unacademy

Code Optimzation Part-1

Complete Course on Compiler Design

Subbarao Lingamgunta • Lesson 21 • Sept 13, 2023

unacademy

Code Optimization

Intermediate Code

⇓

Loop Optimization

⇓

① Loop invariant (code motion)

Ex

```
a=10, b=20, c=30, d=40, e=50
s=7;                f=10;
i=1;
while ( i ≤ 10,00,000)
{
    s = s+10;
    x = a-b
    y = c+d
    i = i+1;
    z = e*f;
}
```

code motion

```
f=60, i=1
d=40, e=50
s=7
a=10, b=20, c=30
x=a-b, y=c+d
z=e*f
while (i ≤ 10,00,000)
{
    s=s+10
    i=i+1
}
```

② Loop Unrolling (or) decreasing comparisions

```
i=1;
while(i <= 10,00,000)
{
    a[i]=i;
    i=i+1;
}
```

⟹

```
i=1
while(i <= 10,00,000)
{
    a[i]=i;      1 3 5
    i++;
    a[i]=i;      2 4 6
    i++;
}
```

unacademy

① j=1
while(j ≤ 10,00,000)
    x = y+2
    z = x+w
    j = j+1

i=1
② while(i ≤ 10,00,000)
    a = b+c;
    b = c+d;
    d = e+g
    i = i+1

i=1
while(i ≤ 10,00,000)
    x = y+2
    z = x+w
    a = b+c
    b = c+d
    d = e+g
    i = i+1

unacademy

induction variable elimination

```
x = 10, y = 20, j = 4, i = 1;
while( i <= 100)
{
    x = y * j;
    i = i + 1;
    j = j + 4;
}
```

⟹

```
x = 10, y = 20, j = 4
while( j <= 400)
{
    x = y * j
    j = j + 4;
}
```

unacademy

Constant Folding

$x = 10 + 20 \flat 3$

$\Downarrow$

$x = 30$

(1)

If (10 < 20) goto hell

$\Downarrow$

goto hell

Copy propagation

$t_2 = t_1$

$t_3 = t_1 \flat t_2$

$t_4 = t_3$

$t_5 = t_3 \flat t_2$

$y = t_5 + t_4$

(2)

$\Longrightarrow$

$t_3 = t_1 \flat t_1$

$t_5 = t_3 \flat t_1$

$y = t_5 + t_3$

Constant Propagation

$\Downarrow$

$p = 360 \quad q = 180$

$x = (p/q) \flat a \Longrightarrow x = 2 \flat a$

(3)

unacademy

Strength Reduction

$2 * x$

$\Downarrow$

$x + x$

$x^2 \Longrightarrow x * x$

$8 * x \Longrightarrow$ Left-Shift 3 times

$x / 8 \Longrightarrow$ RS- 3- times

Dead code Elimination

$a = 50, \quad b = 60$

$t_1 = a * b$

$c = 70 \quad d = 80 \Longrightarrow$

$t_1 = c * d$

print(t_1)

$c = 70 \quad d = 80$

$t_1 = c * d$

print(t_1)

$a = 50, \quad b = 60$

$t_1 = \text{Disp}_{\text{hi}}(a, b)$

$\Longrightarrow$

$a = 50, \quad b = 60$

$t_1 = \text{Disp}_{\text{hi}}(a, b)$

$a = 70 \quad b = 80$

$t_1 = a * b$

print(t_1)

hi

600

500

$a = 70 \quad b = 80$

$t_1 = a * b$

print(t_1)

