CITY
UNIVERSITY OF LONDON
EST 1894

STEM Digital Academy

School of Science & Technology

www.city.ac.uk

Merge Sort

Programming and Algorithms

Lecture by

Dr Daniil Osudin

```
n = 3
for i in range(1,n+1):
    print("Hello World!")
```

```
Hello World!
Hello World!
Hello World!
```

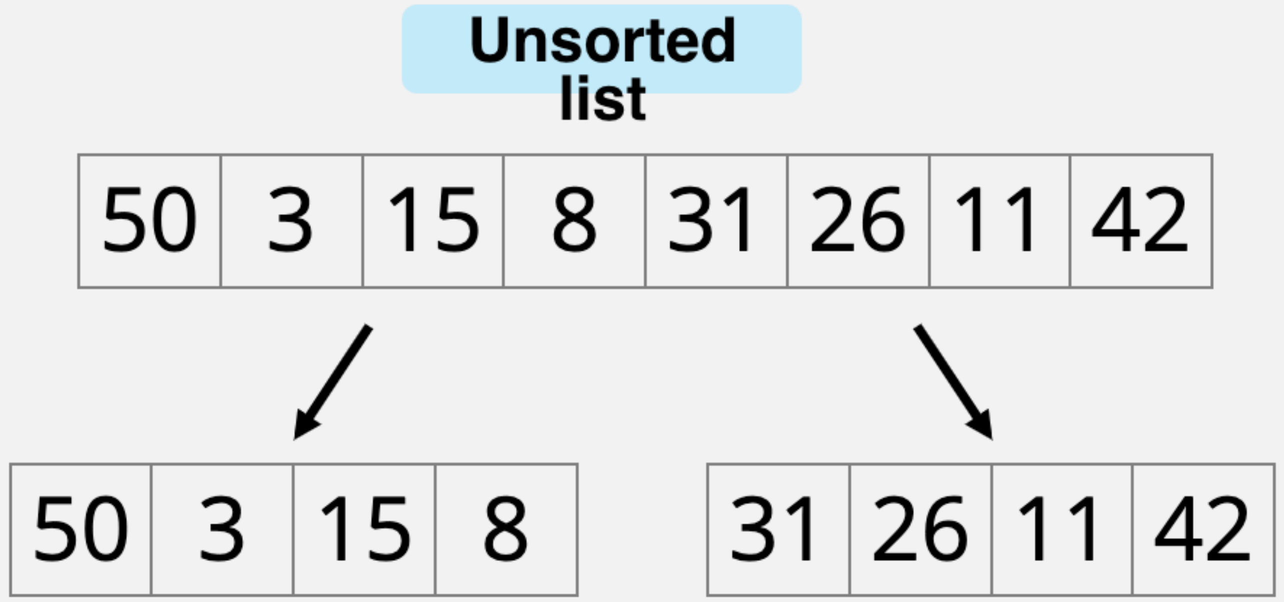
What will we Cover?

- Merge sort algorithm
- Understanding the efficiency of the algorithm using big-O notation

