(6) (a, b)

$S \rightarrow$ while M_1 B do M_2 S_1

{ backpatch (S_1.nextlist, M_1.instr)
backpatch (B.truelist, M_2.list)
S.nextlist = B.falselist
gen (goto M_1.instr)

$S \rightarrow$ begin L end { S.nextlist = L.nextlist }

$S \rightarrow A$ { S.nextlist = null }

$L \rightarrow LMS$ { backpatch (L.nextlist, M.instr)
L.nextlist = S.nextlist

$L \rightarrow S$ { L.nextlist = S.nextlist }

$A \rightarrow id = E$ { gen (top.get (id.lexeme = E.addr)) }

$E \rightarrow E_1 \pm E_2$ { E.addr = new Temp()
gen (E.addr = E_1.addr $\pm$ E_2.addr) }

$B \rightarrow E_1$ relop E_2 { B.truelist = makelist(nextquad)

B.falselist = makelist(nextquad + 1)

gen(if E_1.addr relop E_2.addr goto —)

gen(goto —)

$B \rightarrow (B)$ | B.truelist = B_1.truelist

B.falselist = B_1.falselist

$B \rightarrow B_1$ && MB_2 | backpatch(B_1.truelist, M.instr)

B.truelist = B_2.truelist

B.falselist = merge(B_1.falselist, B_2.falselist)

$B \rightarrow B_1$ || MB_2 { backpatch(B_1.falselist, M.instr)

B.truelist = merge(B_1.truelist, B_2.truelist)

