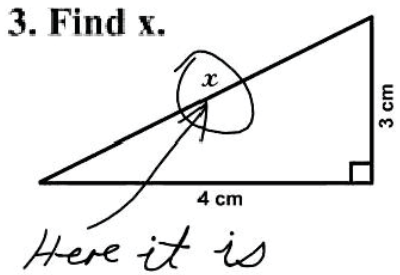
Slides #3

Variables

Chapter 2.3-2.18

CMPT 125/128

"Just a darn minute! — Yesterday you said that X equals two!"



Topics

1) How can we use libraries?
2) How can we store data, such as numbers?
3) What are the different types of variables?

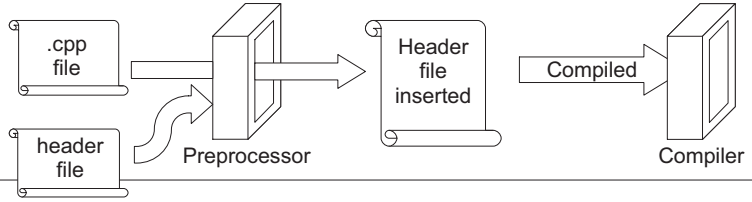
#include

Libraries

- Core C++
  - The "core" C++ provides just the basic language.
- Libraries:
  - Extra functionality comes from...
  - Example: iostream for input/output.
- To use a library, we...
  - Header File:
    declares the functionality provided by the library.

Including a Header File

- #include includes a header file.
  #include <iostream> // For input/output support
- #include is not a C++ statement:
  - The preprocessor starts with your .cpp file.
  - It process the #include directive by copying the header file into a temporary copy of your file.
  - This temporary copy is then compiled.



Variables

Variables

- A variable stores a value.
  - It is:
  - C++ is...
    Each variable is given a type when it is created.
- Example:
  - Declare the variable:
    int numStudents;
  - Use the variable:
    numStudents = 72;

Variable declarations tell the compiler the variable's type (int) and name (numStudents).

All variables must be

"Error: Undeclared identifier"

This assignment statement copies the value (5) into the variable (numStudents).

Example with Variables

```
// Small demonstration of variables.
// From Gaddis et. al., Modified by Dr. Fraser.
#include <iostream>
using namespace std;

int main() {
    // Create the variable, give it a value, and then display it.
    int numStudents;
    numStudents = 5;
    cout << "The value of numStudents is: " << "numStudents" << endl;
    cout << "The value of numStudents is: " << numStudents << endl;

    // Change the value and re-display it.
    numStudents = 7;
    cout << "Now the value of numStudents is: " << numStudents << endl;
    return 0;
}
```

Output:

```
The value of numStudents is: numStudents
The value of numStudents is: 5
Now the value of numStudents is: 7
```

Identifiers

- Identifier: a programmer-defined name which
  - Ex: Variable names, function or classes (later...)
- Valid Identifiers:
  - First character: a-z or A-Z or _
  - Any other characters: a-z or A-Z or _ or 0-9
  - Examples:
    - height, x, numStudents, NUM_PEOPLE
    - place0, t1x235e_23, _and_then_some
- Invalid Identifiers:
  - 2Tall, 11a, test#2, 3dGraphics

Identifiers

- Identifiers cannot be keywords:
  - Keywords are...
  - Ex: int, return, char, using, namespace, for, while...
- Tips:
  - Use meaningfully descriptive names:
    - numStudents is better than n
    - boxHeight is better than x
  - Use camel case for variables names:
    First word is lower case,
    Capitalize first letter of later words.
    - Ex: Students per course: ...

Data Types

Data Types

- There are a few different types of data:
  - Numbers
    - Integers: Whole numbers like 0, -14, 8382.
    - Floating point: Fractional values like -1.1, 3.14
  - Text
    - Character: A single character like 'h', 'i', '!'
    - String: A sequence of characters like "Hello!"

