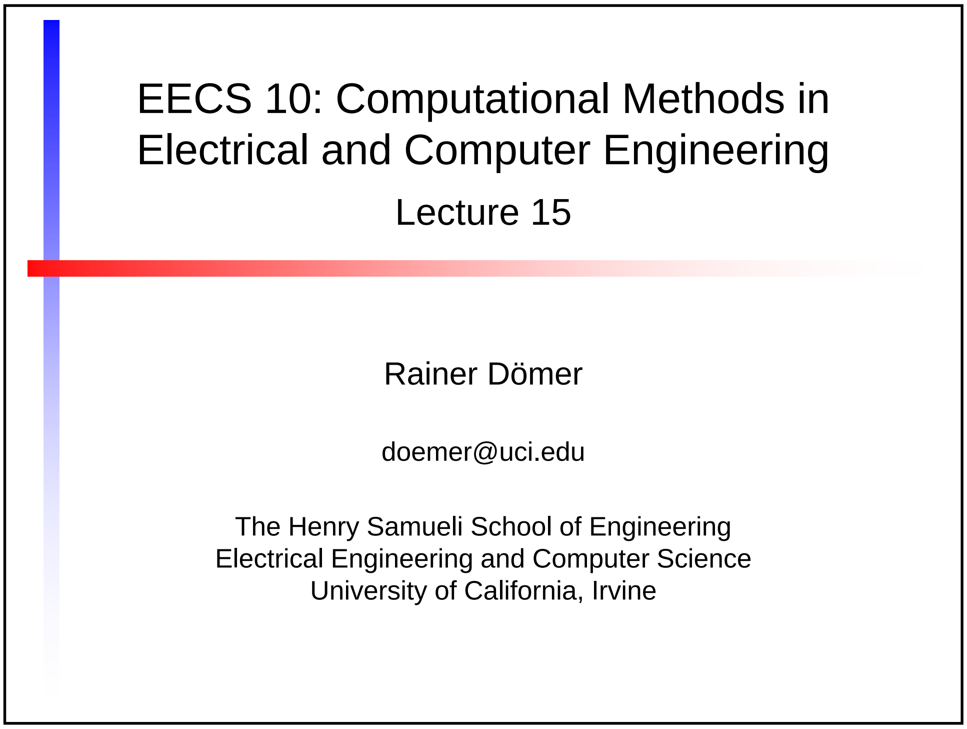


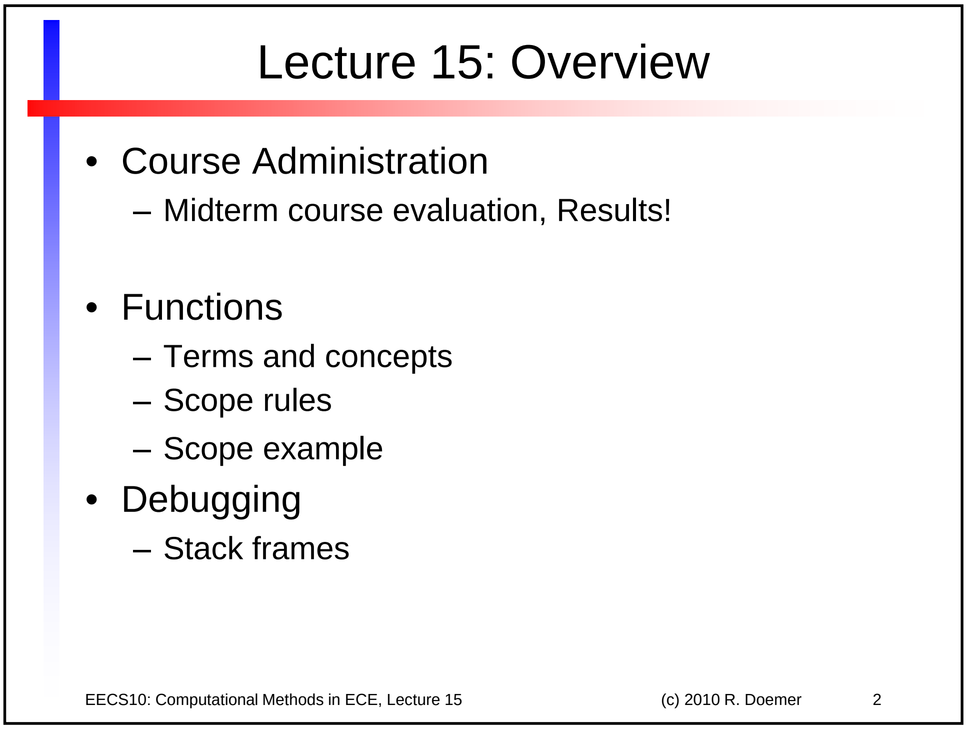
EECS 10: Computational Methods in Electrical and Computer Engineering