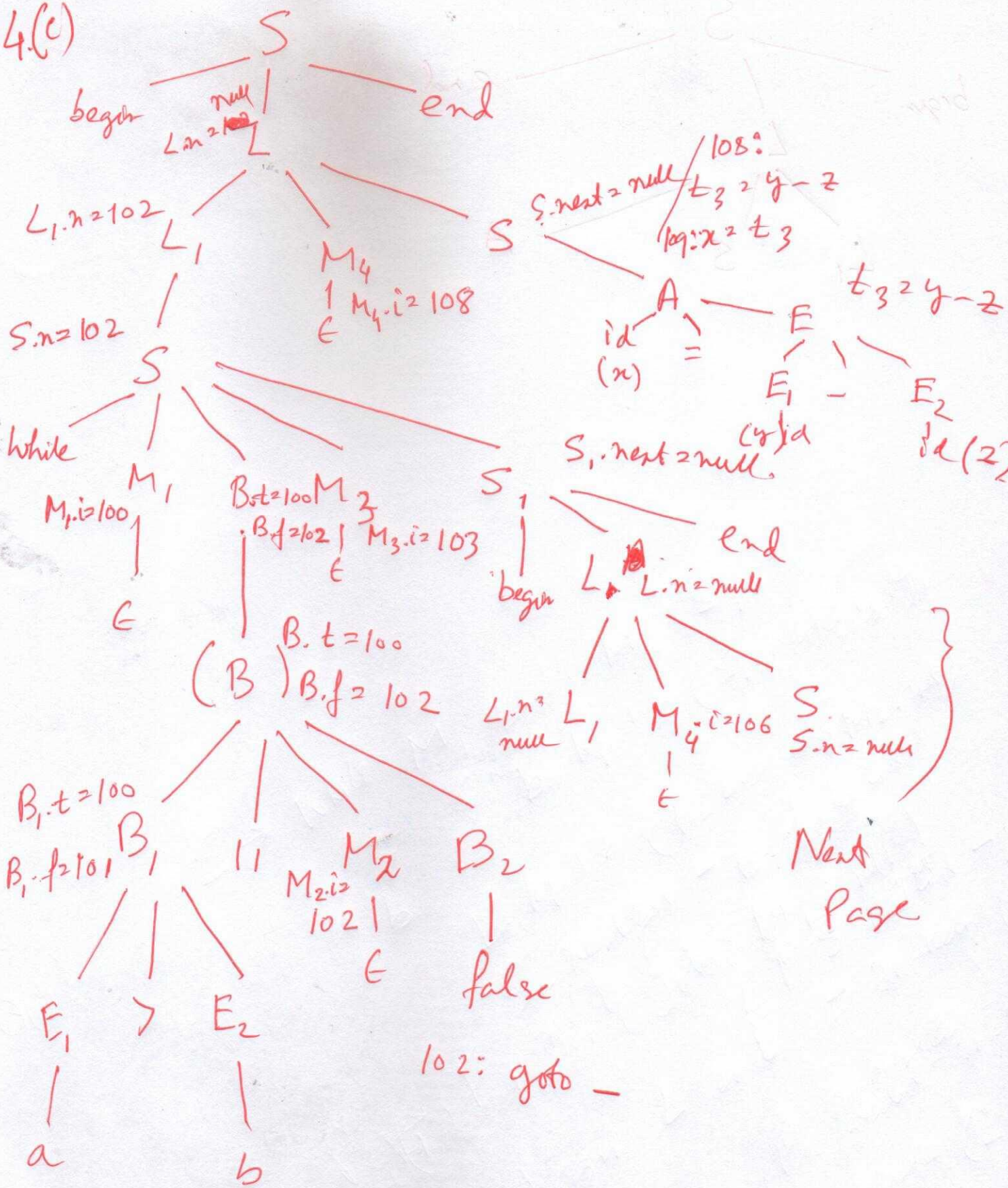
B.falselist = B_2.falselist

B → true | B.truelist = makelist(nextq)
gen(goto _)

B → false | B.falselist = makelist(nextq)
gen(goto _)

