Exam 3 – Math 215

1 True/False

Problem 1.1 (80 points; 10 points each)**.** Decide if each of the following is true or false. You do not need to justify your choice here.

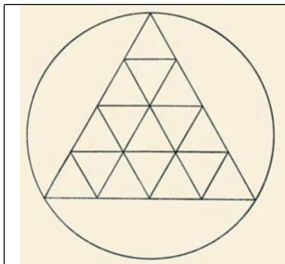


Figure 1

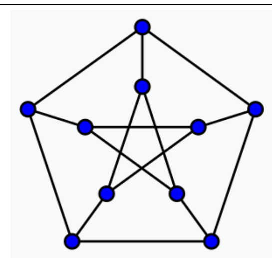


Figure 2

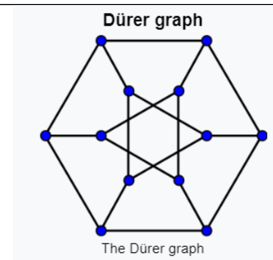


Figure 3

(a) False The relation $|$ (divides) on $\mathbb{Z}$ given by $m \mid n \iff n = q \cdot m$ for some $q \in \mathbb{Z}$ is not an order on $\mathbb{Z}$ since there are incompatible elements, for example, $3 \nmid 5$ and $5 \nmid 3$.

$(\mathbb{Z}, |)$ is one of the examples of orders we have used often.

(b) True Let $G = (V, E)$ be an undirected graph. Define the "*connected by a path*" relation on V to be $u \sim v$ iff there is a path from u to v. $\sim$ is an equivalence relation on V with equivalence classes being the components of G.

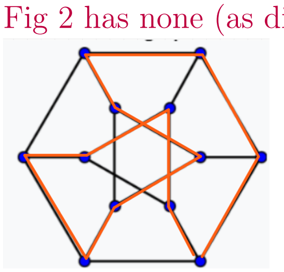
This is the definition of a component.

(c) True There is a way to draw Figure 1 without lifting your pencil.

Yes $\deg(v)$ is even for all nodes v.

(d) False Neither of Figure 2 or Figure 3 has a Hamiltonian cycle. (I didn't want to make this too easy after our last meeting:)

Fig 2 has none (as discussed in class), Fig 3 has an easy to find Hamiltonian path.



(e) False A flood pours through the entrance of a cavern that has only one other opening for the water to flow out of. The water performs essentially a depth-first search to find the exit. (Assume the cavern is "flat," i.e., no real change in elevations.)

This would be breadth-first. The water will just spread out and take every path possible at every possibility.

(f) True A fox enters a rabbit burrow to search for dinner. The fox's search for a rabbit to eat is most like a depth-first search.

This is depth-first, the fox will just keep following her nose until she hits a dead end and backtracks or eats a rabbit.

(g) False 20,000 words are stored in a balanced and full binary search tree. You can decide if "stuck" is in the word list with at most only 13 comparisons, i.e., questions of the form: "Is 'stuck' the word I am currently looking at?" "Is 'stuck' to the left (alphabetically) of the word I am currently looking at?" or "Is 'stuck' to the right of the word I am currently looking at?"

The height of a full-balanced binary tree capable of holding 20,000 items would be $\log_2(20,000) = 14.28$. Or if you like, there are $\leq 2^0 + 2^1 + 2^2 + \cdots + 2^{13} = \frac{2^{14}-1}{2-1} = 2^{14} = 16,384$ nodes in a full-balanced binary tree of height 13. So the tree holding $20,000$ items must have some branches of length 14, and thus, it could take up to 15 comparisons to determine if an item is not in the tree.

(h) True If a connected graph $G = (V, E)$ satisfies $|E| = |V| - 1$, then for any $u, v \in V$, there is a unique path between u and v.

A connected graph always satisfies $|E| \geq |V| - 1$. If G contained a circuit, we could remove an edge and still have a connected graph. In that case, $|E| - 1 \geq |V| - 1$. So there must not be a circuit, i.e., G is a tree, and hence, between any two nodes, there is a unique path.

