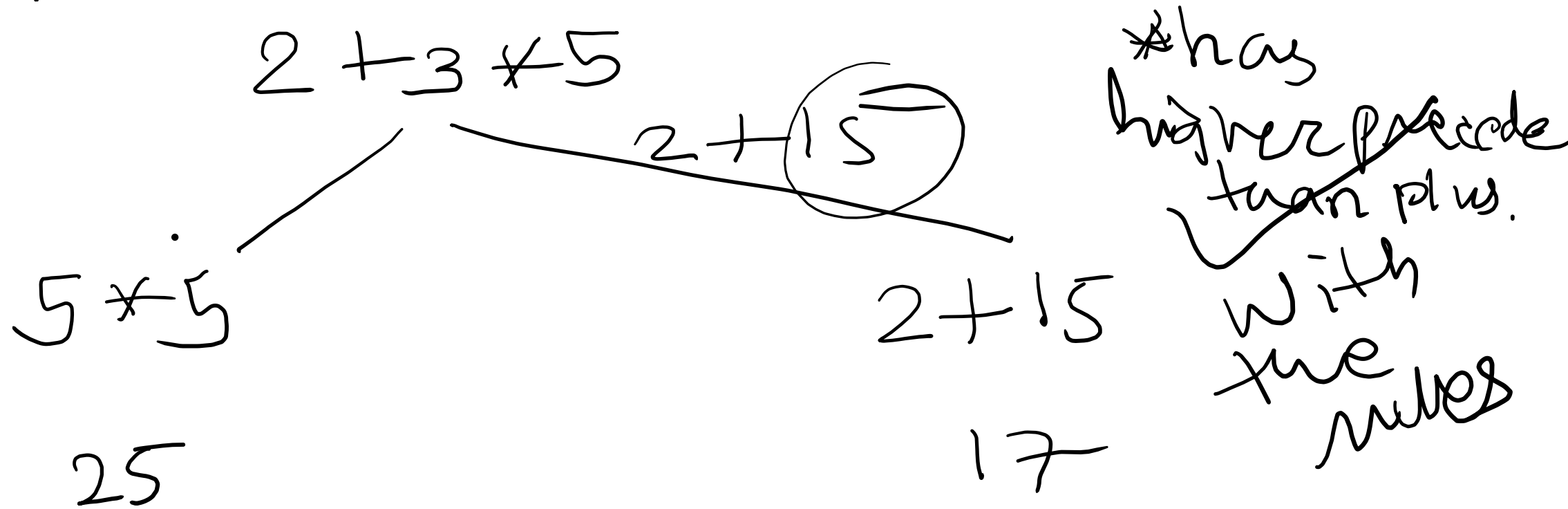


Let's solve it

11

Operator precedence & Associativity



computer has to be deterministic
(Finite Automata)

Associativity:

when there are more than one operator from same level meaning having same priority/rank. At that time computer checks associativity of that level.

i.e.

addition '+'

has left to right associativity

$$\begin{array}{l} L \quad (2+3)+9 \quad R \\ \Rightarrow \quad (5+9) \quad 14 \end{array}$$

$$\begin{array}{l} L \quad (2+3)-9 \quad R \\ \quad 5-9 \quad -4 \end{array}$$

```
int a=10, y=20, z=30;
```

```
x=y=z;
```

```
printf("x is %d", x);
```

```
printf("y is %d", y);
```

```
printf("z is %d", z);
```

~~$x=y$ $x=20$
 $y=20$ $y=30$
 $z=30$~~

Because assignment
operation level
has associativity
right to left.

$y = z ;$
↖

$z = 30$

$y = 30$

x ↖ y

$x = 30$

x 15 30

y 15 30

z 15 30

Precedence has to with priority
which one first
to do
operation

//PUMAS REBL TAC

//P - Parenthesis

//U - Unary

//M - Multiplication

//Af - Addition

//S - Shift

//R - Relational

//E - Equal/Not equal

//B - Bitwise

//L - Logical

//T - Turnary/Conditional

//Al - Assignment

//C - Comma

comparision ==

except shift

Associativity has to do with
direction

left to right

all others

right to left

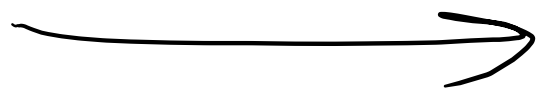
Assignment

Unary

Ternary

ϕ

$$2 - 3 + 1$$



$$\boxed{-1} + 1$$

Binary

Addition binary

$$(-1) + 1$$

0

$$\begin{array}{r}
 520 \\
 \text{Binary} \quad (101)_{2} \Rightarrow (5)_{10} \\
 \quad \quad \quad \frac{000}{000}
 \end{array}$$

$5 \mid 1 \mid 0 \Rightarrow \text{True OR False}$
 logical
 OR \Downarrow
 True

Can you decide result
ahead of time?

Logical AND

True || False || False || True

first false dictates
overall result is
going to be false.

Logical OR

And hence, it does not do
work.

False || True || False || False || False

if you find first True, overall
will be true. Rest no, need
to check.

Write conditions / expressions
in your program accordingly
to make your program
execute faster.

For ~~if~~ write conditions in the ~~begin~~ For //
which tend to ~~begin~~ write conditions
or have more chances of False in the beginning
which have more chances of True

Type Casting

Data type conversion

```
int a;    float f = 30.50;
```

```
a = f;
```

// Down sizing

Data loss

Because ³⁰

float is more bytes,

The int will be truncated.

~~float~~ f; int a = 10;

f = a; // Upstair

f = 10.00

Initialization
default

10. random ?
garbage ?