

## Lecture 4.2: Overview

- Programming Principles
  - Algorithm and control flow
- Structured Programming
  - Control flow chart
  - Sequential execution
  - Conditional execution
    - `if` statement
    - `if-else` statement
    - `switch` statement
  - Structured Program Composition
  - Examples `Grade.c`, `Grade2.c`

EECS10: Computational Methods in ECE, Lecture 4

(c) 2013 R. Doemer

1

## Programming Principles

- Thorough *understanding* of the problem
- *Problem definition*
  - Input data
  - Output data
- *Algorithm*: Procedure to solve the problem
  - Detailed set of *actions* to perform
  - Specification of *order* in which to perform the actions
  - Termination after a *finite* number of steps
- *Pseudo code*: Planning a program
  - Informal (English) description of steps in an algorithm
  - Example: Cake baking recipe
- *Control flow*
  - Detailed execution order of steps in the program
- *Program*: Instructions for the computer
  - Formal description in programming language
    - Statements (steps, actions)
    - Control structures (flow of control)

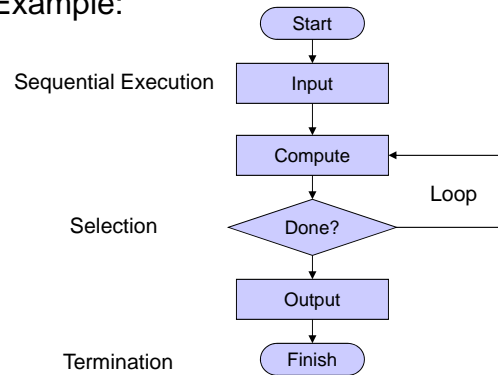
EECS10: Computational Methods in ECE, Lecture 4

(c) 2013 R. Doemer

2

## Structured Programming

- Control Flow Chart
  - Graphical representation of program control flow
  - Example:



EECS10: Computational Methods in ECE, Lecture 4

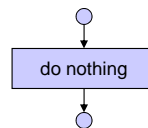
(c) 2013 R. Doemer

3

## Structured Programming

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

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



EECS10: Computational Methods in ECE, Lecture 4

(c) 2013 R. Doemer

4

## Structured Programming

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

```
{
  /* statement 1 */

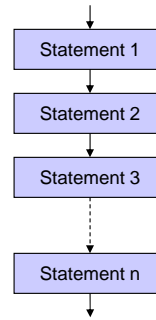
  /* statement 2 */

  /* statement 3 */

  /* ... */

  /* statement n */
}
```

Flow chart:



EECS10: Computational Methods in ECE, Lecture 4

(c) 2013 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... */
```

EECS10: Computational Methods in ECE, Lecture 4

(c) 2013 R. Doemer

6

## 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) {
    printf("%d is negative!", x);
    indentation level 1 → /* handle negative values of x... */
    if (x < -100) {
        printf("%d is too small!", x);
        indentation level 2 → → /* handle the problem... */
    } /* fi */
    indentation level 1 → } /* fi */
indentation level 0 if (x > 0) {
    printf("%d is positive!", x);
    indentation level 1 → /* handle positive values of x... */
    } /* fi */
indentation level 0 /* more statements... */
  
```

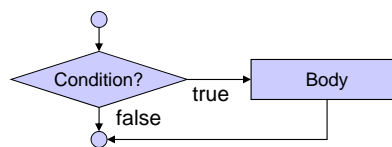
EECS10: Computational Methods in ECE, Lecture 4

(c) 2013 R. Doemer

7

## Structured Programming

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



- Example:

```

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

EECS10: Computational Methods in ECE, Lecture 4

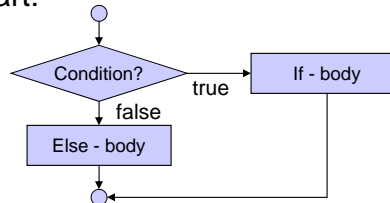
(c) 2013 R. Doemer

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 4

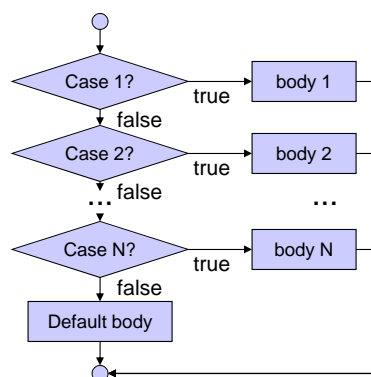
(c) 2013 R. Doemer

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 4

(c) 2013 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 4 (c) 2013 R. Doemer 11

## Structured Program Composition

- Example:
  - Initial flow chart

EECS10: Computational Methods in ECE, Lecture 4 (c) 2013 R. Doemer 12

## Structured Program Composition

- Example:
  - Sequential composition

EECS10: Computational Methods in ECE, Lecture 4
(c) 2013 R. Doemer
13

## Structured Program Composition

- Example:
  - insertion of another sequential statement

EECS10: Computational Methods in ECE, Lecture 4
(c) 2013 R. Doemer
14

## Structured Program Composition

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

The flowchart shows a sequence of operations starting with an oval, followed by a rectangle, then a diamond (decision point). A dashed box encloses the diamond, a rectangle to its right, and a rectangle below the diamond. An arrow points from the diamond to the right-hand rectangle, and another arrow points from that rectangle back to the diamond, forming a loop. Below the diamond is another rectangle, followed by a final oval.

EECS10: Computational Methods in ECE, Lecture 4 (c) 2013 R. Doemer 15

## Structured Program Composition

- Example:
  - insertion of **sequential** statement

The flowchart shows a sequence of operations starting with an oval, followed by a rectangle, then a diamond (decision point). A dashed box encloses two rectangles stacked vertically to the right of the diamond. An arrow points from the diamond to the top rectangle, and another arrow points from the bottom rectangle back to the diamond, forming a loop. Below the diamond is another rectangle, followed by a final oval.

EECS10: Computational Methods in ECE, Lecture 4 (c) 2013 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 node (rectangle), a decision node (diamond), and another process node. A dashed box highlights the insertion of an if-else structure: a decision node (diamond) branches to a process node (rectangle) on the right, which then loops back to the decision node. The other branch of the decision node leads to a process node (rectangle) below it. After the if-else structure, the flow continues through a final process node and ends at a stop node (oval).

EECS10: Computational Methods in ECE, Lecture 4 (c) 2013 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 node (rectangle), a decision node (diamond), and another process node. A dashed box highlights the insertion of a sequential statement: a decision node (diamond) branches to a process node (rectangle) on the right, which then leads to a second process node (rectangle) below it. The other branch of the decision node leads to a process node (rectangle) below it. After the sequential statement, the flow continues through a final process node and ends at a stop node (oval).

EECS10: Computational Methods in ECE, Lecture 4 (c) 2013 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 process node. A decision node branches off to the right, leading to a sequence of three process nodes. The second and third process nodes in this branch are enclosed in a dashed box, indicating they are being inserted. The flow then returns to the main sequence and ends at a final node (oval).

EECS10: Computational Methods in ECE, Lecture 4 (c) 2013 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 process node. A decision node branches off to the right, leading to two parallel paths, each containing two process nodes. The flow then returns to the main sequence. A dashed box at the bottom indicates the insertion of a switch statement, which consists of three decision nodes, each leading to a process node, all of which then merge back into the main flow. The flowchart ends at a final node (oval).

EECS10: Computational Methods in ECE, Lecture 4 (c) 2013 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 4

(c) 2013 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 4

(c) 2013 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.
%
```