CECS 528, Exam 1, Fall 2025, Dr. Ebert

IMPORTANT: READ THE FOLLOWING DIRECTIONS SO YOU WILL NOT LOSE POINTS. Directions: This exam has SIX different problems: one problem for each of LO's 1-4 and two additional problems.

- For each problem, write your solution using **ONE SHEET OF PAPER ONLY (BOTH FRONT AND BACK)**. Write **NAME** and **PROBLEM NUMBER** on each sheet.
- Write solutions to different problems on **SEPARATE SHEETS** of paper.
- For example, if you decide to solve all six problems, then you will submit **SIX** sheets for grading.
- A 20% deduction in points will be applied to each solution that does not follow the above guidelines.

Unit 1 LO Problems (25 pts each)

LO1. Solve the following problems.

(a) Use the Master Theorem to determine the growth of $T(n)$ if it satisfies the recurrence $T(n) = 81T(n/3) + n^4$. Defend your answer. (10 pts)

Solution.

$n^{\log_3 81} = n^4 = f(n)$. $\therefore$ By Case 2 of M.T., $T(n) = \Theta(n^4 \log n)$.

(b) Use the substitution method to prove that, if $T(n)$ satisfies

$$T(n) = 27T(n/3) + 9n,$$

then $T(n) = \mathrm{O}(n^3)$. (15 pts)

The inductive assumption: $T(k) \le ck^3 + dk$, for all $k < n$ and some constants $c \ge 0$ and $d \in I$.

$$T(n) = 27T(n/3) + 9n \le 27\left(c\left(\frac{n}{3}\right)^3 + d\left(\frac{n}{3}\right)\right) + 9n$$

$$= cn^3 + 9dn + 9n \le cn^3 + dn \iff$$

$$8dn \le -9n \iff d \le -9/8$$

LO2. Solve the following problems.

(a) Consider Karatsuba's algorithm which we'll call `multiply` for multiplying two even-length n-bit binary numbers x and y. Let x_L and x_R be the leftmost $n/2$ and rightmost $n/2$ bits of x respectively. Define y_L and y_R similarly. Let P_1 be the result of calling `multiply` on inputs x_L and y_L, P_2 be the result of calling `multiply` on inputs x_R and y_R, and P_3 the result of calling `multiply` on inputs $x_L + x_R$ and $y_L + y_R$. Then return the value

$$P_1 \times 2^n + (P_3 - P_1 - P_2) \times 2^{n/2} + P_2.$$

Prove that the returned value does in fact equal xy. Hint: first provide an arithmetc expression that expresses x (repectively, y) in terms of x_L and x_R (respectively, y_L and y_R). (15 pts)

Solution. $xy = (x_L 2^{n/2} + x_R)(y_L 2^{n/2} + y_R) =$

$x_L y_L 2^n + x_R y_R + x_L y_R 2^{n/2} + x_R y_L 2^{n/2} =$

$P_1 \times 2^n + P_2 + (x_L y_R + x_R y_L) \times 2^{n/2}.$

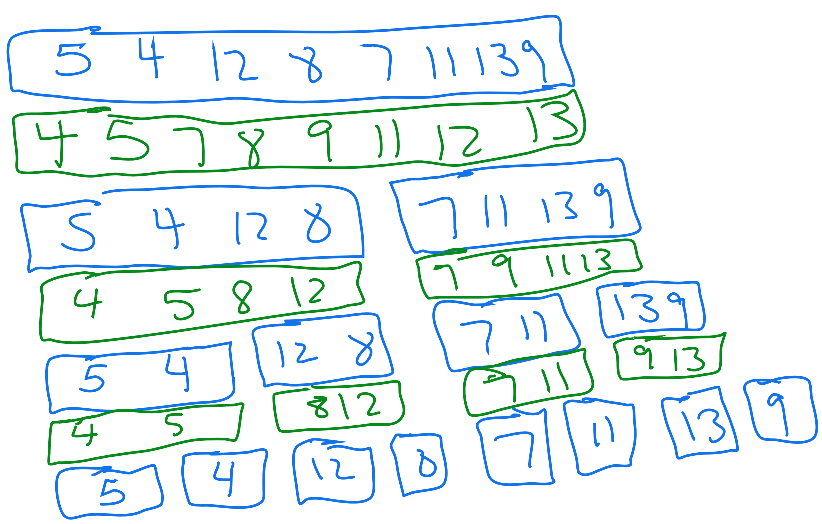
Check that $P_3 - P_1 - P_2 = x_L y_R + x_R y_L$. ✓

(b) Draw the entire recursion tree that results when applying Mergesort to the array

$$a = 5, 4, 12, 8, 7, 11, 13, 9.$$

Label each node with the subproblem instance to be solved at that point of the recursion. Assume that a base case has size equal to one. Above each node in the tree, write the solution to the subproblem instance to which it pertains. (10 pts)

Solution.



