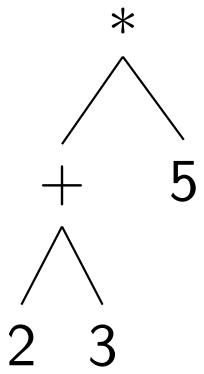
RPN Calculator

Geoffrey Matthews

Department of Computer Science
Western Washington University

November 16, 2018

Expressing arithmetic trees

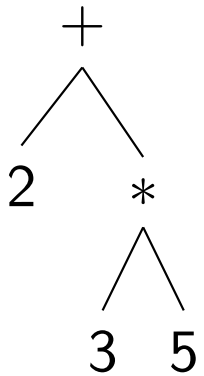



```
((2 + 3) * 5)

2 + 3 * 5          <= Infix

* + 2 3 5          <= Prefix

2 3 + 5 *          <= Postfix
```



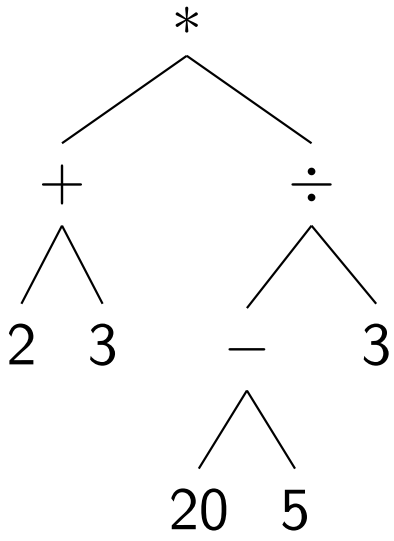

```
(2 + (3 * 5))

2 + 3 * 5          <= Infix

+ 2 * 3 5          <= Prefix

2 3 5 * +          <= Postfix
```

Expressing arithmetic trees



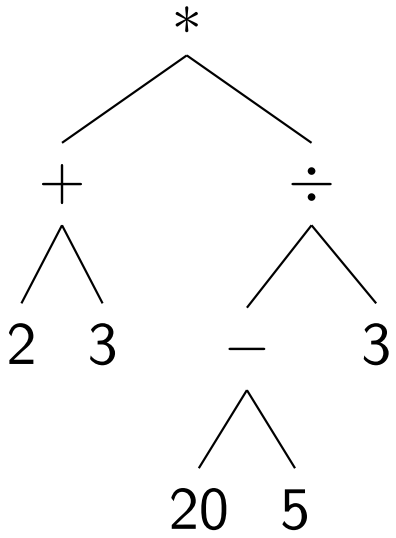

```
((2 + 3) * ((20 - 5) / 3))
```

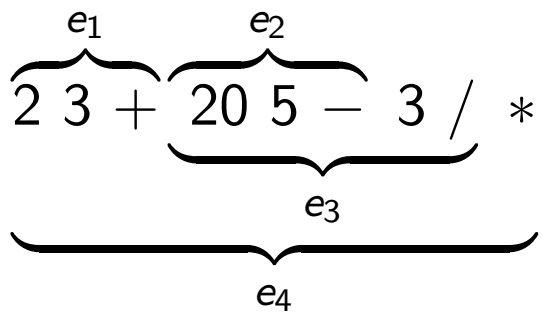
```
2 + 3 * 20 - 5 / 3
```

```
* + 2 3 / - 20 5 3
```

```
2 3 + 20 5 - 3 / *
```

Parsing postfix arithmetic expressions

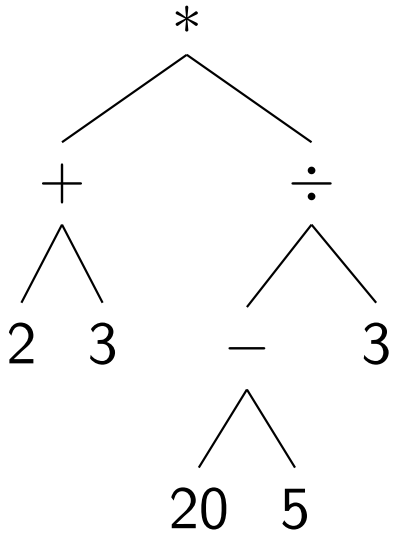




- Postfix grammar is LL(1):

$$D \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid \ldots$$
$$N \rightarrow DN \mid D$$
$$O \rightarrow + \mid - \mid * \mid /$$
$$E \rightarrow N \mid E\ E\ O$$

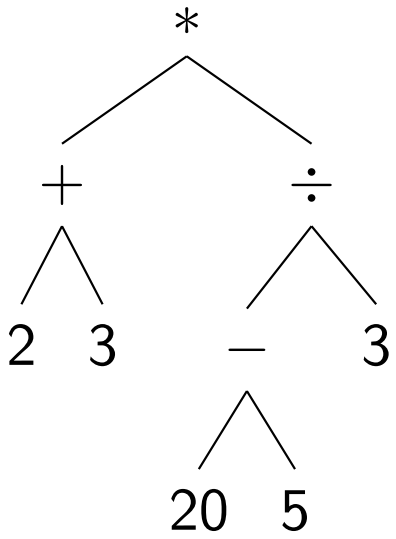
Evaluating arithmetic trees




```
2 3 + 20 5 - 3 / *
```

- Postfix makes it easy to build an interpreter.
- We assume we have a stack for numbers.
- If it's a number, push onto a stack.
- If it's an operator:
  - Pop the top two elements of the stack
  - Apply the operator to the two numbers
  - Push the result onto the stack
- The answer will be on top of the stack.

Evaluating arithmetic trees

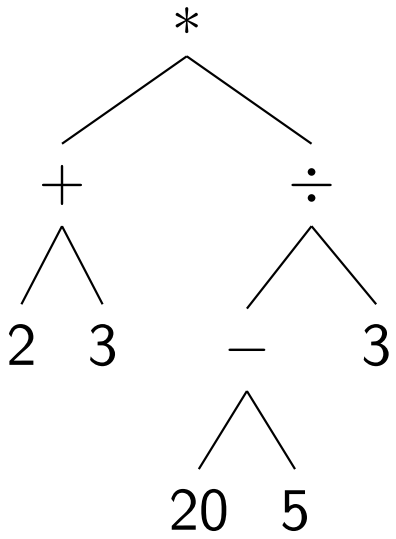



```
2 3 + 20 5 - 3 / *
```

- Note that we can also evaluate numbers while parsing:
  - Initialize an accumulator variable with 0.
  - For each digit:
    - Multiply the accumulator by 10
    - Add the digit

$$6253 = (((6 \times 10) + 2) \times 10) + 5) \times 10) + 3$$

Evaluating arithmetic trees




```
2 3 + 20 5 - 3 / *
```

- Note that we can also evaluate numbers while parsing:
  - Initialize an accumulator variable with 0.
  - For each digit:
    - Multiply the accumulator by 10
    - Add the digit

$$6253 = (((6 \times 10) + 2) \times 10) + 5) \times 10) + 3$$

- We ignore real numbers, negative numbers, *etc.*