unacademy

Algebraic Simplification

$x + 0 = x$

$x - 0 = x$

$x * 1 = x$

$x / 1 = x$

$x * 0 = 0$

$0 / x = 0$

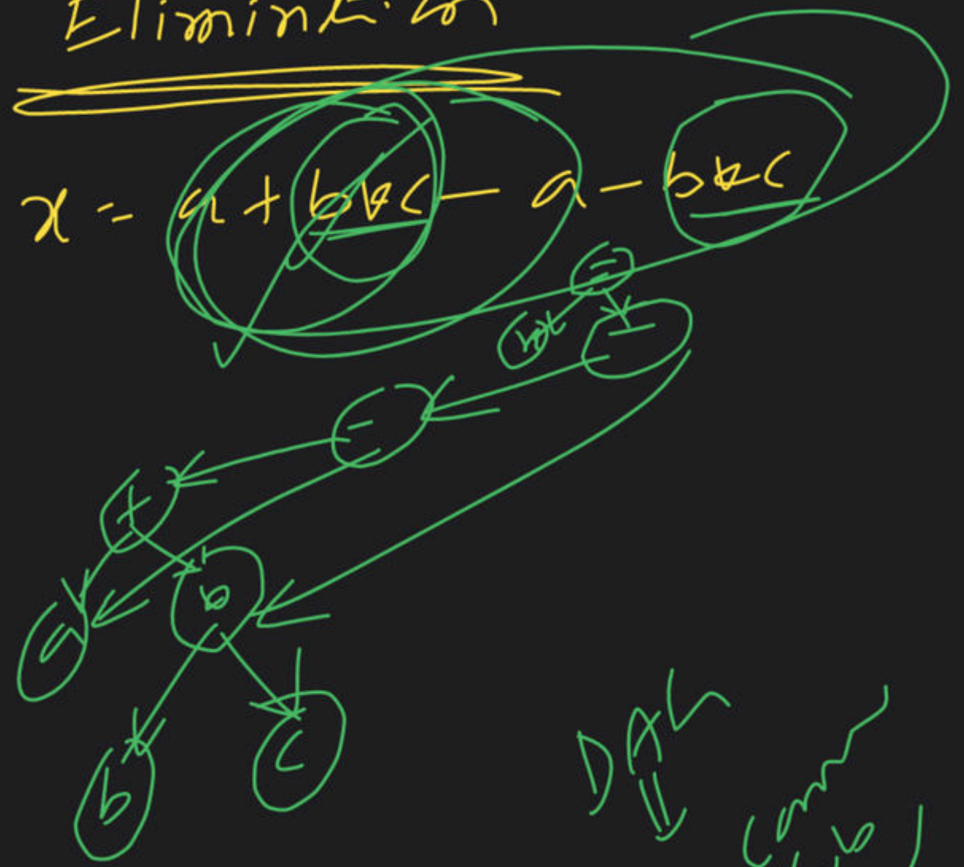
b or True = True

b and True = b

False and True = False

common Subexpression Elimination

$x = a + b * c - a - b * c$



DAG

∴ common subexpression

Code optimization

⇓

Target code

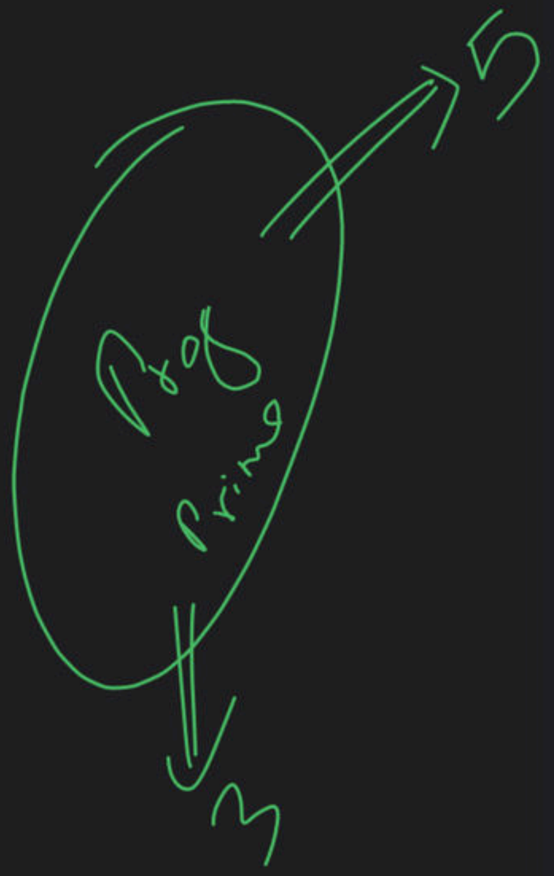
⇓

① Peephole optimization

② Register allocation

⇓

Try to use less register without any change in meaning of it program



Peephole optimization

① Flow control optimization

L_1: goto L_2

L_2: goto L_3 ⟹ L_1: goto L_4

L_3: goto L_4

L_4: ≡

② Removal of unreachable code

#define a 0

if(a)
pf(hi)
else
pf(bi)

⇓

pf(bi)

③ Use 5 machine instructions

```
MOV  R0  i   (R0 = i)
MOV  R1  1   (R1 = 1)        inc i
ADD  R0  R1  (R0 = R0 + R1)
MOV  i   R0  (i = R0)
```

Spilling

④ Redundant Store & Load instructions

Load a R0
Store R0 a

Store R0 a
~~Load a R0~~

5. Algebraic simplification
6. Strength reduction

unacademy

Q

```
z = x+3 + y->f1 + y->f2 ;
for(i=0; i<200; i=i+2)
{
    if(z>i)
    {
        p = p + x+3;
        z = z + y->f1;
    }
    else
    {
        p = p + y->f2;
        z = z + x+3;
    }
}
```

100

Find no. of add & deref operations used

A) 403, 102

B) 203, 2

C) 303, 102

D) 303, 2

⟹ optimized code

```
t1 = x+3
t2 = y->f1
t3 = y->f2
z = t1 + t2 + t3
for(i=0; i<=200; i+=2)
{
    p = p + t1;
    q = q + t2
}
```

