CS 1501 Exam 2 (Take-Home)

Name: **<Kent Kaing>**

Pitt email address: **<kek165>**@pitt.edu

Peoplesoft number: **<4243030>**

Fill in the fields above (name, Pitt username, and Peoplesoft number), read the instructions below and complete the honor statement, and complete your answers within this document. Then, save as a PDF for submission. Submit via Gradescope according to the instructions you received via email.

Avoid making changes to the layout or formatting of this document to ensure your answers align as expected once submitted. In particular, ensure you **do not create any additional pages**. Each problem should be contained on a single page, for a total of 9 pages.

When working on this exam, you are **allowed** to use:

- Lecture slides and recordings
- Your own notes
- The course textbook (Algorithms 4th ed. by Sedgewick and Wayne)
- Blank scrap paper

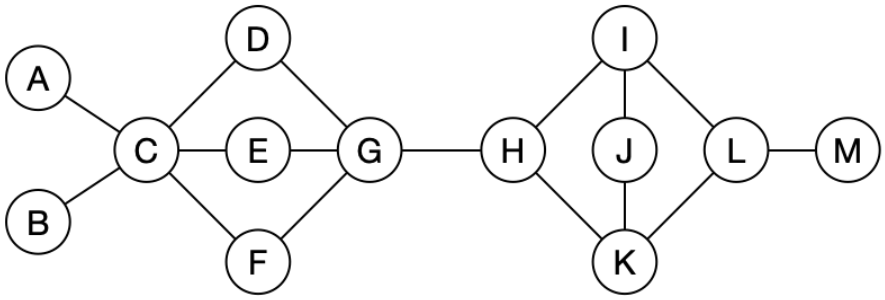
You are **not allowed** to use:

- Online resources beside those stated above
- Code or programs, written by you or others
    - Note that this means you *cannot* write, edit, or run code to solve these problems, but you *can* read (only!) code from the lectures or your notes
- Help from a classmate, former student, or anyone else

Violating these instructions will be considered an academic integrity violation and penalized according to the course policy. Below, please confirm your agreement by adding your name to the honor statement. **Your exam will not be graded without a completed honor statement.**

By submitting this exam, I, **<Kent Kaing>**, pledge on my honor, and with acceptance of the consequences laid out in all applicable policies, that I have neither given nor received any unauthorized assistance on this evaluation, and that the work submitted upholds the highest standards of honesty and academic integrity.

Problem 1. Consider depth-first and breadth-first traversal of the following unweighted graph. Assume neighbors of each vertex are seen in **alphabetical order**.



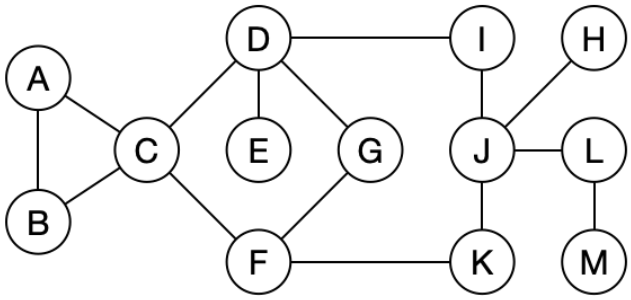
a. State each **edge** (e.g., J-K for the edge connecting J to K) that is added to the spanning tree when completing a depth-first traversal of this graph **starting at Vertex A**. State the edges **in the order they are added**, as numbered below. You may not need all spaces.

1. A-C	2. C-B	3. C-D
4. D-G	5. G-E	6. G-F
7. G-H	8. H-I	9. I-J
10. J-K	11. K-L	12. L-M
13.	14.	15.

b. State each **edge** (e.g., J-K for the edge connecting J to K) that is added to the spanning tree when completing a breadth-first traversal of this graph **starting at Vertex M**. State the edges **in the order they are added**, as numbered below. You may not need all spaces.

1. M-L	2. L-I	3. L-K
4. I-H	5. I-J	6. H-G
7. G-D	8. G-E	9. G-F
10. D-C	11. C-A	12. C-B
13.	14.	15.

Problem 2. Use an articulation point search to determine in the depicted graph is biconnected. **Start from vertex A** and assume that neighbors of each vertex are seen in **alphabetical order**.



a. Give the num value (starting your numbering from 0) and low value for each vertex.

