

# 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**

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

5

## 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

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

6

### 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

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

13

## 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)

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

14

## 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)

## 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)**



## 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	( <i>typename</i> )	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

EECS10: Computational Methods in ECE, Lecture 3

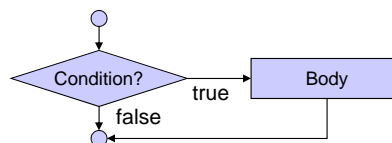
(c) 2014 R. Doemer

17

## 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

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

18

## 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.
...
```

## Lecture 3.2: Overview

- Keywords in C
- Counters
  - Augmented Assignment Operators
  - Increment and Decrement Operators
- Repetition Statements
  - `while` loop
- Counter-controlled repetition
  - Example `Average.c`
- Sentinel-controlled repetition
  - Example `Average2.c`

## Keywords in C

- List of keywords in ANSI-C
  - `auto`      - `double`      - `int`            - `struct`
  - `break`     - `else`         - `long`           - `switch`
  - `case`       - `enum`         - `register`      - `typedef`
  - `char`       - `extern`       - `return`       - `union`
  - `const`      - `float`         - `short`          - `unsigned`
  - `continue`   - `for`           - `signed`        - `void`
  - `default`    - `goto`          - `sizeof`        - `volatile`
  - `do`          - `if`            - `static`        - `while`
- These keywords are reserved!
- Keywords cannot be used as identifiers.
- More keywords are reserved for C++

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

25

## Augmented Assignment Operators

- Assignment operator: `=`
  - evaluates right-hand side
  - assigns result to left-hand side
- Augmented assignment operators: `+=`, `*=`, ...
  - evaluates right-hand side as temporary result
  - applies operation to left-hand side and temporary result
  - assigns result of operation to left-hand side
- Example: Counter
  - `int c = 0; /* counter starting from 0 */`
  - `c = c + 1; /* counting by regular assignment */`
  - `c += 1; /* counting by augmented assignment */`
- Augmented assignment operators:
  - `+=`, `-=`, `*=`, `/=`, `%=`, `<<=`, `>>=`, `||=`, `&&=`

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

26

## Increment and Decrement Operators

- Counting in steps of one
  - increment (add 1)
  - decrement (subtract 1)
- C provides special operators
  - increment operator: ++
    - `count++` post-increment (`count += 1`)
    - `++count` pre-increment (`count += 1`)
  - decrement operator: --
    - `count--` post-decrement (`count -= 1`)
    - `--count` pre-decrement (`count -= 1`)
  - *pre-* increment/decrement
    - value returned is the incremented/decremented (new) value
  - *post-* increment/decrement
    - value returned is the original (old) value

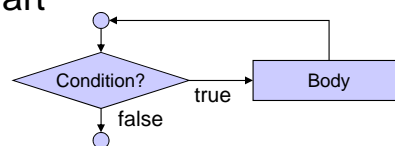
EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

27

## Repetition Statements

- Repetition (aka. iteration, loop)
  - repeated execution of a block of statements
  - counter-controlled
    - counter determines number of repetitions (often predefined at compile time)
  - sentinel-controlled
    - sentinel condition determines number of repetitions (usually determined at run time)
- Control flow chart



EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

28

## Repetition Statements

- **while** loop
  - Control flow statement for repetition (iteration)
    - Repeats execution depending on a specified condition
  - Example:
 

```
int product = 2;
while (product < 1000)
  { product *= 2; }
printf("Product is %d", product);
```
  - Syntax: **while** construct consists of
    - keyword `while`
    - condition expression evaluated to true or false
    - body statement block
  - Semantics: the body is repeatedly executed as long as the condition evaluates to true
    - the condition is evaluated at the *beginning* of each loop

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

29

## Example Program

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

```
/* Average.c: compute the average of a set of numbers */
/*
/* author: Rainer Doemer
/*
/* modifications:
/* 10/10/04 RD initial version
*/

#include <stdio.h>

/* main function */

int main(void)
{
  /* variable definitions */
  int counter;
  double value;
  double total;
  double average;
  ...
}
```

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

30

## Example Program

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

```
...  
  