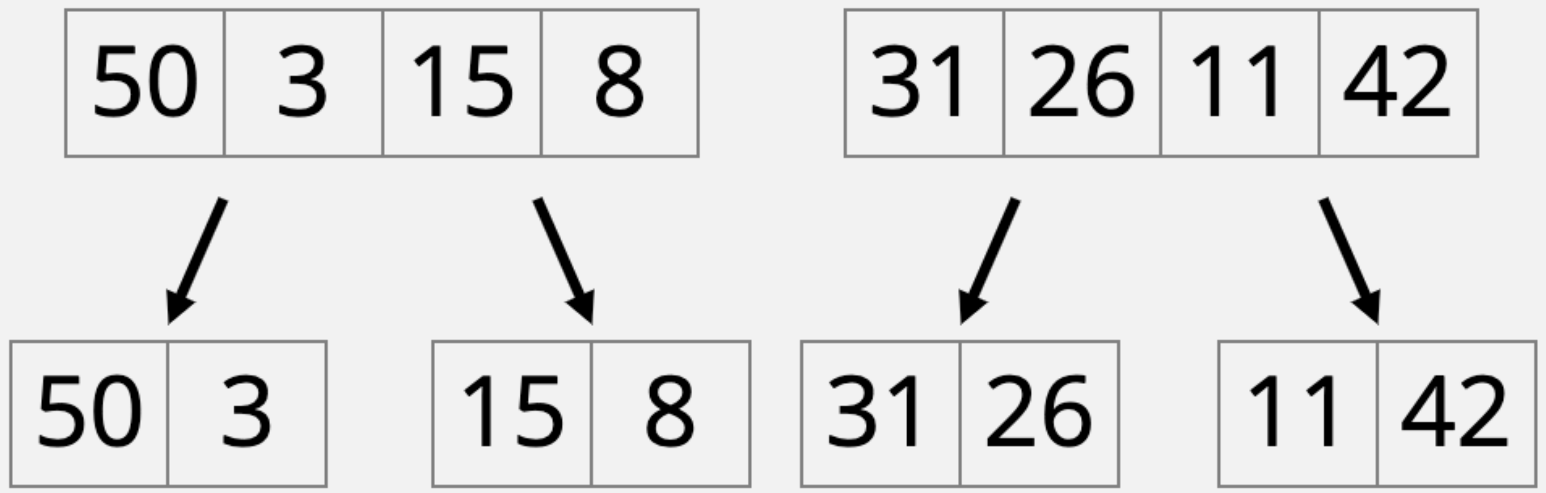
Merge Sort

- Classified as a divide and conquer algorithm
- Divide the initial list into two sub-lists of half the size of the original
- Sort the sub-lists recursively
- Compare the sorted sub-lists and merge them back together

