Arithmetic Calculations

Instructor: Chien-Ho Ko

Outlines

- Basic arithmetic operators
- Increment & decrement operators
- Compound assignment operators
- Type of arithmetic expression
- Style considerations for mathematic statements

Basic Arithmetic Operators

- **Unary arithmetic operators**
  - **Require one operand**
  - **+ (positive), - (negative), ++, --**
- **Binary arithmetic operators**
  - **Require two operands**
  - **+, -, *, /, %**

Example 1

- General C
- Calculate rectangle's diameter and area.
- 08_c01.c

Example 2

- **Robotic C**
  - **Robot GO!**
  - **NXC**
- **Robot control**
- **08_RobotC01.nxc**

Basic Arithmetic Operators

- **Precedence rule**
  - **(a+b/(c-5))/((d+7)/(e-37)/3)**
  - **a=10, b=20, c=15, d=8, e=40**

Step	Reduced expression
1	(a+b/10)/((d+7)/(e-37)/3)
2	(a+b/10)/(15/(e-37)/3)
3	(a+b/10)/(15/3/3)
4	(a+2)/(15/3/3)
5	12/(15/3/3)
6	12/(5/3)
7	12/1.6667
8	7.2

Increment & Decrement Operators

- **Decrement operator**
  - **--**
  - **i--**
- **Increment operator**
  - **++**
  - **i++**

Compound Assignment Operators

- **Simple assignment operator**
  - **=**
- **Compound assignment operators**
  - **Assign sum, +=**

```
int nNumber=1;
nNumber += 3;    // nNumber now is 4
```

  - **Assign difference, -=**
  - **Assign product, *=**
  - **Assign division, /=**
  - **Assign reminder, %=**

Type of Arithmetic Expression

- **Expressions involve *int* & *double***
  - **One operand double, result is double**
    - **10/3=3.333333**
  - **All operands are integers, result is integer**
    - **10/3=3**
- **Explicit type conversion**

Mathematical Library Functions

- **Floating-point type function**
  - **Accept and return *doubles***
  - **#include<math.h>**
- **Integer type function**
  - **Accept and return *integers***
  - **#include<stdlib.h>**
- **() Function call**

Example 3

- General C
- Calculate triangle's diameter and area.
- 08_c02.c

Styles for Mathematic Statements

- **Use parentheses only if necessary**
- **Use simple expressions**
- **Avoid increment operators in complex**
- **Avoid long expressions**