/* input and computation section */  
counter = 1;  
total = 0.0;  
while (counter <= 10)  
{ printf("Please enter value %d: ", counter);  
  scanf("%lf", &value);  
  total += value;  
  counter++;  
} /* elihw */  
  
/* computation section */  
average = total / 10.0;  
  
...
```

## Example Program

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

```
...  
  
/* output section */  
printf("The average is %f.\n", average);  
  
/* exit */  
return 0;  
} /* end of main */  
  
/* EOF */
```



## Example Program

- Example session: `Average.c`

```
% vi Average.c
% gcc Average.c -o Average -Wall -ansi
% Average
Please enter value 1: 23
Please enter value 2: 25
Please enter value 3: 17
Please enter value 4: 18.6
Please enter value 5: 50.8
Please enter value 6: 33.3
Please enter value 7: 12
Please enter value 8: 42
Please enter value 9: 42.2
Please enter value 10: 34
The average is 29.790000.
%
```

## Repetition Statements

- Explicit control flow in loops
  - `break` statement
    - exits the innermost loop
  - `continue` statement
    - jump back to the beginning of the innermost loop

- Example:

```
int i = 0;
int s = 0;
while (1) /* "endless" loop */
{
    i++;
    if (i > 10)
        { break; } /* exit the loop */
    if (i % 2 == 1)
        { continue; } /* next iteration */
    s += i;
} /* elihw */
printf("%d", s);
```

## Example Program

- Average of values: `Average2.c` (part 1/3)

```

/* Average2.c: compute the average of a set of numbers */
/*
/* author: Rainer Doemer */
/*
/* modifications: */
/* 10/10/04 RD sentinel controlled loop */
/* 10/10/04 RD initial version */

#include <stdio.h>

/* main function */

int main(void)
{
    /* variable definitions */
    int counter;
    double value;
    double total;
    double average;
    ...

```

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

35

## Example Program

- Average of values: `Average2.c` (part 2/3)

```

...

/* input and computation section */
counter = 0;
total = 0.0;
while (1)
{ printf("Please enter a value (or -1 to quit): ");
  scanf("%lf", &value);
  if (value == -1.0)
  { break;
    } /* fi */
  total += value;
  counter++;
} /* elihw */

...

```

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

36

## Example Program

- Average of values: `Average2.c` (part 3/3)

```

...

/* computation and output section */
printf("%d values entered.\n", counter);
if (counter >= 1)
    { average = total / (double)counter;
      printf("The average is %f.\n", average);
    } /* fi */

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

/* EOF */

```

## Example Program

- Example session: `Average2.c`

```

% vi Average2.c
% gcc Average2.c -o Average2 -Wall -ansi
% Average2
Please enter a value (or -1 to quit): 2
Please enter a value (or -1 to quit): 3
Please enter a value (or -1 to quit): 4
Please enter a value (or -1 to quit): 5
Please enter a value (or -1 to quit): -1
4 values entered.
The average is 3.500000.
% Average2
Please enter a value (or -1 to quit): -1
0 values entered.
%

```


## Lecture 3.3: Overview

- Review
  - Lecture 1.1: Course administration, setup
  - Lecture 1.2: Unix system environment
  - Lecture 1.3: Introduction to C programming
  - Lecture 2.1: Input, computation, output
  - Lecture 2.2: Basic types, operators
  - Lecture 2.3: Arithmetic expressions
  - Lecture 3.1: Conditional operators, statements
  - Lecture 3.2: Counters, repetition statements
- Review Quiz

## Quiz: Question 16

- Today's computers run at which clock speed?
  - a) 85 MPH
  - b) 1 kHz
  - c) 1 ms
  - d) 1 GHz
  - e) 1 MHz

## Quiz: Question 16

- Today's computers run at which clock speed?
  - a) 85 MPH
  - b) 1 kHz
  - c) 1 ms
  -  d) 1 GHz
  - e) 1 MHz

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

41

## Quiz: Question 17

- Which of the following constructs are valid type names in C? (Check all that apply!)
  - a) `short char`
  - b) `long double`
  - c) `signed long int`
  - d) `unsigned float`
  - e) `signed integer`

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

42

## Quiz: Question 17

- Which of the following constructs are valid type names in C? (Check all that apply!)

a) `short char`

b) `long double`

c) `signed long int`

d) `unsigned float`

e) `signed integer`

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

43

## Quiz: Question 18

- Assume `i` is a variable of type `int` and `d` is a variable of type `double`. Which statement is true for the following assignment? (Check all that apply!)

