

Lecture 16: Algorithms

CS 51P

November 4, 2019

al·go·rithm

/ˈalgəˌrɪθəm/ 

noun

a process or set of rules to be followed in calculations or other problem-solving operations, especially by a computer.

"a basic **algorithm** for division"

Example: Sorting



An aside about memory...

Example: Sorting



Three Possible Sorting Algorithms

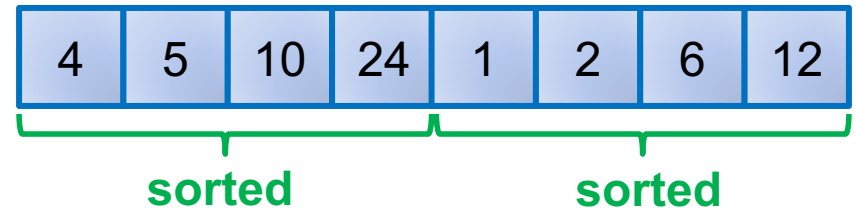
- For each position in the list:
 - Find the object that should be there; put it in the right place

- For each object in the list:
 - If that object should be earlier in the list, put it in the right place

- Recursively:
 - Sort the first half of the list
 - Sort the second half of the list
 - Merge the two halves together

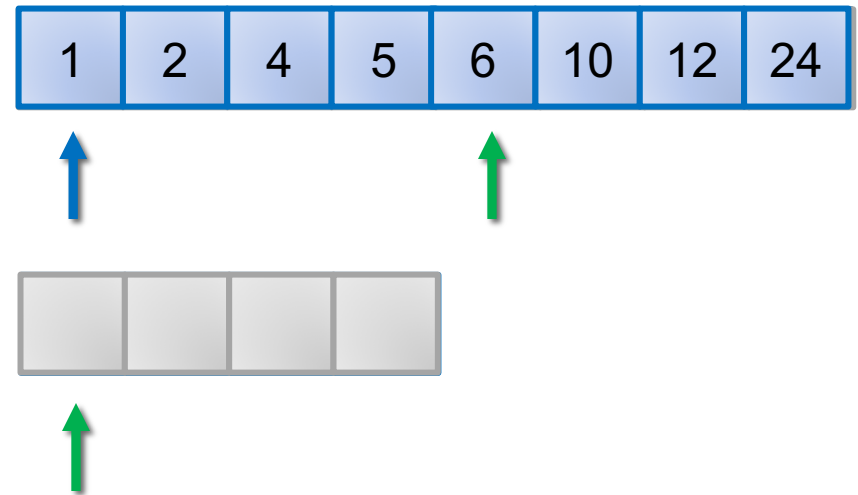
Merging

- What if our list looked like two sorted lists end to end?
- We could sort by merging the two lists!

