1.

1. For ∀b ∈ B,

   R↓A(b) = (by def. of R↓A(b))
   { a ∈ A | R(a, b) } = (by def. of inverse relations)
   { a ∈ A | R¯(b, a) } = (by def. of R¯↓A(b))
   R¯↓A(b)

2. A proof is analogous to 1.

2.
2.

a. child↓Person$_1$(p) = the set of children of person p.
b. child↓Person$_2$(p) = the set of parents of person p.
c. child↓Person$_2$ = the set of parents = the set of persons who have a child.

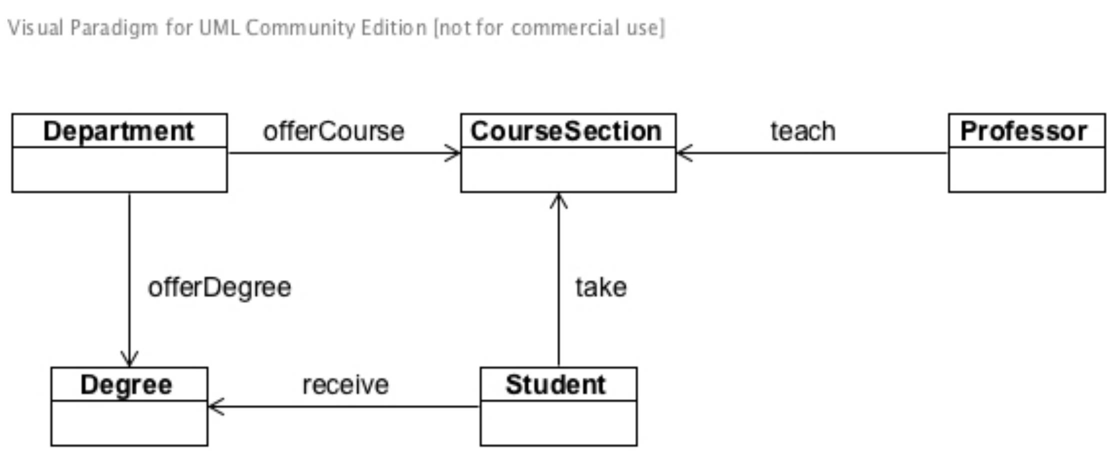
5.

a. contract↓Author(p, b) = the set of authors who signed a contract with publisher p to write book b.
b. contract↓Publisher(a, b) = the set of publishers that have had a contract with author a to write book b.
c. contract↓Book(a, p) = the set of books which author a signed contracts with publisher p to write.
d. contract↓⟨Author, Publisher⟩ = the set of ⟨author, publisher⟩ pairs such that each author in the set signed a contract to write a book with the paired publisher.
e. contract↓Author(p) = the set of authors who signed a contract with publisher p to write a book.

3.

1. (child child child)(p_1, p_2) ⇔ $\exists p_3 \exists p_4$(child(p_1, p_3) ∧ child(p_3, p_4) ∧ child(p_4, p_2)): p_1 is a great-grandchild of p_2.
2. (child sibling)(p_1, p_2) ⇔ $\exists p_3$(child(p_1, p_3) ∧ sibling(p_3, p_2)): p_1 is a nephew or niece of p_2.
3. (sibling child)(p_1, p_2) ⇔ $\exists p_3$(sibling(p_1, p_3) ∧ child(p_3, p_2)): p_1 has a sibling p_3 who is a child of p_2, i.e., p_1 is a child of p_2 and has a sibling.
4. (sibling sibling)(p_1, p_2) ⇔ $\exists p_3$(sibling(p_1, p_3) ∧ sibling(p_3, p_2)): p_1 has a sibling p_3 who is a sibling of p_2, i.e., p_1 is a sibling of p_2 and has another sibling different from p_2, or p_1 is the same as p_2 and has a sibling.

4.



a.