LO3. Do the following.

(a) In relation to the FFT algorithm, why is it essential that, for even n, the nth roots of unity come in additive-inverse pairs? Hint: recall the equation that relates the original $(n-1)$-degree polynomial $A(x)$ to the polynomials $A_e(x)$ and $A_o(x)$. (15 pts)

Solution. The FFT algorithm requires evaluating a $2n-1$ degree polynomial at $2n$ different inputs. It does this by making two recursive calls, each of which entails evaluating a $(n-1)$-degree polynomial at n inputs. Moreover, these n inputs are the squares of the original $2n$ inputs which means that there is indeed only n inputs because the square of a number is the same as the square of its additive-inverse.

(b) If $p(x) = -2 + 3x + x^2 - 5x^3$, then compute $\mathrm{DFT}(p)$ using the FFT algorithm. Show the entire recursion tree as was done in the lecture notes. (10 points)

Solution.

$$\mathrm{DFT}_4(-2,3,1,-5) =$$
$$(-1,3,-1,3) + (1,i,-1,-i) \odot (-2,8,-2,8) = (-3, 3+8i, 1, 3-8i)$$

$$\mathrm{DFT}_2(-2,1) =$$
$$(-2,-2) + (1,-1) \odot (1,1) = (-1,-3)$$

$$\mathrm{DFT}_2(3,-5) =$$
$$(3,3) + (1,-1) \odot (-5,-5) = (-2,8)$$

$$\mathrm{DFT}_1(-2) = -2 \quad \mathrm{DFT}_1(1) = 1 \quad \mathrm{DFT}_1(3) = 3 \quad \mathrm{DFT}_1(-5) = -5$$

LO4. Do the following.

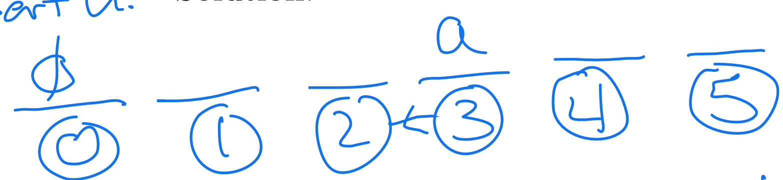
(a) Recall the use of the disjoint-set data structure for the purpose of improving the running time of the `Unit Task Scheduling` algorithm. For the set of tasks

Task	a	b	c	d	e	f
Deadline Index	3	4	5	4	4	4
Profit	60	50	40	30	20	10

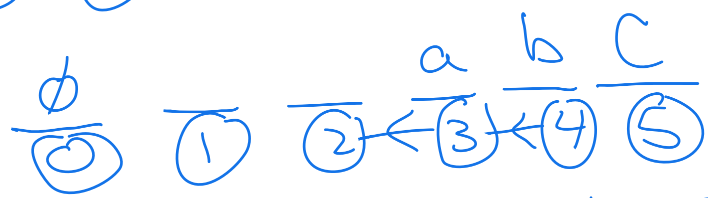
show the M-Tree forest after it has been inserted (or at least has attempted to be inserted in case the scheduling array is full). Note that the earliest possible deadline index is 1, meaning that the earliest slot in the schedule array has index 1. Also, assume that an insert attempt that takes place at index i results in the function call `root`(i), followed by a `union` operation. Finally, to receive credit, your solution should show six different snapshots of the M-Tree forest. (12 pts)

Insert a:

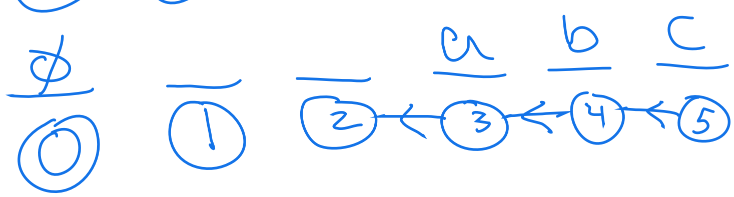
Solution.



