

$$1. \quad a = b[i] + c[j]$$

3AC

$$t_1 = i \times 4$$

$$t_2 = b[t_1]$$

$$t_3 = j \times 4$$

$$t_4 = c[t_3]$$

$$t_5 = t_2 + t_4$$

$$a = t_5$$

$$2. \ a[i] = b * c + b * d$$

3AC

$$t_1 = i * 4$$

$$t_2 = b * c$$

$$t_3 = b * d$$

$$t_4 = t_2 + t_3$$

$$a[t_1] = t_4$$

** $a[t_1] = t_4$

$op :- \quad [] =$

3. $x = f(y+1) + 2$

3AC

$$t_1 = y + 1$$

param t_1

$$t_2 = \text{call } f, 1$$

// return t_2 (optional)

$$t_3 = t_2 + 2$$

$$x = t_3$$

Quadruple Representation

	op	arg1	arg2	result
0	*	i	4	t ₁
1	=[]	b	t ₁	t ₂
2	*	j	4	t ₃
3	=[]	c	t ₃	t ₄
4	+	t ₂	t ₄	t ₅
5	=	t ₅		a

Quadruple Representation

	op	arg1	arg2	result
0	+	y	1	t_1
1	param	t_1		
2	call	f	1	t_2
3	+	t_2	2	t_3
4	=	t_3		x

Compute type and relative address

int x;
 record { int a;
 float b;
 } P;

offset = 0

T.type = int
 width = 4
 id: x
 top

offset = 0 + 4 = 4

offset = 4 + 12 = 16

T.type = record(top)
 width = 12
 id: P
 top

record {
 P (top)
 x int
 }
 top

offset = 4
 { env.p.p(c)
 top

Env
 x int 0

new Env()

Stack
 4

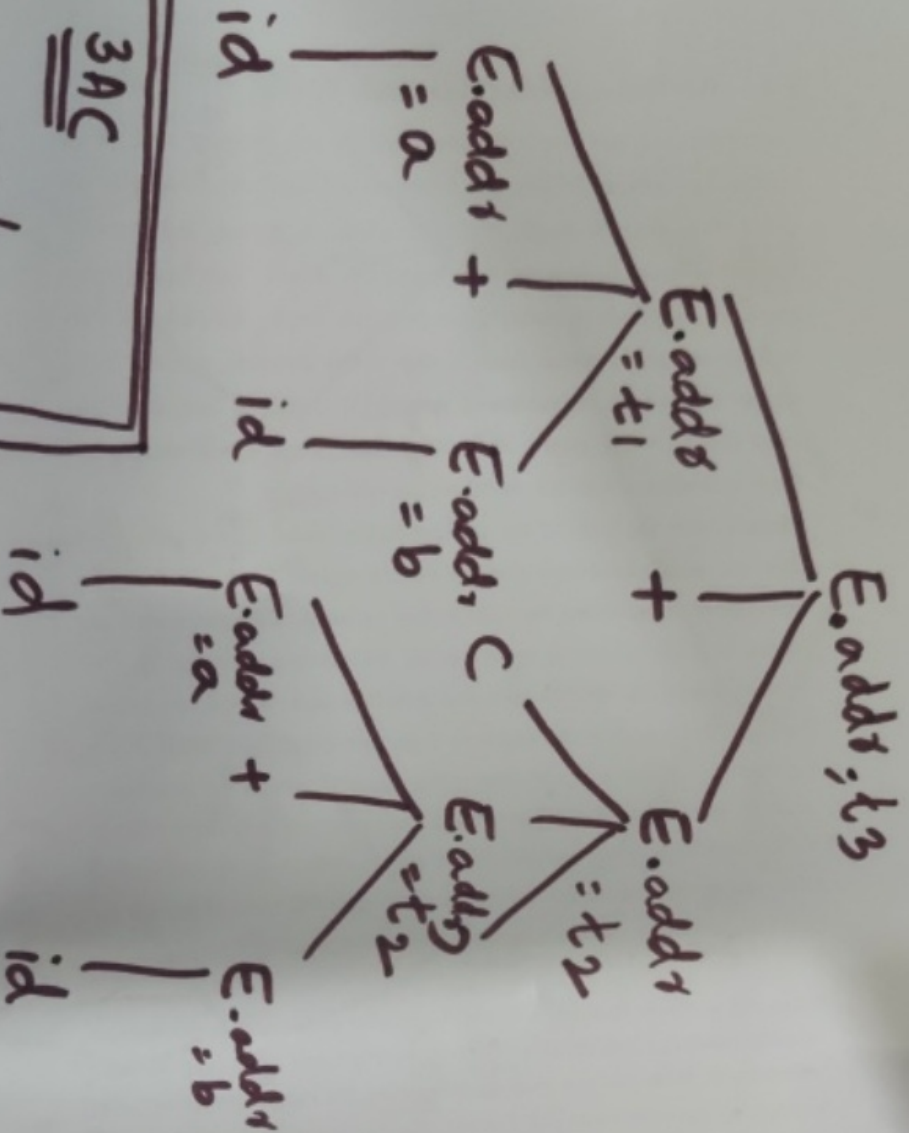
T.type = int
 width = 4
 id: a
 top

offset = 4
 {
 a int 0
 }
 top

T.type = float
 width = 8
 id: b
 top

offset = 12
 {
 float 4
 a int 0
 }
 top

$$a + b + (a + b)$$



3AC

$$t_1 = a + b$$

$$t_2 = a + b$$

$$t_3 = t_1 + t_2$$

Syntax Tree