Integer Data Types

Typical Sizes & Range

	Type	# Bits	Range
Integers	short	16 bits	-32,768 to +32,767
	unsigned short	16 bits	0 to 65,535
	int	32 bits	~ +/- 2 billion
	unsigned int	32 bits	0 to 4.29 billion
	long	32 bits	~ +/- 2 billion
	unsigned long	32 bits	0 to 4.29 billion

- Size (# bits) of each value...
  - int could be 16 bits or 32 bits (or something else!)
- Only guaranty is: size of short <= int <= long

Variable Declaration

- Simple Variable Declaration:
  - int height;
- Can declare multiple variables at once, and Can initialize variables at declaration.
- Examples:
  - short count, sum;
  - unsigned int numStudents = 0,
    daysLeft = 85,
    numClasses = 1;
  - long bigNumber = 4000000000L,
    smallNumber = 0;

Integer constants (85) are of type int.

Values larger than an int must be "forced" to be a long by adding an L: x=10L; // A long type.

Variable Example

```
// Demonstrate variables
#include <iostream>
using namespace std;

int main() {
     int  numStudents = 72,
          numInstructors = 1;
     unsigned int numTAs = 2;
     short lecturesPerWeek = 3;

     cout << "This class has " << numStudents << " students, "
          << numInstructors << " instructors, and "
          << numTAs<<" TAs." << endl;
     cout << "There are " << lecturesPerWeek
          << " lectures each week." << endl;

     return 0;
}
```

```
This class has 72 students, 1 instructors, and 2 TAs.
There are 3 lectures each week.
```


Operators

Operations on Numbers

- Most basic math operations work on numbers.
  int x=10, y=3, z=0;
  - Addition z = x + y;
  - Subtraction z = x - y;
  - Multiplication z = x * y;
  - Division z = x / y;
  - Modulo z = x % y;
  - Negation z = -x;

Negation is Unary: it takes only on argument.

+, -, *, /, % are binary operators: they take two arguments.

Variable Example

```
// Demonstrate variables
#include <iostream>
using namespace std;

int main() {
     int width = 100,
          height = 20;
     int area = width * height;

     cout << "Box dimensions: " << width << " by "
          << height << " gives total area " << area << "." << endl;
     return 0;
}
```

Box dimensions: 100 by 20 gives total area 2000.

Review

1. What is wrong with each of these?
   a) int 1stVar = 10;
   b) long for = 0;

2. What is the value of each of these variables?
   a) int x = 5 / 2;
   b) int y = 21 % 5;

3. What integer data type best fits each range?
   a) Month of year
   b) Cost of new car (in $)
   c) Bytes in an MP3 file

char and string

char

- The char type can hold a single character.
  - Pronounced like "charred" not like "car".
- Characters are represented by the computer
  - 'A' is 65, 'B' is 66, 'C' is 67, ... (ASCII codes)
  - cout outputs char-number (65) as a character ('A').

```
char aLetter = 'A';
cout << aLetter << endl;
aLetter = 70;
cout << aLetter << endl;
aLetter = aLetter + 1;
cout << aLetter << endl;
```

Output:

```
A
F
G
```


string Class

- The string class stores and manipulates strings.
  - string class defined in library: #include <string>

```
// Example for string
#include <iostream>
#include <string>
using namespace std;

int main() {
    string mySaying;
    mySaying = "What's up Doc?";

    cout << "Who says: '" << mySaying << "'" << endl;
    return 0;
}
```

Output:

```
Who says: 'What's up Doc?'
```


Working with strings

- = String Assignment
  string name = "Bond";
- + String Concatenation
  - Use a + to join two strings together.

  string full = // = "James Bond"
- String Length
  - Use the "member-function" length on a string:

  int nameLen = // = 10 chars long.
- [] Get a character in a string
  char firstChar = name[0]; //
  char secondChar = name[1]; //

Floating Point

Floating Point

- Floating point numbers are used to store values like:
  3.1415, -0.03, 0.000000000001, 6.7 x 10^{84}
- They are stored using scientific notation:
  - 3.1415E0, -3.0E-2, 1.0E-12, 6.7E84
- Types:
  - float
    (typically) 7 significant digits, up to 3.4E18
  - double Double precision
    (typically) 16 significant digits, up to 1.7E308
  - long double Ofter larger than double.
    (typically) 16 significant digits, up to 1.7E308

  Note no unsigned floating point types.

