# CS 430 Spring 2019 

$$
\begin{gathered}
x=a+b+c \\
y=\sin (x) \\
E=m c^{2} \\
e^{i x}=\cos x+i \sin x \\
\rho\left(\frac{\partial \boldsymbol{u}}{\partial t}+\boldsymbol{u} \cdot \nabla \boldsymbol{u}\right)=-\nabla \bar{p}+\mu \nabla^{2} \boldsymbol{u}+\frac{1}{3} \mu \nabla(\nabla \cdot \boldsymbol{u})+\rho \boldsymbol{g}
\end{gathered}
$$

## Expressions

## Expressions

- Expression: specification of computation
- Fundamental to high-level languages
- Form/syntax expressed using BNF grammars
- Four main components:

1) Operations
2) Operands
3) Parentheses
4) Function calls

## Expressions

- Operators: symbols representing computation
- Unary vs. binary vs. ternary
- Infix vs. prefix vs. postfix
- Precedence
- Associativity (left or right)
- Overloading
- Short-circuit boolean operators


## Expressions

- Operands: input data for computation
- Evaluation order (left-to-right or right-to-left)
- Type conversions
- Implicit vs. explicit
- Narrowing vs. widening
- Errors
- Overflow and underflow
- Division by zero
- Floating-point issues (e.g., NaN, subnormal)


## Expressions

- Parentheses
- Explicit precedence and associativity
- Tuple creation
- Function calls
- Side effects: a function changes a parameter or a non-local variable
- Referential transparency: expressions with the same value can be substituted for each other


## Assignment Statements

- Symbol and ambiguity with equality operator
- "=" vs. ":=" vs. "==" vs. "↔"
- Assignments as expressions; good idea?
- Conditional targets (ternary LHS)
- ( n > 5 ? a : b) = n*2
- Compound assignments
- Shortened forms of an assignment: "+=" and "++"
- Multiple assignments
- $a, b=c / 2, c \% 2 a, b=b, a$


## Evaluating Expressions

- Construct expression tree (e.g., parse the expression)
- Build tree from root ("which operation is done last?")
- Use precedence and associativity to guide you
- Evaluate using a post-order traversal
- Use evaluation order to guide you
- Track side effects as you go