3. (take offerCourse¯)(s, d): student s completed a course section offered by department d.
6. (offerCourse teach¯)(d, p): department d offered a course section taught by professor p.
8. (take offerCourse¯ offerDegree)(s, d): student s completed a course section offered by a department that offered degree d.
9. (teach take¯ receive)(p, d): professor p taught a course section completed by a student that received degree d.

b.

1. { ⟨*d*, *c*⟩ : department *d* offered a degree received by a student who took course section *c* }
   offerDegree receive¯ take
2. { ⟨*d*, *dg*⟩ : department *d* offered a course section taken by a student who received degree *dg* }
   offerCourse take¯ receive
3. { ⟨*s*, *s′*⟩ : student *s* took a course section offered by a department that offered a degree received by student *s′* }
   take offerCourse¯ offerDegree receive¯
4. { ⟨*c*, *d*⟩ : course section *c* was taken by a student who received degree *d* }
   take¯ receive
5. { ⟨*c*, *d*⟩ : course section *c* was offered by a department that offered degree *d* }
   offerCourse¯ offerDegree
6. { ⟨*s*, *s′*⟩ : students *s* and *s′* took a common course section }
   take take¯
7. { ⟨*p*, *p′*⟩ : professors *p* and *p′* taught a common course section }
   teach teach¯
8. { ⟨*c*, *c′*⟩ : course sections *c* and *c′* were offered by a common department }
   offerCourse¯ offerCourse
9. { ⟨*d*, *d′*⟩ : degrees *d* and *d′* were received by a common student }
   receive¯ receive

5.
4. offerCourse (take↓⟨CourseSection, Program⟩) : Department → Program

is the binary relation of ⟨*d*, *p*⟩ such that department *d* offered a course section taken by a student in program *p* regardless of grades.

6.

1. For ∀x ∈ S,

   $R{\downarrow}S_1(x)$ = (by def. of $R{\downarrow}S_1(x)$)
   { s ∈ S | R(s, x) } = (by def. of symmetry)
   { s ∈ S | R(x, s) } = (by def. of $R{\downarrow}S_2(x)$)
   $R{\downarrow}S_2(x)$

2. Examples of symmetric relations.
   - sibling(Person, Person) relation in Question 2.3.
   - bornInSameYear(Person, Person): p_1 and p_2 were born in the same year.
   - synonymous(Word, Word): word w_1 is synonymous with word w_2.

7. Examples of transitive relations.

1. ancestor(Person, Person).
2. prerequisite(Course, Course)
3. Let Node be the set of all nodes in a certain network.
   path(Node, Node): There exists a path from node n_1 to node n_2 in the network.

8. Proof of *if*: Suppose R = R¯. Then for ∀x, y ∈ S,

R(x, y) ⇔ (by assumption of R = R¯)
R¯(x, y) ⇔ (by def. of R¯)
R(y, x)

Hence R is symmetric.

Proof of *only if*: Suppose R is symmetric. Then for ∀x, y ∈ S,

R(x, y) ⇔ (by symmetry of R)
R(y, x) ⇔ (by def. of R¯)
R¯(x, y)

Hence $R = R^{-}$.

9. For $\forall x, y \in S_1$,

$RR^{-}(x, y) \Leftrightarrow$ (by def. of composition)
$\exists z \in S_2\ (R(x, z) \wedge R^{-}(z, y)) \Leftrightarrow$ (by def. of R^{-})
$\exists z \in S_2\ (R^{-}(z, x) \wedge R(y, z)) \Leftrightarrow$ (by commutativity of $\wedge$)
$\exists z \in S_2\ (R(y, z) \wedge R^{-}(z, x)) \Leftrightarrow$ (by def. of composition)
$RR^{-}(y, x)$

Hence RR^{-} is symmetric.

10. For $\forall x, y, z \in S$,

$R^{-}(x, y) \wedge R^{-}(y, z) \Leftrightarrow$ (by def. of R^{-})
$R(y, x) \wedge R(z, y) \Leftrightarrow$ (by commutativity of $\wedge$)
$R(z, y) \wedge R(y, x) \Leftrightarrow$ (by transitivity of R)
$R(z, x) \Leftrightarrow$ (by def. of R^{-})
$R^{-}(x, z)$

Hence R^{-} is transitive.

