EECS 10: Computational Methods in Electrical and Computer Engineering

Lecture 3

Rainer Dömer

doemer@uci.edu

The Henry Samueli School of Engineering
Electrical Engineering and Computer Science
University of California, Irvine

Lecture 3.1: Overview

- Review Quiz
- Comparison of Values
  - Relational Operators
  - Logical Operators
  - Conditional Operator
- Conditional Statements
  - **`if`** statement
- Conditional Programming
  - Example **`Comparison.c`**

Quiz: Question 11

- What is the value of the integer **x** after the following statement?

```
x = 3 << 2 >> 1;
```

a) **Syntax Error!**
b) **3**
c) **6**
d) **12**
e) **321**

Quiz: Question 11

- What is the value of the integer **x** after the following statement?

```
x = 3 << 2 >> 1;
```

a) **Syntax Error!**
b) **3**
c) **6**
d) **12**
e) **321**

Quiz: Question 12

- Which of the following constants is of type **`double`**?
  (Check all that apply!)
  a) **`42`**
  b) **`.42`**
  c) **`4e2`**
  d) **`4E2`**
  e) **`42f`**

Quiz: Question 12

- Which of the following constants is of type **`double`**?
  (Check all that apply!)
  a) **`42`**
  b) **`.42`**
  c) **`4e2`**
  d) **`4E2`**
  e) **`42f`**

Quiz: Question 13

- What is the result type of the following expression?

```
-1 + 2.3f * (4.5 / 67f) – (short)89
```

a) **short int**
b) **int**
c) **long int**
d) **float**
e) **double**

Quiz: Question 13

- What is the result type of the following expression?

```
-1 + 2.3f * (4.5 / 67f) – (short)89
```

a) **short int**
b) **int**
c) **long int**
d) **float**
e) **double**

Quiz: Question 14

- What is the value of **x** after the following code segment?

```
int    i = 10;
double d = 0.5;
double x;

x = i/3 + d;
```

a) **0.333333**
b) **3.0**
c) **3.333333**
d) **3.5**
e) **3.833333**

Quiz: Question 14

- What is the value of **x** after the following code segment?

```
int    i = 10;
double d = 0.5;
double x;

x = i/3 + d;
```

a) **0.333333**
b) **3.0**
c) **3.333333**
d) **3.5**
e) **3.833333**

Quiz: Question 15

- Given the following code fragment,

```
double x;
double y;

x = (int)(y + 0.5);
```

which of the following statements is true? (Check all that apply!)

a) for **y=5.0**, **x** is set to **5.0**
b) for **y=5.1**, **x** is set to **5.0**
c) for **y=5.49**, **x** is set to **5.0**
d) for **y=5.5**, **x** is set to **6.0**
e) for **y=5.95**, **x** is set to **6.0**

Quiz: Question 15

- Given the following code fragment,

```
double x;
double y;

x = (int)(y + 0.5);
```

which of the following statements is true? (Check all that apply!)

a) for **y=5.0**, **x** is set to **5.0**
b) for **y=5.1**, **x** is set to **5.0**
c) for **y=5.49**, **x** is set to **5.0**
d) for **y=5.5**, **x** is set to **6.0**
e) for **y=5.95**, **x** is set to **6.0**

Comparison of Values

- Relational Operators
  - direct comparison of two values
  - Boolean result: truth value, true or false
- Logical Operators
  - Operations on Boolean values
- Conditional Operator
  - Conditional evaluation of expressions

Relational Operators

- Comparison operations
  - **<** less than
  - **>** greater than
  - **<=** less than or equal to
  - **>=** greater than or equal to
  - **==** equal to (remember, **=** means assignment!)
  - **!=** not equal to
- Comparison is defined for all basic types
  - integer (e.g. **`5 < 6`**)
  - floating point (e.g. **`7.0 < 7e1`**)
- Result type is Boolean, but represented as integer
  - false **`0`**
  - true **`1`** **(**or any other value *not* equal to zero**)**

Logical Operators

- Operation on Boolean/truth values
  - `!` "not" logical negation
  - `&&` "and" logical and
  - `||` "or" logical or
- Truth table:

x	y	!x	x && y	x || y
0	0	1	0	0
0	1	1	0	1
1	0	0	0	1
1	1	0	1	1

- Argument and result types are Boolean, but represented as integer
  - false `0`
  - true `1` (or any other value *not* equal to zero)

