# Practical 4

Finding the following for a subset S of a given partially ordered set P
i. Whether a given element in P is an upper bound (lower bound) of S or not.
ii. Set of all upper bounds (lower bounds) of S.
iii. The least upper bound (greatest lower bound) of S, if it exists.

## 1

```
(%i1) kill(all);
(%o0) done

(%i1) findRelation(A):=block(
        [A2:cartesian_product_list(A, A), R:[]],
        for i:1 thru length(A2) do(
        t:A2[i],
        if(remainder(t[2], t[1])=0) then R:cons(t, R)
        ),
        R
      );
(%o1) findRelation(A):=block([A2:cartesian_product_list(A,A),
      R:[]],for i thru length(A2) do
      (t:A2_i,if remainder(t_2,t_1)=0 then R:cons(t,R) ),R)

(%i2) A1:[2, 4, 5, 10, 12, 20, 25];
(%o2) [2,4,5,10,12,20,25]

(%i3) R1:findRelation(A1);
(%o3) [[25,25],[20,20],[12,12],[10,20],[10,10],[5,25],[
      5,20],[5,10],[5,5],[4,20],[4,12],[4,4],[2,20],[2,12],[2
      ,10],[2,4],[2,2]]

(%i4) P1:[2, 4];
(%o4) [2,4]
```

## 2

```
(%i5) checkUpperBound(A, R, P, e):=block(
        [s:0],
        for i:1 thru length(P) do(
          if(member([P[i], e], R)) then(s:s+1)
        ),
        if(s=length(P)) then(return(true)) else(return(false))
      );
```

(%o5) checkUpperBound(A,R,P,e):=block([s:0],for i thru length(P) do if member([$P_i$,e],R) then s:s+1 ,if s=length(P) then return(true) else return(false))

```
(%i6) checkUpperBound(A1, R1, P1, 10);
```

(%o6) false

```
(%i7) checkUpperBound(A1, R1, P1, 12);
```

(%o7) true

## 3

```
(%i8) checkLowerBound(A, R, P, e):=block(
        [s:0],
        for i:1 thru length(P) do(
          if(member([e, P[i]], R)) then(s:s+1)
        ),
        if(s=length(P)) then(return(true)) else(return(false))
      );
```

(%o8) checkLowerBound(A,R,P,e):=block([s:0],for i thru length(P) do if member([e,$P_i$],R) then s:s+1 ,if s=length(P) then return(true) else return(false))

```
(%i9) checkLowerBound(A1, R1, P1, 5);
```

(%o9) false

```
(%i10) checkLowerBound(A1, R1, P1, 2);
```

(%o10) true

```
(%i11) P2:[4, 12];
```

(%o11) [4,12]

```
(%i12) checkLowerBound(A1, R1, P2, 2);
```

(%o12) true

## 4

```
(%i13) findUpperBounds(A,  R,  P):=block(
         [s,  t,  C:[]],
         for k:1 thru length(A) do(
           s:0,
           t:A[k],
           for i:1 thru length(P) do(
           if(member([P[i],  t],  R)) then(s:s+1)
           ),
           if(s=length(P)) then(C:cons(t,  C))
         ),
         return(C)
       );
```

(%o13) findUpperBounds(A,R,P):=block([s,t,C:[]],for k thru length(A) do (s:0,t:$A_k$,for i thru length(P) do if member([$P_i$,t],R) then s:s+1 ,if s=length(P) then C:cons(t,C) ),return(C))

```
(%i14) findUpperBounds(A1,  R1,  P1);
```

(%o14) [20,12,4]

```
(%i15) P2:[2,  5];
```

(%o15) [2,5]

```
(%i16) findUpperBounds(A1,  R1,  P2);
```

(%o16) [20,10]

## 5

```
(%i17) findLowerBounds(A,  R,  P):=block(
         [s,  t,  C:[]],
         for k:1 thru length(A) do(
           s:0,
           t:A[k],
           for i:1 thru length(P) do(
           if(member([t,  P[i]],  R)) then(s:s+1)
           ),
           if(s=length(P)) then(C:cons(t,  C))
         ),
         return(C)
       );
```

(%o17) findLowerBounds(A,R,P):=block([s,t,C:[]],for k thru length(A) do (s:0,t:$A_k$,for i thru length(P) do if member([t,$P_i$],R) then s:s+1 ,if s=length(P) then C:cons(t,C) ),return(C))

```
(%i18) findLowerBounds(A1, R1, P1);
```
(%o18) [2]

```
(%i19) findLowerBounds(A1, R1, P2);
```
(%o19) []

```
(%i20) P3:[4, 10, 25];
```
(%o20) [4,10,25]

```
(%i21) findUpperBounds(A1, R1, P3);
```
(%o21) []

```
(%i22) findLowerBounds(A1, R1, P3);
```
(%o22) []

```
(%i23) P4:[4, 20];
```
(%o23) [4,20]

```
(%i24) findLowerBounds(A1, R1, P4);
```
(%o24) [4,2]

## 6

```
(%i25) lub(A, R, P):=block(
         [U:findUpperBounds(A, R, P)],
         if(U=[]) then(return(U)) else(
           t:U[1],
           for i:2 thru length(U) do(
             if(member([U[i], t], R)) then(t:U[i])
           ),
           return(t)
         )
       );
```
(%o25) lub(A,R,P):=block([U:findUpperBounds(A,R,P)],if U= [] then return(U) else (t:$U_1$,for i from 2 thru length(U) do if member([$U_i$,t],R) then t:$U_i$ ,return(t)))

```
(%i26) lub(A1, R1, P1);
```
(%o26) 4

```
(%i27) lub(A1, R1, P2);
```
(%o27) 10

```
(%i28) lub(A1, R1, P3);
(%o28) []

(%i29) lub(A1, R1, P4);
(%o29) 20
```

## 7

```
(%i30) glb(A, R, P):=block(
         [L:findLowerBounds(A, R, P)],
         if(L=[]) then(return(L)) else(
           t:L[1],
           for i:2 thru length(L) do(
             if(member([t, L[i]], R)) then(t:L[i])
           ),
           return(t)
         )
       );
(%o30) glb(A,R,P):=block([L:findLowerBounds(A,R,P)],if L=[
       ] then return(L) else (t:L_1,for i from 2 thru length(L) do if
       member([t,L_i],R) then t:L_i ,return(t)))

(%i31) glb(A1, R1, P1);
(%o31) 2

(%i32) glb(A1, R1, P2);
(%o32) []

(%i33) glb(A1, R1, P3);
(%o33) []

(%i34) glb(A1, R1, P4);
(%o34) 4

(%i35) P5:[2, 4, 12];
(%o35) [2,4,12]

(%i36) glb(A1, R1, P5);
(%o36) 2
```