100

unacademy

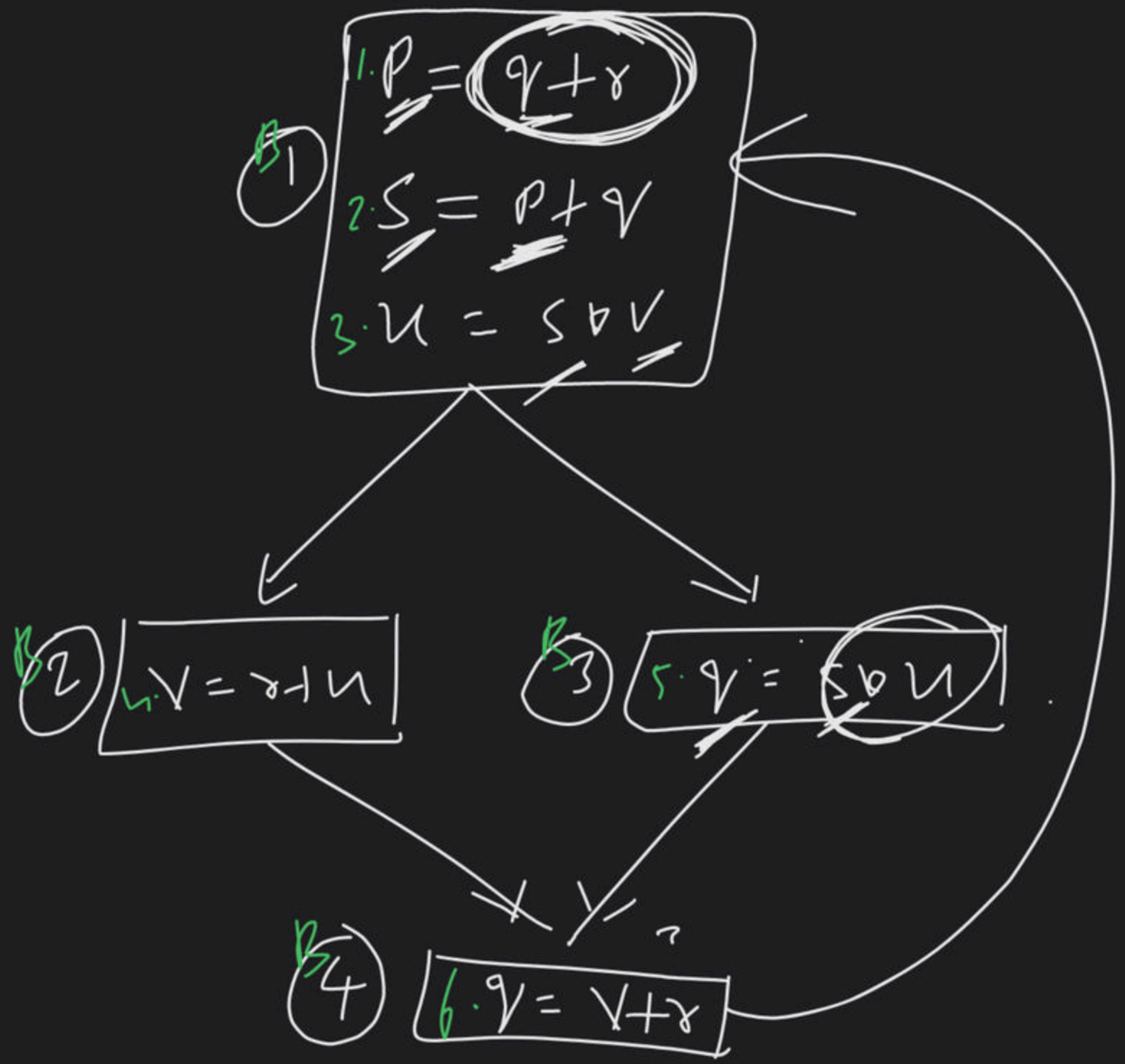
Live Variable Analysis

a Variable x is live at statement i iff