EECS10: Computational Methods in ECE, Lecture 3 (c) 2013 R. Doemer 15

Conditional Operator

- Conditional evaluation of values in expressions
- Question-mark operator:
  `test ? true-value : false-value`
  - evaluates the ***`test`***
  - if ***`test`*** is true, then the result is ***`true-value`***
  - otherwise, the result is ***`false-value`***
- Examples:
  - `(4 < 5) ? (42) : (4+8)` evaluates to `42`
  - `(2==1+2) ? (x) : (y)` evaluates to `y`
  - `(x < 0) ? (-x) : (x)` evaluates to `abs(x)`

EECS10: Computational Methods in ECE, Lecture 3 (c) 2013 R. Doemer 16

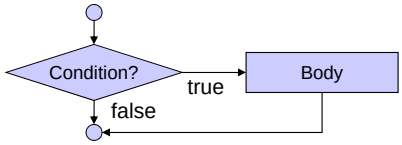
Operator Evaluation Order

- Associativity: left to right or right to left
- Precedence: group-wise, top to bottom

– parentheses	`(`, `)`	n/a
– unary plus, minus, negation	`+`, `-`, `!`	right to left
– type casting	`(`*`typename`*`)`	right to left
– multiplication, division, modulo	`*`, `/`, `%`	left to right
– addition, subtraction	`+`, `-`	left to right
– shift left, shift right	`<<`, `>>`	left to right
– relational operators	`<`, `<=`, `>=`, `>`	left to right
– equality	`==`, `!=`	left to right
– logical and	`&&`	left to right
– logical or	`||`	left to right
– conditional operator	`?:`	left to right
– assignment operator	`=`	right to left

Conditional Statements

- **`if`** statement
  - Control flow statement for decision making
    - Changes control flow depending on a specified condition
  - Control flow chart:



  - Semantics:
    - Body is executed *only if* the condition evaluates to true

Conditional Statements

- **`if`** statement
  - Control flow statement for decision making
    - Changes control flow depending on a specified condition
  - Example:
    - ```
      if (x < 0)
          { printf("%d is negative", x); }
      ```
    - ```
      if (x >= 0)
          { printf("%d is positive", x); }
      ```
  - Syntax: **`if`** construct consists of
    - Keyword **`if`**
    - Condition expression evaluated to true or false
    - Body statement block

Example Program

- Comparison of values: **`Comparison.c`** (part 1/3)

```
/* Comparison.c: arithmetic comparisons         */
/*                                              */
/* author: Rainer Doemer                        */
/*                                              */
/* modifications:                               */
/* 10/07/04 RD  initial version                 */

#include <stdio.h>

/* main function */

int main(void)
{
   /* variable definitions */
   int a, b;

...
```

Example Program

- Comparison of values: **`Comparison.c`** (part 2/3)

```
...
   /* input section */
   printf("Please enter a value for integer a: ");
   scanf("%d", &a);
   printf("Please enter a value for integer b: ");
   scanf("%d", &b);

   /* computation and output section */
   if (a == b)
      { printf("%d is equal to %d.\n", a, b);
       } /* fi */
   if (a != b)
      { printf("%d is not equal to %d.\n", a, b);
       } /* fi */
   if (a < b)
      { printf("%d is less than %d.\n", a, b);
       } /* fi */
...
```


Example Program

- Comparison of values: **`Comparison.c`** (part 3/3)

```
...
   if (a > b)
      { printf("%d is greater than %d.\n", a, b);
       } /* fi */
   if (a <= b)
      { printf("%d is less than or equal to %d.\n", a, b);
       } /* fi */
   if (a >= b)
      { printf("%d is greater than or equal to %d.\n", a, b);
       } /* fi */

   /* exit */
   return 0;
} /* end of main */

/* EOF */
```

Example Program

- Example session: **`Comparison.c`**

```
% vi Comparison.c
% gcc -Wall -ansi Comparison.c -o Comparison
% Comparison
Please enter a value for integer a: 42
Please enter a value for integer b: 56
42 is not equal to 56.
42 is less than 56.
42 is less than or equal to 56.
% Comparison
Please enter a value for integer a: 6
Please enter a value for integer b: 6
6 is equal to 6.
6 is less than or equal to 6.
6 is greater than or equal to 6.
% Comparison
Please enter a value for integer a: 77
Please enter a value for integer b: 6
77 is not equal to 6.
...
```