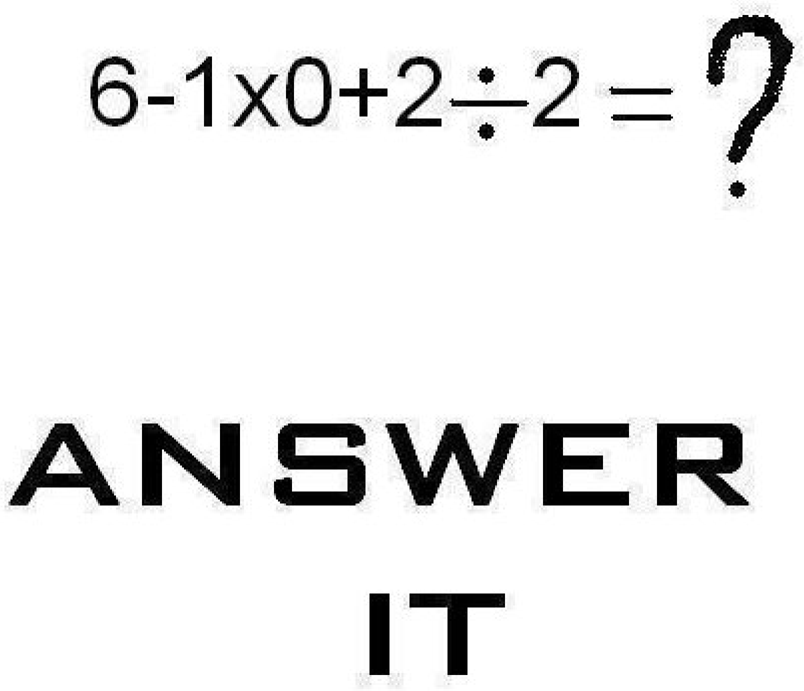
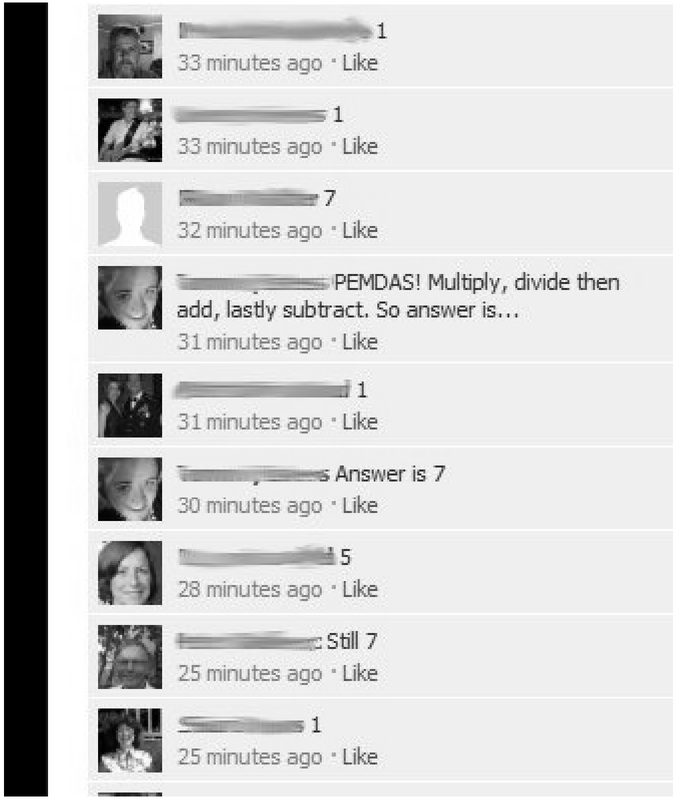
Expressions

Chapter 2.3 (part)
Slides #4





[redacted] 1
33 minutes ago · Like

[redacted] 1
33 minutes ago · Like

[redacted] 7
32 minutes ago · Like

[redacted] PEMDAS! Multiply, divide then add, lastly subtract. So answer is...
31 minutes ago · Like

[redacted] 1
31 minutes ago · Like

[redacted] Answer is 7
30 minutes ago · Like

[redacted] 5
28 minutes ago · Like

[redacted] Still 7
25 minutes ago · Like

[redacted] 1
25 minutes ago · Like

Topics

1) How can we calculate values?
2) What's the best way to work with values like 3600?

Math Expressions

(And not like "Wow! Math is great!")