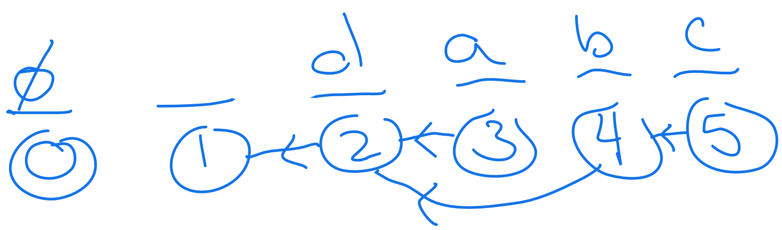
Insert b:



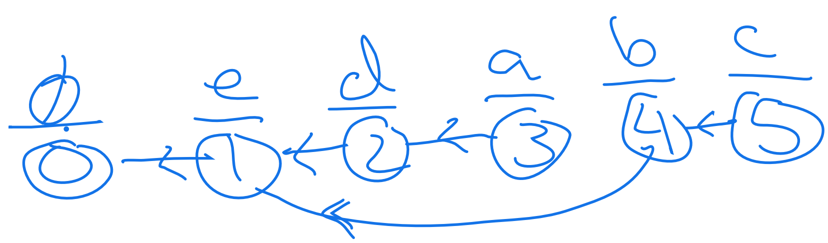
Insert c:



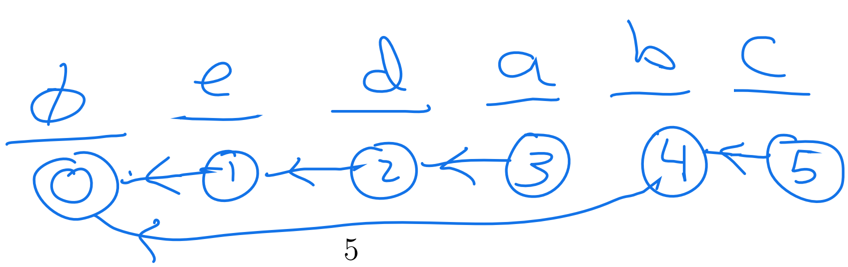
Insert d:



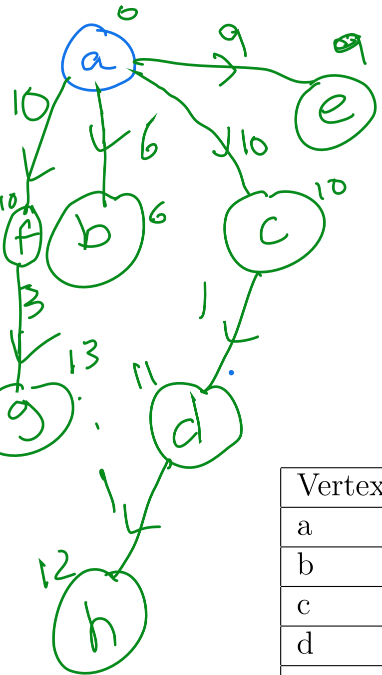
Insert e:

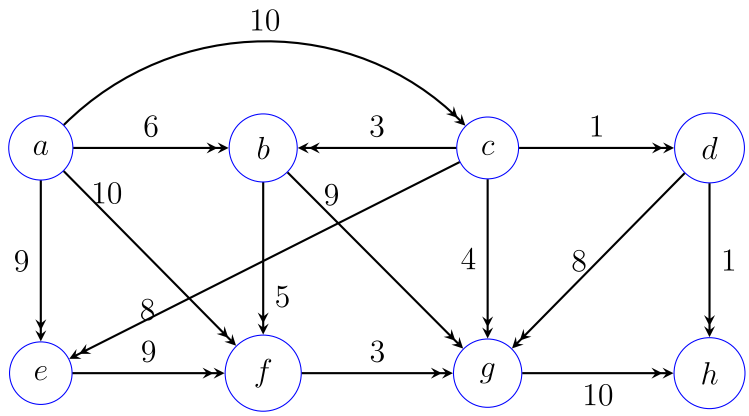


Insert f:



(b) Demonstrate Dijkstra's algorithm on the directed weighted graph shown below. Copy the table on to your solution page and in column i of the table provide the i-neighboring distance from the source a to each vertex v in the graph, as well as the vertex in T_i that is the parent of v in the i-neighboring path. Place a dash "-" in all subsequent cells once a vertex has been added to the DDT. Draw the final DDT. Hint in Round 0, T_0 is an empty tree. (13 pts)





Vertex	0th/par	1st/par	2nd/par	3rd/par	4th/par	5th/par	6th/par	7th/par
a	$0/\emptyset$	—	—	—	—	—	—	—
b	$\infty/\emptyset$	6/a	—	—	—	—	—	—
c	$\infty/\emptyset$	10/a	10/a	10/a	—	—	—	—
d	$\infty/\emptyset$	$\infty/\emptyset$	$\infty/\emptyset$	$\infty/\emptyset$	11/c	11/c	—	—
e	$\infty/\emptyset$	9/a	9/a	—	—	—	—	—
f	$\infty/\emptyset$	10/a	10/a	10/a	10/a	—	—	—
g	$\infty/\emptyset$	$\infty/\emptyset$	15/b	15/b	14/c	13/f	13/f	13/f
h	$\infty/\emptyset$	$\infty\ \emptyset$	$\infty/\emptyset$	$\infty/\emptyset$	$\infty/\emptyset$	∞/h	12/d	—

Solution. See above.