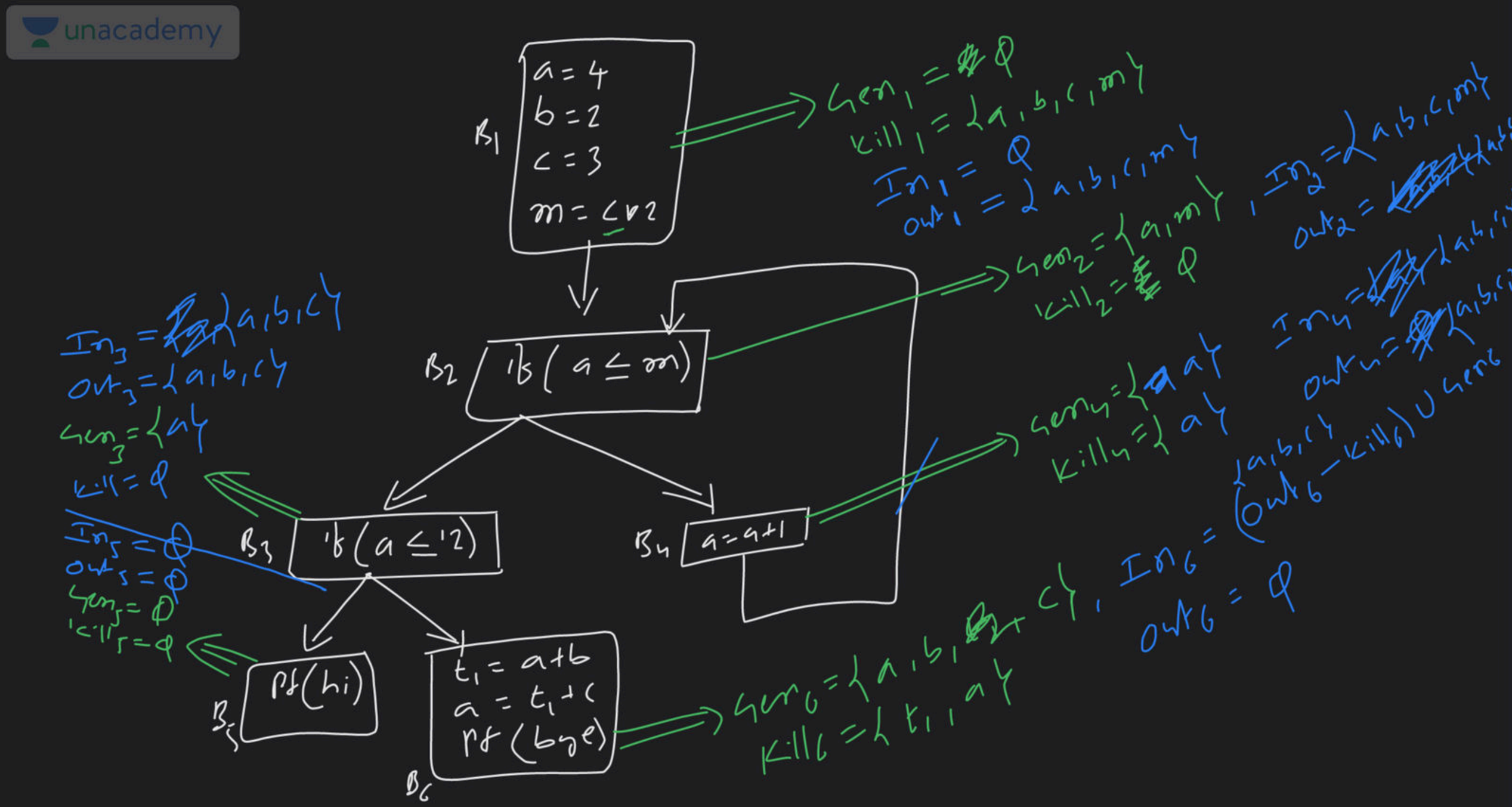
(1) Some statement j should read x

(2) There should be a path from i to j

(3) statement j should read old-x



