CS 161
Intro to CS I

More Programming and Conditional Statements
Odds and Ends...

• Recitation Quiz #1 due today by 11:59pm
  – Email to specific recitation TA
• Assignment #1 due Sunday by 11:59pm
  – Submit on TEACH
  – If off campus, need VPN for mapped network drive
  – It must compile and run on ENGR!!!
• Make demo appointment (signup homepage)
Extra-Credit Exercise #2

Get into groups of 4-5, and each write your name on a piece of paper.

• Each person state:
  – What are you struggling with the most on Assignment #1?

• As a group:
  – Offer advice on how to fix it.
Constants

• What is a constant?
• How do we define a constant?
  – Use of a macro
    • `#define`
    • Placed at top of program
    • No semicolon at end
    • Example: `#define MAX_SIZE 100`
  – Use of `const`
    • Same as declaring variable but `const`
    • Example: `const int MAX_SIZE = 100;`
Intro to Macros

• C++: `<climits>`
• Use MIN and MAX macros from library
  http://www.cplusplus.com/reference/clibrary/climits/
  (Note that the values listed are not the values on our system!!!)
  – INT_MAX
  – INT_MIN
  – LONG_MAX
  – LONG_MIN
  – SHRT_MAX
  – SHRT_MIN
• Remember unsigned too...
```cpp
#include <iostream>
#include <climits>

using namespace std;

int main() {
    cout << "unsigned long max macro: "
        << ULONG_MAX << endl;  // this is a macro from the climits library
    return 0;
}
```
Expressions

• What is an expression?
  – Set of operations producing a value
    • Combining literal values
      12 * 4 + 6 * 10 vs. ((12 * 4) + 6) * 10
    • Combining variables
      var1 * var2 + var3 * var4 vs. ((var1 * var2) + var3) * var4
Expressions cont.

• Pieces of an Expression:
  – Operators
    • Indicate operation, e.g. +, *, /, -, %
  – Operands
    • Values in the expression
  – Evaluation
    • Process of obtaining results from operations on operands
Arithmetic Operators

- Add  
  \[ 34 + 23 \]
- Subtract  
  \[ 34 - 23 \]
- Multiply  
  \[ 2 \times 23 \]
- Divide  
  \[ 40 / 10 \]
- Remainder/Mod  
  \[ 34 \% 5 \]
Precedence

• What is precedence?
  – Binding power of operator
  – (*, /, %) vs. (+, -)

• How do we override precedence?
  – Parenthesis!

• Examples:
  12 * 4 + 6 * 10 vs. ((12 * 4) + 6) * 10
Arithmetic

• Integer Arithmetic
  std::cout << 3/8;    /*prints 0*/
  std::cout << 34/5;   /*prints 6*/
  int age=5;
  std::cout << age/2;  /*prints 2*/

• Floating Point Arithmetic
  std::cout << 34.0/5.0;  /*prints 6.8*/
  std::cout << 3.0/8;    /*prints .375*/
  float years=2.0;
  std::cout << age/years; /*prints 2.5*/
Type Casting

• Casting

```cpp
std::cout << age / (int) years;  /*prints 2*/
std::cout << (int) (age / years);  /*prints 2*/
std::cout << (float) age / 2;    /*prints 2.5*/
```

• What is wrong with these?

```cpp
std::cout << (int) age / years;  /*prints 2.5*/
std::cout << (float) (age/2);   /*prints 2.0*/
```
```cpp
#include <iostream>
#include <climits>  // has macros for ULONG_MAX, LONG_MAX, etc.
#include <cmath>    // has built-in function pow() for exponents
#define BITS_BYTE 8  // create our own constant macros for bits in a byte

using namespace std;

int main() {
    // long long_max;  // signed by default
    unsigned long ulong_max;  // specify unsigned explicitly

    ulong_max=(unsigned long)pow(2,BITS_BYTE*8)-1;  // need to typcast
    cout << ulong_max << endl;
    cout << "unsigned long max macro: "
        << ULONG_MAX << endl;
    ulong_max=ulong_max+1;  // this will only overflow if you reach limit
    cout << ulong_max << endl;

    return 0;
}
```
Extra-Credit Exercise #2

• Can you think of an equation that wouldn’t rely on overflow and would work in all instances?