	A	B	C	D	E	F	G	H	I	J	K	L	M
num	0	1	2	3	4	6	5	9	10	8	7	11	12
low	0	0	0	2	4	2	2	9	3	3	3	11	12

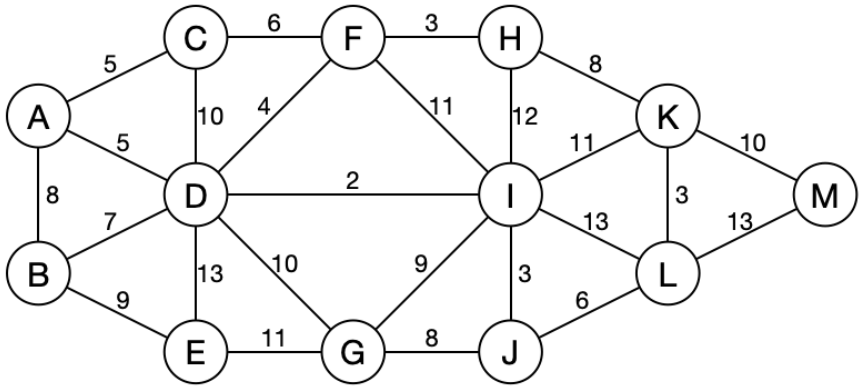
b. Identify all articulation points, and for each one, concisely state how the algorithm determines it is an articulation point.

C- C's num value is 2 and that is equal to its child's(D) low value 2
D- D's num value is 3 and that is less that its child's(E) low value 4
J- J's num value is 8 and that is less than its child's(H) low value 9
L- L's num value is 11 and that is less than its child's(M) low value 12

c. Finally, state if the graph is biconnected.

No

Problem 3. Use Prim's algorithm to determine a minimum spanning tree of the pictured weighted graph. **Start from vertex D** and assume that neighbors of each vertex are seen in **alphabetical order**.



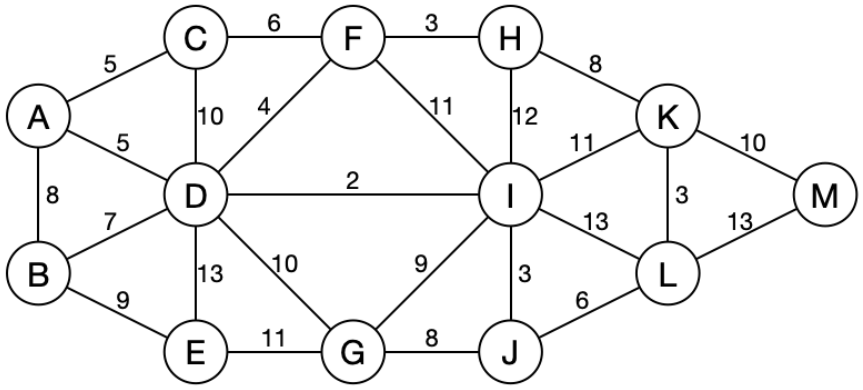
a. State each **edge** (e.g., L-M for the edge connecting L to M) that is included in the minimum spanning tree. Write them in the order they are added according to Prim's algorithm, as numbered below. You may not need all spaces.

1. D-I	2. I-J	3. D-F
4. F-H	5. D-A	6. A-C
7. J-L	8. L-K	9. D-B
10. J-G	11. B-E	12. K-M
13.	14.	15.

b. What is the weight of the MST?

65

Problem 4. Consider using Dijkstra's algorithm to determine weighted shortest paths in the pictured weighted graph, **starting from vertex B**. In the case of a tie, prefer to visit the vertex that appear earlier in the alphabet. Mark **visited vertices with parentheses** as demonstrated.



After visiting 3 vertices, the state of the `distance` and `via` arrays is shown below.

	(A)	(B)	C	(D)	E	F	G	H	I	J	K	L	M
Distance	8	0	13	7	9	11	17	∞	9	∞	∞	∞	∞
Via	B	root	A	B	B	D	D	—	D	—	—	—	—

a. Show the state of the `distance` and `via` arrays (shown below initially as a copy of the data above) after visiting a 5th vertex (i.e., visit **two more** vertices).

	(A)	(B)	C	(D)	(E)	F	G	H	(I)	J	K	L	M
Distance	8	0	13	7	9	11	17	21	9	12	20	22	∞
Via	B	root	A	B	B	D	D	I	D	I	I	I	—

b. Show the state of the `distance` and `via` arrays after visiting an 8th vertex (i.e., visit **three more** vertices after (a)).

