# Spring 219

## Question 1

## Question 2

Suppose you have one machine and n jobs,a1, . . . , an. Each job aj has processing time tj, profitpj, and deadline dj. The machine can only process one job at a time and that job must run uninterruptedly until completion. If job aj is completed by deadline dj, you receive profit pj, but if it is completed after, you receive nothing. Assuming all processing times are integers between 1 and n and dj ≥ tj for all jobs, give an algorithm for computing the maximum profit you can make.What is the run time of your algorithm? Justify the run time.

<https://walkccc.github.io/CLRS/Chap16/Problems/16-2/>

## Question 3

Suppose that a graph G has a minimum spanning tree (MST) already computed. What is the runtime complexity of updating the MST if you add a new vertex and incident edges to G? Justify the complexity.

# Fall 2018

## Question 1

A forensic lab receives a delivery of n samples. They look identical, but in fact, some of them have different chemical composition. There is a device that can be applied to two samples and tells whether they are different or not. It is known in advance that most of the samples (more then 50%) are identical. Find one of the samples in the majority making no more than n comparisons. (Beware: it is possible that two samples are identical but do not belong to the majority of identical samples.)

<https://studylib.net/doc/18212366/homework-4>

## Question 2

Draw a single binary tree T such that

a) each node of T stores a single character from {A,E,F,M,N,U,X},

b) a preorder traversal of T yields EXAMFUN, and

c) a inorder traversal of T yields MAFXUEN.

<https://unihub.files.wordpress.com/2013/04/w05.pdf>

<http://www.cse.unsw.edu.au/~cs2011/05s1/tut/solutions/tut06sol.html>

## Question 3

Each of n keys in an array may have one of the values red, white or blue. Give a linear time algorithm for rearranging the keys so that all the reds come before all the blues and all the blues come before all the whites. The only operations permitted on the keys are

* Examination of a key to find out what color it is.
* A swap (interchange of positions) of two keys specified by their indices.

<https://stackoverflow.com/questions/15186953/an-algorithm-to-sort-an-array-in-linear-time>

<https://www.geeksforgeeks.org/sort-an-array-of-0s-1s-and-2s/>

<https://en.wikipedia.org/wiki/Dutch_national_flag_problem>

<https://www.youtube.com/watch?v=BOt1DAvR0zI>

# Spring 2018

## Problem #1

## Problem #2

As your reward for saving the Kingdom of Bigfunnia from the evil “Exponential symptotic" the king has given you the opportunity to earn some “big cash". Behind the castle (near the hot tub) there is a maze. Along each corridor of the maze there is a bag of gold coins. The amount of gold in each bag varies. You will be given the opportunity to walk through the maze, picking up bags of gold. You may enter only through the door marked “ENTER" and exit through the door marked “EXIT". (These are separate doors). While in the maze you may not retrace your steps. Each corridor of the maze has an arrow painted on the wall. You may only go down the corridor in the direction of the arrow. There is no way to traverse a “loop" in the maze. You will receive a map of the maze, including the amount of gold in and the direction of each corridor. Describe an algorithm to help you pick up the most gold?

## Problem #3

# Spring 2017

## Question 1

You are given an array A of n distinct integers and another integer x. Give an O(nlogn)-time algorithm to determine whether or not there exists two elements in A whose sum is exactly x

<https://ita.skanev.com/02/03/07.html>

<http://www.cse.unsw.edu.au/~cs2011/05s2/Assignment2-answers.pdf>

## Question 2

VLSI databases commonly represent an integrated circuit as a list of rectangles. Assume that each rectangle is axis aligned (sides parallel to the x- and y-axes), so that we represent a rectangle by its minimum and maximum x- and y-coordinates. Give an O(nlogn)-time algorithm to decide whether or not a set of n rectangles so represented contains two rectangles that overlap. Your algorithm need not report all intersecting pairs, but it must report that an overlap exists if one rectangle entirely covers another, even if the boundary lines do not intersect.

(1) Start with an empty set of active intervals called T, let this be an Interval Tree, e.e. Red-Black Tree (making search time O(logn)

(2) Sort the segment endpoints from left to right. Break any ties by using point with lower y-coordinate first. We can use something like merge sort which O(nlogn) which will set the upper-bound of the algorithm

(3) For each point *p* in the sorted list

(4) if *p* is a left endpoint of a segment *s* then

(5) QUERY\_INTERSECTION(T, s); return TRUE if there are

(6) INSERT(*T,s*)

(7) else

(8) QUERY\_INTERSECTION(T, s); return TRUE if there are

(9) DELETE(*T, s*)

(10) return FALSE

<https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=2&ved=2ahUKEwjg2sWx_oHmAhVB7qwKHWzcC58QFjABegQIAxAC&url=https%3A%2F%2Fcourses.csail.mit.edu%2F6.046%2Ffall01%2Fhandouts%2Fps6sol.ps&usg=AOvVaw0chxumJSunUU7AiZNHIkbM>

<https://cw.fel.cvut.cz/b181/_media/courses/cg/lectures/09-intersect.pdf>

<http://www.cs.kent.edu/~dragan/AdvAlg/CG.pdf>

<https://en.wikipedia.org/wiki/Total_order>

https://www.hackerearth.com/practice/math/geometry/line-sweep-technique/tutorial/

## Question 3

**The transitive closure of a graph is a graph which contains an edge (u, v) whenever there is a directed path from node u to node v. Design an O(V E)-time algorithm for computing the transitive closure of a directed graph G = (V, E).**

We can determine the vertices reachable from a particular vertex in O(V+E) time using any basic graph searching algorithm. Thus we can compute the transitive closure in O(VE+V^2) time by searching the graph with each vertex as the source.

If ∣V∣=O(E), we're done as VEVEVE is now the dominating term in the running time bound. If not, we preprocess the graph and mark all degree-0 vertices in O(V+E) time. The rows representing these vertices in the transitive closure are all 0s, which means that the algorithm remains correct if we ignore these vertices when searching. After preprocessing, ∣V∣=O(E) as ∣E∣≥∣V∣ / 2. Therefore searching can be done in O(VE) time.

<https://walkccc.github.io/CLRS/Chap25/25.2/#252-8>

<https://walkccc.github.io/CLRS/Chap25/Problems/25-1/>

<https://www.geeksforgeeks.org/transitive-closure-of-a-graph-using-dfs/>

[www.cnblogs.com/william-cheung/p/3677469.html](http://www.cnblogs.com/william-cheung/p/3677469.html)