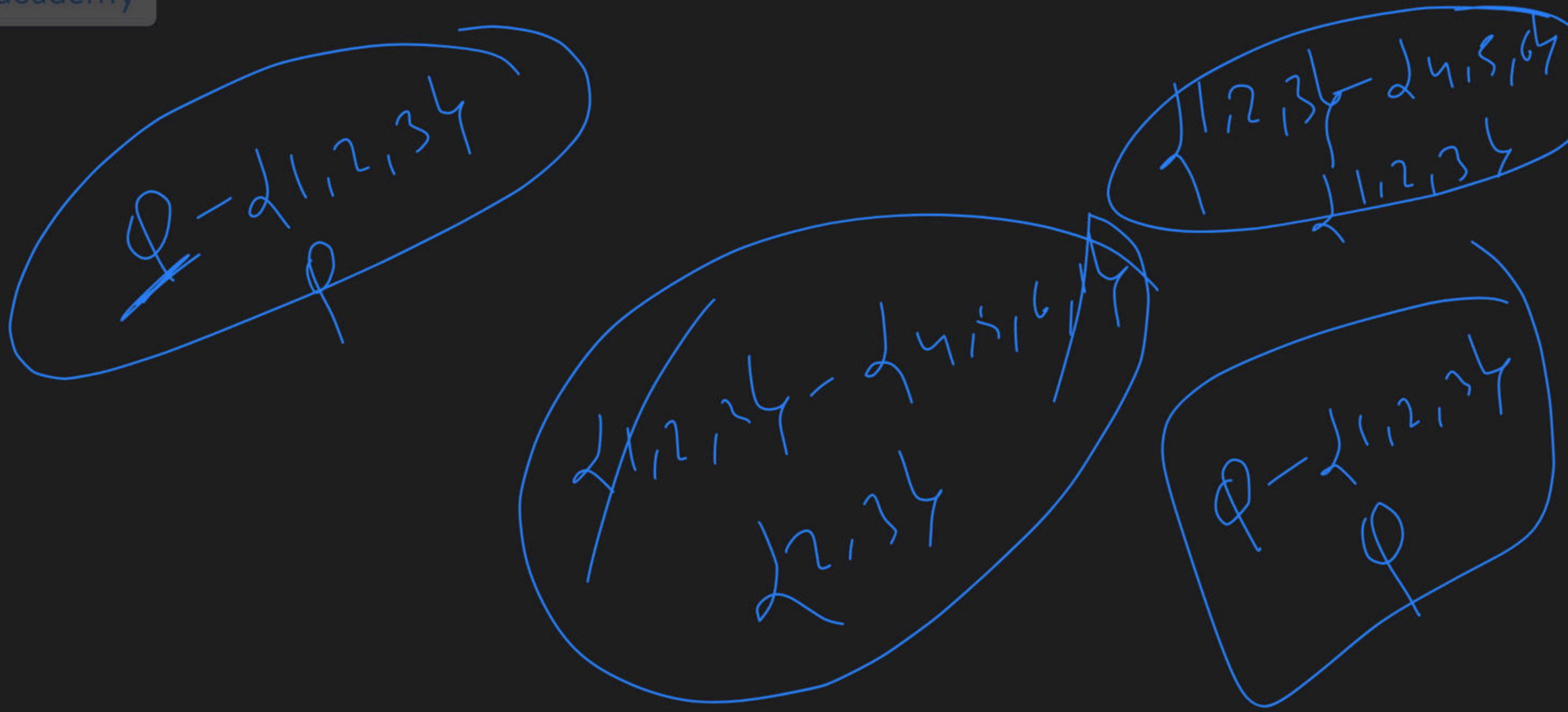
	P	Q	R	S	U	V
1	N	L	L	N	N	L
2	L	L	L	N	N	L
3	N	N	L	L	N	L
4	N	N	L	N	L	N
5	N	N	L	L	L	L
6	N	N	L	N	N	L

