Topics that might be on the third examination on November 22 2022

Repeated topics from the second test study guide:

Asymptotic complexity classes.
A few time complexity and recurrence problems, since not everyone has mastered them.
Be able to write pseudocode:
1. Floyd Warshall algorithm.
2. Bellman Ford algorithm.

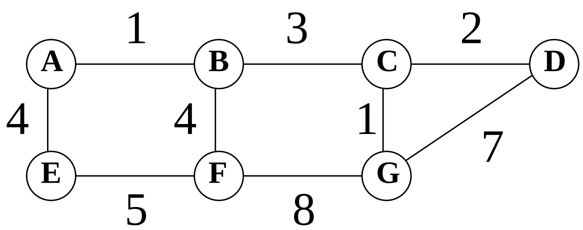
Be able to step through Dijkstra's algorithm for a small graph.
Johnson's algorithm.
Given a figure showing a weighted directed graph with some negative weights, update the weights on edges so that there are no negative weights.
What is the worst-case time complexity for each of those four algorithms?
Dynamic Programming, possibly with memoization.
Find components of an undirected graph using union/find.

Additional topics.

Simple True/False and fill in the blanks questions.
Classes of data structures (search structure, priority queue, etc.)
Implementation of data structures using linked lists and arrays.
Dynamic programming.
Finding strong components of a directed graph using DFS.
Cuckoo hashing.
All kinds of hashing.
Dynamic programming.
Graham scan and amortized analysis.
Median of medians algorithm for selection.
Compressing higher dimensional arrays (including arrays that are not rectangular) into one dimension in main memory.
Kruskal's algorithm using union/find.
A* algorithm.
Sparse structures.
What are the practical implications of a problem being NP-complete?

Example Problems

1. Find a minimum spanning tree of the weighted graph shown below.

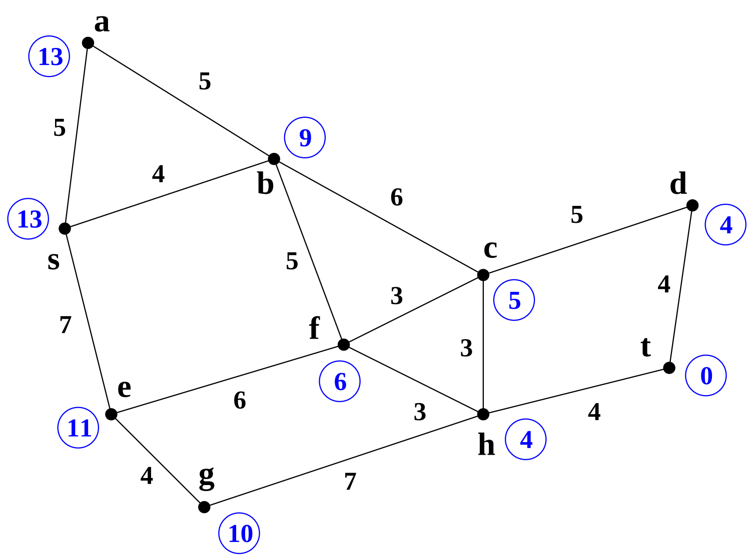


   Use union-find, and show steps.

2. Use the A* algorithm to find the shortest path from **s** to **t** in the weighted graph G shown below. Use the heuristic which is indicated by the circled numbers.

   G can be considered to be a directed graph, where each edge is replaced by two directed edges, one in each direction, both the the same weight. Thus, the directed version of G has 26 directed edges.

   Adjust each directed edge, using the give heuristic, and show the modified weighted graph. Then, find the shortest path from **s** to **t** using the modified weights.



3. Explain how you would use a search structure to implement a sparse array. Your explanation should include the following:

   (a) How array entries are represented.

   (b) How you would set all items in the array to the default value. (Assume that the default value is zero.)

   (c) How you would implement Store.

(d) How you would implement Fetch.

You do not have to write C++ code or pseudocode. If you write more than 20 words for any of the above three explanations, you are probably not answering the question correctly.

4. What does this function compute? ________ (That space is enough to hold the correct answer.)

```
int mystery(int n)
  // input condition: n >= 0
 {
  if(n == 0) return 1;
  else if(n%2) // n is odd
   return 2*mystery(n-1);
  else
   {
    int temp = mystery(n/2);
    return temp*temp;
   }
 }
```

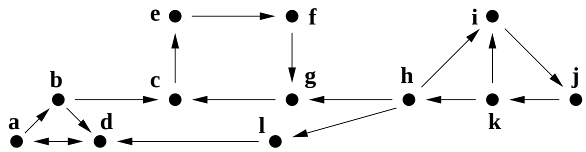
5. A three dimensional $10 \times 8 \times 6$ array `X` is stored in row-major order. The indices range from 0 to 9, 1 to 8, and 0 to 5, respectively. Each item in the array occupies two addressable memory locations. The first item in the array, `X[0,1,0]`, starts at address 1024. Where will the entry `X[5,3,2]` be stored?

