Last name ______________________

First name ______________________

LARSON—MATH 356—HOMEWORK WORKSHEET 13
Test 2 REVIEW.

Definitions. Write the definition **and** an example.

1. A *tree.*
2. *Eulerian circuit.*
3. What is a *coloring* of a graph? What is a *proper coloring*?
4. What is the *chromatic number* $\chi(G)$ of a graph G?
5. What is a *bipartite graph*?
6. What are the *neighbors* of a vertex v in a graph G? What is $Nbhd(v)$?
7. What is a proper K-coloring of a graph G?
8. What is $P(K;G)$?
9. (**Notation**). If $e = (v, w)$ is an edge in graph G, what is $G - \{e\}$?
10. (**Notation**). If $e = (v, w)$ is an edge in graph G, what is $G/\{e\}$?
11. What is a *directed graph*?
12. What is a *capacity* of an edge in a directed graph?
13. What is a *network* $\mathbf{X}$?
14. What is a *flow* in a network?
15. What is the *value* of a flow in a network?
16. What is a *cut* in a network?
17. What is the *capacity* of a cut in a network?

Algorithms

18. Describe the Tarjan-Trojanowski algorithm to find a maximum independent set in a graph. What is the main idea that drives the recursion? What is corresponding equation?
19. What is an algorithm for computing $P(K;G)$? What is the main idea that drives the recursion? What is corresponding equation?

Theorems. State the theorem and give an example.

20. The necessary and sufficient condition for the existence of an Eulerian circuit in a graph.

21. What is the Max Flow Min-Cut Theorem?

Problems

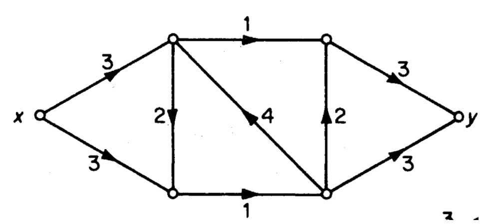
22. Use graph-theoretic induction to show that every tree with at least two vertices has a leaf.

23. Use graph-theoretic induction to show that trees are bipartite.

24. Find the number of proper K-colorings of a complete graph K_n.

25. P_4 is the path on 4 vertices (above). Use the Tarjan-Trojanowski algorithm to find $maxset(P_4)$. Include the recursion tree. What is the stopping condition?

26. Find the chromatic polynomial for P_4. Include the entire recursion tree. What is the stopping condition?



27. In this network $X = (D, x, y, cap)$, x is the *source* and y is the *sink*. Label the remaining vertices anything. The numbers on the edges are capacities. Find a maximum flow f in this network any way you want (its small enough that you may not need our sophisticated algorithm).

28. Check that f is a *flow*. Explain.

29. What is the value of this flow?

30. Given your flow f, what are the vertices that would be scanned in the last iteration of our scan-and-label algorithm?

31. Define the corresponding cut.

32. What is the capacity of this cut?

33. **Prove** that your flow is maximum.