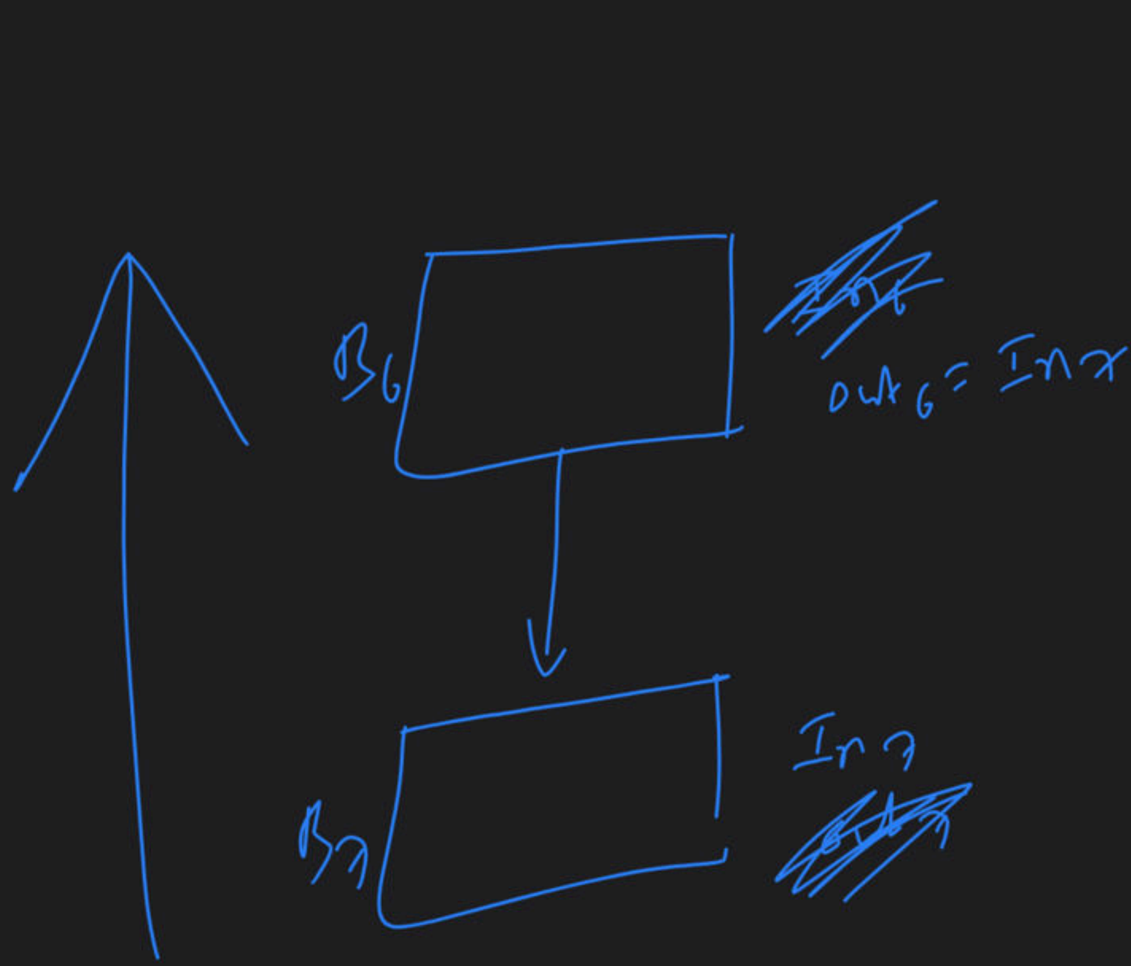

unacademy
B1
a = 4
b = 2
c = 3
m = c+2
B2
if (a ≤ m)
B3
if (a ≤ 12)
B4
a = a+1
B5
pf(hi)
B6
t1 = a+b
a = t1+c
pf(bye)
gen1 = ϕ
kill1 = {a, b, c, m}
In1 = ϕ
out1 = {a, b, c, m}
gen2 = {a, m}
kill2 = ϕ
In2 = {a, b, c, m}
out2 =
In3 = {a, b, c}
out3 = {a, b, c}
gen3 = {a}
kill3 = ϕ
In4 =
out4 =
gen4 = {a}
kill4 = {a}
In5 = ϕ
out5 = ϕ
gen5 = ϕ
kill5 = ϕ
In6 = (out6 − kill6) ∪ gen6
out6 = ϕ
gen6 = {a, b, c}
kill6 = {t1, a}

