

# EECS 10: Computational Methods in Electrical and Computer Engineering

## Lecture 10

Rainer Dömer

doemer@uci.edu

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

## Lecture 10: Overview

- Midterm 1 Review Quiz
  - Top 5 most “difficult” questions
- Structured Programming
  - Control flow charts
  - Sequential statements
  - Conditional statements
    - `if` statement
    - `if-else` statement
    - `switch` statement
  - Structured Program Composition
  - Example `Grade.c`
  - Example `Grade2.c`

## Midterm 1 Review Quiz

- Top 5 most “difficult” questions:
  - Rank 5: Question 6 (65.8% wrong answers)
- Which of the following statements is true about data types in ANSI-C?  
(Check all that apply! 2 pts.)
  - a) `int` has a larger range than `char`
  - b) `char` can store a smaller value than `unsigned int`
  - c) `long` has a smaller range than `unsigned int`
  - d) `float` has a higher precision than `double`
  - e) `float` can store a greater value than `long int`

EECS10: Computational Methods in ECE, Lecture 10

(c) 2007 R. Doemer

3

## Midterm 1 Review Quiz

- Top 5 most “difficult” questions:
  - Rank 5: Question 6 (65.8% wrong answers)
- Which of the following statements is true about data types in ANSI-C?  
(Check all that apply! 2 pts.)
  - a) `int` has a larger range than `char`
  - b) `char` can store a smaller value than `unsigned int`
  - c) `long` has a smaller range than `unsigned int`
  - d) `float` has a higher precision than `double`
  - e) `float` can store a greater value than `long int`

EECS10: Computational Methods in ECE, Lecture 10

(c) 2007 R. Doemer

4

## Midterm 1 Review Quiz



- Top 5 most “difficult” questions:
  - Rank 4: Question 5 (69.9% wrong answers)
- Which of the following constructs denotes a valid type name in C?  
(Check all that apply! 2 pts.)
  - a) `short char`
  - b) `unsigned char`
  - c) `unsigned long int`
  - d) `short double`
  - e) `signed float`

EECS10: Computational Methods in ECE, Lecture 10

(c) 2007 R. Doemer

5

## Midterm 1 Review Quiz

- Top 5 most “difficult” questions:
  - Rank 4: Question 5 (69.9% wrong answers)
- Which of the following constructs denotes a valid type name in C?  
(Check all that apply! 2 pts.)
  - a) `short char`
  -  b) `unsigned char`
  -  c) `unsigned long int`
  - d) `short double`
  - e) `signed float`

EECS10: Computational Methods in ECE, Lecture 10

(c) 2007 R. Doemer

6

## Midterm 1 Review Quiz

- Top 5 most “difficult” questions:
  - Rank 3: Question 13 (75.1% wrong answers)
- Which of the following C expressions yield the same result?  
(Check all that apply!)
  - $4 \ll 8 \% 5 / 2$
  - $(4 \ll 8) \% 5 / 2$
  - $4 \ll 8 \% (5 / 2)$
  - $(4 \ll 8 \% 5) / 2$
  - $4 \ll (8 \% 5) / 2$



EECS10: Computational Methods in ECE, Lecture 10

(c) 2007 R. Doemer

7

## Midterm 1 Review Quiz

- Top 5 most “difficult” questions:
  - Rank 3: Question 13 (75.1% wrong answers)
- Which of the following C expressions yield the same result?  
(Check all that apply!)
 

	a) $4 \ll 8 \% 5 / 2$	$= 8$
	b) $(4 \ll 8) \% 5 / 2$	$= 2$
	c) $4 \ll 8 \% (5 / 2)$	$= 4$
	d) $(4 \ll 8 \% 5) / 2$	$= 16$
	e) $4 \ll (8 \% 5) / 2$	$= 8$

EECS10: Computational Methods in ECE, Lecture 10

(c) 2007 R. Doemer

8

## Midterm 1 Review Quiz

- Top 5 most “difficult” questions:
  - Rank 2: Question 20 (79.8% wrong answers)

```
unsigned int x=0, y=0;
scanf("%d", &x);
while((x>=1) != 0)
  {y += 1;}
printf("%d", y);
```

