Fundamental Data Types

Lecture 4
Sections 2.7 - 2.10

Robb T. Koether

Hampden-Sydney College

Wed, Sep 4, 2013

Outline

Integer Types

Integer Type

```
    cout << 123 << endl;
```

- Integers are stored as binary numbers.
- An n-bit integer can hold any of 2^n different values.
- Integer types are either signed or unsigned.
- Integer literals must not have a decimal point.

Integer Types

Integer Types

```
int a = 123;
int b = -456;
unsigned int c = 789;
```

- The integer types.
  - **`short`** – 2 bytes
  - **`int`** – 4 bytes
  - **`long`** – 4 bytes
- Each type can be either signed or unsigned.

Signed vs. Unsigned Integers

- Unsigned integers.
  - An unsigned integer cannot be negative.
  - All of its bits constitute the number.
  - Values range from 0 to $2^n - 1$.
- Signed integers
  - A signed integer can be positive or negative.
  - One bit is designated as the sign bit.
  - The remaining $n - 1$ bits constitute the number.
  - Values range from -2^{n-1} to $2^{n-1} - 1$.

Example of Signed and Unsigned Integers

Unsigned	
000	0
001	1
010	2
011	3
100	4
101	5
110	6
111	7

Signed	
000	0
001	1
010	2
011	3
100	-4
101	-3
110	-2
111	-1

3-bit integers, unsigned and signed

Ranges of Values of Integer Type

Type	Range	
	From	To
`unsigned short`	0	65535
`short`	−32, 768	32,767
`unsigned int`	0	4,294,967,295
`int`	−2, 147, 483, 648	2,147,483,647
`unsigned long`	0	4,294,967,295
`long`	−2, 147, 483, 648	2,147,483,647

Integer Overflow

- What happens when an integer value becomes too large?
- That is, what if we assign to an integer the largest legal value, and then add 1?
- Example
  - `IntLimitTest.cpp`

Outline

Floating-Point Types

Floating-Point Type

```
cout << 12.34 << endl;
```

- Floating-point numbers are stored in two parts.
- The mantissa contains the significant digits.
- The exponent locates the decimal point.
- Floating-point literals must have a decimal point.

Floating-Point Types

Floating-Point

```
float a = 123.456;
double b = 123.456789012345;
```

- The floating-point types.
  - **`float`** – 4 bytes
  - **`double`** – 8 bytes
- Each type can be either signed or unsigned.

Single-Precision Numbers

float Type

```
float x = 12.34567;
```

- Single-precision floating point numbers (**floats**) occupy 4 bytes of memory.
  - 8-bit exponent, including the sign.
  - 24-bit mantissa, including the sign.
- The positive values range from a minimum of $\pm 1.17549 \times 10^{-38}$ to a maximum of $\pm 3.40282 \times 10^{38}$.
- Approximately 7-place precision.

Double-Precision Numbers

double Type

```
double pi = 3.141592653589793;
```

- Double-precision floating point numbers (**doubles**) occupy 8 bytes of memory.
  - 11-bit exponent, including the sign.
  - 53-bit mantissa, including the sign.
- The values range from a minimum of $\pm 2.22507 \times 10^{-308}$ to a maximum of $\pm 1.79769 \times 10^{308}$.
- Approximately 16-place precision.

Floating-Point Overflow and Underflow

- What happens if we begin with the largest possible **`float`** and then double it?
- What happens if we begin with the smallest possible positive **`float`** and divide it by 2?
- Example
    - `FloatLimitTest.cpp`

Outline

The Character Type

char Type

```
char letter = 'a';
```

- Characters (`chars`) are stored as one-byte integers using the ASCII values (see p. 1140).
- A character object can have any of 256 different values.
- Character literals must use single quotation marks.

Characters as Integers

char Type

```
char letter = 'a';
int value = letter;
letter = value + 1;
```

- Characters are interchangeable with integers in the range 0 to 255.
- The numerical value of a character is its ASCII value.
  - Blank space (ASCII 32).
  - Digits 0 - 9 (ASCII 48 - 57).
  - Uppercase letters A - Z (ASCII 65 - 90).
  - Lowercase letters a - z (ASCII 97 - 122).
- Characters are ordered according to their ASCII values.

Outline

The **string** Type

string Type

```
    string message = "Hello, World";
```

- A string is stored as a sequence of characters.
- A string may hold any number of characters, including none.
- String literals must use double quotation marks.

The **string** Type

string Type

```
string msg = "Hello, World";
cout << msg[9] << msg[1] << msg[11] << endl;
```

- The characters in the string are indexed, beginning with index 0.
- They can be accessed individually by writing the index within square brackets `[...]`.

Floating-Point Overflow and Underflow

- Example
    - `CharCalcs.cpp`

Outline

Assignment

Assignment

- Read Sections 2.7 - 2.10.