Additional Problems

A1. Consider Hoare's `Quicksort` algorithm.

(a) What is its worst-case running time and what must happen in order for it to occur? Explain and defend your answer using basic math. (10 pts)

Solution. Worst case running time is $\mathrm{O}(n^2)$ which occurs when each partitioning step results in one array having size 1, and the other array having size $n-2$. In this case the partitioning steps require a total of

$$\mathrm{O}(n + (n-2) + (n-4) + \cdots + 2) = \mathrm{O}(n^2),$$

since the sum of the first $n/2$ even numbers is

$$\sum_{i=1}^{\frac{n}{2}} 2i = (n/2)(n/2 - 1) = \mathrm{O}(n^2).$$

(b) Based on the different algorithms we've studied, what one change could be made to `Quicksort` that would guarantee that it has an improved worst-case running time? Defend your answer by making use of the Master Theorem. (10 pts)

Solution. Use the `Median-of-Five Find Statistic` algorithm to find the median of the array to be sorted and use this median as the pivot. Now the algorithm's number of steps $T(n)$ satisfies

$$T(n) = 2T(n/2) + n$$

since both the left and right subproblems now have the same size of $n/2$ and `Median-of-Five Find Statistic` requires a linear number of steps.

A2. Do the followng.

(a) Provide each of the 6th roots of unity, writing each one in the standard form $a+bi$. Verify that the square of each 6th root of unity is a 3rd root of unity. Hint: $\sin(60°) = \frac{\sqrt{3}}{2}$. (10 pts)

Solution.

$\omega_6^0 = 1$

$\omega_6^1 = \frac{1}{2} + \frac{\sqrt{3}}{2} i$

$\omega_6^2 = -\frac{1}{2} + \frac{\sqrt{3}}{2} i$

$\omega_6^3 = -1$

$\omega_6^4 = -\frac{1}{2} - \frac{\sqrt{3}}{2} i$

$\omega_6^5 = \frac{1}{2} - \frac{\sqrt{3}}{2} i$

$\omega_3^0 = 1 \qquad \omega_3^1 = -\frac{1}{2} + \frac{\sqrt{3}}{2} i$

$\omega_3^2 = -\frac{1}{2} - \frac{\sqrt{3}}{2} i$

$1^2 = 1 = (-1)^2$ $\checkmark$

$\left(+\frac{1}{2} + \frac{\sqrt{3}}{2} i\right)^2 = \left(-\frac{1}{2} - \frac{\sqrt{3}}{2} i\right)^2 =$

$\frac{1}{4} - \frac{3}{4} + \frac{\sqrt{3}}{2} i = -\frac{1}{2} + \frac{\sqrt{3}}{2} i$ $\checkmark$

$\left(-\frac{1}{2} + \frac{\sqrt{3}}{2} i\right)^2 = \left(\frac{1}{2} - \frac{\sqrt{3}}{2} i\right)^2 =$

$-\frac{1}{2} - \frac{\sqrt{3}}{2} i$ $\checkmark$

(b) In a few sentences, explain how to modify either Prim's or Kruskal's algorithm so that it computes a *maximum spanning tree*, i.e. one whose total cost is a maximum amongst all possible spanning trees. (10 pts)

Solution. For Kruskal's, sort the edges in *decreasing* order of weight. For Prim's, the Heap node having the highest priority is the one that connects to the current mst with the *greatest* weight.

(c) Recall that the `MNode` data structure has the single attribute `MNode parent` which references the `MNode` that is the parent of the given node. Provide a *recursive* implementation of the function

```
MNode root(MNode n)
```

that returns the root of the tree for which n is a member. Furthermore, as a side effect, n and each of its ancestors (except for the root) will have its parent attribute set to the root. In other words, your implementation should support path compression. (10 pts)

Solution.

```
MNode root(MNode n)
{
    MNode parent = n.parent;

    if(parent == NULL)
        return n;

    MNode n2 = root(parent);

    n.parent = n2;  //path compression

    return n2;
}
```