Wed, Feb. 21, 2024
Scientific Computing

Announcements:

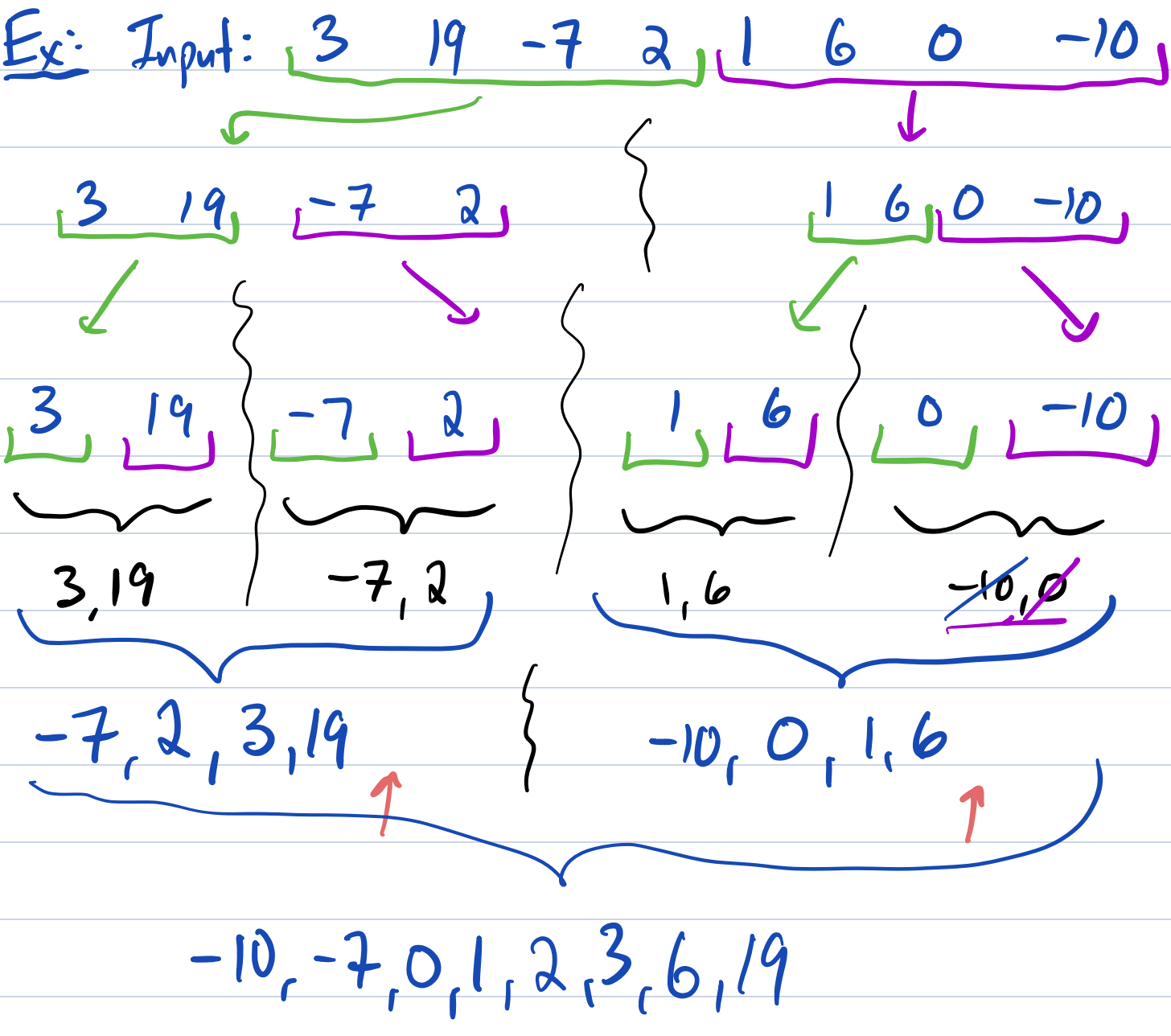
→ HW 2 due on Friday
→ O.H. 2-3 today
10:30-11:30 Thursday } Cudahy 307

Topic 6 - Divide + Conquer (continued)

Sorting a list of n #s
Search space n!
Brute force: $O(n!)$
Insertion Sort: $O(n^2)$
Merge Sort: $O(n \cdot \log(n))$ ← today

Divide + Conquer: (1) Split the input in half
(2) Sort each half (with D+C)
(3) Combine the two sorted half-lists

into one big sorted list.



Pseudocode

```
function merge_sort(Q):   # Q is a list of numbers
    if |Q|=1:
        return Q
    L = left half of Q          L=[3]
    R = right half of Q         R=[19]
    L = merge_sort(L)  } sort each half individually, recursively   L=[3]
    R = merge_sort(R)  }                                            R=[19]

    new_list = []
    while |L|+|R| > 0:
        take L[0] or R[0], whichever is smaller, remove it, and append it to new_list
    return new_list
```

How the computer does this:

merge_sort([3, 19, 2, -7])

- merge_sort([3, 19]) (L)
  - merge_sort([3]) (L)
    - returns [3]
  - merge_sort([19]) (R)
    - returns [19]
  - combines + returns [3, 19]
- merge_sort([2, -7]) (R)
  - merge_sort([2]) (L)
    - returns [2]
  - merge_sort([-7]) (R)
    - returns [-7]
  - combines + returns [-7, 2]

combines + returns [-7, 2, 3, 19]

4 things

Pseudocode

```
function merge_sort(Q):   # Q is a list of numbers
    if |Q|=1:
        return Q
    L = left half of Q
    R = right half of Q
    L = merge_sort(L)
    R = merge_sort(R)

    new_list = []
    while |L|+|R| > 0:
        take L[0] or R[0], whichever is smaller, remove it, and append it to new_list
    return new_list
```

[3,19]

Pseudocode

```
function merge_sort(Q):   # Q is a list of numbers
    if |Q|=1:
        return Q
    L = left half of Q
    R = right half of Q
    L = merge_sort(L)
    R = merge_sort(R)

    new_list = []
    while |L|+|R| > 0:
        take L[0] or R[0], whichever is smaller, remove it, and append it to new_list
    return new_list
```

[3]

Pseudocode

```
function merge_sort(Q):   # Q is a list of numbers
    if |Q|=1:
        return Q
    L = left half of Q
    R = right half of Q
    L = merge_sort(L)
    R = merge_sort(R)

    new_list = []
    while |L|+|R| > 0:
        take L[0] or R[0], whichever is smaller, remove it, and append it to new_list
    return new_list
```