	(A)	(B)	(C)	(D)	(E)	(F)	G	H	(I)	(J)	K	L	M
Distance	8	0	13	7	9	11	17	14	9	12	20	18	∞
Via	B	root	A	B	B	D	D	F	D	I	I	J	—

c. Show the state of the distance and via arrays after completing Dijkstra's algorithm (i.e., visit **all** vertices).

	(A)	(B)	(C)	(D)	(E)	(F)	(G)	(H)	(I)	(J)	(K)	(L)	(M)
Distance	8	0	13	7	9	11	17	14	9	12	20	18	30
Via	B	root	A	B	B	D	D	F	D	I	I	J	K

Problem 5. Consider a graph with 16 vertices (0–15). Give the contents of the `id` and `size` arrays of a Union Find data structure after performing the following operations. Use a **weighted-trees** approach with **path compression** to implement the Union Find. Resolve ties by keeping the numerically lesser connected component ID as the root.

a. `union(10, 5); union(11, 14); union(15, 3); union(7, 1); union(4, 6); union(8, 9); union(12, 8); union(10, 11); union(13, 0); union(6, 7);`

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
id	0	1	2	3	1	5	4	1	8	8	5	5	8	0	11	3
size	2	4	1	2	2	4	1	1	3	1	1	2	1	1	1	1

b. Starting from your answer to the (a):
`union(0, 3); union(9, 7);`

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
id	0	1	2	0	1	5	4	1	1	8	5	5	8	0	11	3
size	4	7	1	2	2	4	1	1	3	1	1	2	1	1	1	1

c. Starting from your answer to (b):
`union(12, 10); union(2, 15);`

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
id	0	1	0	0	1	1	4	1	1	8	5	5	8	0	11	3
size	5	11	1	2	2	4	1	1	3	1	1	2	1	1	1	1

Problem 6. Consider using Karatsuba's algorithm to multiply the following binary numbers:

101110000111 × 100011010111

Assume that your base case is the multiplication of 4-bit integers (or smaller). Specify the multiplications (i.e., subproblems) that need to be completed to produce the following results.

a. M1:	101110	×	100011
b. M2:	101110	×	010111
c. M3:	000111	×	100011

d. To compute the multiplication you stated for (a), what further multiplications (subproblems) need to be solved?

M1:	101	×	100
M2:	101	×	011
M3:	110	×	100

e. To compute the multiplication you stated for (b), what further multiplications (subproblems) need to be solved?

M1:	101	×	010
M2:	101	×	111
M3:	110	×	010

f. To compute the multiplication you stated for (c), what further multiplications (subproblems) need to be solved?

M1:	000	×	100
M2:	000	×	011
M3:	111	×	100

Problem 7. Consider the RSA public key $e = 33, n = 55$. Fill in all Xs below with values.

a. Determine p and q:

Answers: $p = 5$ and $q = 11$

b. Determine $\phi(n)$:

Answer: $\phi(n) = 40$

c. Use XGCD to determine d and z that satisfy $\gcd(\phi(n), e) = \phi(n) * z + e * d$. Complete the XGCD table below (start with $a > b$). Note that you may not need all rows.

	a	b	a/b	a%b	g	x	y
1	40	33	1	7	1	-14	17
2	33	7	4	5	1	3	-14
3	7	5	1	2	1	-2	3
4	5	2	2	1	1	1	-2
5	2	1	2	0	1	0	1
6	1	0	NaN	NaN	1	1	0
7							
8							

Answer: $z = -14$, $d = 17$, and $\gcd(\phi(n), e) = 1$

Problem 8. Consider bringing a pizza with **14 slices** to class to share. Your goal is to maximize the amount of joy that you bring to your classmates by distributing pizza. You estimate the amount of joy brought to a single student when they receive some number of slices as follows.

Slices	0	1	2	3	4	5
Joy	0	1	5	8	9	15

a. Assume that the amount of joy brought to your classmates overall is the sum of the joy brought to each one. Use dynamic programming to populate the following memoization data structure to determine the maximum amount of joy that you can provide to the class. You may not need all spaces in the array.

Slices	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Joy	0	1	5	8	10	15	16	20	23	25	30	31	35	38	40		

b. What is the maximum amount of joy that you can bring to the class?

40