- Which of the following statements are true about the program? (Check all that apply!)
  - a)  $y$  will be the integer part of  $\log_2(x)$
  - b)  $y$  will be equal to  $x$
  - c) It computes the product of  $x$  and  $y$
  - d) It sets  $y$  to the sum of  $x$  and  $y$
  - e) The condition in line 3 is equivalent to  $(x/=2) != 0$

EECS10: Computational Methods in ECE, Lecture 10

(c) 2007 R. Doemer

9

## Midterm 1 Review Quiz

- Top 5 most “difficult” questions:
  - Rank 2: Question 20 (79.8% wrong answers)

```
unsigned int x=0, y=0;
scanf("%d", &x);
while((x>=1) != 0)
  {y += 1;}
printf("%d", y);
```

- Which of the following statements are true about the program? (Check all that apply!)
  - a)  $y$  will be the integer part of  $\log_2(x)$
  - b)  $y$  will be equal to  $x$
  - c) It computes the product of  $x$  and  $y$
  - d) It sets  $y$  to the sum of  $x$  and  $y$
  - e) The condition in line 3 is equivalent to  $(x/=2) != 0$

EECS10: Computational Methods in ECE, Lecture 10

(c) 2007 R. Doemer

10

## Midterm 1 Review Quiz

- Top 5 most “difficult” questions:
  - Rank 1: Question 21 (79.8% wrong answers)

```
unsigned int x=0, y=0;
scanf("%d", &x);
while((x>=1) != 0)
    {y += 1;}
printf("%d", y);
```

- When running the program, which of the following is correct? (Check all that apply!)
  - If the user enters 6, it will print 2.
  - If the user enters 6, it will print 3.
  - If the user enters 4, it will print 2.
  - If the user enters 4, it will print 1.
  - If the user enters 4, it will print 4.

EECS10: Computational Methods in ECE, Lecture 10



(c) 2007 R. Doemer

11

## Midterm 1 Review Quiz

- Top 5 most “difficult” questions:
  - Rank 1: Question 21 (79.8% wrong answers)

```
unsigned int x=0, y=0;
scanf("%d", &x);
while((x>=1) != 0)
    {y += 1;}
printf("%d", y);
```

- When running the program, which of the following is correct? (Check all that apply!)
  -  a) If the user enters 6, it will print 2.
  - b) If the user enters 6, it will print 3.
  -  c) If the user enters 4, it will print 2.
  - d) If the user enters 4, it will print 1.
  - e) If the user enters 4, it will print 4.

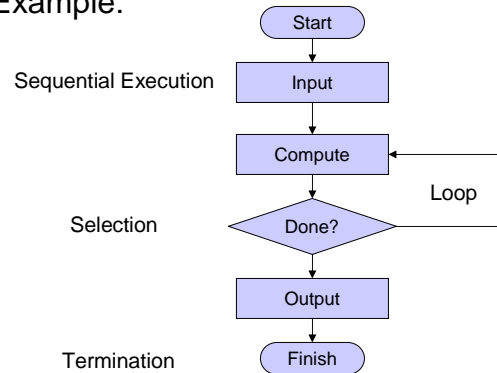
EECS10: Computational Methods in ECE, Lecture 10

(c) 2007 R. Doemer

12

## Structured Programming

- Control flow charts
  - Graphical representation of program control flow
  - Example:



EECS10: Computational Methods in ECE, Lecture 10

(c) 2007 R. Doemer

13

## Structured Programming

- Sequential execution in C
  - Statement blocks: *Compound statements*
  - Sequence of statements grouped by braces: { }
- Example:

```

{
  /* statement 1 */

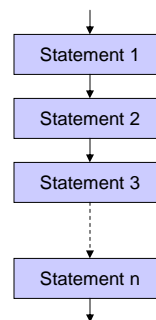
  /* statement 2 */

  /* statement 3 */

  /* ... */

  /* statement n */
}
  
