

# EECS 10: Computational Methods in Electrical and Computer Engineering

## Lecture 12

Rainer Dömer

doemer@uci.edu

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

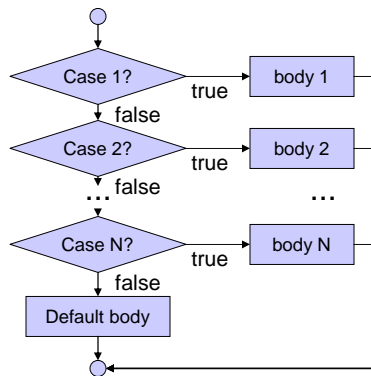
## Lecture 12: Overview

- Structured Programming
  - Structured jump statements
    - `break` statement in `switch` statement
    - `break` and `continue` in `while` loop
    - `break` and `continue` in `do-while` loop
    - `break` and `continue` in `for` loop
- Arbitrary jump statements
  - `goto` statement
- Debugging
  - Source-level debugger `gdb`
  - Example `Interest2.c`

## 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 12

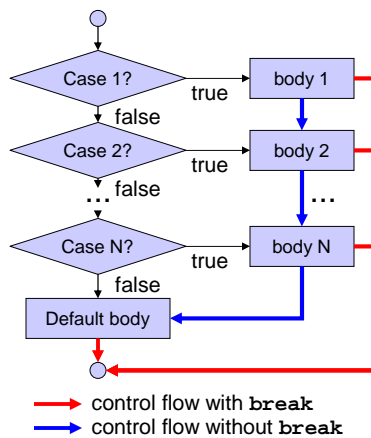
(c) 2006 R. Doemer

3

## Structured Programming

- Selection: **break** in **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 12

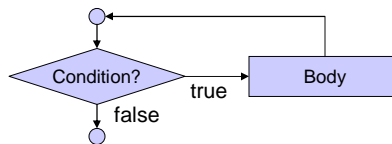
(c) 2006 R. Doemer

4

## Structured Programming

- Repetition: **while** loop

- Flow chart:



- Example:

```
int product = 2;
while (product < 1000)
{ product *= 2;
} /* elihw */
```

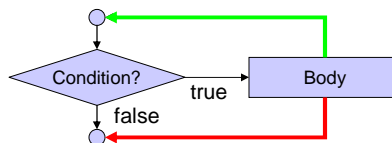
- Note:

- The condition is evaluated at the *beginning* of each loop!

## Structured Programming

- Repetition: **break/continue** in **while** loop

- Flow chart:



- Control flow:

- control flow with **break**
- control flow with **continue**

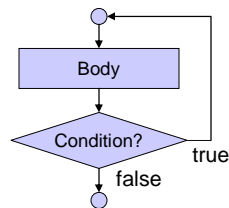
- Note:

- The condition is evaluated at the *beginning* of each loop!

## Structured Programming

- Repetition: **do-while** loop

– Flow chart:



– Example:

```
int product = 2;
do { product *= 2;
    } while (product < 1000);
```

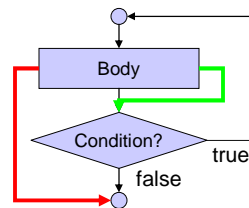
– Note:

- The condition is evaluated at the *end* of each loop!

## Structured Programming

- Repetition: **break/continue** in **do-while** loop

– Flow chart:



– Control flow:

- control flow with **break**
- control flow with **continue**

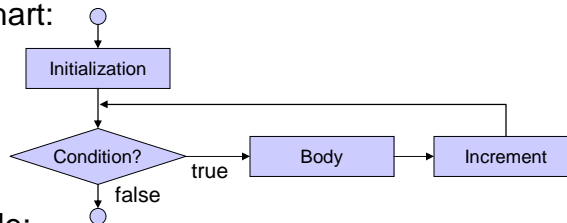
– Note:

- The condition is evaluated at the *end* of each loop!

