

ENGG 233 Fall 2004 L02: Examples of SCOPE and VISIBILITY for Oct. 1

Author: Dr. S. A. Norman

scope1.cc: Shows scope of some variables and parameters ...

```
#include <iostream>
#include <cmath>
using namespace std;

const double MATH_CONSTANT_PI = 3.1415926535897931;
const double RADS_PER_DEG = MATH_CONSTANT_PI / 180.0;
const double DEGS_PER_RAD = 180.0 / MATH_CONSTANT_PI;

int conversion_count = 0;

double rad2deg(double rad);
double deg2rad(double deg);

int main()
{
    double input1;
    cout << "Enter an angle in degrees " << endl;
    cin >> input1;
    cout << "sin(" << input1 << " degrees) is "
         << sin(deg2rad(input1)) << '.' << endl;
    cout << "cos(" << input1 << " degrees) is "
         << cos(deg2rad(input1)) << '.' << endl;
    double input2;
    cout << "Enter a number between -1.0 and 1.0:" << endl;
    cin >> input2;
    cout << "An angle with a sin of " << input2 << " is "
         << rad2deg(asin(input2)) << " degrees." << endl;
    cout << "This program did " << conversion_count
         << " unit conversions of angles." << endl;
    return 0;
}

double rad2deg(double rad)
{
    double deg;
    deg = DEGS_PER_RAD * rad;
    conversion_count++;
    return deg;
}

double deg2rad(double deg)
{
    conversion_count++;
    return RADS_PER_DEG * deg;
}
```

scope of cout and cin

scope of MATH_CONSTANT_PI

scope of RADS_PER_DEG

scope of DEGS_PER_RAD

scope of conversion_count

scope of input1

scope of input2

scope of rad

scope of the other deg

scope2.cc: Shows scope of some variables and parameters, including a variable that is local to a compound statement ...

```
#include <iostream>
using namespace std;

void smallest_first(int& a, int& b);
// If initial value of a > initial value of b,
// then a and b get swapped.

int main()
{
    scope of x and y
    int x = 27, y = 12;
    cout << "before function call: " << x << ' ' << y << endl;
    smallest_first(x, y);
    cout << "after function call: " << x << ' ' << y << endl;
    return 0;
}

void smallest_first(int& a, int& b)
{
    scope of a and b
    {
        if (a > b) {
            scope of temporary
            int temporary = a;
            a = b;
            b = temporary;
        }

        // It would be an error to try to access temporary here.

        return;
    }
}
```

hiding.cc: Shows a logic error. The author thinks he is harmlessly repeating type information about a and b, but in fact he is creating local variables that *hide* the parameters a and b ...

```
#include <iostream>
using namespace std;

void smallest_first(int& a, int& b);
// If initial value of a > initial value of b,
// then a and b get swapped.

int main()
{
    int x = 27, y = 12;
    cout << "before function call: " << x << ' ' << y << endl;
    smallest_first(x, y);
    cout << "after function call: " << x << ' ' << y << endl;
    return 0;
}

void smallest_first(int& a, int& b)
{
    if (a > b) {
        int a, b;           // MISTAKE! Local variables hide arguments.
        int temporary = a;
        a = b;
        b = temporary;
    }
    return;
}
```