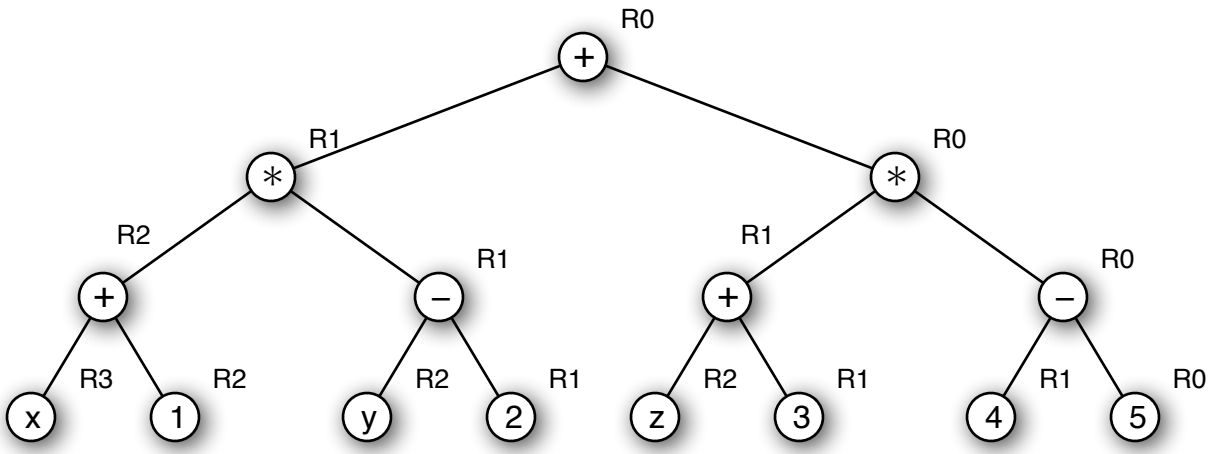
Allocating Registers during Expression Evaluation

Sample tree



Assembly code generated

```
R0=5
R1=4
SUB  R0,R1
COPY R1,R0
; res in R0; R1 free
R1=3
R2=z
ADD  R2,R1
; res in R1; R2 free
MUL  R1,R0
; res in R0; R1 free
R1=2
R2=y
SUB  R1,R2
COPY R2,R1
; res in R1; R2 free
R2=1
R3=x
ADD  R3,R2
; res in R2; R3 free
MUL R2,R1
; res in R1; R2 free
ADD R1,R0
; res in R0; R1 free
```

Notes:

- think of pushing and popping on a stack
- note the reversal of arguments in subtractions
  (could also reverse reg. usage **first**)
- MUL must really be a macro or a subroutine call
- variables must be LOADED, or SET, or DATA'ed
- constants can be SET, but only if they are ≤ 15
- can store result register in the tree node