How programs crunch numbers

CS 10A – BASIC MATH IN C/C++

Allen Zhao – SRJC CS 10A

Introduction

- Math in programming is nearly completely identical as how you would do it, but with one key difference.
  - Integer division (requires use of int variables)
- All programs follow PEMDAS rules
- + for addition
- - for subtraction
- * for multiplication
- / for division

Integer Division

- To us, a problem like ¾ would result in 0.75. However, a computer calculates that as 0.
- Standard integer division in a program will instead calculate the largest number that can be multiplied to the divisor without exceeding the original dividend.
  - TL;DR – the answer is rounded down to the nearest integer
  - 8/9 = 0
  - 8/8 = 1
  - 8/3 = 2

Modulo Division

- Where does the lost value in integer division go?
- Modulo division: gets the remainder from division
- % for modulo division
- Can only be used with integers
  - 8 % 9 = 8
  - 8 % 8 = 0
  - 8 % 7 = 1
  - 8 % 3 = 2
- So for integers, / and % can be used in conjunction to get a full solution for a division problem.

Using Math in Code

Program

```
int main()
{
        int num;
        cout << “Enter a number: ”;
        cin >> num;
        cout << 3*num + 2 << endl;
        num = num/2; // reassigns num
        cout << num << endl << num%2;
        return 0;
}
```

Console

➢ ./a.exe

```
Enter a number: 5
17
2
0
```

Variables for Non-Integer Math

- To do regular division, we'll need a variable type that supports decimal places: float and double
  - float – short for floating point accuracy, 7 decimal places
  - double – has double the precision of float, 15 decimal places
- To tell the program to do regular and not integer division, at least one number involved must NOT be an integer.
  - In the case of whole numbers, use one 0 in the decimal place.
  - i.e. ¾ = 0, 3.0/4 = 0.75, 3/4.0 = 0.75

Utilizing Non-Integer Math

Program

```
int main()
{
        float x = 2.5, a;
        double y = 3;           // actually 3.0
        int z = 3, b;

        cout << x*z << endl;    // non-integer result
        a = y/z;
        cout << a << endl;      // actually 1.0
        b = x*3;                // conversion back to int
        cout << b << endl;
        cout << 2.0*b << endl;          // actually 14.0
        return 0;
}
```

Console

- ./a.exe

```
7.5
1
7
14
```

The Equal Sign

- The equal sign is used a lot in code, but it's used differently in programming than it is in math!
- = is the assign operator, NOT an equal operator
  - x = 5 // this assigns the value 5 to variable x
  - x = x + 5 // this reassigns x as 5 plus what x currently is
- == is the operator used to define equal in C/C++
  - x == y // this asks the program if x is equal to y
- Be careful! This is an extremely common error that even veteran programmers sometimes overlook.