```

Flow chart:



EECS10: Computational Methods in ECE, Lecture 10

(c) 2007 R. Doemer

14

## Structured Programming

- Sequential execution in C
  - Statement blocks: *Compound statements*
  - Sequence of statements grouped by braces: { }
- *Indentation* increases readability of the code
  - proper indentation is highly recommended!
- Example:

```

/* some statements... */
if (x < 0) {
    printf("%d is negative!", x);
    /* handle negative values of x... */
    if (x < 100) {
        printf("%d is too small!", x);
        /* handle the problem... */
    } /* fi */
} /* fi */
if (x > 0) {
    printf("%d is positive!", x);
    /* handle positive values of x... */
} /* fi */
/* more statements... */

```

EECS10: Computational Methods in ECE, Lecture 10

(c) 2007 R. Doemer

15

## Structured Programming

- Sequential execution in C
  - Statement blocks: *Compound statements*
  - Sequence of statements grouped by braces: { }
- *Indentation* increases readability of the code
  - proper indentation is highly recommended!
- Example:

```

/* some statements... */
if (x < 0) {
    printf("%d is negative!", x);
    /* handle negative values of x... */
    if (x < 100) {
        printf("%d is too small!", x);
        /* handle the problem... */
    } /* fi */
} /* fi */
if (x > 0) {
    printf("%d is positive!", x);
    /* handle positive values of x... */
} /* fi */
/* more statements... */

```

indentation level 0

indentation level 1 →

indentation level 2 → →

indentation level 1 →

indentation level 0

indentation level 1 →

indentation level 0

EECS10: Computational Methods in ECE, Lecture 10

(c) 2007 R. Doemer

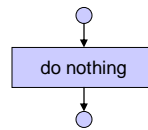
16



## Structured Programming

- Empty statement blocks
  - empty compound statement
  - does nothing (no operation, no-op)
  - Example: Flow chart:

```
{
  /* nothing */
}
```



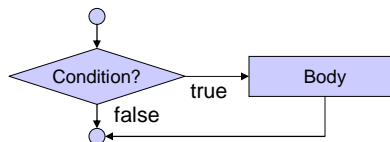
EECS10: Computational Methods in ECE, Lecture 10

(c) 2007 R. Doemer

17

## Structured Programming

- Selection: **if** statement
  - Flow chart:



- Example:

```
if (grade >= 60)
{ printf("You passed.");
} /* fi */
```

EECS10: Computational Methods in ECE, Lecture 10

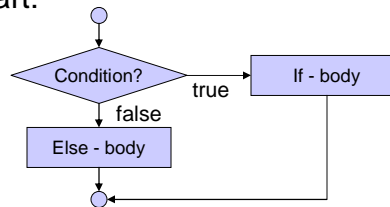
(c) 2007 R. Doemer

18

## Structured Programming

- Selection: **if-else** statement

– Flow chart:



– Example:

```

if (grade >= 60)
{ printf("You passed.");
} /* fi */
else
{ printf("You failed.");
} /* esle */
  
```

EECS10: Computational Methods in ECE, Lecture 10

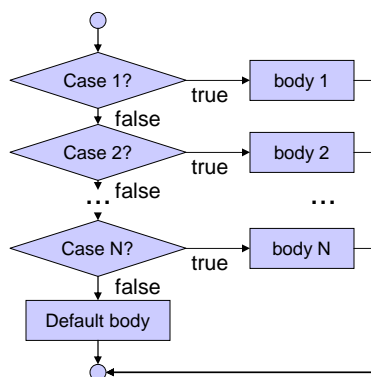
(c) 2007 R. Doemer

19

## Structured Programming

- Selection: **switch** statement

– Flow chart:



Example:

```

switch(LetterGrade)
{ case 'A':
  { printf("Excellent!");
    break; }
  case 'B':
  case 'C':
  case 'D':
  { printf("Passed.");
    break; }
  case 'F':
  { printf("Failed!");
    break; }
  default:
  { printf("Invalid grade!");
    break; }
} /* hctiws */
  
