**simple arithmetic**

**LAB 3**

**SECTION A**

**SUBMITTED BY:**

**Brian Reber**

**SUBMISSION DATE:**

**9/25/2009**

**Lab Problem**

The purpose of this lab is to introduce C programming data types and operators. We will use simple arithmetic to show the danger of not paying attention to what data type is being used.

**Analysis**

In this lab, we will look at three different problems.

1. Mysterious Output – The given code will contain mistakes that will still allow the program to run, but will give incorrect program output.
2. Simple Arithmetic – we are to write a program that will do a variety of simple math problems, and will store them in different data types. We are to notice what happens with each data type.
3. Wiimote – We will run the given code, and work with the Wiimote to figure out what the code does, and why it is helpful.

**Inputs**:

None

**Outputs**:

1. Mysterious Output
   1. Float
   2. Int
   3. Int
2. Simple Arithmetic – multiple values according to the Lab Instructions
3. Wiimote - <Float, Float, Float> 🡪Float

**Design**

Algorithm:

1. Get first int value.
2. Get second int value.
3. Calculate the value of c using given formula.
4. Display the value of c.

**Testing**

1. Mysterious Output

For this section, we compiled and ran the program. First off, when we compiled we got a few errors, but none of them prevented the program from being run. The reason for these errors was either because there wasn’t an argument in the prinf function where there should have been, or the wrong type was expected. For the first part of problem 1, the error was in the integer division. We divided two whole numbers and stored it as a short (a shorter int). Dividing 1 by 5 therefore gave us 0.0000000. Part two, there wasn’t an argument where there was one expected. This caused it to print out 8388608. This is obviously not the answer for 2 + 3. The third part, the printf function was expecting an int, but instead it was given a float. This also caused it to print out 8388608.

1. Simple Arithmetic

On this section, there were many problems that arose. Most of them had to do with integer division, or wrong type for the problem.

* 1. No error.
  2. Should have given us 27361080, but since this number is bigger than a short can handle, it gave us the max number a short can handle, 32568.
  3. Here we had an int division problem. 12/5 will give us 2.00 for a float, instead of 2.4 which is what should have been given. Solve this problem by doing 12.0/5.
  4. No error.
  5. Gives us 21 instead of 22 because of integer division. 22/3 gives us 7.00.
  6. Int division problem. The correct answer in double format is 2.44, but since we stored it as an int, it gave us 2.
  7. Int division problem. Same as f. Solve this by doing 22.0 / (3\*3).
  8. Int division error. Should give us 22.00, but instead 21.00. Solve this by 22.0/3\*3.
  9. No error.
  10. Since we stored this as a short, it cut off the last two numbers. If we had stored it as a float, it would have worked fine.
  11. No error.
  12. No error.
  13. No error.
  14. No error.

1. Wiimote

For this section, we were supposed to use a given program to test the Wiimotes. This program calculated the magnitude of the G-Force. To do this, the program took the square root of each component (x, y, z) squared. So when the Wiimote was at rest, the magnitude was 1. This is to be expected because only one g is acting on it. When we dropped the Wiimote, the values were below 1, which makes sense because it was in free fall – less than one g. When we caught it, the value spiked above at about 3 or so g’s. That means that it stopped quickly and experienced three times the force of gravity.

**Comments**

I liked this lab because it showed me in great detail that knowing which type of number I am dealing with will make a huge difference. This also requires the programmer to think through what the output will be, so that they can program the right type. Also, it was interesting to see how the program in C can read the input from the Wiiwrap program – being able to input data from an external device was interesting. Working with the formulas and conversions in part two helped because I had to think about how to do it on paper, and then convert that knowledge into a program.

**Implementation**

// CprE 185: Lab 3

// Problem 1: Mysterious Output

#include "stdio.h"

int main()

{

short IntegerResult;

float DecimalResult;

IntegerResult = 1 / 5;

printf("The value of 1/5 is %f\n", IntegerResult);

IntegerResult = 2 + 3;

printf("The value of 2+3 is %d\n");

DecimalResult = 1.0 / 22.0;

printf("The value 1.0/22.0 is %d\n", DecimalResult);

return 0;

}

Lab3 Output

The value of 1/5 is 0.000000

The value of 2+3 is 3883

The value 1.0/22.0 is 3907

/\*

\* CprE 185: Lab 3

\* lab3\_2.c

\*

\*

\* Programmer: Brian Reber Date: 9/18/09

\* Instructor: Daniels Class: CprE185

\*

\*/

#include <stdio.h>

int main()

{

short a = 6427 + 1725;

printf("6427 + 1725 = %d\n", a);

short b = (6971 \* 3925) - 95;

printf("(6971 \* 3925) - 95 = %d\n", b);

float c = 79 + 12 / 5;

printf("79 + 12 / 5 = %.2f\n", c);

float d = 3640.0 / 107.9;

printf("3640.0 / 107.9 = %.2f\n", d);

short e = (22 / 3) \* 3;

printf("(22 / 3) \* 3 = %d\n", e);

int f = 22 / (3 \* 3);

printf("22 / (3 \* 3) = %d\n", f);

float g = 22 / (3 \* 3);

printf("22 / (3 \* 3) = %.2f\n", g);

float h = 22 / 3 \* 3;

printf("22 / 3 \* 3 = %.2f\n", h);

float i = (22.0 / 3) \* 3.0;

printf("(22.0 / 3) \* 3.0 = %.2f\n", i);

short j = 22.0 / (3.0 \* 3.0);

printf("22.0 / (3.0 \* 3.0) = %d\n", j);

float k = 22.0 / 3.0 \* 3.0;

printf("22.0 / 3.0 \* 3.0 = %.2f\n", k);

float radius = 23.567 / (2 \* 3.14);

float area = 3.14 \* radius \* radius;

printf("The area of a circle with circumference 23.567 = %.2f\n", area);

float feet = 14;

float meters = feet \* .3;

printf("14 ft = %.2f meters \n", meters);

float degrees = 76;

float centigrade = (degrees - 32) / 1.8;

printf("76 degrees Fahrenheit = %.2f degrees Centigrade\n", centigrade);

}

Lab 3\_2 Output

6427 + 1725 = 8152

(6971 \* 3925) - 95 = 32568

79 + 12 / 5 = 81.00

3640.0 / 107.9 = 33.73

(22 / 3) \* 3 = 21

22 / (3 \* 3) = 2

22 / (3 \* 3) = 2.00

22 / 3 \* 3 = 21.00

(22.0 / 3) \* 3.0 = 22.00

22.0 / (3.0 \* 3.0) = 2

22.0 / 3.0 \* 3.0 = 22.00

The area of a circle with circumference 23.567 = 44.22

14 ft = 4.20 meters

76 degrees Fahrenheit = 24.44 degrees Centigrade

/\*

\* CprE 185: Lab 3

\* lab3\_3.c

\*

\* Instructor: Daniels Class: CprE185

\*

\*/

#include <stdio.h>

#include <math.h>

int main() {

float x, y, z;

while (1) {

scanf("%f,%f,%f", &x, &y, &z);

printf("Magnitude of (%5.2f,%5.2f,%5.2f) is: %6.2f\n",

x, y, z, sqrt(x\*x+y\*y+z\*z));

}

return 0;

}