6. The *rod-cutting* problem is a classic dynamic programming problem, explained on the webpage https://www.techiedeligh cutting/

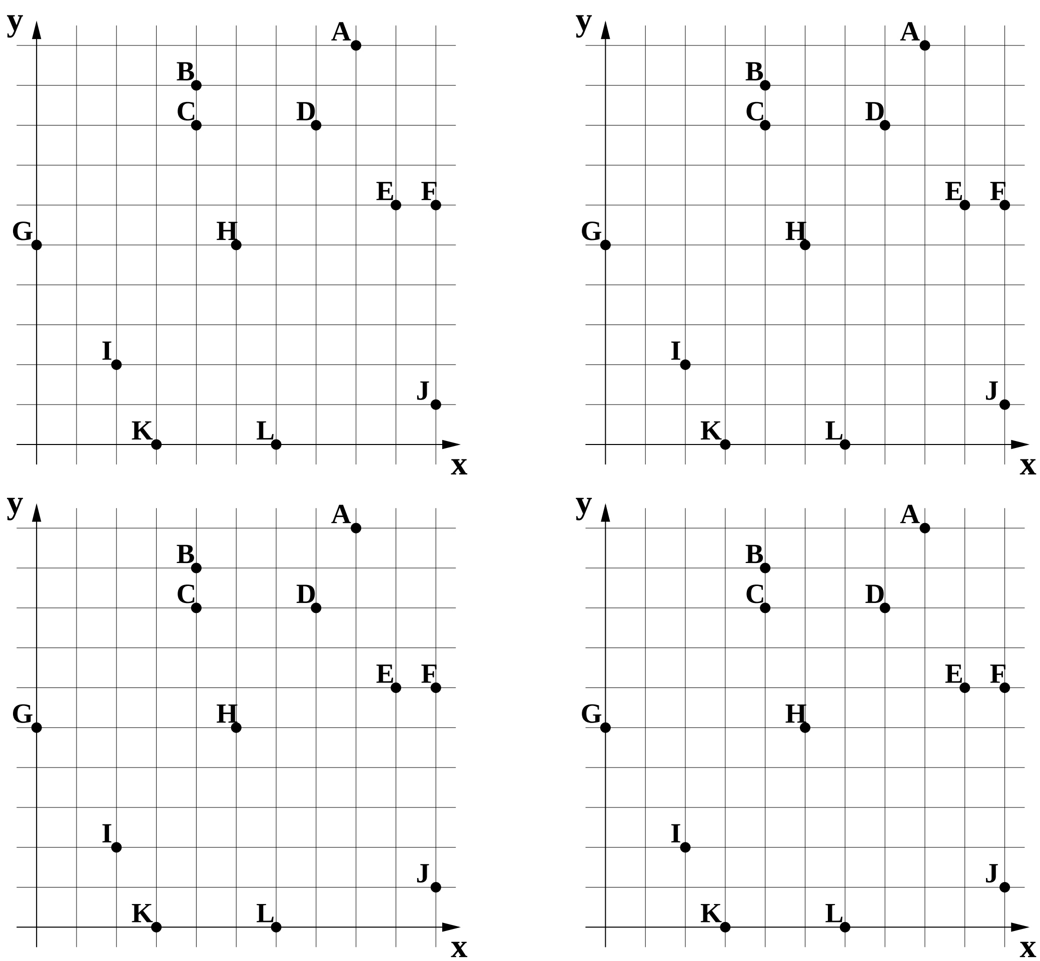
You are given one rod of length N. You want to cut the rod into smaller rods of integer length in order to maximize the total value of the pieces.

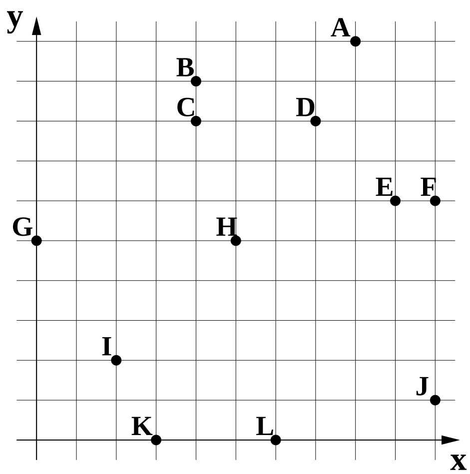
You are given an array V, where $P[i]$ is the price at which you can sell a rod of length i, for all integers $1 \leq i \leq N$. Assume $V[i] > 0$ for all i. Write a dynamic program which computes the way to cut the rod so as to maximize your revenue. The time complexity of your algorithm should be $O(N^2)$.

