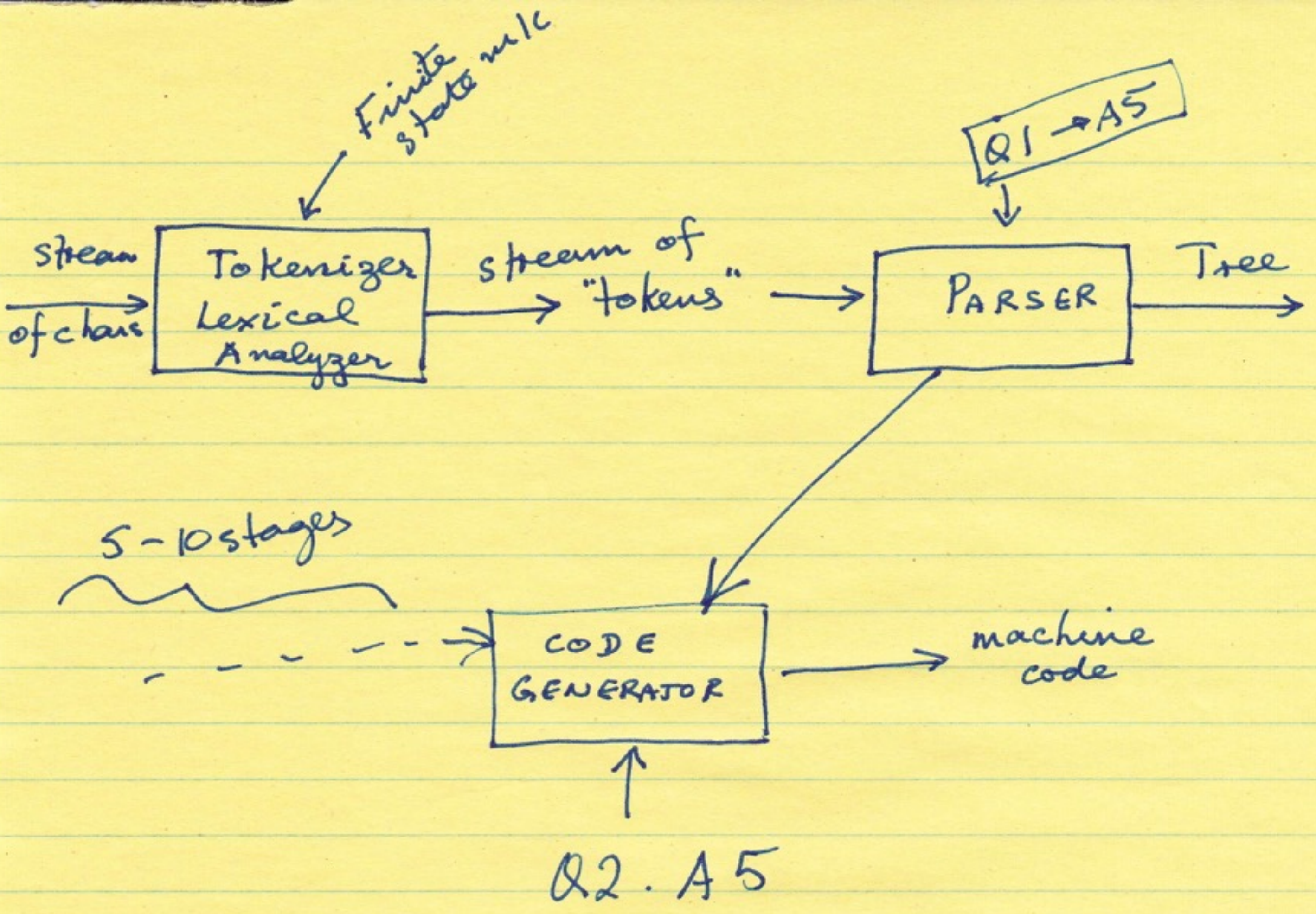


Machine Language

1 Register m/c → ACCUMULATOR

Variable names ⤳ memory locations
Other temp storage ⤳ ints

$$\left.\begin{array}{lll} \text{LOAD} & \text{X} & /\text{LOAD}\ n \\ \text{STORE} & \text{X} & /\text{STORE}\ n \\ \text{ADD} & \text{X} & /\text{ADD}\ n \\ \text{MUL} & \text{X} & /\text{MUL}\ n \end{array}\right\}\ \text{M/c instruction set}$$

no control flow

Code generation is done by
<u>recursively</u> traversing the tree.

$$A \longrightarrow \text{LOAD } A$$

$$\begin{tikzcd}[column sep=small] & + \arrow[dl, no head] \arrow[d, no head] & & \\ B & \textcircled{A} \arrow[rr] & & \text{ADD } A \end{tikzcd}$$

Need to indicate the context.

