

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 “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: Question 5

- In our Unix environment, which of the following statements is true about the following command line?
(Check all that apply! 2 pts.)

```
% gcc Test.c -Wall -o Test -ansi
```

- a) The compiler will read the file `Test.c`
- b) The compiler will write the file `Test`
- c) The compiler will ignore all warnings
- d) The compiler will follow ANSI-C rules
- e) The compiler will test the file `Test.c`

Midterm 1 Review Quiz: Question 5

- In our Unix environment, which of the following statements is true about the following command line?
(Check all that apply! 2 pts.)

```
% gcc Test.c -Wall -o Test -ansi
```

- a) The compiler will read the file `Test.c`
- b) The compiler will write the file `Test`
- c) The compiler will ignore all warnings
- d) The compiler will follow ANSI-C rules
- e) The compiler will test the file `Test.c`

Midterm 1 Review Quiz: Question 6

- In C, which constructs represent valid operators?
(Check all that apply! 2 pts.)

a) ==
b) <=
c) +=
d) +=
e) <<=

EECS10: Computational Methods in ECE, Lecture 10

(c) 2005 R. Doemer

5

Midterm 1 Review Quiz: Question 6

- In C, which constructs represent valid operators?
(Check all that apply! 2 pts.)

a) ==
 b) <=
c) +=
 d) +=
 e) <<=

EECS10: Computational Methods in ECE, Lecture 10

(c) 2005 R. Doemer

6

Midterm 1 Review Quiz: Question 8

- Which of the following names are valid keywords in C?
(Check all that apply! 2 pts.)

a) `key`
b) `long`
c) `return`
d) `cont`
e) `scanf`



EECS10: Computational Methods in ECE, Lecture 10

(c) 2005 R. Doemer

7

Midterm 1 Review Quiz: Question 8

- Which of the following names are valid keywords in C?
(Check all that apply! 2 pts.)

a) `key`
 b) `long`
 c) `return`
d) `cont`
e) `scanf`

EECS10: Computational Methods in ECE, Lecture 10

(c) 2005 R. Doemer

8

Midterm 1 Review Quiz: Question 9




- Which of the following constructs denotes a valid type name in C?
(Check all that apply! 2 pts.)
 - a) `long char`
 - b) `signed char`
 - c) `unsigned long long int`
 - d) `long double`
 - e) `unsigned float`

EECS10: Computational Methods in ECE, Lecture 10

(c) 2005 R. Doemer

9

Midterm 1 Review Quiz: Question 9

- Which of the following constructs denotes a valid type name in C?
(Check all that apply! 2 pts.)
 - a) `long char`
 -  b) `signed char`
 -  c) `unsigned long long int`
 -  d) `long double`
 - e) `unsigned float`

EECS10: Computational Methods in ECE, Lecture 10

(c) 2005 R. Doemer

10

Midterm 1 Review Quiz: Question 16

- 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! 2 pts.)

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

EECS10: Computational Methods in ECE, Lecture 10

(c) 2005 R. Doemer

11

Midterm 1 Review Quiz: Question 16

- 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! 2 pts.)

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

EECS10: Computational Methods in ECE, Lecture 10

(c) 2005 R. Doemer

12

Structured Programming

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

```

graph TD
    Start([Start]) --> Input[Input]
    Input --> Compute[Compute]
    Compute --> Done{Done?}
    Done -- Loop --> Compute
    Done --> Output[Output]
    Output --> Finish([Finish])
            
```

Sequential Execution
Selection
Termination

EECS10: Computational Methods in ECE, Lecture 10
(c) 2005 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:

```

graph TD
    S1[Statement 1] --> S2[Statement 2]
    S2 --> S3[Statement 3]
    S3 -.-> Sn[Statement n]
            
```

