Compiler Construction

BITS Pilani
Pilani Campus

Vinti Agarwal
March 2021

innovate achieve lead

BITS Pilani

Pilani Campus

CS F363, Compiler Construction

Lecture topics: Three Address code generation

Code Generation for Arrays expression

A [100]

→ Consider an array element A[i] 0, ..., 99

→ Start location of element A[i] 1, ..., 100

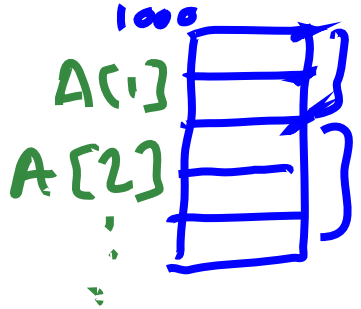
low, high.

⇒ base + (i − low) * w

⇒ (base − low * w) + i * w

constant offset

for w=2



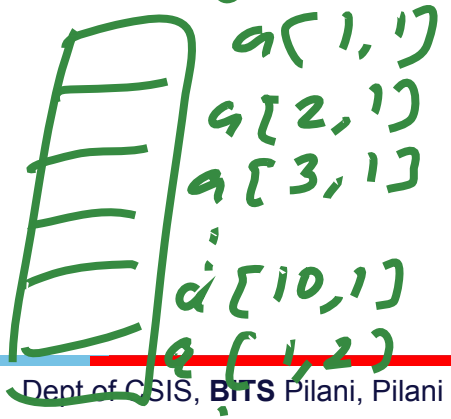
e.g. A[2]

(1000 − 1*2) + 2*2 = 1002

for 2-D Array

i) Row Major Storage

ii) Column major Storage



$a[i_1, i_2]$ (low_1, low_2)

start location of $a[i_1, i_2]$ $n_2 \rightarrow$ no. of colums.

$\Rightarrow$ base $+((i_1 - low_1) * n_2 + i_2 - low_2) * w$

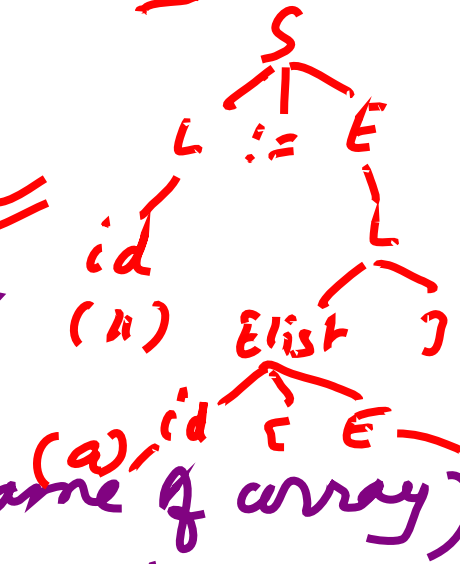
$\Rightarrow$ base $- (low_1 * n_2 - low_2) * w + (i_1 * n_2 + i_2) * w$

constant | offset

Grammer :-

1) $S \rightarrow L := E$
2) $E \rightarrow E + E \mid (E) \mid L$
3) $L \rightarrow$ Elist] | id
4) Elist $\rightarrow$ Elist, E | id [E

$x = a[b]$



Attributes :-

L.place $\rightarrow$ holds name of var (or name of array)

L.offset $\rightarrow$ NULL for simple var (offset of element of array)

E.place → name of var holding value of exp E

Elist.place → name of var holding value of index exp

Elist.array → holds the name of array

Elist.dim → holds the current dim under consideration.

$S \rightarrow L := E$

if L.offset = NULL

gen (L.place := E.place)

else

gen (L.place [L.offset] = E.place

$E \rightarrow E_1 + E_2$

E.place = new Temp( )

E.code = E_1.code || E_2.code || gen(E.place =

$E_1.place + E_2.place$

innovate achieve lead

$E \rightarrow (E_1)$

$E.place = E_1.place$

$E \rightarrow L$

if $L.offset = NULL$

$E.place = L.place$

else $E.place = L.place[L.offset]$

$L \rightarrow id$

$L.place = id.place$

$L.offset = NULL$

$L \rightarrow E\,list\,]$

$L.place = newTemp()$

$L.offset = $ "

gen ($L.place = C(E\,list.array)$

A.offset = Elist.place * width(Elist.array)

innovate achieve lead

Elist → id [E

Elist.array = id.place

Elist.place = E.place

Elist.dim = 1

Elist → Elist1 , E

t = new temp()

