SOUTHERN ADVENTIST UNIVERSITY
School of Computing

Values, Variables, Types, Expressions, and Arithmetic

Chapters 3 and 4

1

For Next Time

- Read Chapters 3 and 4

SOUTHERN ADVENTIST UNIVERSITY
School of Computing

2

Numeric Values

- Integers
  - int, long, short, unsigned, char
- Floating point
  - float, double
- Limitations of numeric values

Integer Limits

- Visual C++

Type	Storage	Minimum Value	Maximum Value
short	16 bits	-32,768	32,767
int	32 bits	-2,147,483,648	2,147,483,647
long	32 bits	-2,147,483,648	2,147,483,647

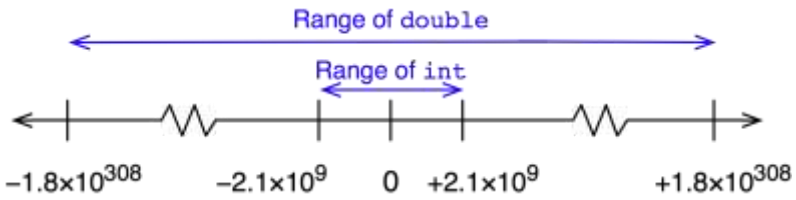
Floating-point Limits

For Visual C++

Type	Storage	Smallest Magnitude	Largest Magnitude	Minimum Precision
`float`	32 bits	1.17549×10^{-38}	$3.40282\times10^{+38}$	6 decimal digits
`double`	64 bits	2.22507×10^{-308}	$1.79769\times10^{+308}$	15 decimal digits

SOUTHERN ADVENTIST UNIVERSITY School of Computing

Data Ranges



SOUTHERN ADVENTIST UNIVERSITY School of Computing

Scientific Notation

- Floating point values can be expressed in exponential form
- 4.7×10^{-5} is expressed as `4.7e-5`
- Use `e` instead of `E` to improve readability

Variables and Assignment

- Variables store values
- The assignment operator (=) assigns a value to a variable
- All variables within a program must be declared

Constants

- "Variables" that do not change
- The `const` specifier denotes a constant
- Compiler ensures a constant will not change
- Attempts to modify a constant results in an error

Identifiers

- Used to name variables
  - Used to name other things, also
- Must start with a letter or underscore
- The rest of the name can a mixture of letters, underscores, and digits
- C++ is case sensitive
- The exceptions?

asm	auto	bool	break
case	catch	char	class
const	const_cast	continue	default
delete	do	double	dynamic_cast
else	enum	explicit	export
extern	false	float	for
friend	goto	if	inline
int	long	mutable	namespace
new	operator	private	protected
public	register	reinterpret_cast	return
short	signed	sizeof	static
static_cast	struct	switch	template
this	throw	true	try
typedef	typeid	typename	union
unsigned	using	virtual	void
volatile	wchar_t	while	

Reserved Words

- Also called *keywords*
- You'll find a list in Chapter 3
- You may not use a reserved word as an identifier (e.g., variable name)
- No need to memorize them
  - The compiler will let you know if you use one accidentally

Type Conversions

- int → double, is this OK
- double → int, is this OK?

Arithmetic

Operator	Operation
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus

Operator Precedence

- Just as in normal mathematics:
  - `*` and `/` are applied before `+` and `–`
  - Parentheses can override the order of application

SOUTHERN ADVENTIST UNIVERSITY School of Computing

Operator Associativity

- Is `x + y + z` evaluated as
  - `(x + y) + z`
  - `x + (y + z)`
- Does it matter?

SOUTHERN ADVENTIST UNIVERSITY School of Computing

Operator Arity

- Unary:
  - `-x`
  - `+sum`
- Binary
  - `sum + diff`
  - `2 - ev`

SOUTHERN ADVENTIST UNIVERSITY School of Computing

Precedence and Associativity

Precedence from high to low

Arity	Operators	Associativity
Unary	`+, -`	
Binary	`*, /, %`	Left
Binary	`+, -`	Left
Binary	`=`	Right

SOUTHERN ADVENTIST UNIVERSITY School of Computing

Mixed Arithmetic

- In an expression involving different numeric types, the less dominant types are converted to the more dominant types for the purpose of evaluating the expression
- Example

```
int x = 5;
double y = 4.0, z;
z = x + y;
```

SOUTHERN ADVENTIST UNIVERSITY School of Computing

Comments

- Helpful to human readers
- Ignored by the compiler
- Single-line comments

```
//  This is a brief note
```

- Block comments

```
/*  This is a longer remark
    that covers
    several lines.  */
```

SOUTHERN ADVENTIST UNIVERSITY School of Computing

Source Code Formatting

- Like comments: unimportant to compiler but very important to human readers
- Some guidelines:
  - Each statement on its own line
  - Align curly braces in a standard way
  - Indent bodies of functions and structured statements
  - Use spaces around binary operators

SOUTHERN ADVENTIST UNIVERSITY School of Computing

Errors

- Compile-time errors
  - Violation of the rules of the C++ language
  - Compiler tells you about them
  - Plentiful when you are first learning a language
  - Generally easy to fix
- Runtime errors
  - Depends on the situation of the running program: like dividing by a variable that has been assigned zero
  - The run-time environment terminates the program's execution

SOUTHERN ADVENTIST UNIVERSITY School of Computing

Logic Errors

- Not detectable by the compiler
- Not detectable by the run-time environment
- The program's logic is not correct
  - The hardest kinds of errors to diagnose and repair
  - The compile can provide no feedback
- Can result from carelessness, lack of understanding of the problem, lack of understanding of the way the language works; for example:
  - Wrote `x - y`, meant `y - x`

SOUTHERN ADVENTIST UNIVERSITY School of Computing

Logic Error?

$$^{\circ}C = \frac{5}{9} \times (^{\circ}F - 32)$$

```
double degrees_F, degrees_C;
cin >> degrees_F;
degrees_C = 5/9*(degrees_F - 32);
```

SOUTHERN ADVENTIST UNIVERSITY School of Computing

Additional Arithmetic Operators

- Increment: **++**
- Decrement: **--**
- Increase: **+=**
- Other operations similar to increase:
  - **-=**
  - ***=**
  - **/=**
  - **%=**
  - Others . . .

SOUTHERN ADVENTIST UNIVERSITY School of Computing

Next . . .

Conditional execution