

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

- 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`

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) 2009 R. Doemer 3

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) 2009 R. Doemer 4

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) 2009 R. Doemer

5

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

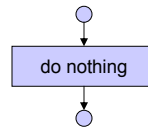
(c) 2009 R. Doemer

6

Structured Programming

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

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



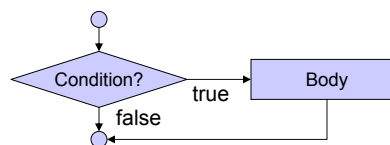
EECS10: Computational Methods in ECE, Lecture 10

(c) 2009 R. Doemer

7

Structured Programming

- Selection: `if` statement
 - Flow chart:



- Example:

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

EECS10: Computational Methods in ECE, Lecture 10

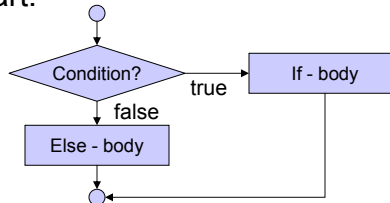
(c) 2009 R. Doemer

8

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

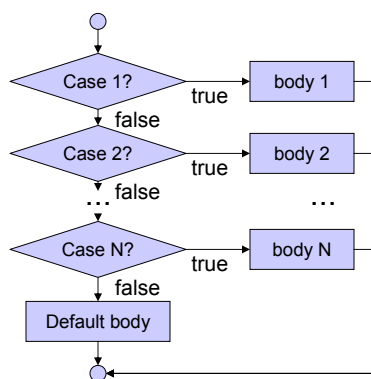
(c) 2009 R. Doemer

9

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) 2009 R. Doemer

10

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) 2009 R. Doemer 11

Structured Program Composition

- Example:
 - Initial flow chart

EECS10: Computational Methods in ECE, Lecture 10 (c) 2009 R. Doemer 12

Structured Program Composition

- Example:
 - Sequential composition

```

graph TD
  Start([Start]) --> Node1[ ]
  Node1 --> Node2[ ]
  Node2 --> End([End])
  subgraph Box [ ]
    Node1
    Node2
  end
  
```

EECS10: Computational Methods in ECE, Lecture 10 (c) 2009 R. Doemer 13

Structured Program Composition

- Example:
 - insertion of another sequential statement

```

graph TD
  Start([Start]) --> Node1[ ]
  Node1 --> Node2[ ]
  Node2 --> Node3[ ]
  Node3 --> End([End])
  subgraph Box [ ]
    Node1
    Node2
  end
  
```

EECS10: Computational Methods in ECE, Lecture 10 (c) 2009 R. Doemer 14

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 from the decision node to the first process node, then to the second process node, and finally to a final process node and an end node (oval). An arrow from the right side of the decision node loops back to the top of the second process node, representing an if-else structure.

EECS10: Computational Methods in ECE, Lecture 10 (c) 2009 R. Doemer 15

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 the decision node and two sequential process nodes. The flow goes from the decision node to the first process node, then to the second process node, and finally to a final process node and an end node (oval). An arrow from the right side of the decision node loops back to the top of the first process node, representing a sequential structure.

EECS10: Computational Methods in ECE, Lecture 10 (c) 2009 R. Doemer 16

Structured Program Composition

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

The flowchart shows a sequence of operations starting from a start node (oval). It proceeds through a process box, a decision diamond, and another process box. A dashed box highlights the insertion of an if-else structure: a decision diamond branches to a process box (the 'if' branch), which then leads to a process box (the 'else' branch), which finally loops back to the main flow. The flowchart concludes with a final process box and an end node (oval).

EECS10: Computational Methods in ECE, Lecture 10 (c) 2009 R. Doemer 17

Structured Program Composition

- Example:
 - insertion of sequential statement

The flowchart shows a sequence of operations starting from a start node (oval). It proceeds through a process box, a decision diamond, and another process box. A dashed box highlights the insertion of a sequential statement: a decision diamond branches to a process box, which then leads to a process box, which finally loops back to the main flow. The flowchart concludes with a final process box and an end node (oval).

EECS10: Computational Methods in ECE, Lecture 10 (c) 2009 R. Doemer 18

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 process node (rectangle), a decision node (diamond), another process node, and a third decision node. From the second decision node, the flow branches to a process node, then to a third decision node. From this third decision node, the flow branches to a sub-structure enclosed in a dashed box, which contains three sequential process nodes. After exiting the dashed box, the flow goes to a fourth decision node, then to a process node, and finally to an end node (oval).

EECS10: Computational Methods in ECE, Lecture 10 (c) 2009 R. Doemer 19

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 process node, a decision node, another process node, and a third decision node. From the third decision node, the flow branches to a process node, then to a fourth decision node. From this fourth decision node, the flow branches to two parallel paths, each consisting of a process node followed by a decision node. Both paths then merge back into a single path that leads to a sub-structure enclosed in a dashed box. This sub-structure represents a switch statement, consisting of three sequential decision nodes, each followed by a process node. All three paths from the decision nodes merge into a single path that leads to an end node (oval).

EECS10: Computational Methods in ECE, Lecture 10 (c) 2009 R. Doemer 20

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) 2009 R. Doemer

21

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) 2009 R. Doemer

22

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.
%
```