## Structured Programming

- Repetition: **for** loop

– Flow chart:



– Example:

```

for(i = 0; i < 10; i++)
{ printf("i = %d\n", i);
} /* rof */
  
```

– Syntax:

- `for(initialization; condition; increment)`  
`body`

EECS10: Computational Methods in ECE, Lecture 12

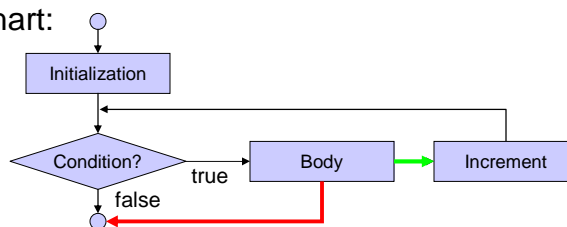
(c) 2006 R. Doemer

9

## Structured Programming

- Repetition: **break/continue** in **for** loop

– Flow chart:



– Control flow:

→ control flow with **break**

→ control flow with **continue**

– Syntax:

- `for(initialization; condition; increment)`  
`body`

EECS10: Computational Methods in ECE, Lecture 12

(c) 2006 R. Doemer

10

## Arbitrary Jump Statements

- Arbitrary jumps: `goto` statement
  - `goto` statement jumps to the specified *labeled* statement (within the same function)

– Example:

```
begin:  count = 0;
        goto next;
repeat: if (count > 100)
        { goto end; }
next:   count++;
        if (count == 77)
        { goto next; }
        goto repeat;
end:    printf("%d", count);
```

– Warning:

- `goto` statement allows *un-structured* programming!
- `goto` statement should be avoided whenever possible!

## Debugging

- Source-level Debugger `gdb`
  - Debugging features
    - run the program under debugger control
    - follow the control flow of the program during execution
    - set breakpoints to stop execution at specific points
    - inspect (and adjust) the values of variables
    - find the point in the program where the “crash” happens
  - Preparation:
 

compile your program with debugging support on

    - Option `-g` tells compiler to add debugging information (symbol tables) to the generated executable file
    - `gcc -g Program.c -o Program`
    - `gdb Program`

## Debugging

- Source-level Debugger `gdb`
  - Basic `gdb` commands
    - `run`
      - starts the execution of the program in the debugger
    - `break function_name`
      - inserts a breakpoint at *function\_name*
      - program execution will stop at the breakpoint
    - `list line_numbers`
      - lists the current or specified *line\_numbers*
    - `print variable_name`
      - prints the current value of the variable *variable\_name*
    - `next`
      - executes the next statement (one statement at a time)
    - `quit`
      - exits the debugger (and terminates the program)
    - `help`
      - provides helpful details on debugger commands

EECS10: Computational Methods in ECE, Lecture 12

(c) 2006 R. Doemer

13

## Example Program

- Compound interest: `Interest2.c` (part 1/2)

```

/* Interest2.c: compound interest on savings account */
/* author: Rainer Doemer */
/* modifications: */
/* 10/23/05 RD version to demonstrate debugging */
/* 10/19/04 RD initial version */

#include <stdio.h>

/* main function */
int main(void)
{
    /* variable definitions */
    double amount, balance, rate, interest;
    int year;

    /* input section */
    printf("Please enter the initial amount in $:\n");
    scanf("%lf", &amount);
    printf("Please enter the interest rate in %:\n");
    scanf("%lf", &rate);

```

EECS ...

## Example Program

- Compound interest: `Interest2.c` (part 2/2)

```

...

/* computation and output section */
for(year = 1; year <= 10; year++)
{
    interest = amount * (rate/100.0);
    balance = amount + interest;
    printf("Interest for year%3d is $%8.2f.\n", year,
           interest);
    printf("The new balance is      $%8.2f.\n", balance);
    amount = balance;
} /* rof */

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

/* EOF */

```

EECS10: Computational Methods in ECE, Lecture 12

(c) 2006 R. Doemer

15

## Example Program

- Example session: `Interest2.c` (part 1/6)

```

% vi Interest2.c
% gcc Interest2.c -o Interest2 -g -Wall -ansi
% Interest2
Please enter the initial amount in $:
1000
Please enter the interest rate in %:
1.5
Interest for year 1 is $ 15.00.
The new balance is $ 1015.00.
Interest for year 2 is $ 15.22.
The new balance is $ 1030.22.
...
Interest for year 10 is $ 17.15.
The new balance is $ 1160.54.
% gdb Interest2
% gdb 6.3
Copyright 2004 Free Software Foundation, Inc.
GDB is free software, ...
There is absolutely no warranty for GDB.
This GDB was configured as "sparc-sun-solaris2.7"...
(gdb)
...

