 50, 85, 74, 60, 59, 53, 49, 88, 54, 63, 65, 82, 52, 87, 70,
84, 71, 63, 51, 84, 71, 74, 68, 57, 82, 57, 47, 59, 41, 62, 86, 69, 65, 68, 64, 61, 61, 47, 73, 50, 75,
75, 70, 89, 67, 78, 59, 73, 79, 67, 69, 86, 47, 81, 64, 100, 65, 65, 53, 37, 79, 53, 100, 72, 53, 54,
71, 77, 75, 84, 26, 72, 77, 91, 83, 63, 68, 59 }

print("Original Numbers:")
PrintNumbers(numbers)

QuickSort(numbers, 0, length of numbers - 1)

print("\nSorted Numbers:")
PrintNumbers(numbers)
```

Sorting time: 0.08 Seconds

Merge Sort Algorithm:
What is it?

- It's an algorithm that uses divide-and-conquer to split up a data set into multiple subclasses and sort them.

Best-case asymptotic notation: O(n log n)
Worst-case asymptotic notation: O(n log n)

PseudoCode:

```
function MergeSort(arr):
    n = length of arr

    if n <= 1:
        return

    middle = n / 2

    left = new array of size middle
    right = new array of size n - middle

    copy elements from arr[0] to arr[middle - 1] to left
    copy elements from arr[middle] to arr[n - 1] to right

    MergeSort(left)
    MergeSort(right)

    Merge(arr, left, right)

function Merge(arr, left, right):
    leftLength = length of left
    rightLength = length of right

    i = 0, j = 0, k = 0

    while i < leftLength and j < rightLength:
        if left[i] <= right[j]:
            arr[k] = left[i]
            i = i + 1
        else:
            arr[k] = right[j]
            j = j + 1
        k = k + 1

    while i < leftLength:
        arr[k] = left[i]
        i = i + 1
        k = k + 1
```

```
    while j < rightLength:
        arr[k] = right[j]
        j = j + 1
        k = k + 1

numbers = { 72, 69, 90, 47, 76, 71, 88, 40, 64, 38, 58, 40, 65, 78, 50, 69, 88, 18, 46, 54, 66, 65,
44, 69, 74, 73, 69, 67, 70, 62, 69, 63, 56, 40, 97, 81, 74, 50, 75, 57, 55, 58, 53, 59, 50, 65, 55, 66,
57, 82, 53, 77, 53, 88, 71, 33, 82, 52, 58, 0, 79, 39, 62, 69, 59, 67, 45, 60, 61, 39, 58, 63, 41, 61,
49, 44, 30, 80, 61, 62, 47, 49, 50, 72, 42, 73, 76, 71, 58, 73, 65, 27, 71, 43, 79, 78, 65, 63, 58, 65,
79, 68, 85, 60, 98, 58, 87, 66, 52, 70, 77, 62, 54, 51, 99, 84, 75, 78, 51, 55, 79, 91, 88, 63, 83, 87,
72, 65, 82, 51, 89, 53, 87, 75, 74, 58, 51, 70, 59, 71, 76, 59, 42, 57, 88, 22, 88, 73, 68, 100, 62,
77, 59, 54, 62, 70, 66, 60, 61, 66, 82, 75, 49, 52, 81, 96, 53, 58, 68, 67, 72, 94, 79, 63, 43, 81, 46,
71, 52, 97, 62, 46, 50, 65, 45, 65, 80, 62, 48, 77, 66, 76, 62, 77, 69, 61, 59, 55, 45, 78, 67, 65, 69,
57, 59, 74, 82, 81, 74, 58, 80, 35, 42, 60, 87, 84, 83, 34, 66, 61, 56, 87, 55, 86, 52, 45, 72, 57, 68,
88, 76, 46, 67, 92, 83, 80, 63, 64, 54, 84, 73, 80, 56, 59, 75, 85, 89, 58, 65, 68, 47, 71, 60, 80, 54,
62, 64, 78, 70, 65, 64, 79, 44, 99, 76, 59, 63, 69, 88, 71, 69, 58, 47, 65, 88, 83, 85, 59, 65, 73, 53,
45, 73, 70, 37, 81, 97, 67, 88, 77, 76, 86, 63, 65, 78, 67, 46, 71, 40, 90, 81, 56, 67, 80, 74, 69, 99,
51, 53, 49, 73, 66, 67, 68, 59, 71, 77, 83, 63, 56, 67, 75, 71, 43, 41, 82, 61, 28, 82, 41, 71, 47, 62,
90, 83, 61, 76, 49, 24, 35, 58, 61, 69, 67, 79, 72, 62, 77, 75, 87, 52, 66, 63, 46, 59, 61, 63, 42, 59,
80, 58, 85, 52, 27, 59, 49, 69, 61, 44, 73, 84, 45, 74, 82, 59, 46, 80, 85, 71, 66, 80, 87, 79, 38, 38,
67, 64, 57, 62, 73, 73, 77, 76, 57, 65, 48, 50, 85, 74, 60, 59, 53, 49, 88, 54, 63, 65, 82, 52, 87, 70,
84, 71, 63, 51, 84, 71, 74, 68, 57, 82, 57, 47, 59, 41, 62, 86, 69, 65, 68, 64, 61, 61, 47, 73, 50, 75,
75, 70, 89, 67, 78, 59, 73, 79, 67, 69, 86, 47, 81, 64, 100, 65, 65, 53, 37, 79, 53, 100, 72, 53, 54,
71, 77, 75, 84, 26, 72, 77, 91, 83, 63, 68, 59 }

print("Original Numbers:")
PrintNumbers(numbers)

MergeSort(numbers)

print("\nSorted Numbers:")
PrintNumbers(numbers)
```

Sorting time: 0.13 seconds