11 and 14. See Course Notes #1, Section 2: Characteristics.

15.

1. storage manager
2. transaction manager
3. query processor/optimizer
4. transaction manager

16.

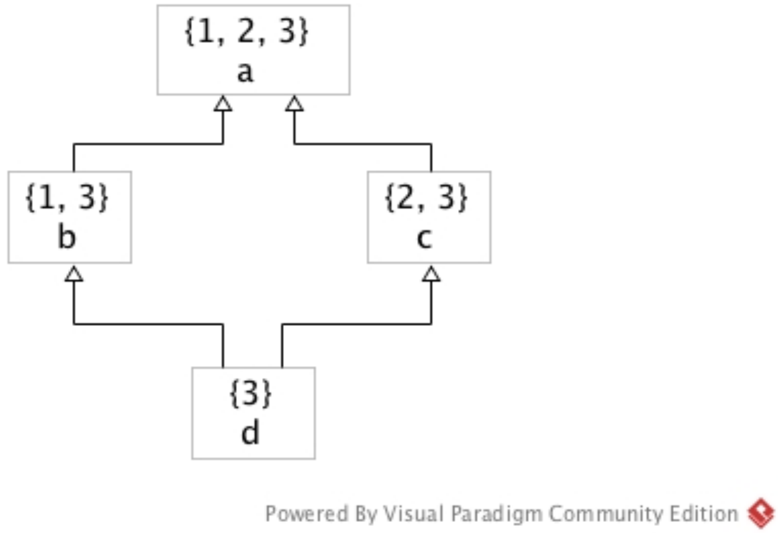
1. (i), (iii)
2. (ii)
3. (i)
4. (ii), (iii)
5. (i), (ii)
6. (i)
7. (ii), (iii)
8. (ii)

17. Build an inheritance hierarchy for each of the following lists of classes with their feature sets.

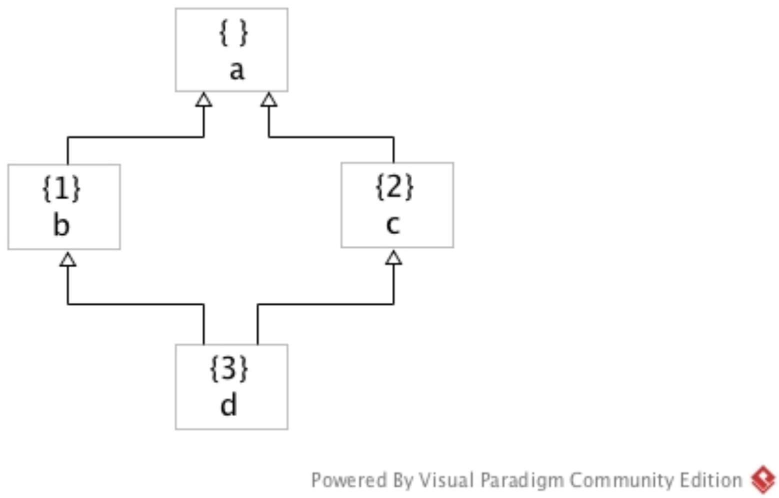
a. Input: Classes 1, ..., 3 and the following F_i:

   $F_1 = \{a, b\}$
   $F_2 = \{a, c\}$
   $F_3 = \{a, b, c, d\}$

   1. $C_a = \{1, 2, 3\}$
      $C_b = \{1, 3\}$
      $C_c = \{2, 3\}$
      $C_d = \{3\}$
   2.