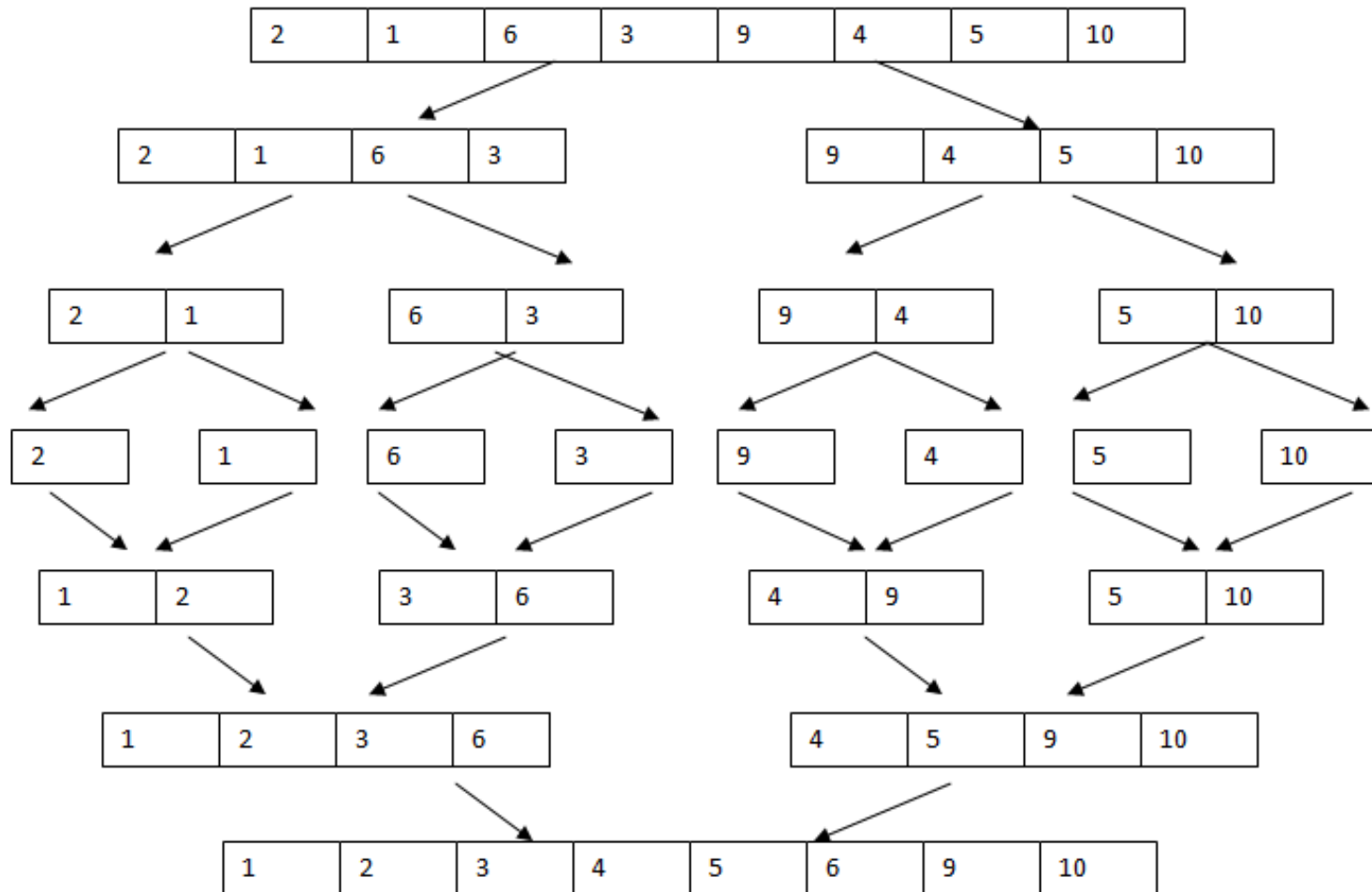


Merging

- What if our list looked like two sorted lists end to end?
- We could sort by merging the two lists!



Mergesort



Sorting Algorithms

Selection Sort

```
def selection_sort(lst):  
  
    # for each pos in list  
    for pos in range(len(lst)):  
  
        # find obj that should be there  
        min_pos = pos  
        for i in range(pos+1, len(lst)):  
            if lst[i] < lst[min_pos]:  
                min_pos = i  
  
        # swap that obj with the one at pos  
        swap(lst, pos, min_pos)
```

Insertion Sort

```
def insertion_sort(lst):  
  
    # for each obj in list  
    for pos in range(len(lst)):  
  
        # move obj to correct position  
        curr_obj = lst[pos]  
        curr_pos = pos  
        while curr_pos > 0 and  
              lst[curr_pos-1] > curr_obj:  
            lst[curr_pos] = lst[curr_pos-1]  
            curr_pos = curr_pos - 1  
        lst[curr_pos] = curr_obj
```

Merge Sort

```
def merge_sort_helper(lst, start, end):  
    # Base Case  
    if end-start < 2:  
        return  
  
    # Recursive Case  
    middle = start + int((end-start) / 2)  
    merge_sort_helper(lst, start, middle)  
    merge_sort_helper(lst, middle, end)  
    merge(lst, start, end)
```

Which algorithm is better?