```
i = (int)d;
```

a) The comparison checks whether `d` is an integer.

b) The precision of `i` is doubled.

c) The parentheses should go around `d`.

d) The value in `d` is converted to an integer value and then assigned to `i`.

e) Any fractional part in `d` is truncated off.

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

44

## Quiz: Question 18

- Assume `i` is a variable of type `int` and `d` is a variable of type `double`. Which statement is true for the following assignment? (Check all that apply!)

```
i = (int)d;
```

- a) The comparison checks whether `d` is an integer.
- b) The precision of `i` is doubled.
- c) The parentheses should go around `d`.
- d) The value in `d` is converted to an integer value and then assigned to `i`.
- e) Any fractional part in `d` is truncated off.

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

45

## Quiz: Question 19

- Which of the following statements correctly computes the polynomial  $p = 2x^2 - 3x + 4$ ? (Check all that apply!)

- a) `p = 2x^2 - 3x + 4;`
- b) `p = 2xx - 3x + 4;`
- c) `p = x*x*2 - 3*x + 4.0;`
- d) `p = 2*(x*x + 3)*x + 4;`
- e) `p = (2*x - 3)*x + 4;`

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

46

### Quiz: Question 19

- Which of the following statements correctly computes the polynomial  $p = 2x^2 - 3x + 4$ ? (Check all that apply!)

a) `p = 2x^2 - 3x + 4;`

b) `p = 2xx - 3x + 4;`

c) `p = x*x*2 - 3*x + 4.0;`

d) `p = 2*(x*x + 3)*x + 4;`

e) `p = (2*x - 3)*x + 4;`

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

47

### Quiz: Question 20

- Which of the following names are valid keywords in C? (Check all that apply!)

a) `do`

b) `when`

c) `void`

d) `main`

e) `Int`

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

48



## Quiz: Question 20

- Which of the following names are valid keywords in C? (Check all that apply!)

- a) `do`
- b) `when`
- c) `void`
- d) `main`
- e) `Int`

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

49

## Quiz: Question 21

- Which of the following names are valid identifiers in C? (Check all that apply!)

- a) `xyz123`
- b) `IBM`
- c) `dollar amount`
- d) `My_Very_Long_Variable_Name`
- e) `2fast4you`

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

50

## Quiz: Question 21

- Which of the following names are valid identifiers in C? (Check all that apply!)

- a) `xyz123`
- b) `IBM`
- c) `dollar amount`
- d) `My_Very_Long_Variable_Name`
- e) `2fast4you`

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

51

## Quiz: Question 22

- What is the result of the evaluation of the following expression?

```
1 == 2 || 3 < 4 && 5 > 6
```

- a) `123456`
- b) `true`
- c) `false`
- d) `1`
- e) `0`

EECS10: Computational Methods in ECE, Lecture 3


(c) 2014 R. Doemer

52

## Quiz: Question 22

- What is the result of the evaluation of the following expression?

```
1 == 2 || 3 < 4 && 5 > 6
```

- a) 123456
- b) true
- c) false
- d) 1
-  e) 0

## Quiz: Question 23

- What is the result of the evaluation of the following expression?


```
17 < 42 ? 17 : 42
```

- a) 1742
- b) 17
- c) 42
- d) true
- e) false

## Quiz: Question 23

- What is the result of the evaluation of the following expression?

```
17 < 42 ? 17 : 42
```

- a) 1742
-  b) 17
- c) 42
- d) true
- e) false

## Quiz: Question 24

- For integer  $x = 1$  at the beginning, what is the value of  $x$  after the following statement?