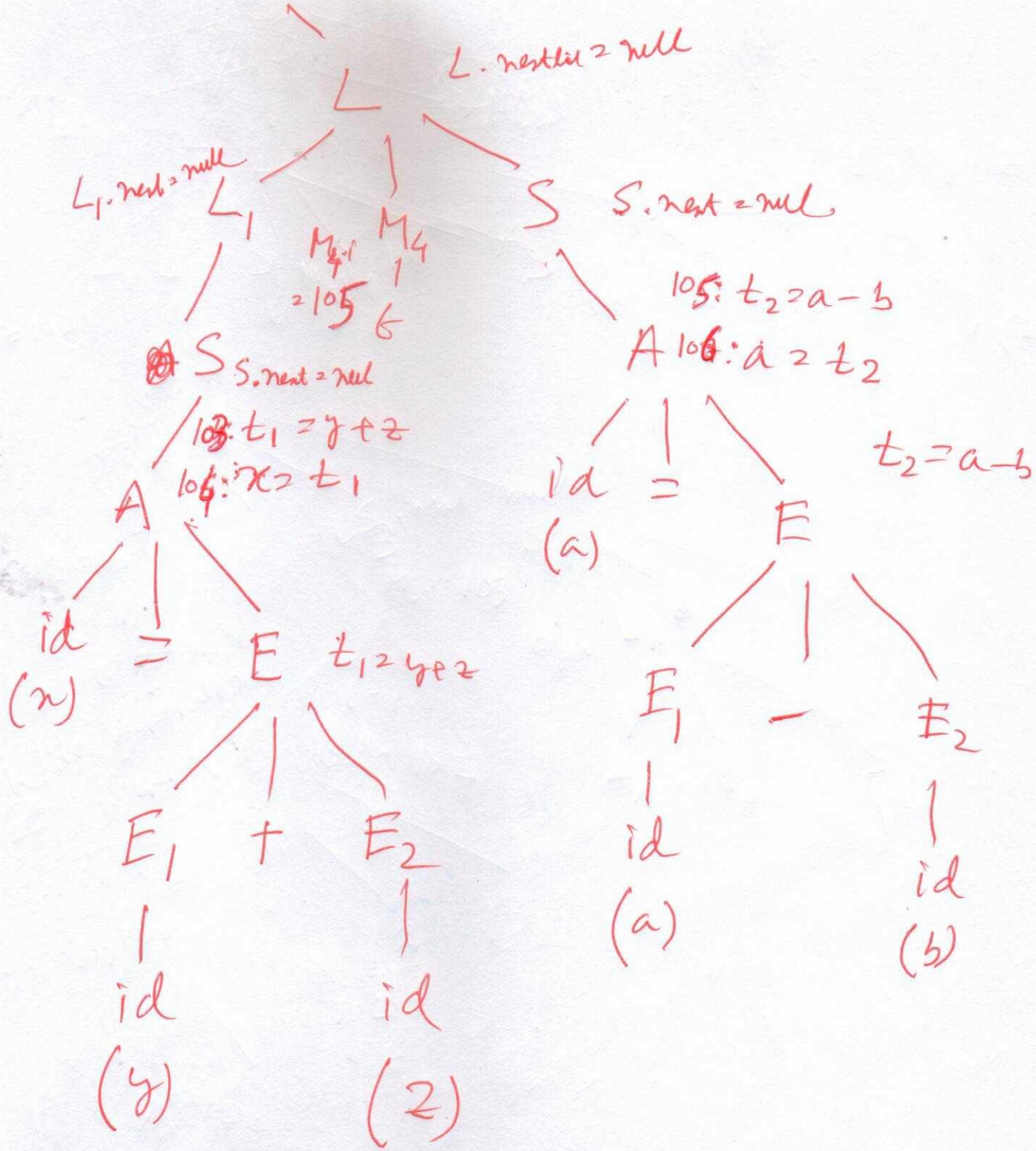
M → ε | M.quad = nextquad

Q4.(c)



100: if a > b goto _

101: goto 102


L
L.nextlist = null
L1
L1.next = null
M4
M4.l = 105
S
S.next = null
S
S.next = null
105: t1 = y + z
106: x = t1
A
id
(x)
=
E
t1 = y + z
E1
id
(y)
+
E2
id
(z)
105: t2 = a - b
A
106: a = t2
id
(a)
=
E
t2 = a - b
E1
id
(a)
-
E2
id
(b)

100: if a > b goto 103

101: goto 102

102: goto 108

103: $t_1 = y + z$

104: $x = t_1$

105: $t_2 = a - b$

106: $a = t_2$

107: goto 100 ← while generates this

108: $t_3 = y - z$

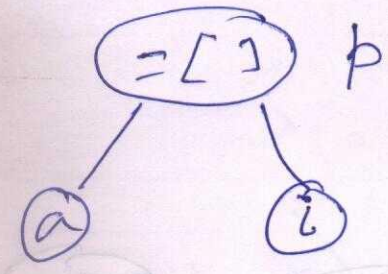
109: $x = t_3$

Q(5)

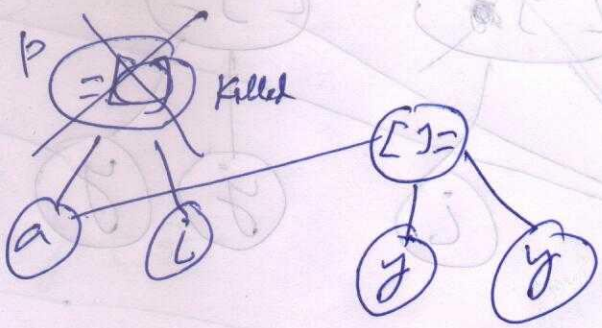
		R_1	R_2	R_3	R_4	a	b	c	d	t_1	t_2	t_3
LD R_1, a LD R_2, b	both R_1, R_2 are free.	a	–	–	–	a, R_1	b	c	d	–	–	–
		a,	b,	–	–	a, R_1	b, R_2	c	d			
SUB R_3, R_1, R_2	Choose R_3 as R_3 is free.	ä	b	t_1	–	a, R_1	b, R_2	c	d	R_3		
b is available in R_2. LD R_4, c	R_4 is free.	a	b	t_1	c	a, R_1	b, R_2	c, R_4	d	R_3		
MULT R_1, R_2, R_4	Choose R_1 for t_2 as a will not be used further.	t_2	b	t_1	c	a	b, R_2	c, R_4	d	R_3	R_1	
ADD R_3, R_3, R_1 Choose R_3 for t_3, as b & c will be used further.	R_1 also could have been used	t_2	b	t_3	c	a	b, R_2	c, R_4	d	–	R_1	R
ADD R_1, R_2, R_4	b, c are available on R_2, R_4. t_3 will be needed if further. Safe option for t_4 is R_1	t_4	b	t_3	c	a	b, R_2	c, R_4	d	–	–	R
ADD R_2, R_3, R_1 ST d, R_2	Choose any Reg for d	t_4	d	t_3	c	a	b	c, R_4	R_2	–	–	R_3