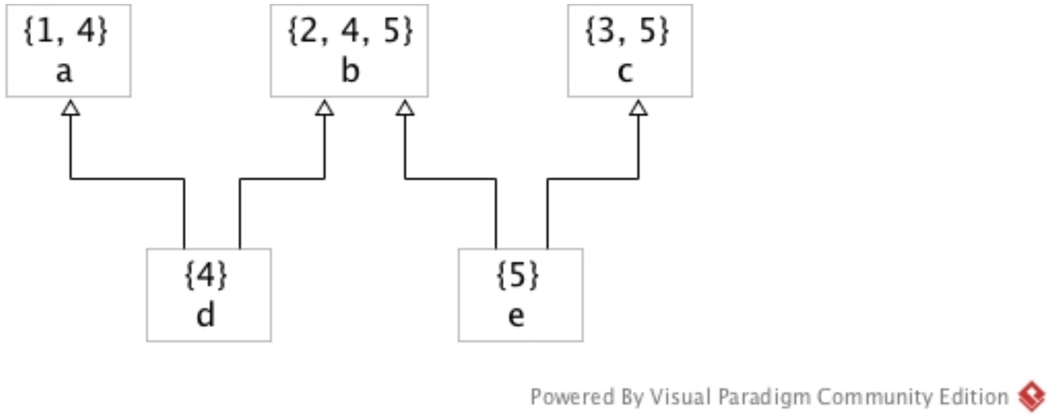
3.



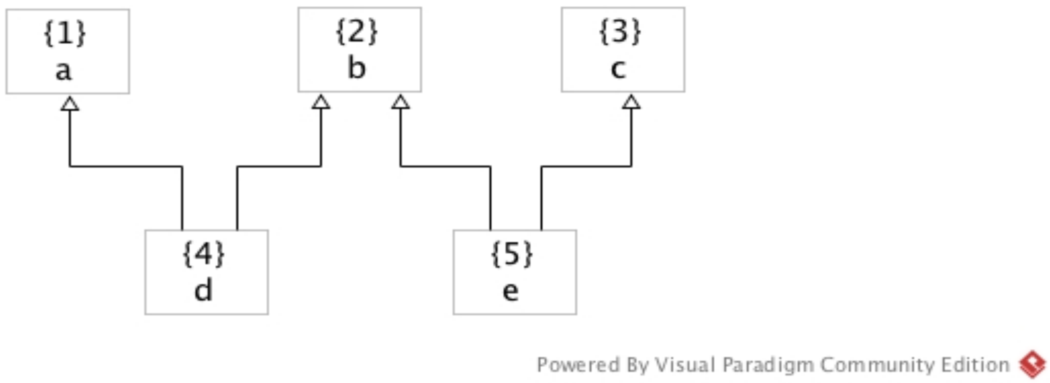
b. Input: Classes 1, …, 5 and the following F_i:

F_1 = {a}
F_2 = {b}
F_3 = {c}
F_4 = {a, b, d}
F_5 = {b, c, e}

1. C_a = {1, 4}
   C_b = {2, 4, 5}
   C_c = {3, 5}
   C_d = {4}
   C_e = {5}
2.



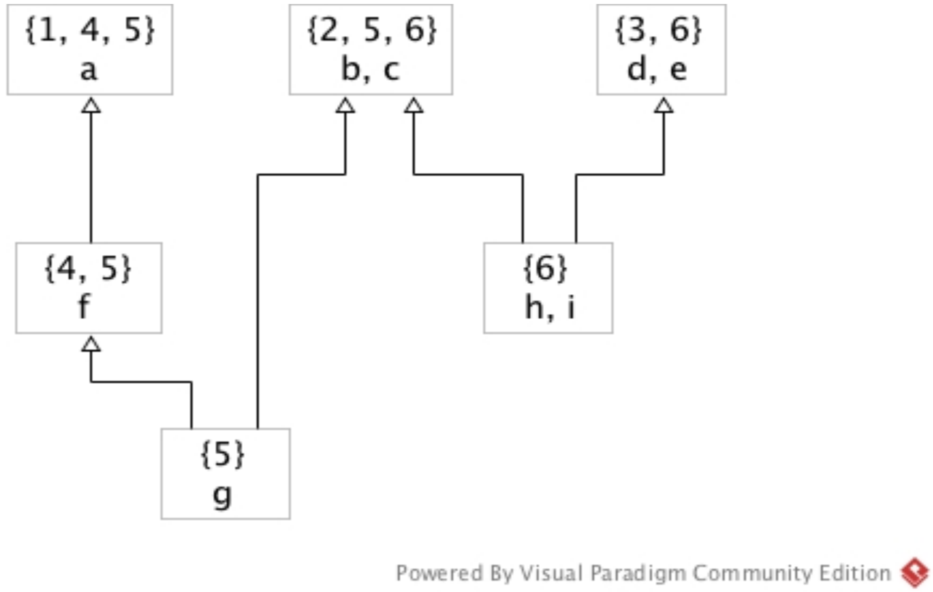
3.