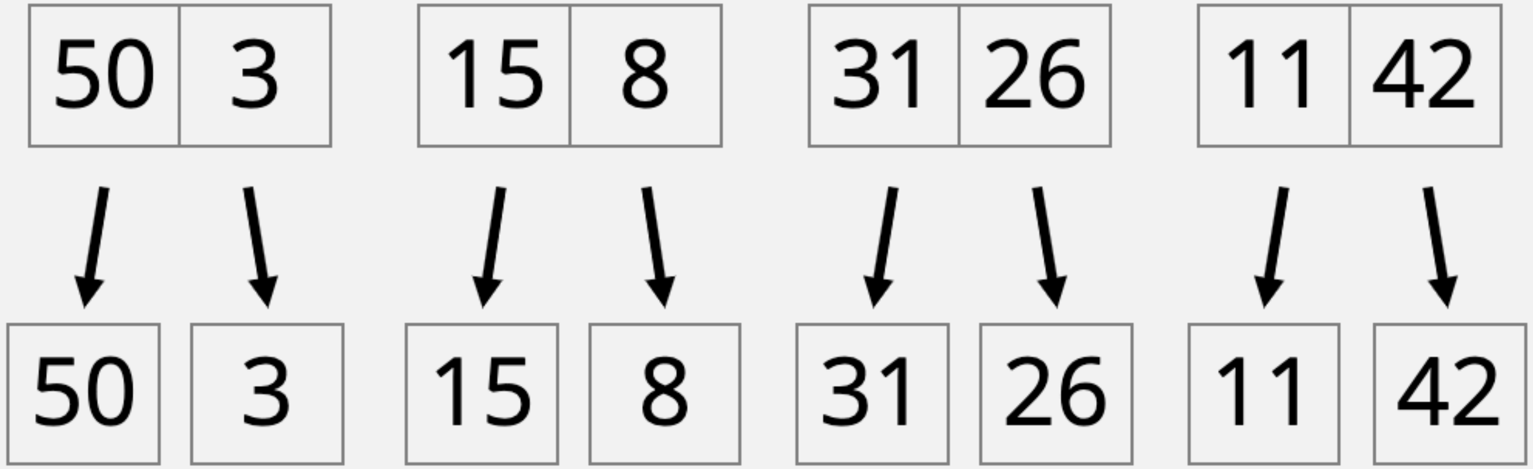
Merge Sort – Divide



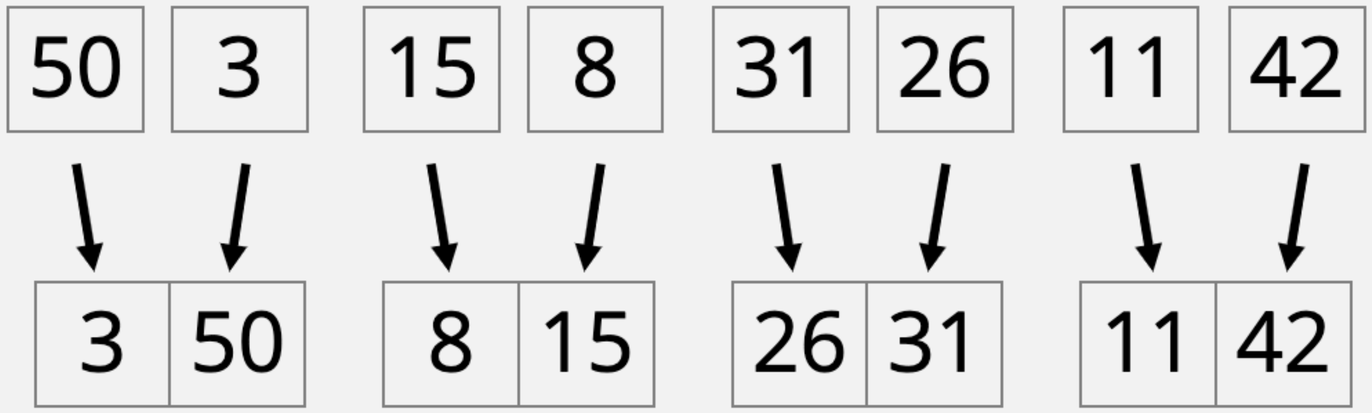
Merge Sort – Divide



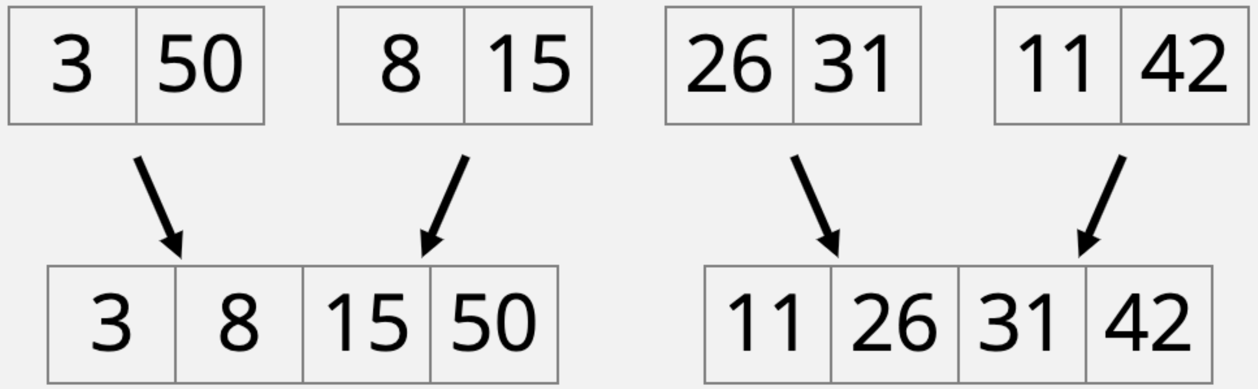
Merge Sort – Divide



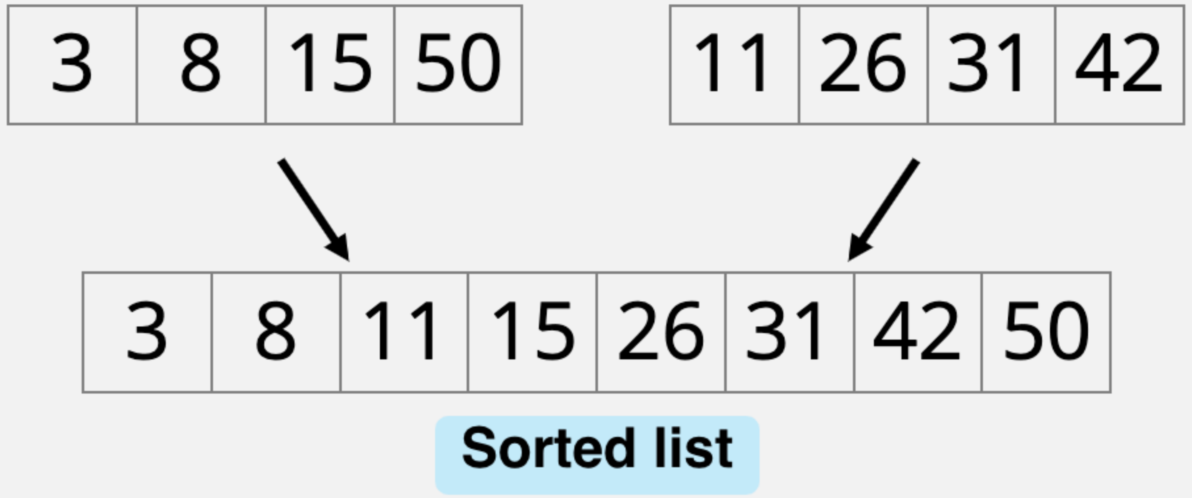
Merge Sort – Conquer



Merge Sort – Conquer