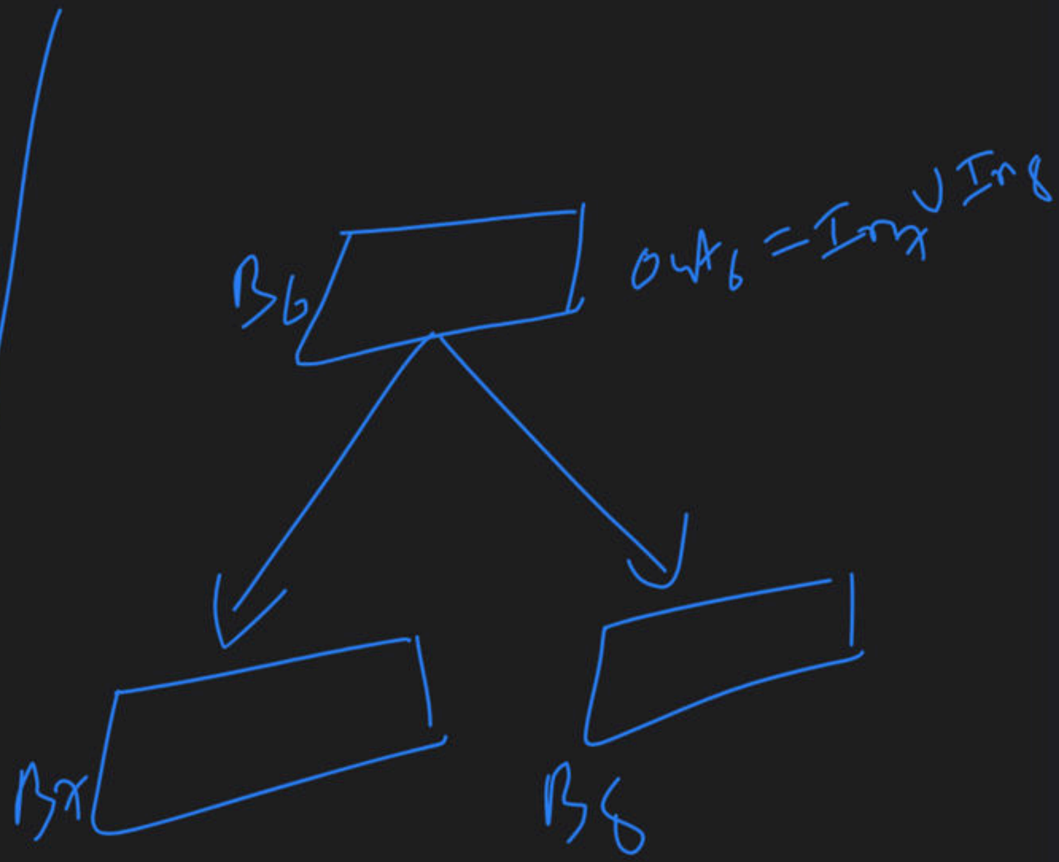
$$In_K = (Out_K - Kill_K) \cup Gen_K$$

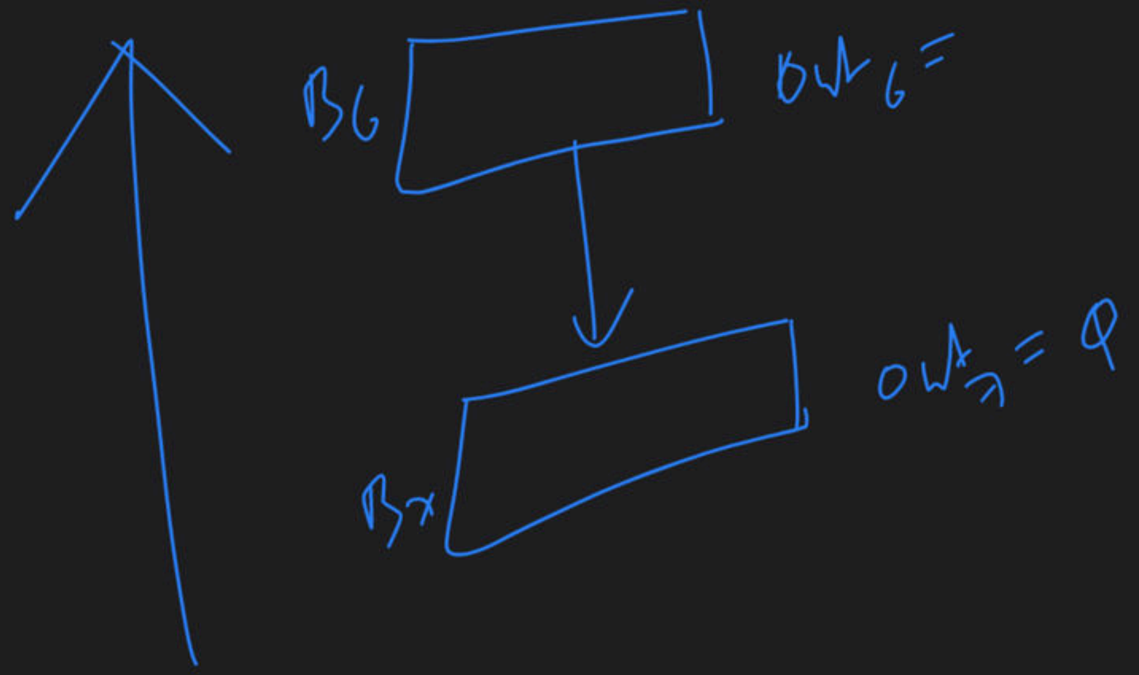
$\varepsilon \rightarrow \varepsilon + \varepsilon$

$\varepsilon \rightarrow \varepsilon + T$








Bg
Bx