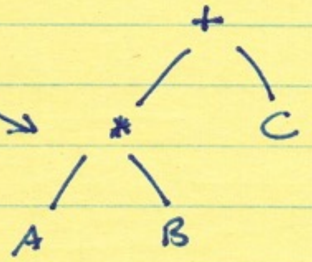
codegen($\underset{\uparrow}{\bullet}$, tree)

some char to
indicate the context.

($) $\underset{\substack{\uparrow \\ \text{fresh}}}{=}$ $\underset{\uparrow}{+}$ $*$

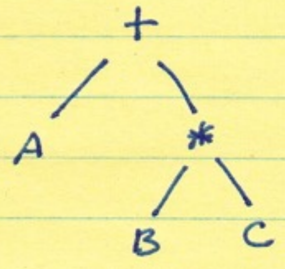
$$\text{codegen}(=, A) \longrightarrow \text{LOAD } A$$
$$\text{codegen}(*, A) \longrightarrow \text{MUL } A$$
$$\text{codegen}(+, A) \longrightarrow \text{ADD } A$$

"A * B + C"



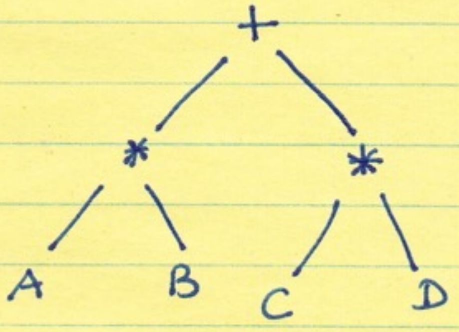
```
LOAD  A
MUL   B
ADD   C
```

A + B * C

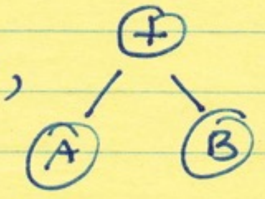


```
LOAD  A
STORE 1
LOAD  B
MUL   C
ADD   1
```

A * B + $\underline{C * D}$

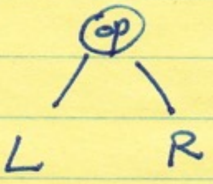


```
LOAD  A
MUL   B
STORE 1
LOAD  C
MUL   D
ADD   1
```

codegen (=, 

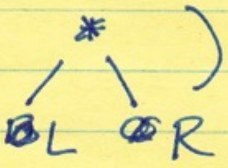
) $\longrightarrow$

codegen (=, A)
Codegen (+, B)

codegen (=, 

) $\longrightarrow$

codegen (=, L)
codegen (op, R)

codegen (~~+~~ op, 

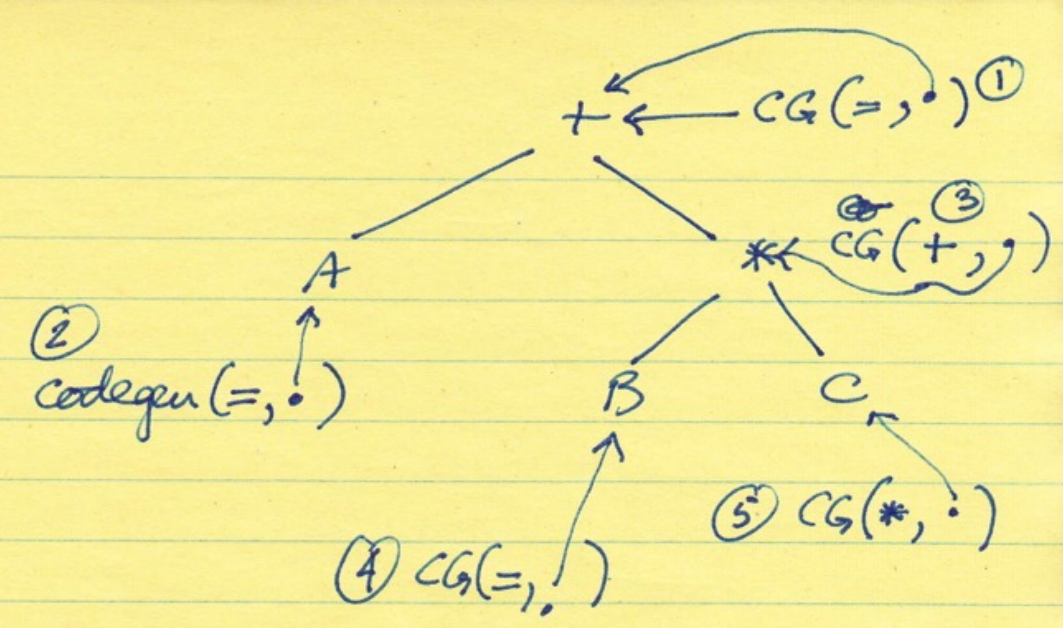
)

temp. storage needed.