Merge Sort – Conquer

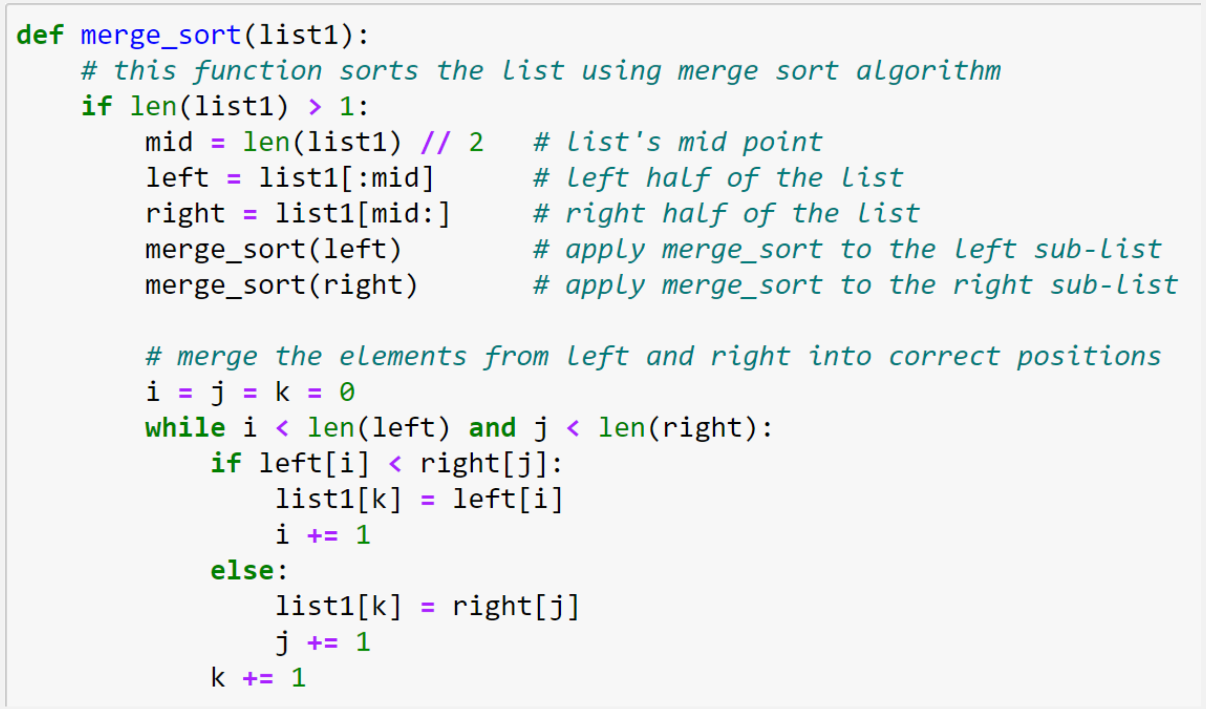


Merge Sort Algorithm

Use merge_sort function repeatedly to divide the list into two halves, sort them and then combine them

1. find mid, left and right
2. merge_sort(left)
3. merge_sort(right)
4. merge(left, right)