EECS10: Computational Methods in ECE, Lecture 10
(c) 2005 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) 2005 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... */
indentation level 0 if (x < 0) {
indentation level 1   printf("%d is negative!", x);
indentation level 1   →| /* handle negative values of x... */
indentation level 2   →| if (x < 100) {
indentation level 2   →| →| printf("%d is too small!", x);
indentation level 1   →| →| /* handle the problem... */
indentation level 1   →| →| } /* fi */
indentation level 0   →| } /* fi */
indentation level 0   if (x > 0) {
indentation level 1   →| printf("%d is positive!", x);
indentation level 1   →| /* handle positive values of x... */
indentation level 0   →| } /* fi */
indentation level 0   /* more statements... */

```

EECS10: Computational Methods in ECE, Lecture 10

(c) 2005 R. Doemer

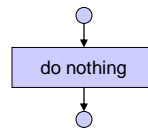
16

Structured Programming

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

Flow chart:

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



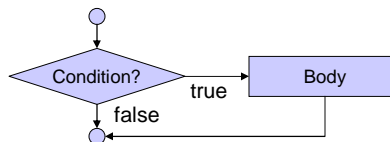
EECS10: Computational Methods in ECE, Lecture 10

(c) 2005 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

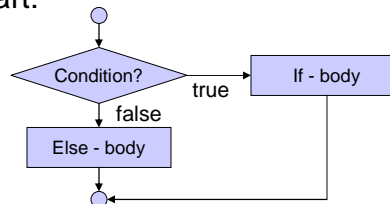
(c) 2005 R. Doemer

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

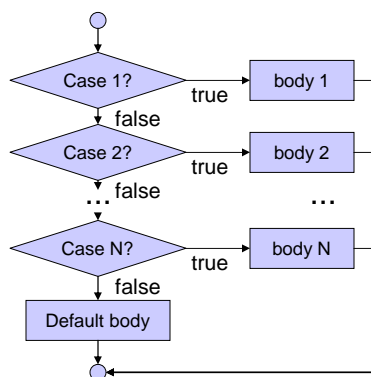
(c) 2005 R. Doemer

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) 2005 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) 2005 R. Doemer 21

Structured Program Composition

- Example:
 - Initial flow chart

EECS10: Computational Methods in ECE, Lecture 10 (c) 2005 R. Doemer 22

Structured Program Composition

- Example:
 - Sequential composition

```

    graph TD
      Start([Start]) --> P1[ ]
      P1 --> P2[ ]
      P2 --> End([End])
      subgraph Box [ ]
        P1
        P2
      end
    
```

EECS10: Computational Methods in ECE, Lecture 10 (c) 2005 R. Doemer 23

Structured Program Composition

- Example:
 - insertion of another sequential statement

```

    graph TD
      Start([Start]) --> P1[ ]
      P1 --> P2[ ]
      P2 --> P3[ ]
      P3 --> End([End])
      subgraph Box [ ]
        P1
        P2
      end
    
```

EECS10: Computational Methods in ECE, Lecture 10 (c) 2005 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 encloses the decision node and two process nodes. The flow goes down from the decision node to the first process node, then right to the second process node, and finally down back to the main flow. After the dashed box, there is another process node and finally an end node (oval).

EECS10: Computational Methods in ECE, Lecture 10 (c) 2005 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 encloses two process nodes. The flow goes down from the decision node to the first process node, then right to the second process node, and finally down back to the main flow. After the dashed box, there is another process node and finally an end node (oval).

EECS10: Computational Methods in ECE, Lecture 10 (c) 2005 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 the insertion of an if-else structure: a decision node leads to a process node (the 'if' branch), which then leads to a process node (the 'else' branch), which finally loops back to the decision node. After the if-else structure, the flow continues to a final process node and ends at a stop node (oval).

EECS10: Computational Methods in ECE, Lecture 10 (c) 2005 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 the insertion of a sequential statement: a decision node leads to a process node, which then leads to two more process nodes in sequence, before looping back to the decision node. After the sequential statement, the flow continues to a final process node and ends at a stop node (oval).

EECS10: Computational Methods in ECE, Lecture 10 (c) 2005 R. Doemer 28

Structured Program Composition

- Example:
 - insertion of sequential statement (twice)

The flowchart shows a sequence of operations starting from a start node (oval). It proceeds through a rectangular process block, a diamond-shaped decision block, and another rectangular process block. From the decision block, the flow branches to a third rectangular process block, then to a second diamond-shaped decision block. This second decision block branches into two paths: one leading to a third rectangular process block, and another leading to a dashed box containing three sequential rectangular process blocks. Both paths from the second decision block merge back into a single path that leads to a fourth rectangular process block, and finally to an end node (oval).

EECS10: Computational Methods in ECE, Lecture 10 (c) 2005 R. Doemer 29

Structured Program Composition

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

The flowchart shows a sequence of operations starting from a start node (oval). It proceeds through a rectangular process block, a diamond-shaped decision block, and another rectangular process block. From the decision block, the flow branches to a third rectangular process block, then to a second diamond-shaped decision block. This second decision block branches into two paths: one leading to a third rectangular process block, and another leading to a fourth rectangular process block. Both paths from the second decision block merge back into a single path that leads to a dashed box containing a switch statement structure. This structure consists of three diamond-shaped decision blocks in a vertical sequence, each followed by a rectangular process block. All three process blocks have arrows pointing to a single merge point below the dashed box. The flow then continues to a final rectangular process block and ends at an oval node.

EECS10: Computational Methods in ECE, Lecture 10 (c) 2005 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) 2005 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 */

...

```

EECS10: Computational Methods in ECE, Lecture 10

(c) 2005 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.
%
```