Expressions

- Expression:
  - A statement that...
  - Usually has an operator.
- Examples:

```
result = 3;
result = x * 2;
result = 1 * x + 2;
```

- Expressions usable anywhere a value is needed:
  - `cout << "Big number " << (1 + 2) << endl;`

Order of Operations

- What is the value of result?
  int result = 4 + 10 / 2;
  - Is it 7 or 9?   (4 + 10) / 2   or   4 + (10 / 2)
- Each operator is given a precedence:
  - Higher precedence operators are applied first.
  - / is higher than +, so the answer is..
  - Add brackets to force an ordering.
- Associativity:
  - Apply the operators from right-to-left, or left-to-right?
  - +, - are left to right: do the one on the..
  - =, += are right to left: do the one on the..

Operator precedence

- Operators at same

  evaluated based on associativity.
  - * and / from L to R
  - = and += from R to L
- Examples:
  - result = -20 + 9 / 5;
  - result = (-20 + 9) / 5;
  - val = 6 + 5 * 4 / 3 * 2;
  - sum = sum + 10;

Prec. Level	Op.	Operation	Associates
1	[ ]	Array Index	L to R
2	+ -	unary plus unary minus	R to L
3	* / %	mult, div, remainder	L to R
4	+ -	add subtract	L to R
5	<< >>	stream ins. extract.	L to R
6	< <= > >=	comparisons	L to R
7	= += -= *= ...	assignments	R to L

Order can be forced by parentheses.
See text Appendix 2 for full table.

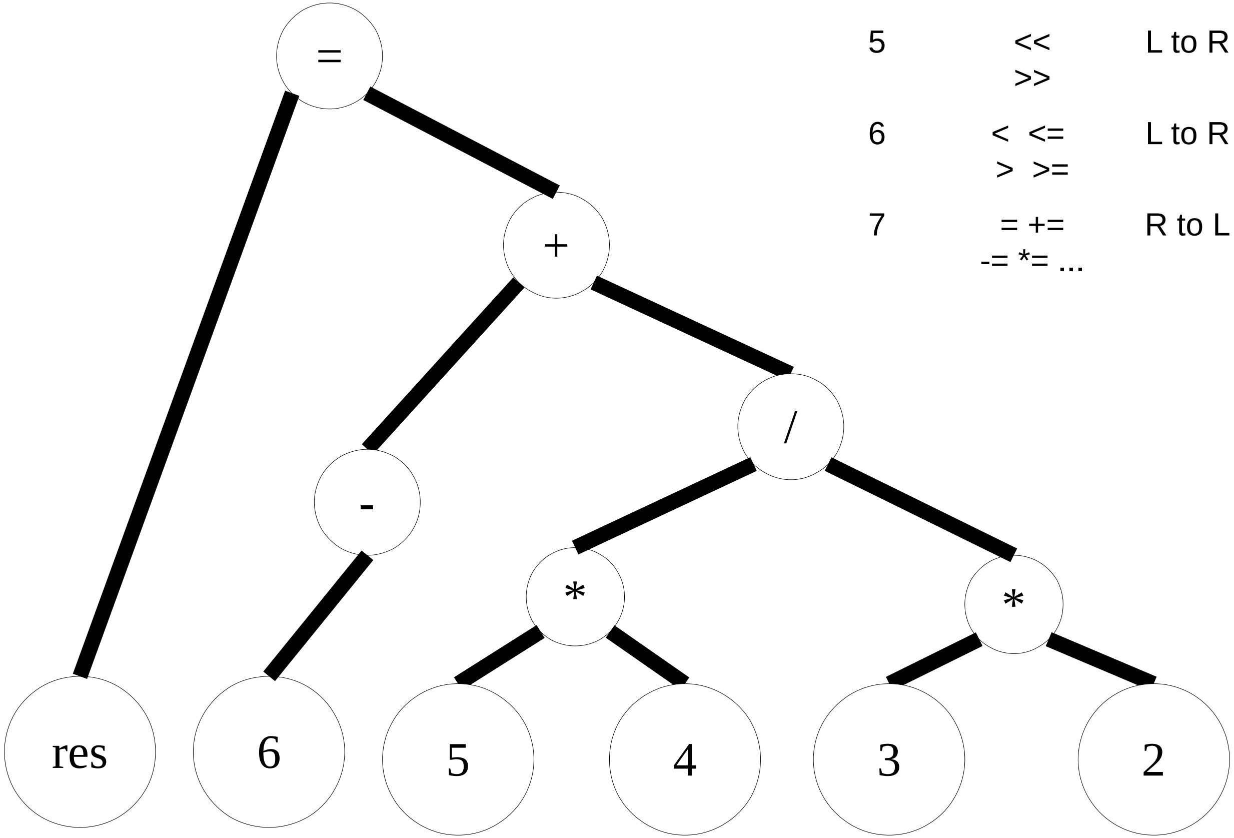
Brackets

- A statement can be correct, but unreadable:
  - result = 1 + 2 / 6 - 1 * 3 / 4 - 3 - -3 * +4;
- Add brackets to make it clear:
  - result = 1 + (2 / 6) - (1 * 3 / 4) - 3 - ((-3) * (+4));

Expression tree

Prec. Level	Op.	Asso.
1	[ ]	L to R
2	unary + unary -	R to L
3	* / %	L to R
4	+ -	L to R
5	<< >>	L to R
6	< <= > >=	L to R
7	= += -= *= ...	R to L

- Represent res = (-6 + 5 * 4 / (3 * 2)) as a tree:
- Operands as leaves.
- Operators as branching nodes.
- Operations lower in the tree have..
- Evaluate from the..
- Write values on internal nodes.