Lecture 15

Rainer Dömer

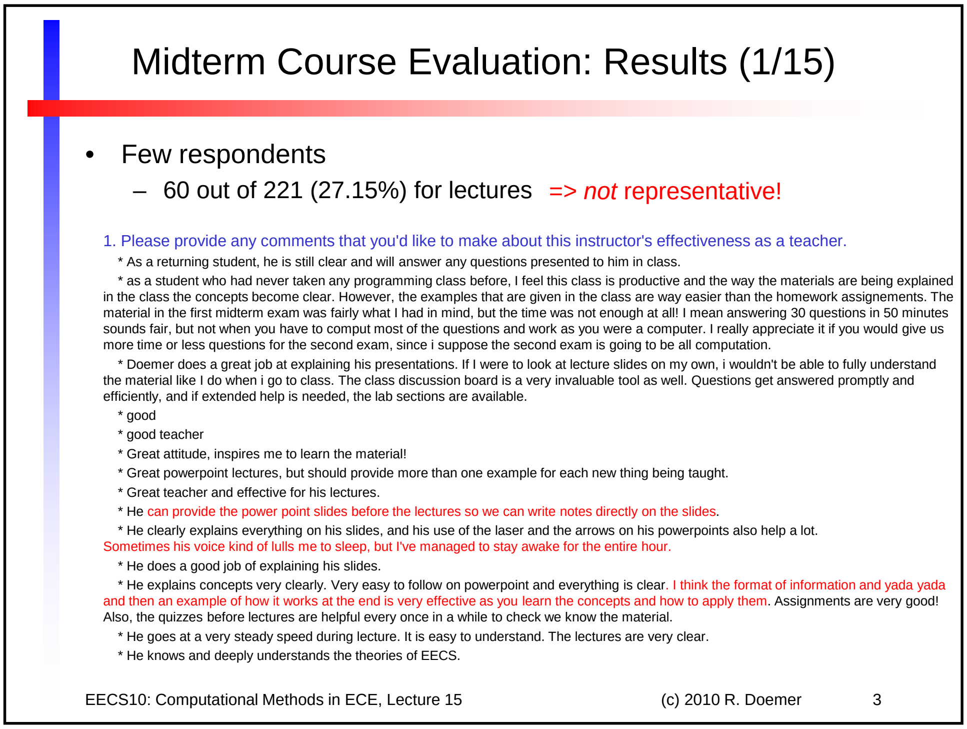
doemer@uci.edu

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



Lecture 15: Overview

- Course Administration
  - Midterm course evaluation, Results!
- Functions
  - Terms and concepts
  - Scope rules
  - Scope example
- Debugging
  - Stack frames



Midterm Course Evaluation: Results (1/15)

- Few respondents
  - 60 out of 221 (27.15%) for lectures => *not* representative!

1. Please provide any comments that you'd like to make about this instructor's effectiveness as a teacher.

* As a returning student, he is still clear and will answer any questions presented to him in class.