```
x += x + 1;
```

- a) 0
- b) 1
- c) 2
- d) 3
- e) 4

### Quiz: Question 24

- For integer  $x = 1$  at the beginning, what is the value of  $x$  after the following statement?

```
x += x + 1;
```

- a) 0
- b) 1
- c) 2
-  d) 3
- e) 4

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

57

### Quiz: Question 25

- Assuming that  $x$  is a variable of type `int`, which values of  $x$  satisfy the following condition?

```
x % 2 == 1
```

- a) no value
- b) any value
- c) any value less than 2
- d) any odd value
- e) any even value

EECS10: Computational Methods in ECE, Lecture 3


(c) 2014 R. Doemer

58

## Quiz: Question 25

- Assuming that  $x$  is a variable of type `int`, which values of  $x$  satisfy the following condition?

```
x % 2 == 1
```

- a) no value
- b) any value
- c) any value less than 2
-  d) any odd value
- e) any even value

## Quiz: Question 26

- Assume that  $x$  is an integer in the range of 1 through 10 inclusively. Which of the following expressions can be used as a test for  $x$  being an even number?

(Check all that apply!)

- a) `x % 2 == 0`
- b) `x / 2 > 1`
- c) `x % 2 == 1`
- d) `x / 2 * 2 == x`
- e) `x==2 || x==4 || x==6 || x==8 || x==10`

## Quiz: Question 26

- Assume that  $x$  is an integer in the range of 1 through 10 inclusively. Which of the following expressions can be used as a test for  $x$  being an even number?

(Check all that apply!)

- a)  $x \% 2 == 0$
- b)  $x / 2 > 1$
- c)  $x \% 2 == 1$
- d)  $x / 2 * 2 == x$
- e)  $x==2 \ || \ x==4 \ || \ x==6 \ || \ x==8 \ || \ x==10$

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

61

## Quiz: Question 27

- Given the following program fragment, what is printed when it gets executed?

- a) nothing
- b) 0
- c) 10
- d) 20
- e) 30

```
int i = 1;
int s = 0;
while (1)
{
    i++;
    if (i >= 10)
        { break; }
    if (i % 2 == 1)
        { continue; }
    s += i;
}
printf("%d", s);
```


EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

62

## Quiz: Question 27

- Given the following program fragment, what is printed when it gets executed?

- a) nothing
- b) 0
- c) 10
-  d) 20
- e) 30

```
int i = 1;
int s = 0;
while (1)
{
    i++;
    if (i >= 10)
        { break; }
    if (i % 2 == 1)
        { continue; }
    s += i;
}
printf("%d", s);
```

## Quiz: Question 28

- Which of the following variable declarations is valid in ANSI-C?  
(Check all that apply!)

- a) `double xyz;`
- b) `double xy, z;`
- c) `double x = .1;`
- d) `double x = 1.1, y = 2.2, z = 3.3;`
- e) `double x,y,z = 1.0,2.0,3.0;`



## Quiz: Question 28

- Which of the following variable declarations is valid in ANSI-C?

(Check all that apply!)


- a) `double xyz;`
- b) `double xy, z;`
- c) `double x = .1;`
- d) `double x = 1.1, y = 2.2, z = 3.3;`
- e) `double x,y,z = 1.0,2.0,3.0;`

## Quiz: Question 29

- Which of the following data types has the largest range of representable numbers?

- a) `char`
- b) `short int`
- c) `long long int`
- d) `unsigned int`
- e) `signed long int`

### Quiz: Question 29

- Which of the following data types has the largest range of representable numbers?
  - a) `char`
  - b) `short int`
  -  c) `long long int`
  - d) `unsigned int`
  - e) `signed long int`

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

67

### Quiz: Question 30

- Which of the following data types can store the greatest value?
  - a) `long int`
  - b) `long long int`
  - c) `unsigned long long int`
  - d) `float`
  - e) `double`

EECS10: Computational Methods in ECE, Lecture 3

(c) 2014 R. Doemer

68

## Quiz: Question 30

- Which of the following data types can store the greatest value?
  - a) `long int`
  - b) `long long int`
  - c) `unsigned long long int`
  - d) `float`
  - e) **`double`**