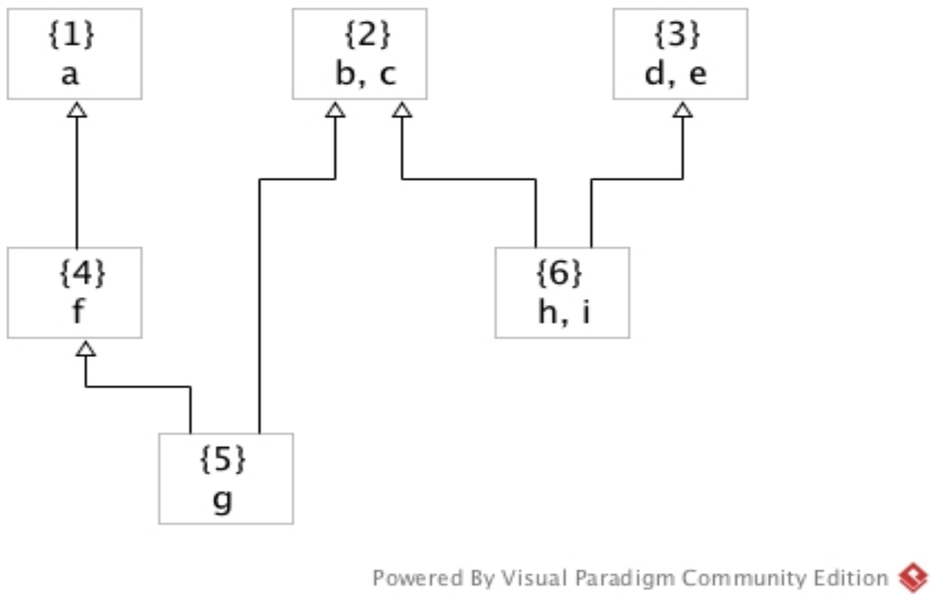
c. Input: Classes 1, …, 6 and the following F_i:

F_1 = {a}
F_2 = {b, c}
F_3 = {d, e}
F_4 = {a, f}
F_5 = {a, b, c, f, g}
F_6 = {b, c, d, e, h, i}

1. C_a = {1, 4, 5}
   C_b = {2, 5, 6}
   C_c = {2, 5, 6}
   C_d = {3, 6}
   C_e = {3, 6}
   C_f = {4, 5}
   C_g = {5}
   C_h = {6}
   C_i = {6}
2.