* as a student who had never taken any programming class before, I feel this class is productive and the way the materials are being explained in the class the concepts become clear. However, the examples that are given in the class are way easier than the homework assignements. The material in the first midterm exam was fairly what I had in mind, but the time was not enough at all! I mean answering 30 questions in 50 minutes sounds fair, but not when you have to comput most of the questions and work as you were a computer. I really appreciate it if you would give us more time or less questions for the second exam, since i suppose the second exam is going to be all computation.

* Doemer does a great job at explaining his presentations. If I were to look at lecture slides on my own, i wouldn't be able to fully understand the material like I do when i go to class. The class discussion board is a very invaluable tool as well. Questions get answered promptly and efficiently, and if extended help is needed, the lab sections are available.

* good

* good teacher

* Great attitude, inspires me to learn the material!

* Great powerpoint lectures, but should provide more than one example for each new thing being taught.

* Great teacher and effective for his lectures.

* He can provide the power point slides before the lectures so we can write notes directly on the slides.

* He clearly explains everything on his slides, and his use of the laser and the arrows on his powerpoints also help a lot. Sometimes his voice kind of lulls me to sleep, but I've managed to stay awake for the entire hour.

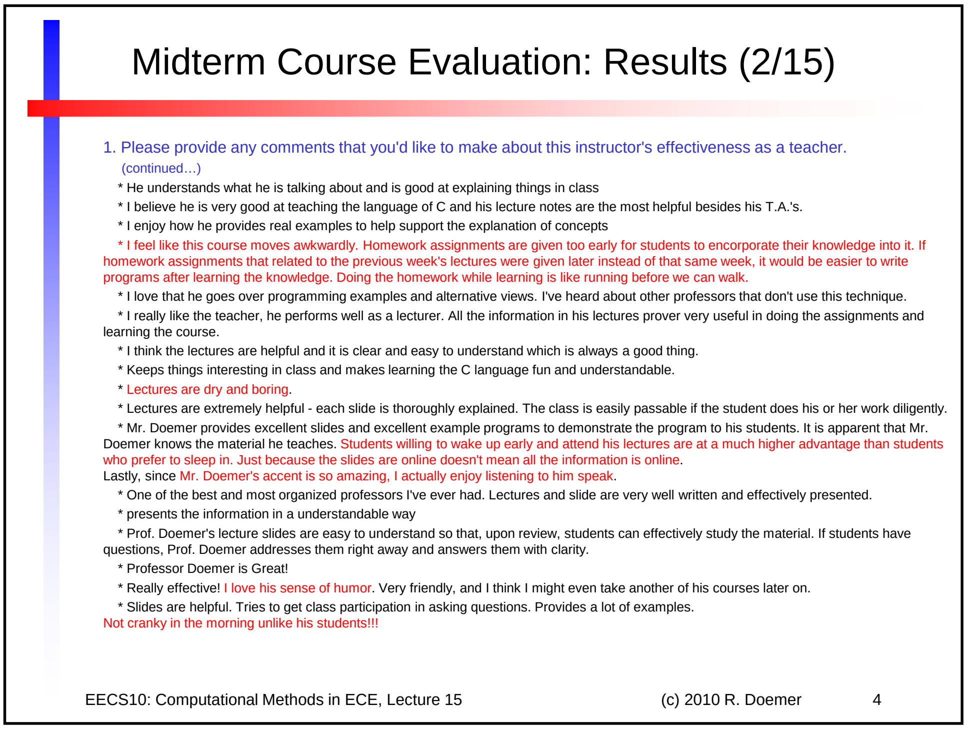
* He does a good job of explaining his slides.

* He explains concepts very clearly. Very easy to follow on powerpoint and everything is clear. I think the format of information and yada yada and then an example of how it works at the end is very effective as you learn the concepts and how to apply them. Assignments are very good! Also, the quizzes before lectures are helpful every once in a while to check we know the material.

* He goes at a very steady speed during lecture. It is easy to understand. The lectures are very clear.