Review

- Draw an expression tree for the following:

  answer = 5 * x + 6 * (1 – x);

Assume x=2

Prec. Level	Op.	Asso.
1	[ ]	L to R
2	unary + unary -	R to L
3	* / %	L to R
4	+ -	L to R
5	<< >>	L to R
6	< <= > >=	L to R
7	= += -= *= ...	R to L

Constants

Constants

- We have already used literal constants:

```
int x = 10;                    // Numeric constant
cout << "Hello world!\n";      // String literal
```

- Raw number in code are..

```
int w = d / 7;
int c = s / 72;
```

- Use named constants like variables:

```
const int MIN_PER_HOUR = 60;
int h = m / MIN_PER_HOUR;
```

const

- const qualifier makes variable...

  ```
  const int PAY_PER_DAY = 475;
  const int DAYS_PER_WEEK = 7;
  ```

  - Constants must be given a value when created.
  - Name is upper case by..
  - Program cannot modify value of a constant:

    ```
    PAY_PER_DAY = 99999; // ERROR!
    ```

- Advantages:
  - Program becomes more...
  - Can change value in entire program in one spot.
    - Ex: change tax rate that's used in 100 calculations!

Example with const

```
// Convert days to weeks/years/fortnights.
// #includes/uses... omitted for space.
const int DAYS_PER_WEEK = 7;
const int DAYS_PER_YEAR = 365;

int main()
{
    const int DAYS_PER_FORTNIGHT = 14;

    cout << "Enter # days: ";
    int numDays = 0;
    cin >> numDays;

    int numWeeks = numDays / DAYS_PER_WEEK;
    int numYears = numDays / DAYS_PER_YEAR;
    int numFortnight = numDays / DAYS_PER_FORTNIGHT;

    cout << "# Days:        " << setw(4) << numDays << endl;
    cout << "# Weeks:       " << setw(4) << numWeeks << endl;
    cout << "# Years:       " << setw(4) << numYears << endl;
    cout << "# Fortnights: " << setw(4) << numFortnight << endl;
}
```

Constants can be..

```
Enter # days: 4641
# Days:       4641
# Weeks:       663
# Fortnights:  331
```

= daysPer.cpp

Guide to Constants

- Which of the following literal constants would be best made into named constants?
  - int numStudents = 0;
  - int next = numStudents + 1;
  - int waitlist = numStudents – 72;

Combined Assignments
& Overflow

Assignment Operators

- Combine an operation with assignment:
    - +=, -=, *=, /=, %=
- Examples:
    - a += b;
      // means a = a + b;
    - a *= b;
      // means a = a * b;
    - a /= 2 + 3;
      // means...

```
const int MAX_COUNT = 10;
int main()
{                                   ..
    int sum = 0;
    int i = 0;
    while (i < MAX_COUNT) {
        sum += i;
        i++;
    }
    cout << "Sum from 0 to "
            << MAX_COUNT - 1
            << " = " << sum << endl;
}
```

Overflow & Underflow

- Each type has a maximum value it can store.
  - Maximum + 1 overflows to the most negative.
  - Minimum – 1 underflows to the most positive.

```
// Work with overflow/underflow
#include <iostream>
#include <climits>
using namespace std;
int main()
{
	int test = INT_MAX;
	cout << "Test starts out at:  " << test << endl;
	test += 1;
	cout << "Adding one gives us: " << test << endl;
	test -= 1;
	cout << "Now subtracting 1:   " << test << endl;
}
```

```
Test starts out at:  2147483647
Adding one gives us: -2147483648
Now subtracting 1:   2147483647
```

#include <climits>..

..

INT_MAX, INT_MIN,
CHAR_MAX, CHAR_MIN,
....

= over_underflow.cpp

Suggested Review Questions

- Draw an expression tree for the following:
  result = 8 * - 1 + 3 / 2
  - Solve it by writing values on the nodes.
  - Write a C++ program to double check your answer.

Summary

- Expressions calculate values using operators.
  - Operator precedence gives us expression trees.
- Use named constants (const), not magic numbers.
- Combined assignment operators like x += 2;