```

EECS10: Computational Methods in ECE, Lecture 10

(c) 2007 R. Doemer

20

## Structured Program Composition

- Initial flow chart
  - Start
  - Program body
  - Finish
- Statement sequences
  - Statement blocks can be concatenated
  - Sequential execution
- Nested control structures
  - control structures can be placed wherever statement blocks can be placed in the code

EECS10: Computational Methods in ECE, Lecture 10 (c) 2007 R. Doemer 21

## Structured Program Composition

- Example:
  - Initial flow chart

EECS10: Computational Methods in ECE, Lecture 10 (c) 2007 R. Doemer 22

## Structured Program Composition

- Example:
  - Sequential composition

EECS10: Computational Methods in ECE, Lecture 10
(c) 2007 R. Doemer
23

## Structured Program Composition

- Example:
  - insertion of another sequential statement

EECS10: Computational Methods in ECE, Lecture 10
(c) 2007 R. Doemer
24

## Structured Program Composition

- Example:
  - insertion of **if - else** statement

The flowchart shows a sequence of operations starting from a start node (oval), followed by a process node (rectangle), then a decision node (diamond). A dashed box highlights the decision node and two process nodes. One process node is connected to the 'Yes' branch of the decision node, and the other is connected to the 'No' branch. Both branches merge back into a single path that continues through another process node and ends at a stop node (oval).

EECS10: Computational Methods in ECE, Lecture 10 (c) 2007 R. Doemer 25

## Structured Program Composition

- Example:
  - insertion of sequential statement

The flowchart shows a sequence of operations starting from a start node (oval), followed by a process node (rectangle), then a decision node (diamond). A dashed box highlights the decision node and two process nodes. The 'Yes' branch of the decision node leads to the first process node, and the 'No' branch leads to the second process node. Both branches merge back into a single path that continues through another process node and ends at a stop node (oval).

EECS10: Computational Methods in ECE, Lecture 10 (c) 2007 R. Doemer 26

## Structured Program Composition

- Example:
  - insertion of **if - else** statement

The flowchart shows a sequence of operations starting from a start node (oval), followed by a process node (rectangle), a decision node (diamond), and another process node. A dashed box highlights a section where a decision node is inserted, leading to two parallel process nodes (if and else branches), which then merge back into a single path before the final process node and end node (oval).

EECS10: Computational Methods in ECE, Lecture 10 (c) 2007 R. Doemer 27

## Structured Program Composition

- Example:
  - insertion of sequential statement

The flowchart shows a sequence of operations starting from a start node (oval), followed by a process node, a decision node, and another process node. A dashed box highlights a section where a decision node is inserted, leading to two sequential process nodes (one after the other), which then merge back into a single path before the final process node and end node (oval).

EECS10: Computational Methods in ECE, Lecture 10 (c) 2007 R. Doemer 28

## Structured Program Composition

- Example:
  - insertion of sequential statement (twice)

The flowchart shows a main vertical sequence of nodes: a start oval, a rectangle, a diamond, a rectangle, a diamond, a rectangle, a diamond, a rectangle, a diamond, a rectangle, and an end oval. A loop is formed by a diamond node branching to a rectangle node, which then loops back to the diamond. A dashed box highlights a sub-section of the loop containing three sequential rectangle nodes. Arrows indicate the flow of control through these nodes.

EECS10: Computational Methods in ECE, Lecture 10 (c) 2007 R. Doemer 29

## Structured Program Composition

- Example:
  - insertion of **switch** statement
  - etc. ...

The flowchart shows a main vertical sequence of nodes: a start oval, a rectangle, a diamond, a rectangle, a diamond, a rectangle, a diamond, a rectangle, a diamond, a rectangle, and an end oval. A loop is formed by a diamond node branching to a rectangle node, which then loops back to the diamond. A dashed box highlights a sub-section of the loop containing three sequential diamond nodes, each branching to a rectangle node, which then loops back to the diamond. Arrows indicate the flow of control through these nodes.

