Official, Imperial Final Examination Review Worksheet

by order of Darth Wolfman

March 7, 2000

Sources have uncovered an Unofficial, bootleg, underground, blackmarket and generally hush-hush final review worksheet circulated. All rebels involved in the illegal and invalid “worksheet” have been sentenced to a lifetime of hard labor in the Graph Mines of Dijkstra-3. However, Darth Wolfman, in his infinite wisdom and patience understands that students may wish an acceptable worksheet, to prepare for the final examination. Therefore, he has proposed the following Official Worksheet for those students who understand the folly of the rebel cause.

Let the following be known... officially:

- The content and style of questions of this worksheet are intended to prepare you for the final examination.
- Some questions on this worksheet may be more open-ended than related questions on the final.
- The final exam will be comprehensive, and will cover topics from the entire quarter.
- The final will focus on topics after the midterm. Note that we did not officially finish Splay Trees before the midterm.
- Do not underestimate the power of the Dark Side.
- To make going through this Official Worksheet easier, and to make things more like the final exam, distracting (and treasonous) pro-rebel Star Wars references will be kept to a minimum for the remainder of this worksheet. See how considerate the Imperial Forces are of your needs?

1 Misconceptions

You have constructed a library of data structures and algorithms for a client; unfortunately, your client turns out to have no idea how to use the things. Explain why each of the following ideas your client has is a misconception and correct the idea. You must contradict something in the quote. Hypothetically, if this were on a final exam, each question would be worth 2 pts.

1.1

On Dijkstra's algorithm to find the Single-Source Shortest Path for a graph: "I don't want to use Dijkstra's Algorithm because it's a greedy algorithm; greedy algorithm can get stuck in locally-optimal solutions, thereby missing the truly optimal solution."

1.2

On Leftist heaps: "I need a priority queue with merge, but I also need an efficient decreaseKey. I can't use Leftist Heaps because they just implement the priority queue with merge ADT. Besides, I'm not using no stinkin' commie data structures." (you might be able to find two misconceptions here, as well as a bit of prejudice...)

1.3

On hashing with large data sets: "I'd like to use some kind of hashing scheme, because I like their constant-time expected performance. But, I have way more data than can fit in RAM, and hash tables can't make good use of disk blocks (in the way that B-Trees can)."

1.4

On improved time bounds for heaps: "You said that your heap provides $O(\log n)$-time Insert and DeleteMin operations. Well, I stayed up all night and re-wrote the data structure in machine code, so that it's 3 times faster. So, now it does the operations way better than $O(\log n)$ time."

2 Graphology

Having retired from tennis, Steffi Graf has re-immersed herself in her old fondness for computer science. She runs a retraining center, to retrain people who are knowledgeable in one discipline to another one. She started with computer science, of course, but has branched out (ahem) to other fields.

The following chart shows the cost of the course to retrain from one discipline to another. Keep in mind that, as students, you only have a finite amount of money (this may change if you join a startup).

		To				
		Graph-ology	Tree-ology	Network-ology	Ornith-ology	Anthrop-ology
	Graphology	∞	1	2	∞	∞
	Treeology	∞	∞	∞	3	4
From	Networkology	∞	∞	∞	∞	1
	Ornithology	∞	∞	∞	∞	2
	Anthropology	18	∞	∞	2	∞

1. Draw the graph that this adjacency matrix represents.

2. Write the adjacency list representation of this graph.

3. Edgar Dijkstra is an expert on graphology. However, he wishes to learn another field. Use his algorithm to find the cheapest way for him to learn each of the other fields (starting with his knowledge of graphology only) as cheaply as possible, using Prof. Graf's courses.

4. Joseph Kruskal is hired to streamline Prof. Graf's operations. For the purposes of this part, assume that the edges are undirected. Dr. Kruskal will make sure that it's possible to get from every course to every other one, but he'll do this using the cheapest set of courses possible. Apply his Minimum Spanning Tree algorithm to find these courses.

3 Project IV

Pick one of the following. Each is based on one of the choices for Project IV.

3.1 Compression - lOSSy TeXt compRESsIoN

The handout describing the Compression option said that lossy compression isn't acceptable for text files. However, in some cases this might be okay. For example, it may not be important to preserve the case of letters (i.e. we don't care if letters are upper or lower case). tHE resUlT May Be diFFICulT to rEAd. AT THe sAME TimE, iT's StIlL poSSIBLE to fIguRe ouT WHAt It sayS.