5(c)

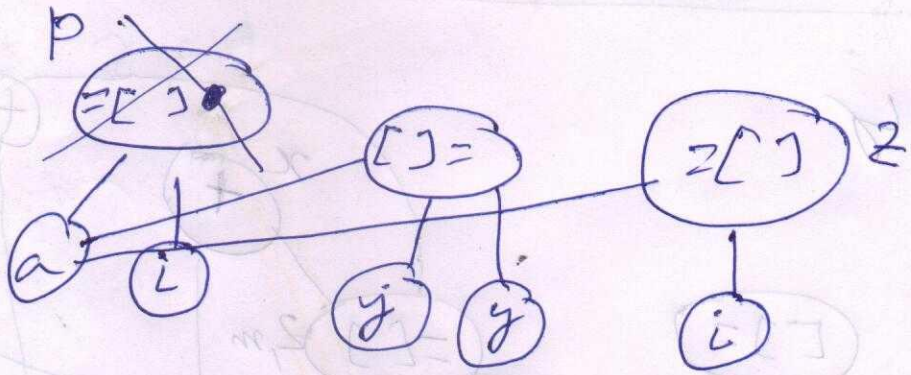
$p = a[i]$



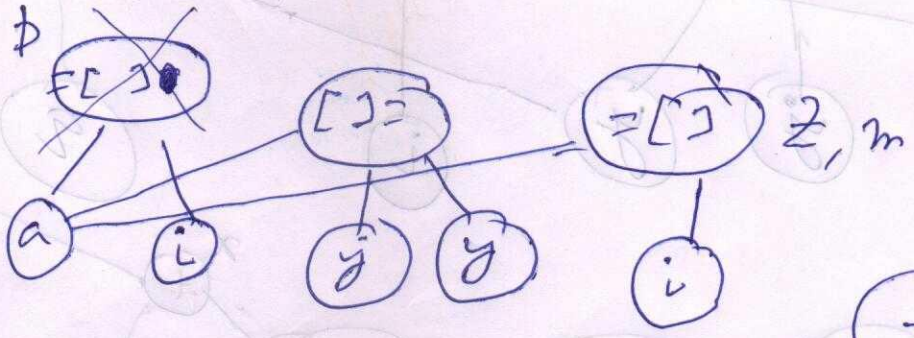
$a[y] = y$



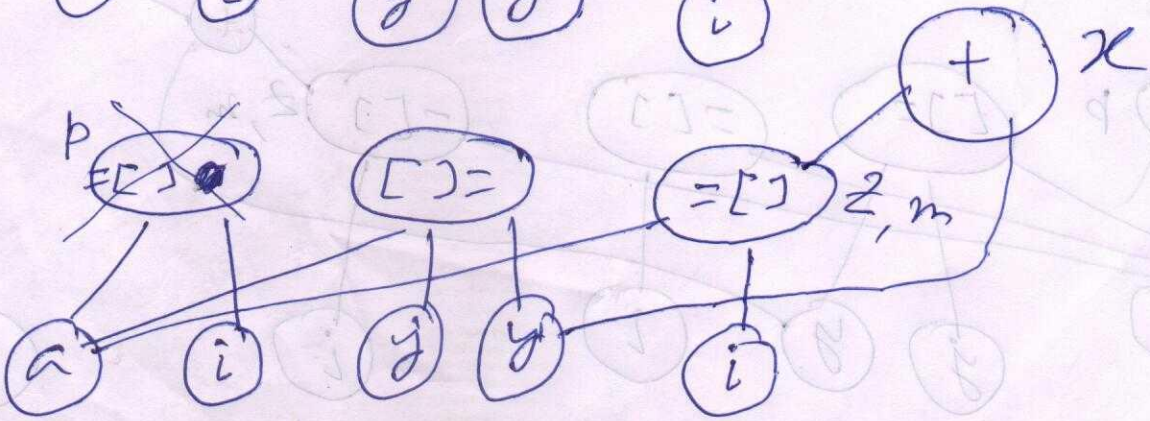