EECS10: Computational Methods in ECE, Lecture 10 (c) 2007 R. Doemer 30

## Example Program

- Grade calculation: `Grade.c` (part 1/3)

```

/* Grade.c: convert score into letter grade    */
/* author: Rainer Doemer                      */
/* modifications:                             */
/* 10/17/04 RD  initial version              */

#include <stdio.h>

/* main function */
int main(void)
{
    /* variable definitions */
    int  score = 0;
    char grade;

    /* input section */
    while (score < 1 || score > 100)
    { printf("Please enter your score (1-100): ");
      scanf("%d", &score);
    } /* elihw */

    ...

```

EECS10: Computational Methods in ECE, Lecture 10

(c) 2007 R. Doemer

31

## Example Program

- Grade calculation: `Grade.c` (part 2/3)

```

...
/* computation section */
if (score >= 90)
    { grade = 'A'; }
else
    { if (score >= 80)
      { grade = 'B'; }
      else
        { if (score >= 70)
          { grade = 'C'; }
          else
            { if (score >= 60)
              { grade = 'D'; }
              else
                { grade = 'F'; }
            } /* esle */
          } /* esle */
        } /* esle */
    } /* esle */
...

```

EECS10: Computational Methods in ECE, Lecture 10

(c) 2007 R. Doemer

32



## Example Program

- Grade calculation: `Grade.c` (part 3/3)

```
...  
  
/* output section */  
printf("Your letter grade is %c.\n", grade);  
  
/* exit */  
return 0;  
} /* end of main */  
  
/* EOF */
```

## Example Program

- Example session: `Grade.c`

```
% vi Grade.c  
% gcc Grade.c -o Grade -Wall -ansi  
% Grade  
Please enter your score (1-100): 111  
Please enter your score (1-100): 99  
Your letter grade is A.  
% Grade  
Please enter your score (1-100): 85  
Your letter grade is B.  
% Grade  
Please enter your score (1-100): 71  
Your letter grade is C.  
% Grade  
Please enter your score (1-100): 69  
Your letter grade is D.  
% Grade  
Please enter your score (1-100): 55  
Your letter grade is F.  
%
```

## Example Program

- Grade calculation: `Grade2.c` (part 1/3)

```

/* Grade2.c: convert score into letter grade */
/* author: Rainer Doemer */
/* modifications: */
/* 10/18/04 RD use 'switch' statement */
/* 10/17/04 RD initial version */

#include <stdio.h>

/* main function */
int main(void)
{
    /* variable definitions */
    int score = 0;
    char grade;

    /* input section */
    while (score < 1 || score > 100)
    { printf("Please enter your score (1-100): ");
      scanf("%d", &score);
    } /* elihw */

```

EECS ...

## Example Program

- Grade calculation: `Grade2.c` (part 2/3)

```

.../* computation section */
switch (score / 10)
{ case 10:
  case 9:
    { grade = 'A';
      break; }
  case 8:
    { grade = 'B';
      break; }
  case 7:
    { grade = 'C';
      break; }
  case 6:
    { grade = 'D';
      break; }
  default:
    { grade = 'F';
      break; }
} /* hctiws */

```

EECS ...

## Example Program

- Grade calculation: `Grade2.c` (part 3/3)

```
...  
  
/* output section */  
printf("Your letter grade is %c.\n", grade);  
  
/* exit */  
return 0;  
} /* end of main */  
  
/* EOF */
```

## Example Program

- Example session: `Grade2.c`

```
% cp Grade.c Grade2.c  
% vi Grade2.c  
% gcc Grade2.c -o Grade2 -Wall -ansi  
% Grade2  
Please enter your score (1-100): 111  
Please enter your score (1-100): 99  
Your letter grade is A.  
% Grade2  
Please enter your score (1-100): 85  
Your letter grade is B.  
% Grade2  
Please enter your score (1-100): 71  
Your letter grade is C.  
% Grade2  
Please enter your score (1-100): 69  
Your letter grade is D.  
% Grade2  
Please enter your score (1-100): 55  
Your letter grade is F.  
%
```