1. Fractional Knapsack problem
2. **Interval overlap detection** :

An interval is represented as a combination of start time and end time. Given a set of intervals, check if any two intervals overlap.Think about brute force O(n2) and O(nlogn).Print the overlapping intervals.

Hint - Brute force : Consider every pair of intervals and check if the pair overlaps or not. The time complexity of this solution is O(n2).

Sorting : Sort all intervals in increasing order of left end (start time). This step takes O(nLogn) time.In the sorted array, if start time of an interval is less than end of previous interval, then there is an overlap. This step takes O(n) time.(Do it by sorting the finish times).

Follow up : Merge all overlapping intervals into one and output the result which should have only mutually exclusive intervals.(Sort by start time, use a stack for holding non-overlapping intervals, keep updating the top’s finish time if the start of next interval is less than the finish time of top interval.Now do without stack by sorting start times in descending order.

<http://www.geeksforgeeks.org/merging-intervals/>

**Interval Partitioning in case of overlap/Optimal Partitioning:** Find the point of maximum overlap and the number of overlaps at that point(depth ‘d’ of the set of intervals is defined as the maximum no of intervals overlapping at any given point.) This can also be phrased as create minimum number of partitions to hold all the intervals such that intervals within a partition are pairwise non-overlapping while intervals of different partitions may overlap. It can be proved that exactly ‘d’ such partitions are required. Two approaches :

*[Approach 1:]*. Keep two sorted arrays(left or start and right or finish). Traverse to check number of overlaps present at any point, print point of max overlap.

<http://www.geeksforgeeks.org/find-the-point-where-maximum-intervals-overlap/)>

Eg. Given arrival and departure time of guests, find the time when maximum number of guests are present in the party.(use two sorted arrays start, end.).DP approach.

Eg 2: Given arrival and departure times of all trains that reach a railway station, find the minimum number of platforms required for the railway station so that no train waits

<http://www.geeksforgeeks.org/minimum-number-platforms-required-railwaybus-station/>

*[Approach 2:]* Use Priority Queue(min heap) : nlogn worst case time.d=number of insertions in the PQ. See classroom scheduling problem : Given start and finish times of lectures, find the minimum number of classrooms required to schedule “all” classes.(minimum d classrooms required to schedule all classes, =depth of intervals).

<https://www.youtube.com/watch?v=i_G8hZYcKnI>

<https://www.youtube.com/watch?v=0o4kwIU7_aE>

<http://algorithmsandme.in/2016/05/interval-partitioning-greedy-algorithm/>

Find intersections of all overlapping intervals.

Interval tree : Given n intervals, print all overlapping intervals. Give nlogn solution using sorting. Give interval tree solution as well.

**Find the largest set(max-sized) of pairwise non-overlapping intervals**.

(Each interval has left and right end. Sort by right end,include the first interval in the result. Now process from the 2nd interval by comparing with the latest interval for overlap. Sorting by left end not optimal because the leftmost intervals may be large enough to overlap with next intervals, thereby excluding them from being taken if selected.Comparing with last interval selected b/c if ith one doesn’t overlap with i-1, it won’t overlap with i-2,i-3,etc as well. This is concept is used in interval scheduling maximization problem.

1. Activity/Interval scheduling problem(each activity having equal priority/profit) - greedy.
2. Activity/Interval scheduling problem(priorities/profits can vary, i.e each activity has some profit associated with it) - Here profit earned has to be optimized. So greedy won’t work. DP in the order of sorted finish time.
3. Job sequencing with deadlines to maximize profit earned(duration=1, varying profit if completed before deadline)- greedy/disjoint set approach. Make n time slots,n=max deadline. Sort by profit.Start processing in sorted order, allocating the rightmost slot available, skip if not available. Count the total profit made. Some jobs will miss their deadlines resulting in zero profit.

<https://www.youtube.com/watch?v=yHsDLU3ZqNM>

1. Job sequencing with deadlines to Minimize Maximum Lateness(durations vary)- Since all jobs to be done.

Sort by deadline. Process in sorted order, keep adding lateness , lateness(i)=max(0,finish-deadline). finish=start+duration given. Job 1 starts at t=0.<https://www.youtube.com/watch?v=zH4z15UpABs>

1. Job sequencing with deadlines(duration and profit both vary) - Will this combination be asked ? B/C if we want to maximize the profit then some jobs may miss their deadline b/c more profitable tasks will be preferred over less profitable tasks(which may miss their deadline resulting in zero profit in which case we’ll be better off without doing them). On the contrary, if we are interested in finishing all the tasks without missing deadlines or missing them with as small margin as possible, profit shouldn’t be of any concern to us.
2. **OptaPlanner :** Optimal room count and sizes for N overlapping Meeting Schedules-Given N weighted (Wi) and possibly overlapping intervals (representing meeting schedules) , find the minimum number "&" capacity of meeting rooms needed to conduct all meetings.http://stackoverflow.com/questions/24657695/optimal-room-count-and-sizes-for-n-overlapping-meeting-schedules
3. Selecting breakpoints-truck driver’s algorithm. Do it.
4. Stable marriage/matching problem(SMP) very interesting.
5. Stable room mate problem (SRP).