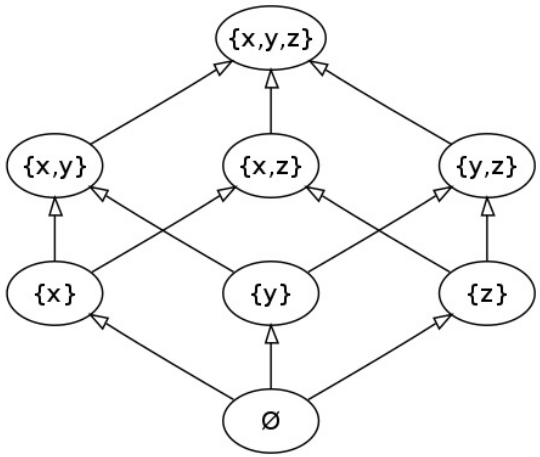
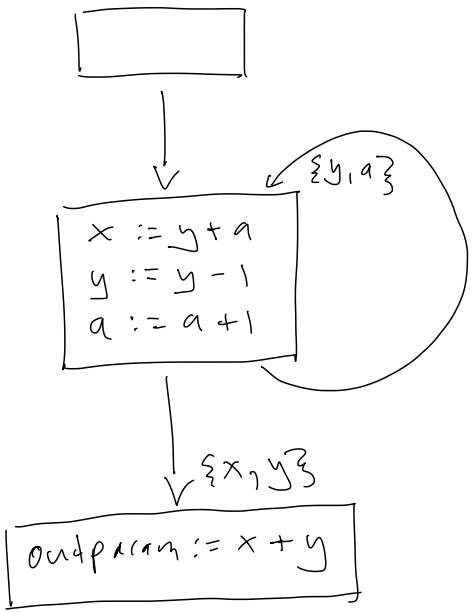


https://en.wikipedia.org/wiki/Partially_ordered_set


{y, a}
x := y + a
y := y - 1
a := a + 1
{x, y}
outparam := x + y

$$\text{IN}[B] = \bigcup_{P \text{ a predecessor of } B} \text{OUT}[P]$$

```
1)  OUT[ENTRY] = ∅;
2)  for (each basic block B other than ENTRY) OUT[B] = ∅;
3)  while (changes to any OUT occur)
4)          for (each basic block B other than ENTRY) {
5)                  IN[B] = ⋃_{P a predecessor of B} OUT[P];
6)                  OUT[B] = gen_B ∪ (IN[B] − kill_B);
            }
```

Figure 9.14: Iterative algorithm to compute reaching definitions

```
IN[EXIT] = ∅;
for (each basic block B other than EXIT) IN[B] = ∅;
while (changes to any IN occur)
        for (each basic block B other than EXIT) {
                OUT[B] = ⋃_{S a successor of B} IN[S];
                IN[B] = use_B ∪ (OUT[B] − def_B);
        }
```

Figure 9.16: Iterative algorithm to compute live variables

```
OUT[ENTRY] = ∅;
for (each basic block B other than ENTRY) OUT[B] = U;
while (changes to any OUT occur)
        for (each basic block B other than ENTRY) {
                IN[B] = ⋂_{P a predecessor of B} OUT[P];
                OUT[B] = e_gen_B ∪ (IN[B] − e_kill_B);
        }
```

Figure 9.20: Iterative algorithm to compute available expressions