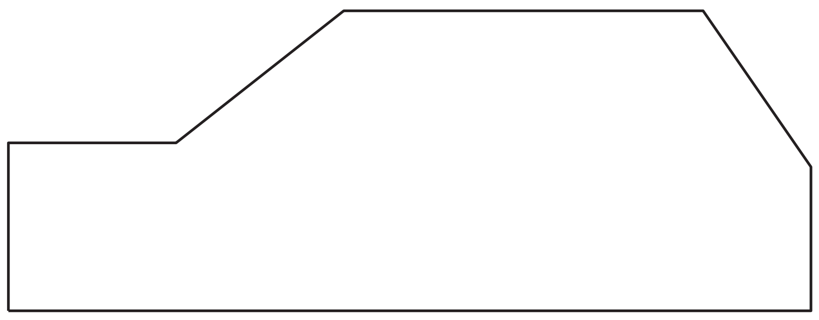
1. How could you take advantage of this relaxed requirement in order to make an LZW/Huffman compressor compress as well as possible? Describe at a high level how you would modify the program.

2. Suppose that it was desirable to preserve the case of letters as much as possible, but this could be fudged sometimes to achieve a better compression ratio. So, we want better compression, but we also ideally want to preserve case.

   Give one approach to modifying an LZW/Huffman compressor so that it balances these two goals well. (Note that there is no mathematical description of what a "good" balance between the two goals are; you just need to come up with a reasonable idea.)

3.2 Maze De-Generation

The follow is the outline of a car (of a certain style):



Find a set of points so that the Voronoi diagram of those points would include the lines in the car drawing (you will probably have to have additional lines that aren't in the drawing - that's okay). Congratulations! You've designed the new Honda Voronoi.

3.3 Maze Visualization - Collision Detection

While the user is exploring a maze, they may bump into walls. You might not want to just let the user go through the wall, but wish to keep them in the same place, and maybe display some insulting message like "You moron, you hit a wall!".

The animation in a maze visualization program can work using time slices. At each time slice, you display the user's view. Then, you calculate movements, and display the result on the next time slice. Typically, time slices are small enough to allow for only a small amount of movement (otherwise things look kind of jerky). So, you can assume that the user hit a wall if on one time slice they're on one side of that wall, and on the next time slice they're on the other side of the wall.

Describe a way of using your BSP tree to do this as efficiently as possible. The efficiency analysis should be based on an expected case (e.g. we don't expect that users will travel from one end of the maze to the other in one time step).

4 Operations

Here's a sequence of operations for a dictionary ADT:

- Insert 4
- Insert 14
- Insert 9
- Find 14
- Delete 4

- Insert 8
- Insert 7

Show how these operations would be performed by:

- A Splay Tree.
- An AVL Tree.
- A 2-3 Tree.
- A closed Hash Table using Quadratic Probing.

5 I love making change - let me count the ways...

Design a dynamic programming algorithm to find out how many ways you can make change. The algorithm should assume that you have 1¢, 5¢, 10¢ and 25¢ coins. Then you have to make a certain amount of change, like say 27¢. The algorithm should then say how many ways there are of making that amount of change.

The sequence of coins you can give back is ordered. In other words, giving a penny and then giving back a nickel is different from giving back a nickel and then giving back a penny (if you say that they're the same, the problem gets somewhat harder).

For example, you can make change for 10¢ in 9 different ways:

- 10 pennies
- All 6 ways of distinctly ordering 5 pennies and 1 nickel
  - nickel,penny,penny,penny,penny,penny
  - penny,nickel,penny,penny,penny,penny
  - penny,penny,nickel,penny,penny,penny
  - penny,penny,penny,nickel,penny,penny
  - penny,penny,penny,penny,nickel,penny
  - penny,penny,penny,penny,penny,nickel
- 2 nickels
- 1 dime

Design a dynamic programming algorithm to solve this problem efficiently.

1. Write pseudocode for the algorithm
2. Analyze its run time
3. Show what the dynamic programming table would look like for an input of 9¢.

6 Steffi Graf's Questions

Steffi Graf enthusiastically suggested the following:

1. Draw an undirected graph with 3 connected components. A "connected component" is a subset of the graph that is in itself connected but is not connected to the rest of the graph.
2. Draw a directed graph that can't be topologically sorted. What property is this graph lacking?
3. Draw a graph that has 1 node of degree 4 and 4 nodes of degree 3.

7 Mr. Simple plays 52 Pick Up

I often play cards, and I love it. Unfortunately, I'm extremely clumsy, so I often drop the cards. Then I want to sort the cards so that I know if anything's missing. I've heard that computer science has a bunch of ways of sorting cards.