Merge Sort Example



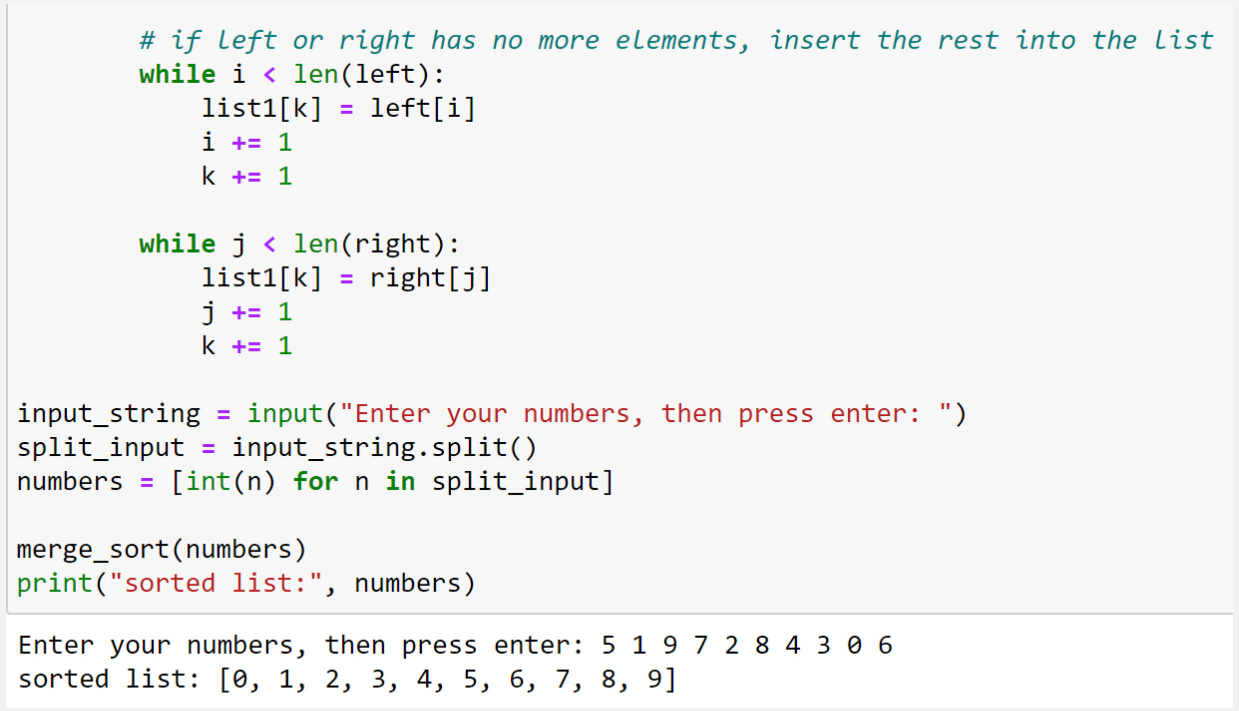
```python
def merge_sort(list1):
    # this function sorts the list using merge sort algorithm
    if len(list1) > 1:
        mid = len(list1) // 2   # list's mid point
        left = list1[:mid]      # left half of the list
        right = list1[mid:]     # right half of the list
        merge_sort(left)        # apply merge_sort to the left sub-list
        merge_sort(right)       # apply merge_sort to the right sub-list

        # merge the elements from left and right into correct positions
        i = j = k = 0
        while i < len(left) and j < len(right):
            if left[i] < right[j]:
                list1[k] = left[i]
                i += 1
            else:
                list1[k] = right[j]
                j += 1
            k += 1
```

…

Merge Sort Example

…



```
        # if left or right has no more elements, insert the rest into the list
        while i < len(left):
            list1[k] = left[i]
            i += 1
            k += 1

        while j < len(right):
            list1[k] = right[j]
            j += 1
            k += 1

input_string = input("Enter your numbers, then press enter: ")
split_input = input_string.split()
numbers = [int(n) for n in split_input]

merge_sort(numbers)
print("sorted list:", numbers)
```

```
Enter your numbers, then press enter: 5 1 9 7 2 8 4 3 0 6
sorted list: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

Analysis

- To sort a list of length n, we keep dividing and sorting the sub-lists of half the size, i.e., 2 O(n/2)
- The loop is O(log n)
- We also perform some additional work to split and combine the lists as well as to compare the values and this is O(n)
- Thus, the big-O notation for merge sort is O(n log n)
- This is an efficient method for sorting

Try It Yourself

Write a program in python environment that takes a string as an input and sorts in alphabetical order using the merge sort algorithm above