* He knows and deeply understands the theories of EECS.

EECS10: Computational Methods in ECE, Lecture 15 (c) 2010 R. Doemer 3



Midterm Course Evaluation: Results (2/15)

1. Please provide any comments that you'd like to make about this instructor's effectiveness as a teacher.

(continued…)

* He understands what he is talking about and is good at explaining things in class

* I believe he is very good at teaching the language of C and his lecture notes are the most helpful besides his T.A.'s.

* I enjoy how he provides real examples to help support the explanation of concepts

* I feel like this course moves awkwardly. Homework assignments are given too early for students to encorporate their knowledge into it. If homework assignments that related to the previous week's lectures were given later instead of that same week, it would be easier to write programs after learning the knowledge. Doing the homework while learning is like running before we can walk.

* I love that he goes over programming examples and alternative views. I've heard about other professors that don't use this technique.

* I really like the teacher, he performs well as a lecturer. All the information in his lectures prover very useful in doing the assignments and learning the course.

* I think the lectures are helpful and it is clear and easy to understand which is always a good thing.

* Keeps things interesting in class and makes learning the C language fun and understandable.

* Lectures are dry and boring.

* Lectures are extremely helpful - each slide is thoroughly explained. The class is easily passable if the student does his or her work diligently.

* Mr. Doemer provides excellent slides and excellent example programs to demonstrate the program to his students. It is apparent that Mr. Doemer knows the material he teaches. Students willing to wake up early and attend his lectures are at a much higher advantage than students who prefer to sleep in. Just because the slides are online doesn't mean all the information is online. Lastly, since Mr. Doemer's accent is so amazing, I actually enjoy listening to him speak.

* One of the best and most organized professors I've ever had. Lectures and slide are very well written and effectively presented.

* presents the information in a understandable way

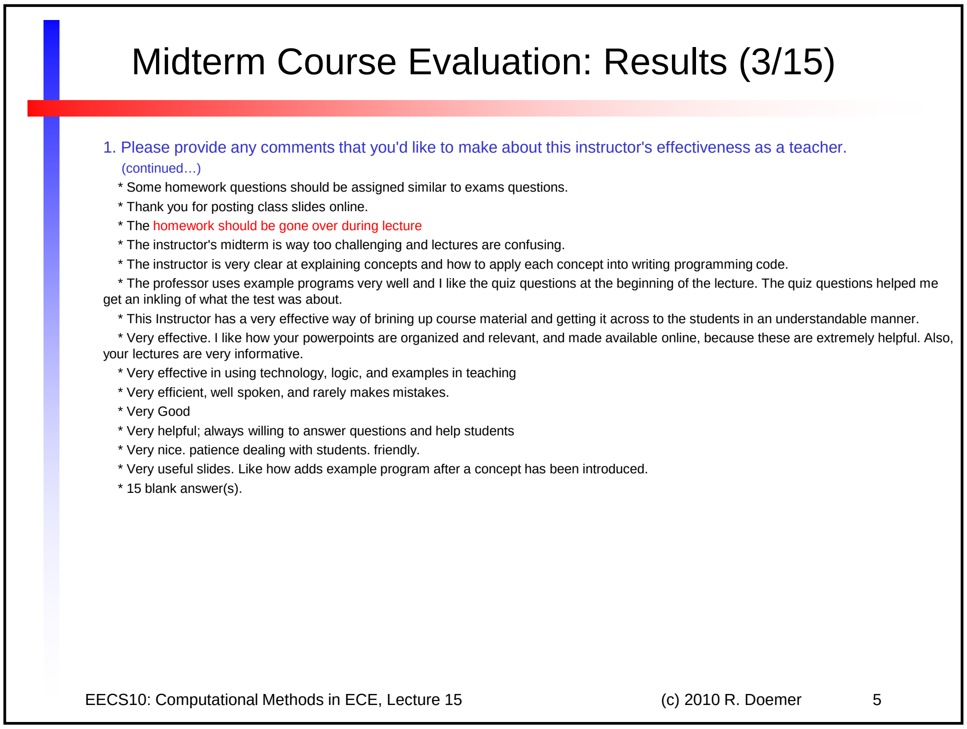
* Prof. Doemer's lecture slides are easy to understand so that, upon review, students can effectively study the material. If students have questions, Prof. Doemer addresses them right away and answers them with clarity.