$z = a[i]$



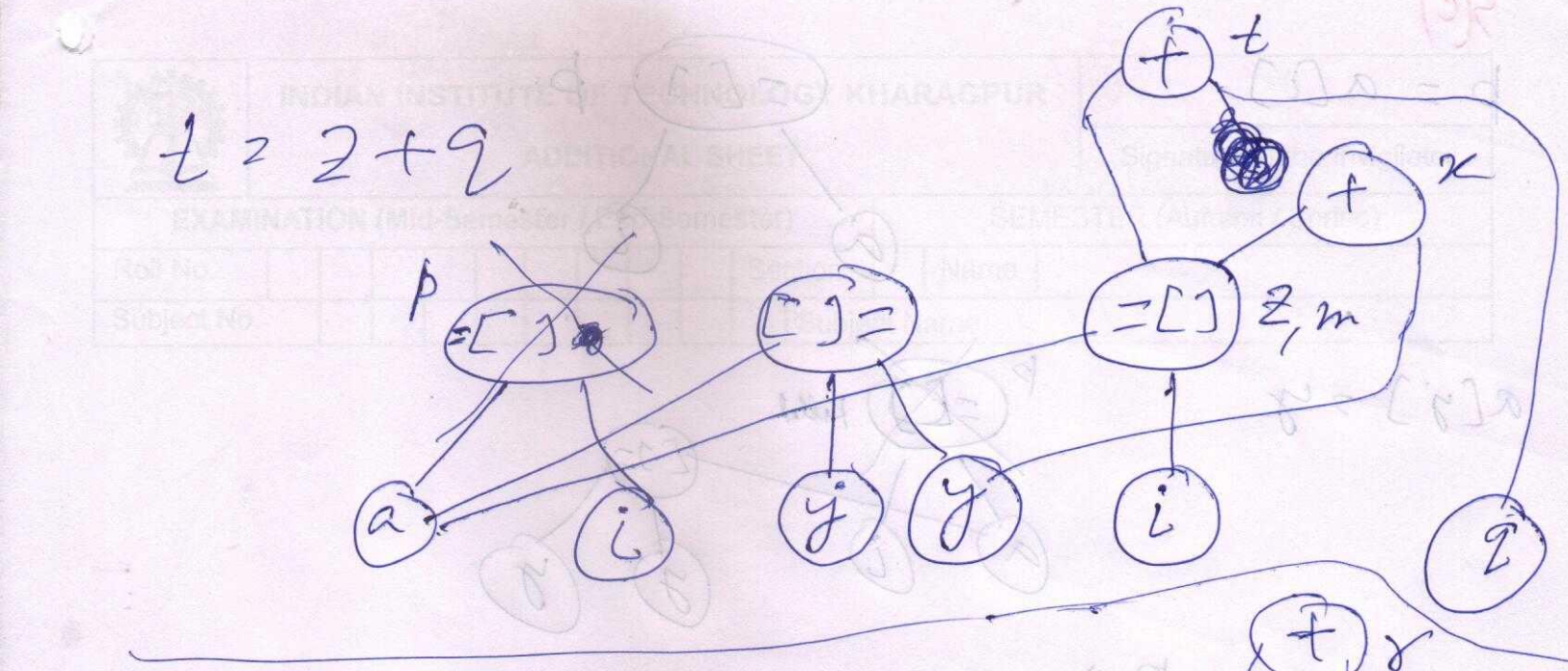
$m = a[i]$



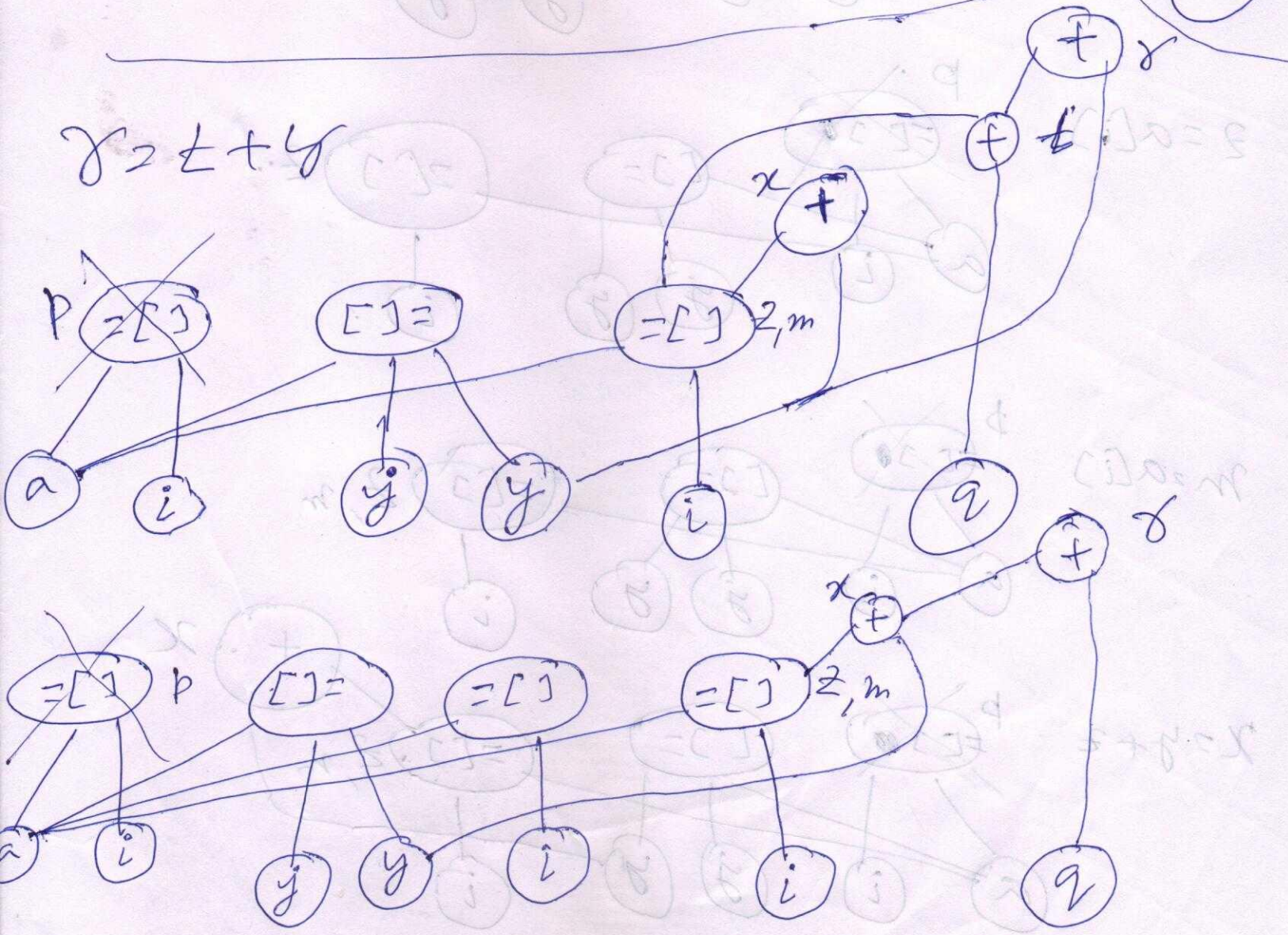
$x = y + z$



$t = z + q$



$y = t + y$



$y = t + y$

$= z + q + y$

$= y + z + q$

$= x + q$

$p = a[i]$

$y = a[j]$

$z = a[i]$

$m = z$

$x = y + z$

$r = x + q$