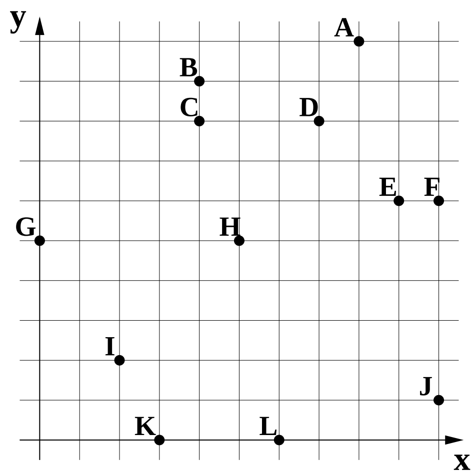
7. Find the strong components of the following graph, using DFS search. Circle the strong components.

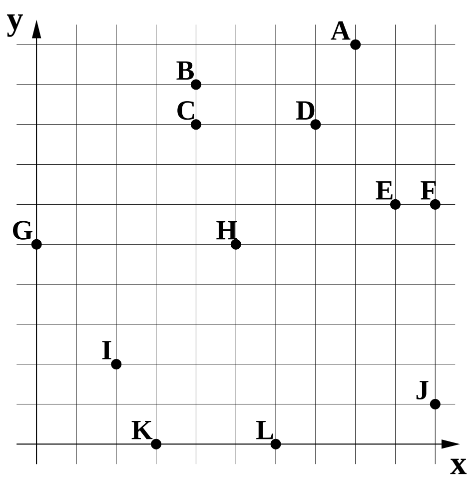


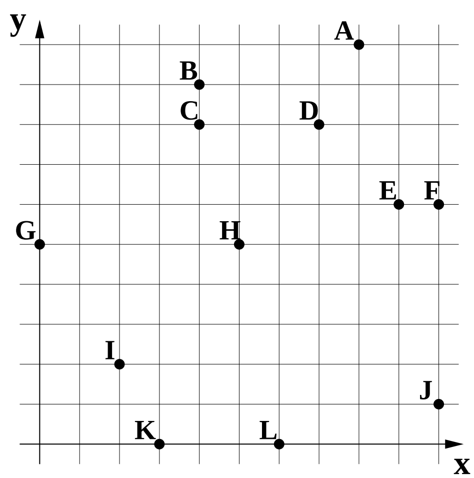
8. Walk through Graham Scan to find the convex hull of the set of points {**A,B,C,D,E,F,G,H,I,J,K,L**} in the plane, as shown in the figure. Use the space below, and as many copies of the figure as you need, to show the steps.