* Professor Doemer is Great!

* Really effective! I love his sense of humor. Very friendly, and I think I might even take another of his courses later on.

* Slides are helpful. Tries to get class participation in asking questions. Provides a lot of examples. Not cranky in the morning unlike his students!!!

EECS10: Computational Methods in ECE, Lecture 15 (c) 2010 R. Doemer 4

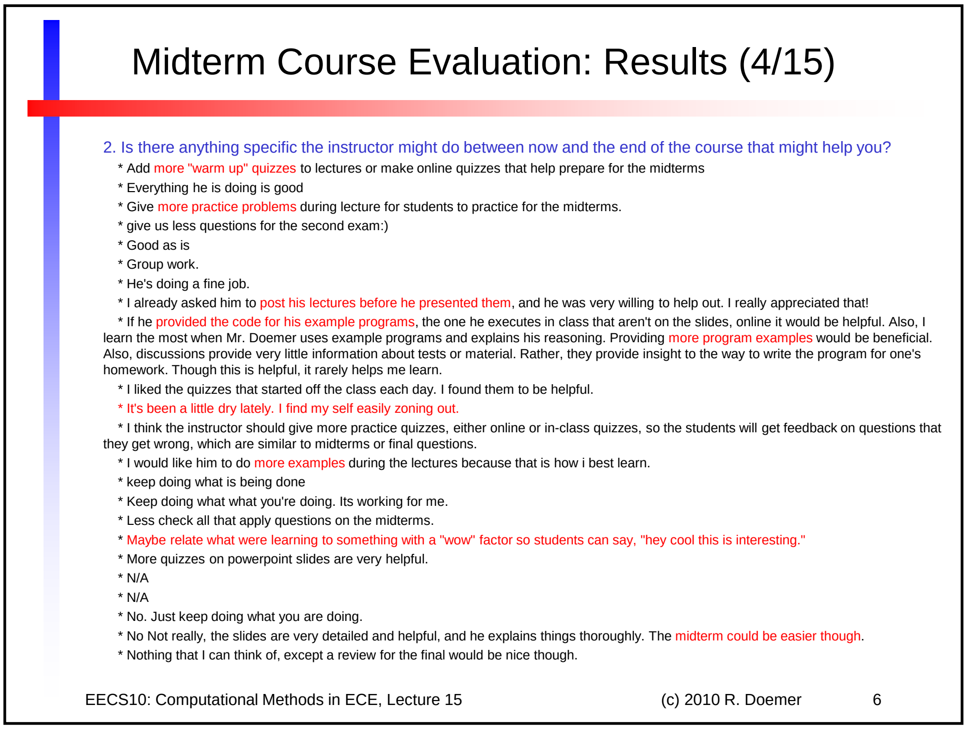


Midterm Course Evaluation: Results (3/15)

1. Please provide any comments that you'd like to make about this instructor's effectiveness as a teacher.

(continued…)

