Homework #16

Name ______________________ Sec ____

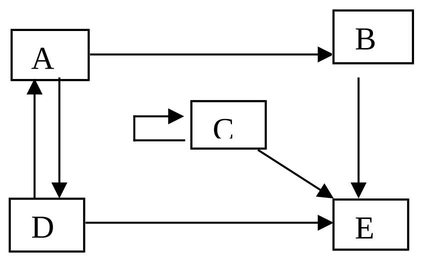
Questions:	Answers:

1. A graph is formally defined as G = (V, E, f). A simple graph (which we usually just call a graph, except when we compare the two, as we do here) is formally defined as G = (V, E).

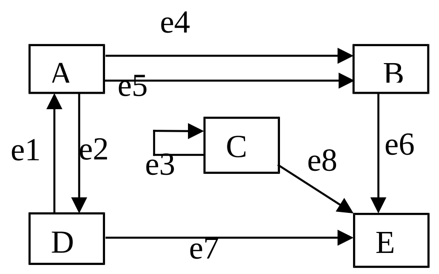
a) Draw the simple graph ({1, 2, 3, 4}, {{1, 2}, {1, 3}, {1, 1}, {2, 4}, {3, 4}}).

b) Draw the graph V = {a, b, c}, E = {e1, e2, e3}, f = {(e1,{a, b}), (e2,{a, b}), (e3,{a, c})}.

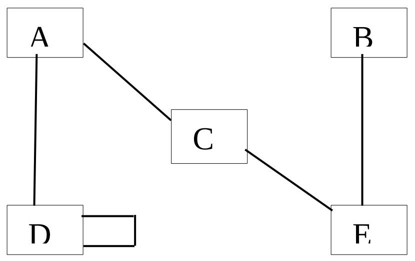
c) Give (V, E) for the following graph.



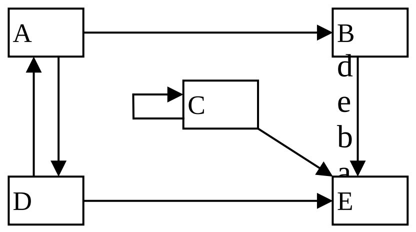
d) Give (V, E, f) for the following graph.



e) Give (V, E) for the following graph.



2. Consider the following graph G.



a) Give the incident edges of the node B.

b) Give the initiating node of the edge (A, B).

c) What is the out degree of A?

d) What is the degree of C?

e) What nodes are adjacent to C?

f) Is G complete?

g) Give the minimum path from D to B and its length.

h) Give the path relation for G.

i) Is D reachable from B?

j) Is the graph connected?

k) Is <E, B, A> a simple path in G?

l) Is <D, A, D, A, D> a cycle in G?

m) Give all simple cycles of G

n) Let H be the undirected graph having the same nodes and edges as G, but with only one edge between A and D. Consider the paths in H that begin and end with the same node:

i. Give one of these paths that should be a cycle for any cycle definition.

ii. Give one of these paths that should not be a cycle for any cycle definition.

d
e
b
a
1
(1, 6)
(2, 9)
(3, 1)
(4, 4)
2
(1, 9)
(3, 5)
(5, 6)
3
(1, 1)
(2, 5)
(5, 5)
4
(1, 4)
5
(2, 6)
(3, 5)
1
2 3
4
2
5
3
3 5
4
1
5
4
3
2
5
1
4
4
5
6

1
9
6
5
4
5
B

3. Draw the “call graph” of the following program fragment. (The nodes of a call graph are methods and the edges are method calls)

a) What do cycles in a call graph mean?
b) What does it mean if the graph is disconnected?

```
MainProgram
  main(String[])
    DatalogProgram.evaluateQueryList(StringBuffer)
DatalogProgram
  evaluateQueryList(StringBuffer)
    QueryList.evaluate(StringBuffer)
FactList
  canProve(Predicate)
    Fact.equals(Object)
Predicate
  set(int, Constant)
Fact
  equals(Object)
RuleList
  canProve(Predicate)
    Rule.prove(Predicate)
Rule
  prove(Predicate)
    Head.matches(Predicate)
    PredicateList.evaluate()
    Head.unify(Predicate)
Head
  unify(Predicate)
  matches(Predicate)
PredicateList
  PredicateList(PredicateList)
    PredicateList.initializeVariableInformation()
    Query.initializeVariableInformation()
  evaluate()
    PredicateList.recurse(int)
  recurse(int)
    PredicateList.keepOnGoing(Boolean)
    Query.keepOnGoing(Boolean)
    PredicateList.recurse(int)
    PredicateList.checkToSeeIfTrue()
  checkToSeeIfTrue
    FactList.canProve(Predicate)
    RuleList.canProve(Predicate)
    PredicateList.saveResult()
    Query.saveResult()
  saveResult()
  setUpVariableLocationMapping()
  initializeVariableInformation
    PredicateList.setUpVariableLocationMapping()
  keepOnGoing(Boolean)
QueryList
  evaluate(StringBuffer)
    Query.evaluate(StringBuffer)
Query
  initializeVariableInformation()
  evaluate(StringBuffer)
    PredicateList.evaluate()
  saveResult()
  keepOnGoing(Boolean)
```

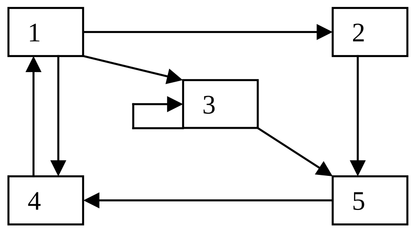
4. Let G be a graph with node set {1, 2, 3, 4, 5, 6, 7, 8} and edge set { {1, 2}, {1, 3}, {2, 5}, {2, 6}, {2, 7}, {3, 5}, {4, 5}, {4, 1}, {4, 7}, {6, 3}, {6, 4}, {7, 3} , {5, 1} }.

a) Is G Planar?

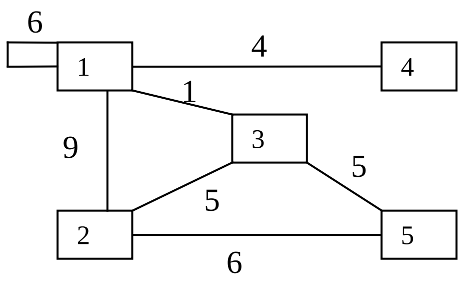
b) If so, draw it without crossing lines; if not, which subgraph does it contain: K_5 or B_3?

5. Give the adjacency list representation for the following graphs. (You may use the simple notation for adjacency lists that appears in the class notes.)

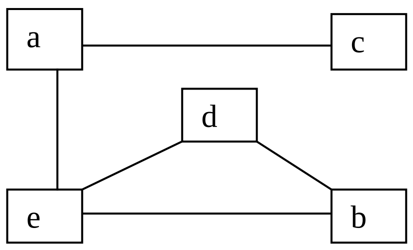
a)



b)



6. Consider the following graph.



a) List the nodes in the order they would be visited in a breadth-first search. Start with node a, and when there is a choice of nodes to visit, choose the one that is alphabetically first among those that can be chosen.

b) Assuming an adjacency list representation, explain why the algorithm runs in O(m) time where m is the number of edges, n is the number of nodes, and m >> n.

7. Consider again the graph in problem #6.

a) List the nodes in the order they would be visited in a depth-first search. Start with node a, and when there is a choice of nodes to visit, choose the one that is alphabetically first among those that can be chosen.

b) Assuming an adjacency list representation, explain why the algorithm runs in O(m) time where m is the number of edges, n is the number of nodes, and m >> n.