CONVERTING FORMULAS TO C++

Variables, Operators, and Functions
Variables

- Variables must be defined and initialized before they may be used.
- All of the following examples assume that the variables are defined and, where necessary, are initialized.
- Variable names must be unique within a scope.
- Variables in formulas may have subscripts but variables in C++ may not:
  - $m_0$ may be converted to $m_0$;
  - $F_n = F_{n-1} + F_{n-2}$ may be converted to $F_n = F_{n1} + F_{n2}$; or $f = f_1 + f_2$;
MULTIPLICATION

- Formulas denote multiplication by placing variables next to each other: PV
- C++ requires an explicit operator: *
- The formula $T = PV$ is translated into C++ as
  - $T = P \times V$
  - Temperature = Pressure * Volume
Formulas denote division in two ways:

- \( v = \frac{x}{t} \)
- \( v = \frac{x}{t} \)

The second way can imply grouping: \( \frac{P}{T_2 - T_1}, T_2 - T_1 \) must be done before the division

C++: \( P \div (T_2 - T_1) \)
INTEGER DIVISION

• Integer division can cause unexpected results
  
  \[ c = \frac{5}{9}(f - 32) \]

  \[ c = 5 / 9 \ast (f - 32), \text{always produces a 0} \]

• Problem is easily corrected

  \[ c = 5.0 / 9.0 \ast (f - 32) \]

  \[ c = 5 \ast (f - 32) / 9 \]

  \[ c = (f - 32) \ast 5 / 9 \]
UNARY MINUS

- C++ has both a unary minus and a unary plus (plus isn’t really useful)
- Both convert from formulas straight to C++
  - +N
  - -N
  - -N can be implemented as -1 * N but this looks cluttered and amateurish

\[
payment = \frac{PR}{1 - (1 + R)^{-N}}
\]

```
double payment = P * R / (1 - pow(1 + r, -N));
```
Like Java, C++ does not have an exponentiation operator

When squaring or even cubing an integer, it’s just about as fast and easy to multiply the number by itself

\[ x^2 = x \times x; \quad x^3 = x \times x \times x; \]

For higher powers, or variable, negative or fractional exponents, use the pow function

\[ y = x^{p/q} \quad y = x^{-n}; \quad y = \text{pow}(x, p/q); \]

The arguments form a group and so don’t require parentheses

Remember that pow returns a double

The return value is a single value that doesn’t require parentheses
The `sqrt` function calculates and returns a square root.

Everything under the radical is part of the function’s non-negative argument (i.e., the argument is self-grouping).

The return value also acts as a group.

\[ m = \frac{m_0}{\sqrt{1 - \frac{v^2}{c^2}}} \]

```c
m = m0 / sqrt(1 - v*v / (c*c));
```

```c
m = m0 / sqrt(1 - pow(v, 2) / pow(c, 2));
```
SYMBOLS OF INCLUSION

• Mathematical formulas can use ( ), [ ], and { } for grouping
• C++ can only use ( )
• You can always use parentheses even when precedence and associativity resolve all ambiguity
  • No magically correct number of parentheses
  • Too many parentheses make the statement harder to read and increase the likelihood of mismatched or unbalanced parentheses

\[ P = F \left[ \frac{r}{(1+r)^n - 1} \right] \left[ \frac{1}{1+r} \right] \]

\[ P = F * \left( \frac{r}{(\text{pow}(1 + r, n) - 1)} \right) * \left( \frac{1}{1 + r} \right); \]