Floating point example

```
// Example for floating point numbers
#include <iostream>
using namespace std;

int main() {
    float distanceSun = 1.49E8;          // in km
    double massSun = 1.989E30;           // in kg
    float timeVisible = 12.3;            // in hours

    cout << "The sun is "<<distanceSun<<" km away."<<endl;
    cout << "It weighs "<<massSun
         << " and we can see it for "<<timeVisible<<" hours per day."<<endl;
    return 0;
}
```

Output:

```
The sun is 1.49e+08 km away.
It weighs 1.989e+30 and we can see it for 12.3 hours per day.
```


Floating point to integer

- Floating point values hold more information than integer values:
  - How could you store 8.254 as an integer?
- Truncate:
  - int num = 8.254; // num actually holds 8.
  - long height = 2.9999; // height actually holds 2.
  - short time = -9.51; // time actually holds -9.

Review

1. What data type best fits each of these:
   a) "To be or not to be?"
   b) 3.1415
   c) 123456
   d) 'a'
2. What is output for each of these cout statements?
   string msg = "Welcome!";
   a) cout << msg[3];
   b) cout << msg.length();
   c) cout << "i" + "8" + "(3.1415)";

bool, Assignment, and Scope

bool

- bool stands for Boolean:
  - Can hold a value of either:...
  - C++ represents true as 1, false as 0.
- Example:

```
bool   lightOn = true;
bool   endOfTime = false,
       foundAnswer = true;
cout << endOfTime << endl << foundAnswer;
```

Output:

```
0
1
```


Assignment Terminology

```
int numStars;

numStars = 1084;
```

On the left is the

Must be a location in memory, such as a variable, where the value can be written.

On the right is the

Must evaluate to a value which can be assigned to the lvalue location.

Uninitialized Variables

- Variables which are not initialized...
  - That value is garbage (unknown).

```
short g1, g2, g3, g4, g5, g6, g7, g8;
cout << g1 << "\t";
cout << g2 << "\t";
cout << g3 << "\t";
cout << g4 << "\n";
cout << g5 << "\t";
cout << g6 << "\t";
cout << g7 << "\t";
cout << g8 << "\n";
```

Output:

```
2052   -29221  114    8240
51     25765   -16446 2216
```

Scope

- Scope is the region of the program where

```
int main() {
    int height = 10;
    cout << height;     // OK.

    cout << width;      // ERROR: not defined yet!
    int width = 10;
    return 0;
}
```

More on this later!

Comments

Comments

- Good comments tell you
- Which comment is best?
  - float rate = 0.12; // Set to 0.12.
  - float rate = 0.12; // Set to current tax rate.
- Rule of thumb:
  - Comment the purpose of every 3-4 lines of code.

Comment Style

- Single line comments use double slash:

```
int i=2;  // Insert meaningful comment here.
```

- Multiple line comments use /* ... and ... */

```
/*
 These are good for larger comments.

 For example, describing a function's purpose,
 Arguments, return value, and errors.
*/
```

- When changing the code...
  - An incorrect comment is worse than no comment!

Summary

- #include allows us to use libraries.
- C++ variables are strongly typed.
  - Must declare variables before use.
  - Operators: +, -, *, /, %
- Types:
  - Integer: short, int, and long.
    - Can be unsigned to store twice as large a value.
  - Text: char holds a character, string holds a string.
  - Floating point: float, double, and long double.
  - True/False: bool is either true (1) or false (0)
- Include meaningful comments!