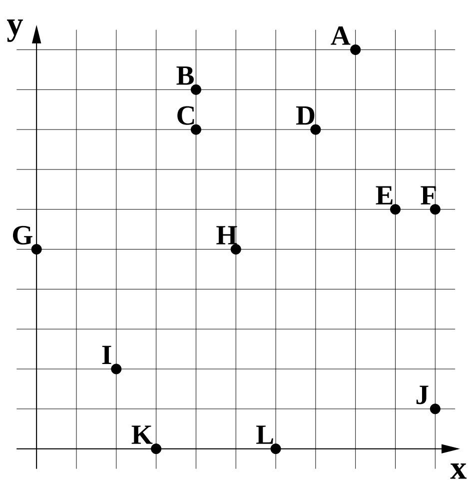


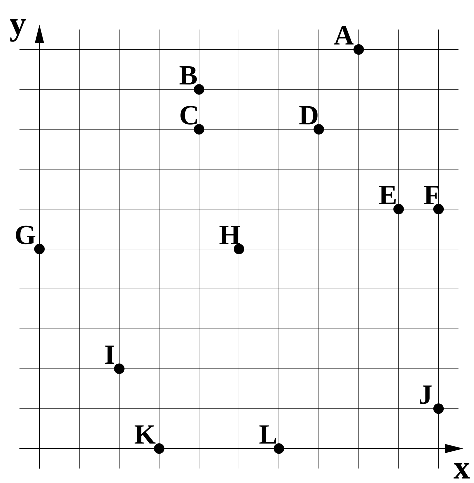

y
A
B
C
D
E
F
G
H
I
J
K
L
x

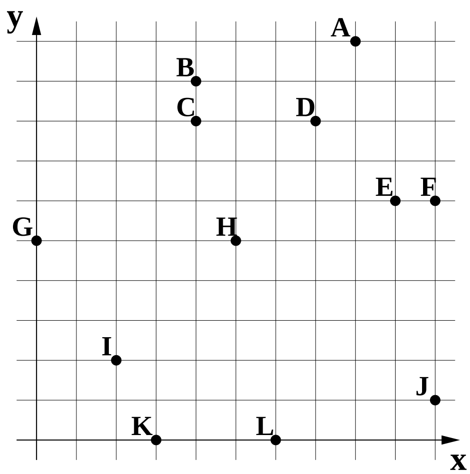
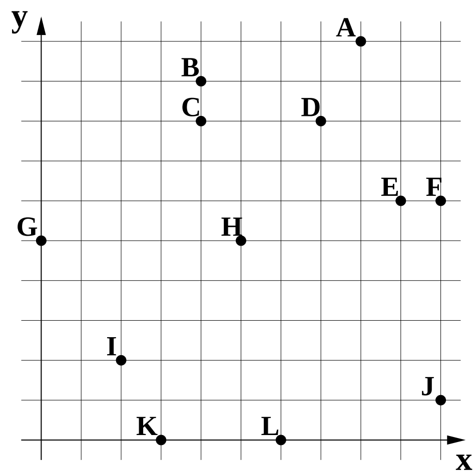
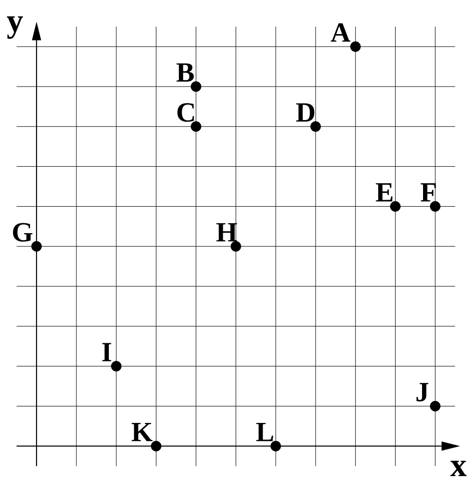
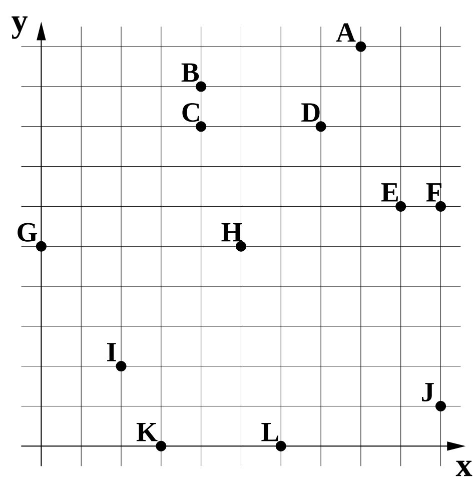
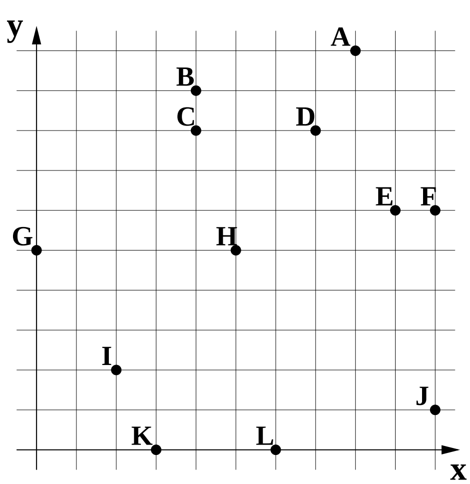
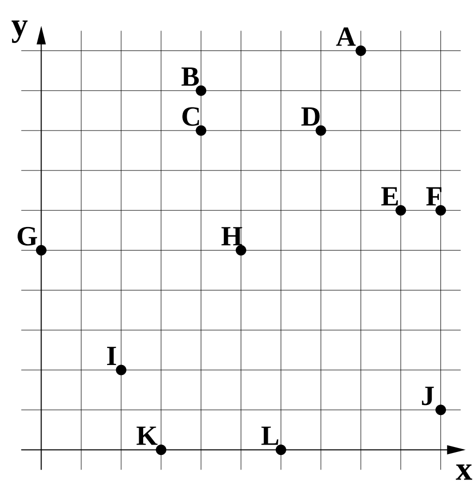

y
A
B
C
D
E
F
G
H
I
J
K
L
x


y
A
B
C
D
E
F
G
H
I
J
K
L
x


y
A
B
C
D
E
F
G
H
I
J
K
L
x


y
A
B
C
D
E
F
G
H
I
J
K
L
x


y
A
B
C
D
E
F
G
H
I
J
K
L
x


y
A
B
C
D
E
F
G
H
I
J
K
L
x

y
A
B
C
D
E
F
G
H
I
J
K
L
x

y
A
B
C
D
E
F
G
H
I
J
K
L
x

y
A
B
C
D
E
F
G
H
I
J
K
L
x

y
A
B
C
D
E
F
G
H
I
J
K
L
x

y
A
B
C
D
E
F
G
H
I
J
K
L
x