



# C++ Programs

- Simple Program
- Statement Execution
- C++ Statements
- Name Spaces

# [ Simple Program ]

## ■ Program Features

- Comments:
  - `//`
  - `/* */`
- Preprocessor directives: `#`
- Libraries: `<..>`, `" "`
- Name spaces: `using...`
- Program: `main(){ }`
- Variable Declaration ;

# [ Names for Variable, etc. ]

- Names: Collection of upper and lower case letters, numbers, and underscores not beginning with a number.
  - My\_Salary, MySalary
  - \_mySalary, salary9
  - sampleData

# [ Keywords ]

- Names that have predefined meaning in C++
  - cout, cin, endl
  - for, main
  - const, int, double, float
  - using, return
  - (and others)

# [Using DDD]

- A debugger is a program that lets you trace the behavior of a program
- Data Display Debugger
  - Part of many unix distributions
  - Note: when cin is used, one must type enter at the end of each line so that the program will know that the line is complete.

# [ Compiling for DDD ]

- `g++ -g -o fig02_02 fig02_02.cpp`
  - `-g` adds additional information to object code required by debugger.
  - `-o` tells the compiler to name the linked file `fig02_02`

# [ Running Example ]




- ddd fig02\_02
- Locate window with “Run” at the top, right click on the X icon at the top left of the window, and select “always on top”
- In the middle window of the large display, click on the line “int main()”
- Click of the “Break” icon

# [ Running Example ]

- Click on the “Run” button.
- In turn, highlight each of the variables, click on the “Display” icon and move the display in the top window to a convenient position
- Repeatedly, click on the “Next” button and watch what happens to the variables.



# [ Data Types ]

- Char  usually one byte
- Double  usually 8 bytes
  - $1.3 \times 10^{-8}$
  - $4.584848238 \times 10^{20}$
  - About 16 decimal digit mantissa and exponent range -307 to 308
- Integer  usually 4 bytes -2,147,483,648 to +2,147,483,647.

# [ Other Data Types ]

- Float: usually four bytes, about 7 digit mantissa and exponent range -37 to 38
- Short: usually 2 bytes -32,767 to 32767
- Unsigned short int: usually 2 bytes 0 to 65,535
- Unsigned int: usually 4 bytes 0 to 4,294,967,295.

# [ Numerical Errors ]

- Since a computer uses base two to represent floating point numbers common decimal numbers cannot be represented exactly in computer memory.
  - Exact 2.5, 7.25, 19.125
  - Not exact 4.3, 7.9
- In general one can only expect 16 significant digit accuracy for double

# [ Numerical Errors ]

- In general one can only expect at most 16 significant digit accuracy for double precision
- In general one can only expect at most 7 significant digit accuracy for single precision.
- In double precision computer arithmetic if  $x=3.2e13$  and  $y=3.2e13$ , then  $(x+0.00008)-y=0$  (cancellation error)

# [ Character Data ]

- A byte holds 0-255.
- Each character is assigned a number between 0-255. The most common code for representing number in a computer is **ASCII**
- In this code
  - 0 'a' is 97, 'A' is 65
  - 0 '~' is 126, '}' is 125
- **char2dec.cpp**

# [ Arithmetic Expressions ]

- Is  $x+y*z$  the same as  $(x+y)*z$ ?
- Is  $x*y*z$  the same as  $(x*y)*z$ ? What about  $x*(y*z)$ ?
- Is  $-2^4$  equal to  $(-2)^4$

# Arithmetic Expression

## Answers

○ No, yes, yes (but sometimes not on computers), no.

**TABLE 2.3** Evaluation of Expressions

1. Parentheses	Evaluate parenthesized subexpressions first.
2. Precedence	Evaluate operators according to this precedence: (i) unary +          unary – (ii) *                    /                    % (iii) binary +          binary –
3. Associativity	If there is a sequence of two or more operators of the same precedence, evaluate unary operators right to left (right associativity) and binary operators left to right (left associativity).

# [ Arithmetic Expression ]

- Moral: when in doubt use parentheses to direct the computer to do the calculation in the order you want it done.
- $-p/s*q-s+d*-p/t+r =$   
 $((((( (-p)/s ) * q ) - s ) + ((( d * (-p) ) / t ) ) ) + r$



# [ Arithmetic Expression ]

■  $-p/s*q-s+d*-p/t+r =$

○  $a=-p$

○  $a=a/s$

○  $a=a*q$

○  $a=a-s$

○  $b=-p$

○  $b=d*b$

○  $b=b/t$

○  $ans=(a+b)+r$

# [ Arithmetic Expression ]

- Integer division

  - $5/6=0$ ,  $11/6=1$ ,  $55/6=9$

  - $4/-3=-1$ ,  $-4/3=-1$

  - $-4/-3=1$

- Modulus ( $a \% b = a - (a/b) * b$ )

  - $11 \% 6=5$ ,  $55 \% 6=1$

  - $4 \% -3 = -1$ ,  $-4 \% 3 = -1$

  - $-4 \% -3 = -1$

- `intarith.cpp`

# [ Mixed Type Expressions ]

- Integer op double produces a double
- Integer op float produces a float.
- Unsigned op signed produces unsigned

# [ Assignment Statements ]

- `int a=15.9-0.91, a=14`
- `float a= 9 / 5, a=1`
- C++ evaluates the expression then stores it to the variable.
- `double a= double(9)/double(5), a=1.8`
- `double a= (double) 9/(double)5, a=1.8`

# [Exercises]

- Odd numbered exercise
  - 2.1
  - 2.2
  - 2.3
  - 2.4
  - 2.5