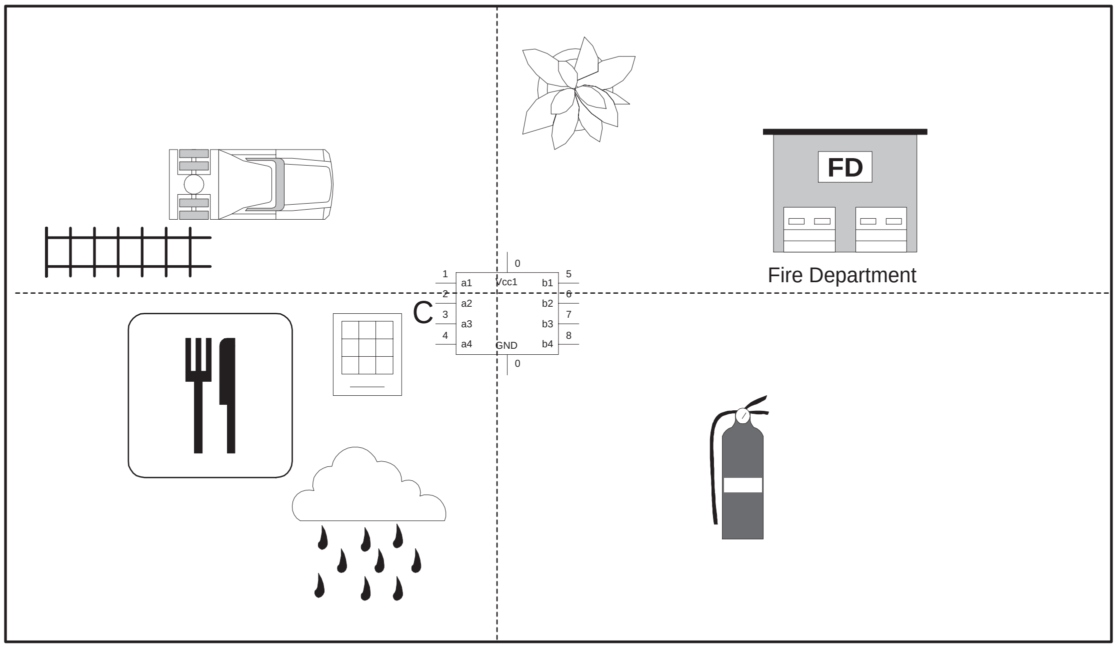
Could you tell me how I would sort cards, in the style of the following computer science algorithms I've heard about. Please remember that I'm totally clueless about computer science, so make sure you describe things in a way that I can understand. Things like "recursively", "comparison function", "array" and so forth confuse me terribly. Also, I'd appreciate it if you could tell me which sorting algorithm you think is best for me.

1. Selection Sort
2. Quick Sort
3. Merge Sort
4. Bin Sort
5. Radix Sort

Also, my friend Reginald said that you could use HeapSort with a Skew Heap to sort things in $O(n \log n)$ even in the worst case. Is this true? For this question, can you load me up with computer science jargon, so that I can impress Reginald?

8 Fun with Visio

The following picture shows a bunch of shapes that come with Visio.


FD
Fire Department
C
1
2
3
4
a1
a2
a3
a4
Vcc1
GND
0
0
b1
b2
b3
b4
5
6
7
8

You'll need to create a Quad Tree and 2-d Tree to store these pictures. One difficulty is that some of the lines in the multidimensional tree may cut through the shapes. This would not happen if we were using points, like in lecture.

There are many way of dealing with this issue, and you're free to use any reasonable method. The recommended solution is that, if you split a picture, just put it on both sides of the line. These pictures will then appear in multiple leaves of the resulting tree.

1. What lines will a Quad Tree create in the picture, corresponding to its nodes.

2. What will a 2-d Tree made from these objects look like.

9 The Big Last Question

Algorithmic Global Enterprises is a new worldwide company. They have offices in thousands of places all around the world. They want to connect their offices with networking cable, so that they're connected to each other. Since they're secretive and weird, they want to use their own lines - not already-existing network cable.

Wow! That's expensive! They need to cut costs. They've got a list of the precise latitute and longitude of every one of their offices, as well as the name of the office. They want you to tell them where to put wires such that every office is connected to every other one (not necessarily directly), and the total length of wire you used is as short as possible (they're using super expensive wire).

For network troubleshooting, they also want to be able to type in the names of two offices, and have your program tell them the route between these two offices along the wires.

Finally, occasionally they create new offices, or decommission old ones. They need your program to suggest wiring for the new offices, and update its information on the offices.

You have at your disposal a library of well-crafted data structures and algorithms including each of the ones we have discussed.

1. Briefly summarize the application that Algorithmic Global Enterprises wants and their requirements for that application. In other words, what ADTs and algorithms do they need, how will they work together, and what time and space requirements are there for the ADTs?

2. Describe a high-level solution to Algorithmic Global Enterprise's problem. You shouldn't write any code, but do pick specific data structures to implement any ADTs in the problem and specify any parameters for those structures. Also, describe when and how any algorithms called are used (and what data structures they use). Explain your choices briefly (i.e., what motivated you to choose one structure over another).