CS 161, Lecture 2: Variables and Math – 12 January 2018
How to Name

• Names also known as identifiers are given to variables and function
• Start with letter: upper case, lower case, underscore
• Followed by sequence of letters and digits
  • Good: myVar, result_of_eq1, _hello
  • Bad: 1234, my-Var, 2eq_res
• Can’t use keywords
Assigning Values to Variables

- Point of variables is to hold data
- Declare a variable
  - `int my_num;
- Use ‘=’ followed by the data you want to store (data must be same type as what was declared)
  - `my_num = 5;
- ‘=’ is the assignment operator not a test for equivalence
  - say `my_num “is assigned” or “gets” 5
- Can declare and assign on same line
  - `int my_num = 5;
  - `char letter = ’a’;
  - `string my_str = ”my string”;

Printing Variables

• cout << “The result is: “ << result << endl;
• Alters out stream
Constants

• Constants do not change

• Two ways to create a constant
  • Define a macro
    • At top of program, no semicolon
    • #define MAX_SIZE 10000
    • MAX_SIZE will always be 10000 through out the entire program
  • Use const keyword
    • Same as declaring variable
    • const int MAX_SIZE 10000;
Predefined Macros

• Some macros already exist for things, typically import in library
• C++: <limits>
• Use MIN and MAX
Expressions

• Set of operations producing a value
• $12\times 4 + 6\times 10$
• $((12\times 4) + 6)\times 10$
• $\text{var1}\times \text{var2} + \text{var3}\times \text{var4}$
• $((\text{var1}\times \text{var2}) + \text{var3})\times \text{var4}$
Pieces of an Expression

• Operators: indicate operation
  • Add +
  • Subtract –
  • Multiply *
  • Divide /
  • Remainder/Mod %

• Operands: values in the expression

• Evaluation: process of obtaining results from operations on operands
Precedence and Division Types

• Precedence: binding power of operator
  • Override with parenthesis

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

• Floating Point Arithmetic
  • std::cout << 3.0/8.0;
  • std::cout << 34.0/5.0;
  • float age = 5.0;
  • std::cout << age/2.0;
Type Casting

• Casting:
  • `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?
  • `std::cout << (int) age / years; /*prints 2.5*/`
  • `std::cout << (float) (age/2); /*prints 2.0*/`
Additional Operators

• Fetch/store same variable
  • var = var + 2
  • var = var * 2

• Assignment/operator combination
  • var += 2
  • var *= 2

• Pre/Post increment/decrement: ++ and –
  • age++ vs. ++age
•https://tinyurl.com/yb9uzp28
```cpp
#include <iostream>
#include <cmath>
using namespace std;

int main ()
{
    float num1, num2;
    num1 = 3.14;
    num2 = 2.0;

    float result = cos(num1/num2);
    cout << "The value of result is: " << result << endl;
    num1 = 5.0;
    result = cos(num1/num2);
    cout << "The value of result is: " << result << endl;

    return 0;
}
```