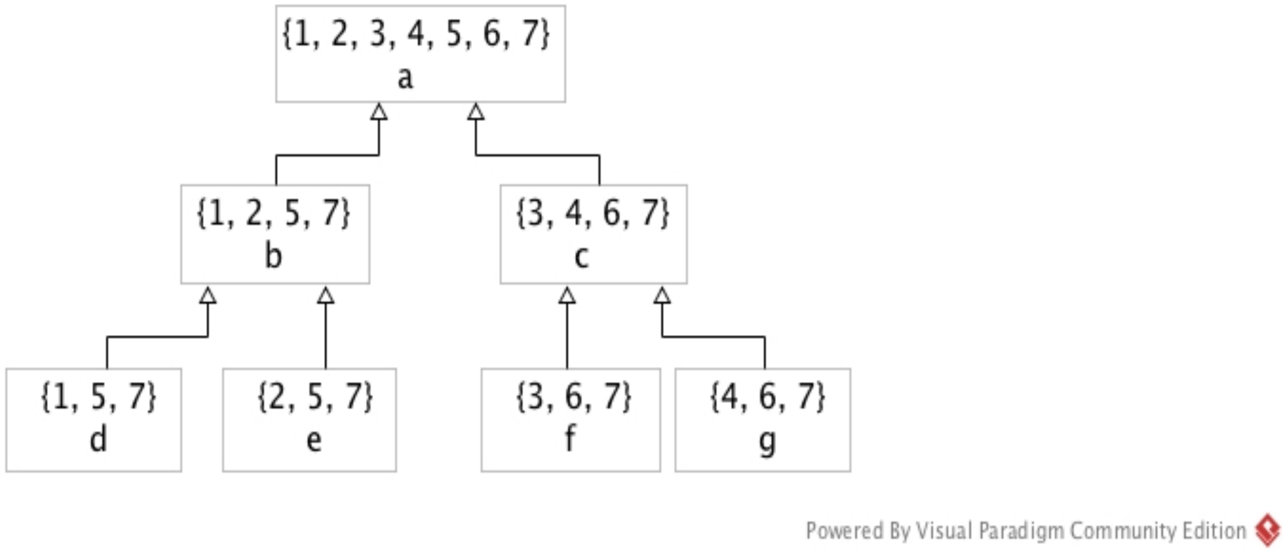
3.



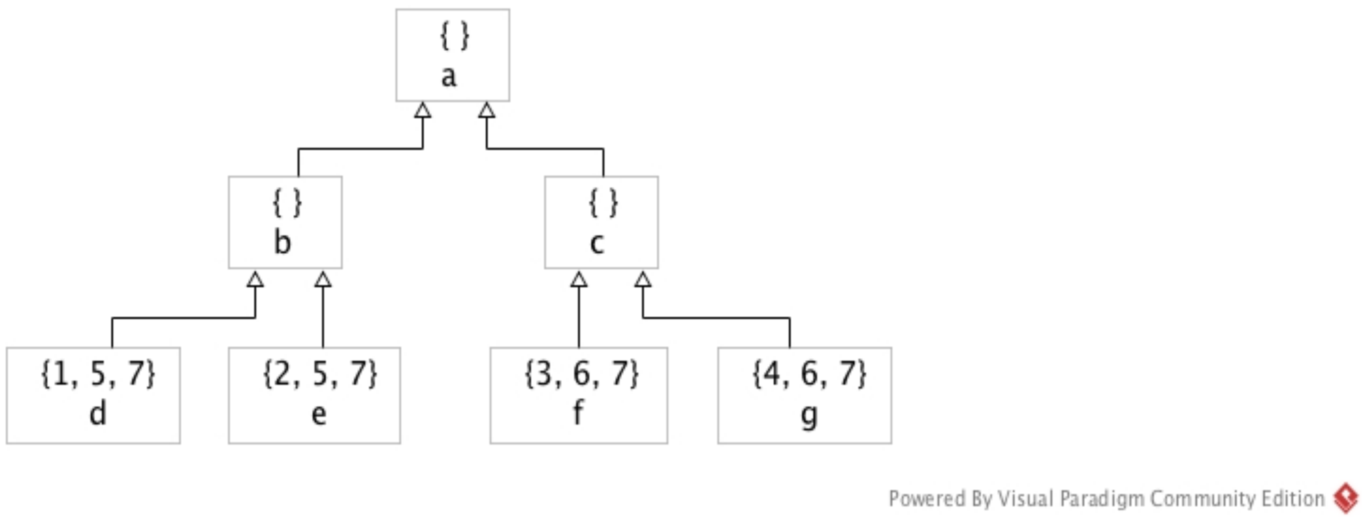
d. Input: Classes 1, …, 7 and the following F_i:

