

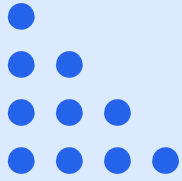
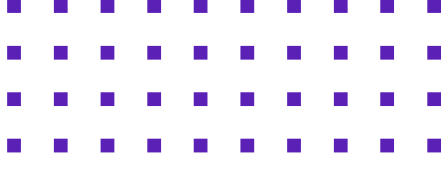
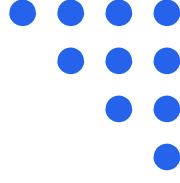


# Greedy Algorithms with Sorting

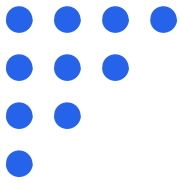
<https://usaco.guide/silver/greedy-sorting>



**CP Initiative**  
joincpi.org



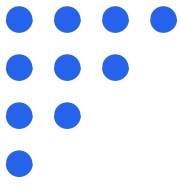
# Greedy and Sorting



- Recall that a greedy algorithm is one that repeatedly uses a local optimum.
  - For example, consider the Coin Change Problem with US denominations.
    - The local optimum is the largest coin you can take, which is repeatedly taken in greedy problems.
- In certain greedy problems, sorting the given input by some quantity will make it easy to see what that local optimum is.

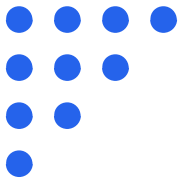
# Greedy with Sorting Example Problem 1

[Codeforces - Studying Algorithms](#)



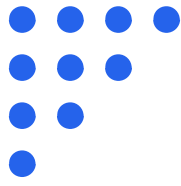
# Solution Sketch

- Intuitively, you want to study the least-time algorithms first.
- Sorting the algorithms by their time-to-learn makes this easy to do.
- $O(n \log n)$



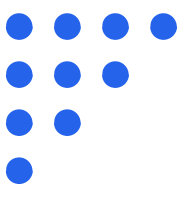
# Solution Code

[USACO Guide - Studying Algorithms](#)



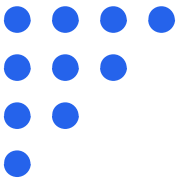
# Greedy with Sorting Example Problem 2

[CSES - Movie Festival](#)



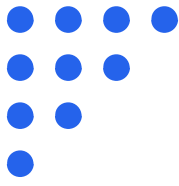
# Solution Sketch

- Greedily choosing intervals in order of start time won't work.
- But greedily choosing intervals in order of end time DOES work.
  - The end time is the “important” quantity because it determines how early until we are free to choose another movie.
- Sort the intervals by their end times, iterate over them in that order, and use them whenever possible.



# Movie Festival Solution

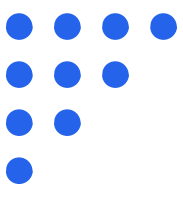
[USACO Guide - The Scheduling Problem](#)





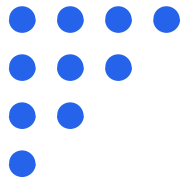
# Greedy with Sorting Example Problem 3

[CSES - Ferris Wheel](#)



# Solution Sketch

- Greedily try to assign the heaviest and lightest children to a gondola.
- If this isn't possible, assign only the heaviest child to a gondola.
- Repeat this process until finished.



# Greedy with Sorting Challenge Problems

[Stick Divisions](#)

[USACO - Berry Picking](#)