```

EECS10: Computational Methods in ECE, Lecture 12

(c) 2006 R. Doemer

16



## Example Program

- Example session: `Interest2.c` (part 2/6)

```

...
(gdb) break main
Breakpoint 1 at 0x106ac: file Interest2.c, line 20.
(gdb) run
Starting program: /users/faculty/doemer/eecs10/Interest/Interest2
Breakpoint 1, main () at Interest2.c:20
20     printf("Please enter the initial amount in $:\n");
(gdb) next
Please enter the initial amount in $:
21     scanf("%lf", &amount);
(gdb) next
1000
22     printf("Please enter the interest rate in %:\n");
(gdb) next
Please enter the interest rate in %:
23     scanf("%lf", &rate);
(gdb) next
1.5
26     for(year = 1; year <= 10; year++)
(gdb) next
...

```

EECS10: Computational Methods in ECE, Lecture 12

(c) 2006 R. Doemer

17

## Example Program

- Example session: `Interest2.c` (part 3/6)

```

...
27     { interest = amount * (rate/100.0);
(gdb) next
28     balance = amount + interest;
(gdb) print interest
$1 = 15
(gdb) print amount
$2 = 1000
(gdb) print balance
$3 = -7.3987334479772013e+304
(gdb) next
29     printf("Interest for year%3d is $%8.2f.\n", year, interest);
(gdb) print balance
$4 = 1015
(gdb) next
Interest for year 1 is $ 15.00.
30     printf("The new balance is      $%8.2f.\n", balance);
(gdb) next
The new balance is      $ 1015.00.
31     amount = balance;
(gdb) next
...

```

EECS10: Computational Methods in ECE, Lecture 12

(c) 2006 R. Doemer

18

## Example Program

- Example session: `Interest2.c` (part 4/6)

```

...
26 for(year = 1; year <= 10; year++)
(gdb) next
27 { interest = amount * (rate/100.0);
(gdb) print year
$5 = 2
(gdb) list
22 printf("Please enter the interest rate in %:\n");
23 scanf("%lf", &rate);
24
25 /* computation and output section */
26 for(year = 1; year <= 10; year++)
27 { interest = amount * (rate/100.0);
28   balance = amount + interest;
29   printf("Interest for year%3d is $%8.2f.\n", year, interest);
30   printf("The new balance is      $%8.2f.\n", balance);
31   amount = balance;
(gdb) list 35
30   printf("The new balance is      $%8.2f.\n", balance);
31   amount = balance;
32 } /* eof */
...

```

EECS10: Computational Methods in ECE, Lecture 12

(c) 2006 R. Doemer

19

## Example Program

- Example session: `Interest2.c` (part 5/6)

```

...
33
34 /* exit */
35 return 0;
36 } /* end of main */
37
38 /* EOF */
(gdb) break 35
Breakpoint 2 at 0x1079c: file Interest2.c, line 35.
(gdb) cont
Continuing.
Interest for year  2 is $   15.22.
The new balance is      $ 1030.22.
Interest for year  3 is $   15.45.
The new balance is      $ 1045.68.
...
Interest for year 10 is $   17.15.
The new balance is      $ 1160.54.

Breakpoint 2, main () at Interest2.c:35
35 return 0;
...

```

EECS10: Computational Methods in ECE, Lecture 12

(c) 2006 R. Doemer

20

## Example Program

- Example session: `Interest2.c` (part 6/6)

```
...  
(gdb) print balance  
$6 = 1160.5408250251503  
(gdb) cont  
Continuing.  
  
Program exited normally.  
(gdb) quit  
%
```