F_1 = {a, b, d}
F_2 = {a, b, e}
F_3 = {a, c, f}
F_4 = {a, c, g}
F_5 = {a, b, d, e}
F_6 = {a, c, f, g}
F_7 = {a, b, c, d, e, f, g}

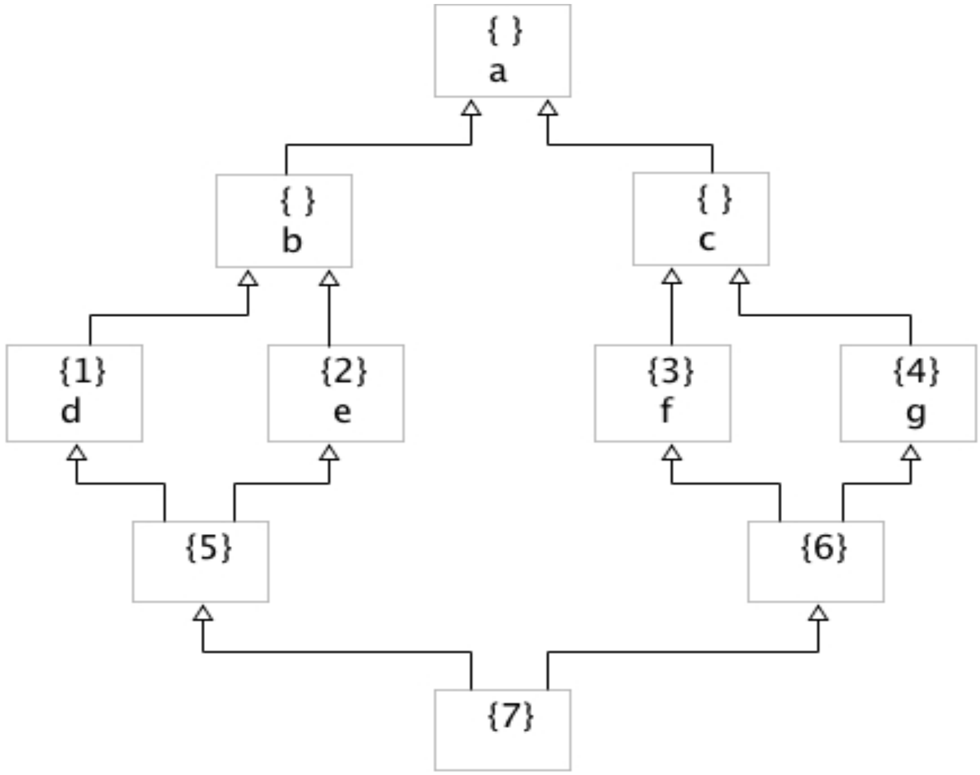
1. C_a = {1, 2, 3, 4, 5, 6, 7}
   C_b = {1, 2, 5, 7}
   C_c = {3, 4, 6, 7}
   C_d = {1, 5, 7}
   C_e = {2, 5, 7}
   C_f = {3, 6, 7}
   C_g = {4, 6, 7}
2.


{1, 2, 3, 4, 5, 6, 7}
a
{1, 2, 5, 7}
b
{3, 4, 6, 7}
c
{1, 5, 7}
d
{2, 5, 7}
e
{3, 6, 7}
f
{4, 6, 7}
g
Powered By Visual Paradigm Community Edition

3.


{ }
a
{ }
b
{ }
c
{1, 5, 7}
d
{2, 5, 7}
e
{3, 6, 7}
f
{4, 6, 7}
g
Powered By Visual Paradigm Community Edition

4.


{ }
a
{ }
b
{ }
c
{1}
d
{2}
e
{3}
f
{4}
g
{5}
{6}
{7}