

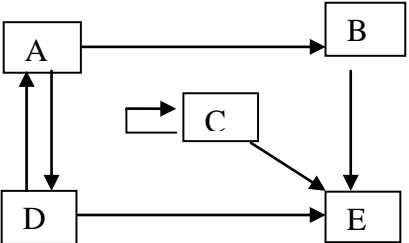
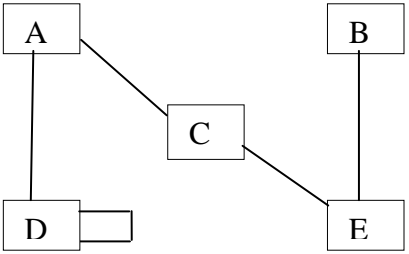
•Grading: 3 = correct
 2 = almost
 1 = an attempt
 0 = nothing
 •Score: Points / Possible

Homework #15

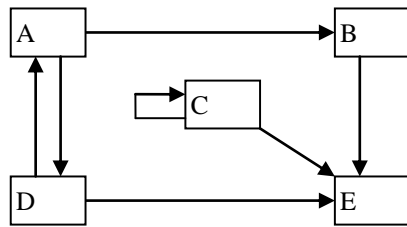
_____ Name

_____ Sec

Discussion 31

Questions:	Answers:
<p>1. A simple graph (which we usually just call a graph) is formally defined as $G = (V, E)$.</p> <p>a) Draw the simple graph $(\{1, 2, 3, 4\}, \{\{1, 2\}, \{1, 3\}, \{1, 1\}, \{2, 4\}, \{3, 4\}\})$.</p> <p>b) Give (V, E) for the following graph.</p>  <pre> graph TD A[A] --> B[B] A --> D[D] D --> A D --> E[E] C[C] --> E B --> E </pre> <p>c) Give (V, E) for the following graph.</p>  <pre> graph TD A[A] --- D[D] A --- C[C] B[B] --- E[E] C --- E </pre>	

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 $\langle E, B, A \rangle$ a simple path in G?
- l) Is $\langle D, A, D, A, D \rangle$ a cycle in G?

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 would 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)}
Fact
  { equals(Object)}
RuleList
  canProve(Predicate)
  { Rule.prove(Predicate)}
Rule
  prove(Predicate)
  { PredicateList.evaluate()}
PredicateList
  evaluate()
  { PredicateList.recurse(int)}
  recurse(int)
  { PredicateList.recurse(int)
    PredicateList.checkToSeeIfTrue()}
  checkToSeeIfTrue
  {FactList.canProve(Predicate)
    RuleList.canProve(Predicate)}
QueryList
  evaluate(StringBuffer)
  { Query.evaluate(StringBuffer)}
Query
  evaluate(StringBuffer)
  { PredicateList.evaluate()}
```

4. Create the adjacency matrix A for the graph below. Compute A^2 , A^3 , and A^4 and then the union of A , A^2 , A^3 , and A^4 , yielding the reachability matrix for A .