m = Elist1.dim + 1

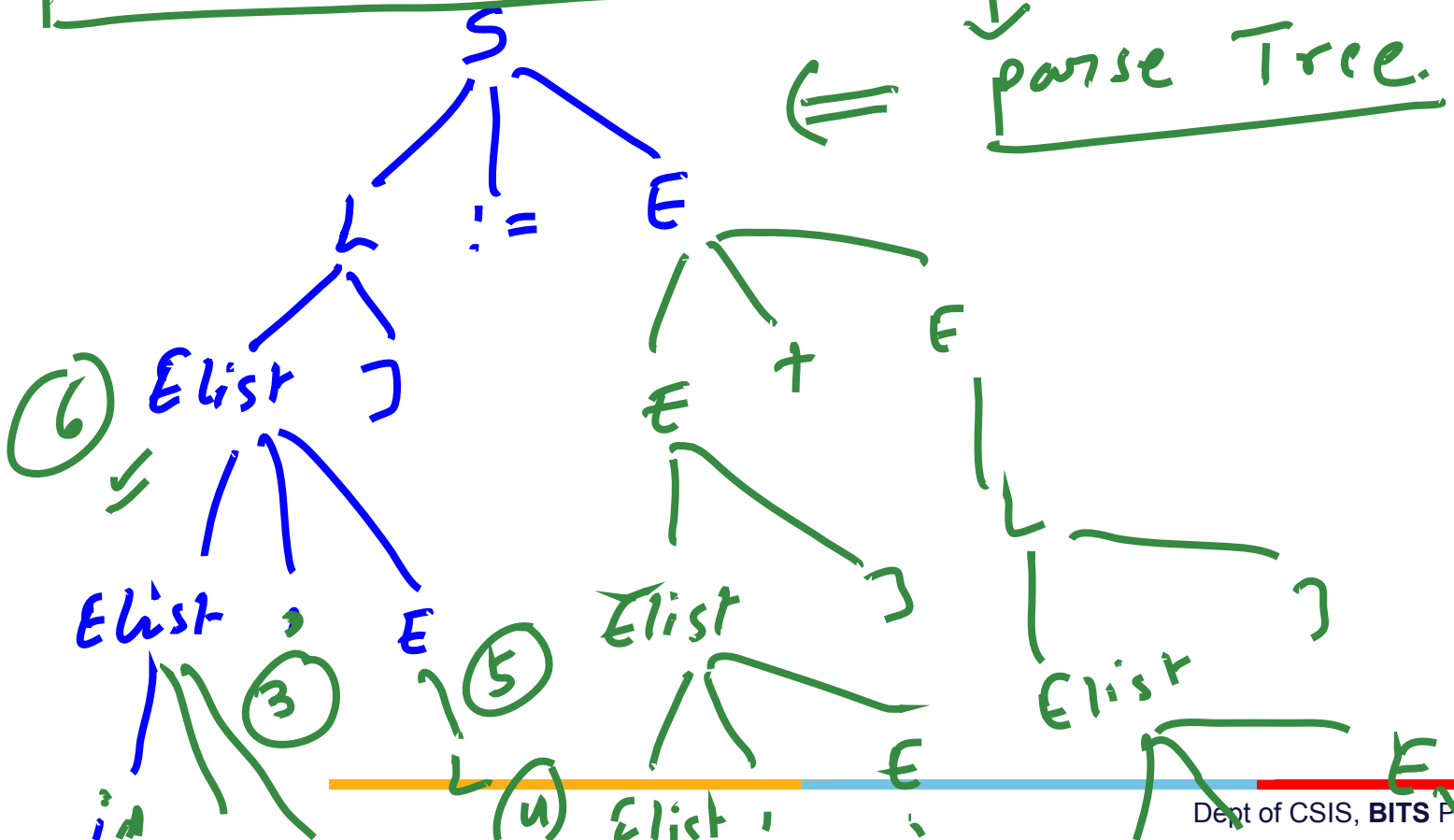
gen(t := Elist1.place * limit(Elist1.array, m))