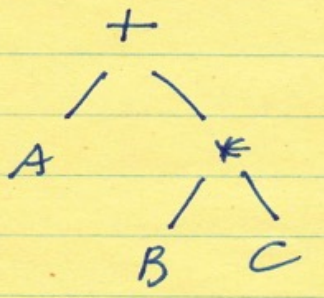
only place where you need memory allocation.

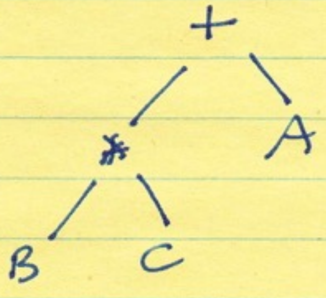
```
memory := memory + 1
output (STORE  memory)
 codegen (=, L)  }  work by
 Codegen (op, R) }    magic
if  op = '+' Then
   output (ADD, memory)
else if op = '*' Then
   output (MUL, memory)
memory := memory - 1
```

$A + B * C$

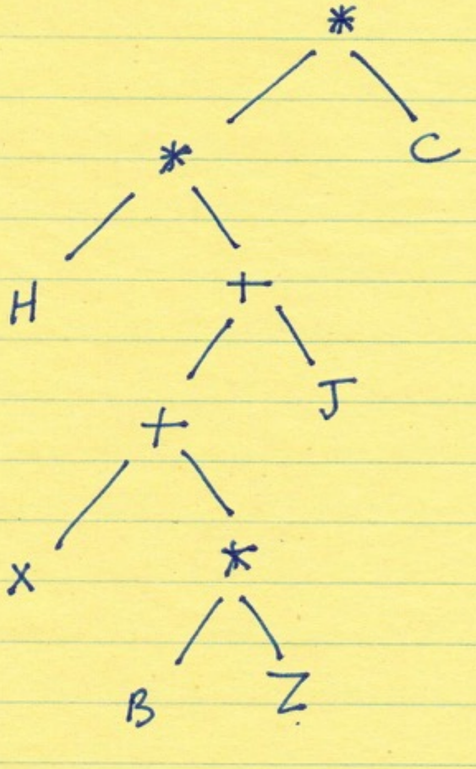


```
LOAD  A
STORE 1
LOAD  B
MUL   C
ADD   1
```
$\}$ 5

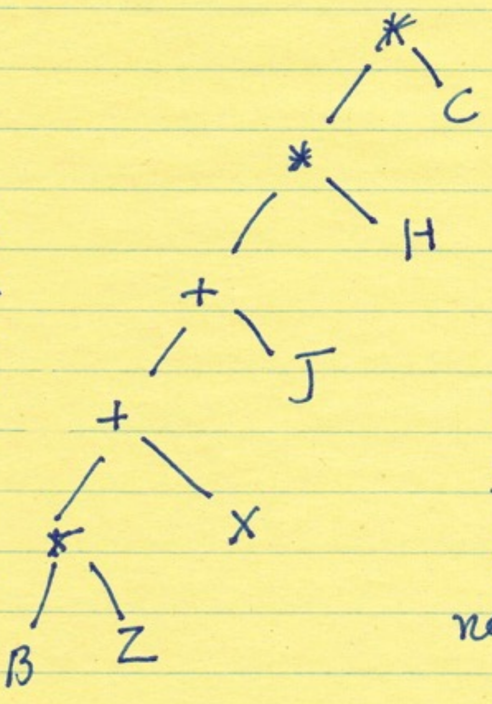




```
LOAD  B
MUL   C
ADD   A
```
$\}$ 3 inst



```
LOAD H
STORE 1
LOAD X
STORE 2
LOAD  B
MUL   Z
ADD   2
ADD   J
MUL   1
MUL   C
```
10 steps
2 extra cells



```
LOAD B
MUL  Z
ADD  X
ADD  J
MUL  H
MUL  C
```
6 steps
no extra memory.