2 Free Response

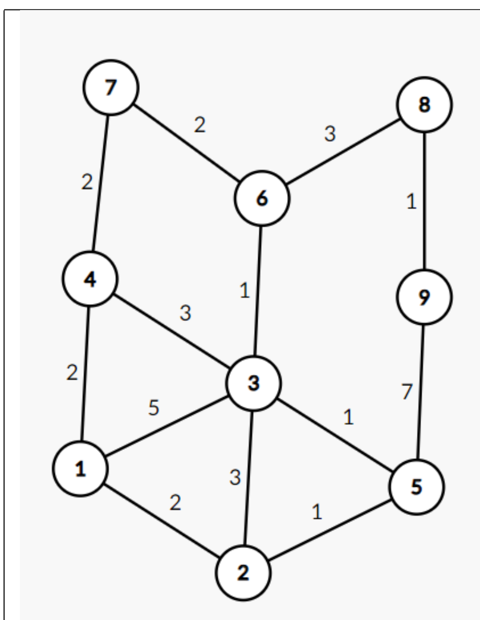


Figure 4

Problem 2.1 (25 points)**.** Use Dijkstra's algorithm to find the shortest path (Figure 4) from the node 1 node to node 9. Make sure that I can see how you are using Dijkstra. Give a "trace" like we did in class, or draw the graph and indicate the re-labelings. What is the shortest path, and what is the weight of the shortest path?

Here is a visual solution. Here is the trace as a table:

bt			0	0	2	5	3	4	6	8
1	0	$\boxed{0}$	0	0	0	0	0	0	0	0
2	∞	2	2	$\boxed{2}$	2	2	2	2	2	2
3	∞	5	5	5	4	$\boxed{4}$	4	4	4	4
4	∞	2	$\boxed{2}$	2	2	2	2	2	2	2
5	∞	∞	∞	3	$\boxed{3}$	3	3	3	3	3
6	∞	∞	∞	∞	∞	4	$\boxed{4}$	4	4	4
7	∞	∞	4	4	4	4	4	$\boxed{4}$	4	4
8	∞	∞	∞	∞	∞	∞	7	7	$\boxed{7}$	7
9	∞	∞	∞	∞	10	10	10	10	9	$\boxed{9}$

So the distance from 1 to 9 is 9 and the path is:

$$0 \to 2 \to 5 \to 3 \to 6 \to 8 \to 9$$

Problem 2.2 (25 points)**.** Use either Prims's or Kruskal's algorithm to find a minimal spanning tree (Figure 4). Make sure that I can tell what algorithm you have chosen and can see how you are using the algorithm. Make clear what the weight of the minimal spanning tree is, as well as what the tree itself is.

Here is a solution with Kruskal. The weight is 13, which is unique, but the tree itself is not.

Here is a solution using Prim.

Problem 2.3 (25 points)**.** Build the binary search tree for the words in the sentence:

> Every choice branches out like a tree, each decision a node guiding our path.

Use the order in which the words are provided to build the tree. You should employ a binary-search for each word to determine where it goes in the tree.

Note: This sentence was generated by ChatGPT and here is what it says about it's creation:

> This sentence not only reflects the structure of a binary search tree, with each choice leading to further branches (or nodes), but also offers a metaphorical view on how decisions shape our journey, much like how nodes determine the structure of a binary search tree.

Here is the solution.

Problem 2.4 (25 points). Below is an 8x8 pixel drawing. One byte (0-255) is used per pixel, so the "size" of the coding is 64x8 = 512 bits. Use Huffman coding to code the values (indicate what you are doing). What is the final size of the encoding in bits?

225	225	225	225	225	225	225	225
225	225	225	225	225	150	150	175
225	225	225	225	225	150	150	175
200	200	200	200	200	150	150	175
200	200	200	200	200	150	150	175
200	200	100	125	125	150	175	175
50	75	100	125	125	150	175	175
50	75	100	100	125	150	175	175

Value	Frequency	Relative Frequency
50	2	3.13%
75	2	3.13%
100	4	6.25%
125	5	7.81%
150	11	17.19%
175	10	15.63%
200	12	18.75%
225	18	28.13%

Figure 5

Here is the solution. The image actually mirrors the Huffman tree.

225	225	225	225	225	225	225	225
225	225	225	225	225	150	150	175
225	225	225	225	225	150	150	175
200	200	200	200	200	150	150	175
200	200	200	200	200	150	150	175
200	200	100	125	125	150	175	175
50	75	100	125	125	150	175	175
50	75	100	100	125	150	175	175

Value	Frequency	Relative Frequency	Codes	Bits
50	2	3.13%	00000	10
75	2	3.13%	00001	10
100	4	6.25%	0001	16
125	5	7.81%	001	15
150	11	17.19%	100	33
175	10	15.63%	101	30
200	12	18.75%	01	24
225	18	28.13%	11	36
				174

See here for the coding.