

Lecture 3: CS2400 Introduction to Computer Science

- Assignment statements
- I/O
- Formatting output
- Data Types

Assignment Statements

To change the value of a variable.

Syntax:

```
Variable_Name_1 = Expression;
```

Examples:

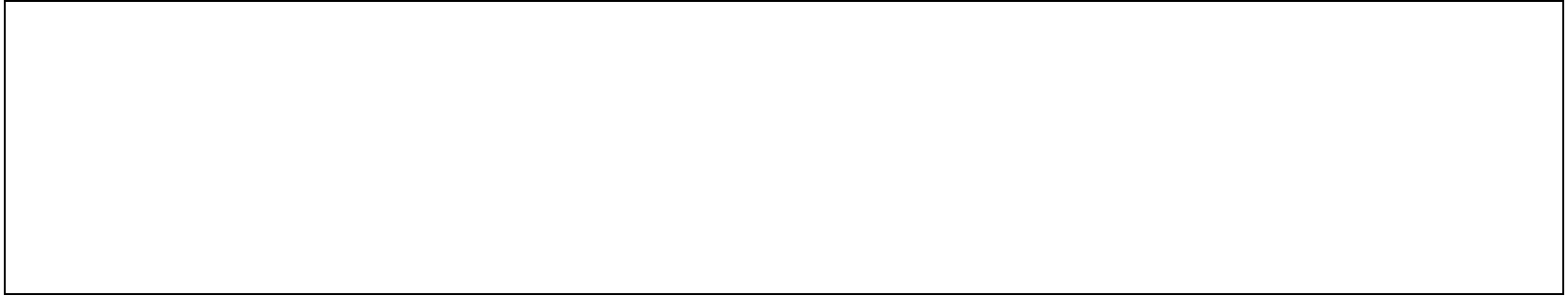
What happens when a variable is on both sides of the equals sign?

Consider the following program:

```
#include <iostream>
#include <cstdlib>
using namespace std;
int main()
{
    int x, y, z;
    x = 42;
    y = x;
    x = z;
    cout << x;
    cout << "\n";
    cout << y;
    cout << "\n";
    cout << z;
    // Leave out next line, what happens?
    cout << "\n";

    return(EXIT_SUCCESS);
}
```

What is the output?



What happened??

How can we avoid this problem?

Initializing variables in declarations:

Syntax:

```
Type_Name Variable_Name_1 = Expression_1,  
Variable_Name_2 = Expression_2,  
...;
```

Examples:

Variables revisited

How should variables be named?

Streams

In C++ input and output is generally processed in terms of streams. A stream flows by and you do not know its source. Each bit of water flows by sequentially, you see one bit at a time, and then the next.

For now, input will be assumed to be from the keyboard, and output is assumed to go to the screen, although that will change later in the class.

Output using cout

Examples:

```
cout << distance << " miles\n";
```

What does this statement do?

```
cout << "The circumference is ";
```

```
cout << (PI * diameter) << "\n";
```

parentheses are required by some compilers, so be sure to include them for portability.

<< is called the *insertion operator*.

Multiple line cout statements

We can re-write:

```
cout << "The circumference is ";  
cout << (PI * diameter);  
cout << "inches\n";
```

as:

```
cout << "The circumference is "  
    << (PI * diameter)  
    << "inches\n";
```

Note: do not break strings across lines!

End-of-line

There are two ways to output a new-line character in C++:

- What way do you know?
- use the special constant `endl`

```
cout << endl;
```

Formatting numbers

When the computer outputs a number of type `double`, this is what happens:

```
price = 17.5;  
cout << "The price is $" << price << endl;
```

generates:

```
The price is $17.5
```

On other systems we might get different outputs. In general, however, we will not get the usual way of outputting dollars and cents -- `$17.50`. How can we achieve the correct format?

Formatting Doubles

To get two digits to the right of the decimal point use the following:

```
cout.setf(ios::fixed);  
cout.setf(ios::showpoint);  
cout.precision(2);
```

Any output statement that follows these three statements will output values of type `double` in dollars and cents format.

In chapter 6 we'll see why this works, and how to change the output format in different ways.

Input

Input with `cin` is analogous to output with `cout`.

How would you input two values (the radius and height of a cylinder) with one statement?

One can also break input over multiple lines.

What happens when a running program encounters a `cin` statement?

`cin` skips over whitespace in the input. It will also skip over returns (blank lines).

Examples:

```
double price;  
int number;
```

```
cin >> price >> number;
```

```
cout << "price " << price << " number "  
      << number << endl;
```

Produces the output:

12

13

price 12 number 13

12

.1

Designing input and output

- Prompt the user for input
- Echo the values back to the user to ensure proper entry of data.
- If the prompt is short, it is sometimes much easier to notice that the computer is waiting for input if the computer does not go on to the next line.

```
Enter the radius of the  
circle:10.0
```

Data Types

What is the difference between 31.0 and 31?

How are they coded in the computer?

Number Types

Note: these differ on different systems.

integer types:

`int` 4 bytes

`long` 4 bytes

`long long` 8 bytes

floating point types:

`float` 4 bytes

`double` 8 bytes

`long double` 16 bytes

(see program lec3-3)

Type char

This type is for characters -- single symbols.

Characters must be enclosed in single quotes, as opposed to strings which are enclosed in double quotes.

```
char asterix='@';  
cout << asterix << endl;
```

would produce what output?

How much storage does a variable of type char take up?

Type Compatibilities

Consider:

```
int i = 3.14159265359;  
cout << i << endl;
```

What is the output?

Why did this happen?