t = t + E.place

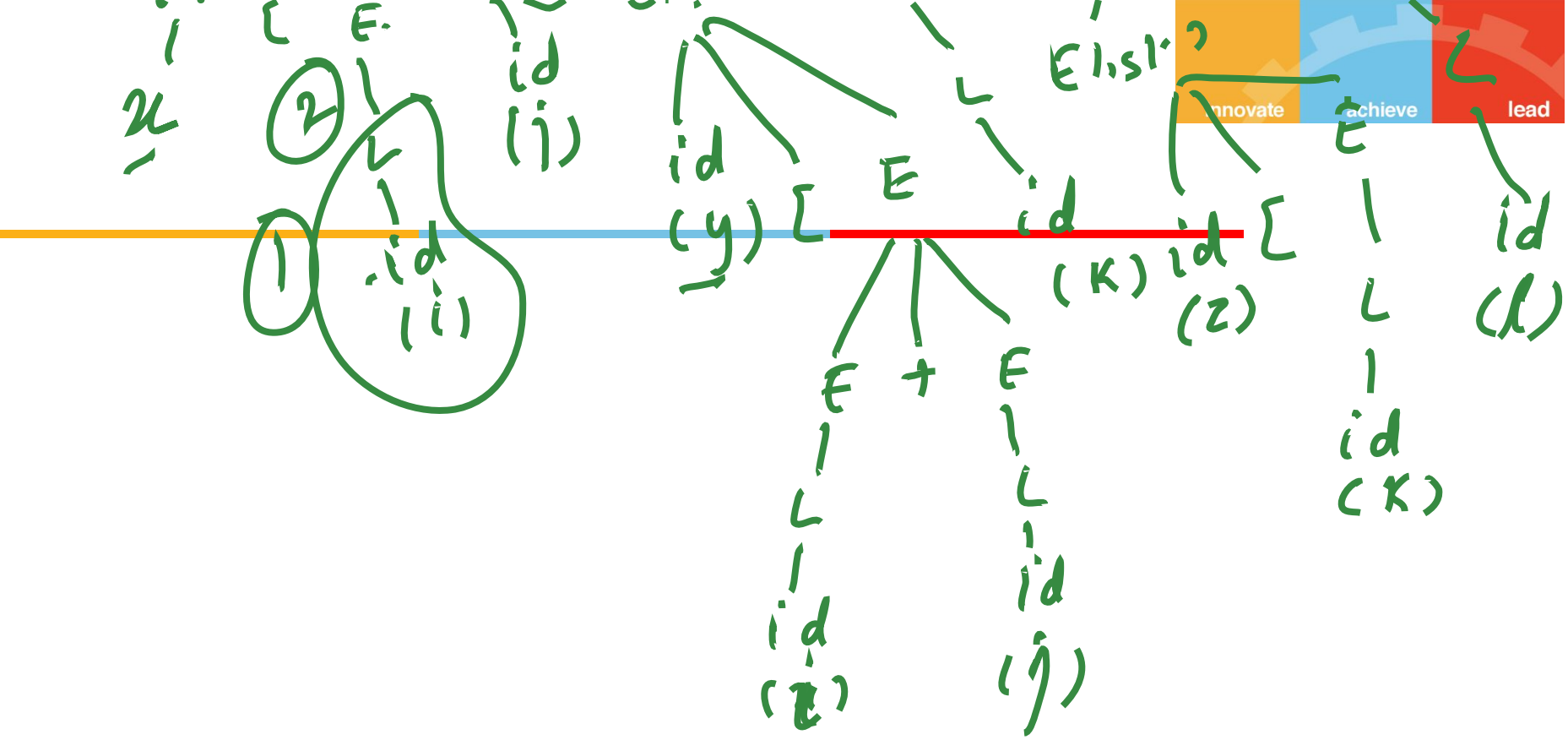
Elist.array = Elist1.array
Elist.place = t
Elist.dim = m

x(d1, d2) y(d3, d4) z(d5, d6)

x[i, j] = y[i + j, k] + z[k, l]   e.g.



⇐ parse Tree.



⇒ Semantic Actions.

(1) { E.place = i
{ E.offset = null

(2) E.place : E.place = t

(3) ·dist.array = x ⟸
·place = i
·dim = 1

(4) h.place = j , L.offset = null.

(5) t.place = j

(6) Elist.array = x
Elist.place = t_1 ⟹
$t_1 = i * limit(x,m)$
$\Rightarrow t_1 = i * d_2$
$t = t_1 + j$

(7) L.place = t_2
L.offset = t_3 ⟹
$t_2 = c(x)$
$t_3 = t_1 * w$

5 h.place = i
L.offset = null

4 .... so on.

final IR code for given 2D array Exp.

$t_1 = i * d_2$
$t_1 = t_1 + j$

$t_2 = c(x)$
$t_3 = t_1 * w$
$t_4 = i + j$

$t_5 = t_4 * d_4$

$t_5 = t_5 + K$

$t_6 = C(Y)$

$t_7 = t_5 * w$

$t_8 = t_6[t_7]$

$t_9 = K * d_6$

$t_9 = t_9 + l$

$t_{10} = C(Z)$

$t_{11} = t_9 * w$

$t_{12} = t_{10}(t_{11})$

$t_2[t_3] = t_{13}$

Thank You!