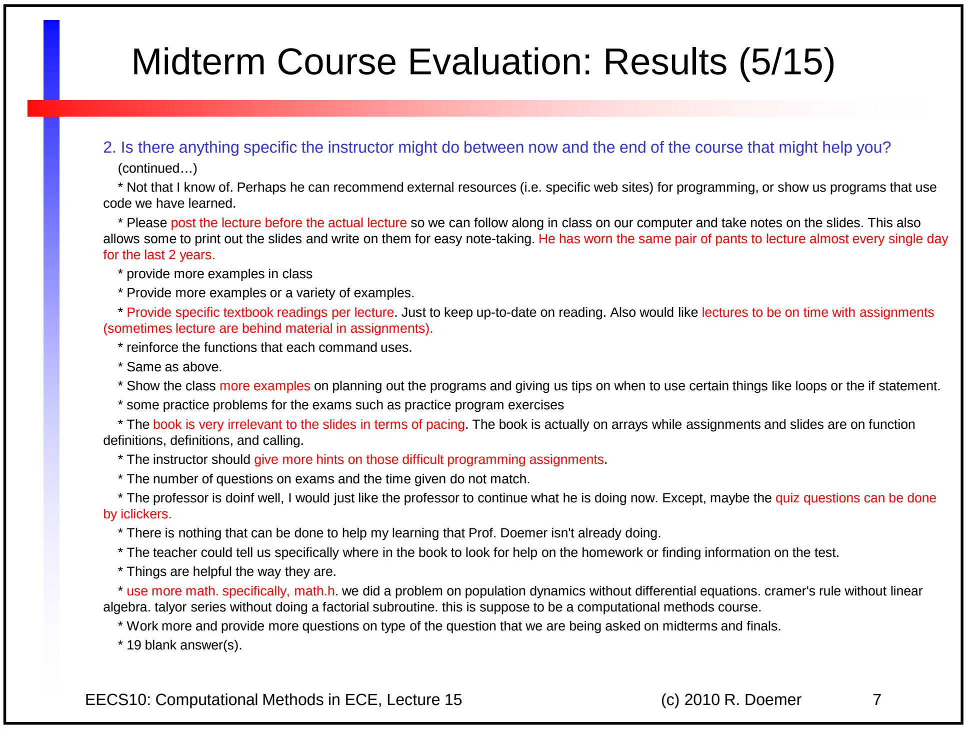
* Some homework questions should be assigned similar to exams questions.
* Thank you for posting class slides online.
* The homework should be gone over during lecture
* The instructor's midterm is way too challenging and lectures are confusing.
* The instructor is very clear at explaining concepts and how to apply each concept into writing programming code.
* The professor uses example programs very well and I like the quiz questions at the beginning of the lecture. The quiz questions helped me get an inkling of what the test was about.
* This Instructor has a very effective way of brining up course material and getting it across to the students in an understandable manner.
* Very effective. I like how your powerpoints are organized and relevant, and made available online, because these are extremely helpful. Also, your lectures are very informative.
* Very effective in using technology, logic, and examples in teaching
* Very efficient, well spoken, and rarely makes mistakes.
* Very Good
* Very helpful; always willing to answer questions and help students
* Very nice. patience dealing with students. friendly.
* Very useful slides. Like how adds example program after a concept has been introduced.
* 15 blank answer(s).



Midterm Course Evaluation: Results (4/15)

2. Is there anything specific the instructor might do between now and the end of the course that might help you?

* Add more "warm up" quizzes to lectures or make online quizzes that help prepare for the midterms
* Everything he is doing is good
* Give more practice problems during lecture for students to practice for the midterms.
* give us less questions for the second exam:)
* Good as is
* Group work.
* He's doing a fine job.
* I already asked him to post his lectures before he presented them, and he was very willing to help out. I really appreciated that!
* If he provided the code for his example programs, the one he executes in class that aren't on the slides, online it would be helpful. Also, I learn the most when Mr. Doemer uses example programs and explains his reasoning. Providing more program examples would be beneficial. Also, discussions provide very little information about tests or material. Rather, they provide insight to the way to write the program for one's homework. Though this is helpful, it rarely helps me learn.
* I liked the quizzes that started off the class each day. I found them to be helpful.
* It's been a little dry lately. I find my self easily zoning out.
* I think the instructor should give more practice quizzes, either online or in-class quizzes, so the students will get feedback on questions that they get wrong, which are similar to midterms or final questions.
* I would like him to do more examples during the lectures because that is how i best learn.
* keep doing what is being done
* Keep doing what what you're doing. Its working for me.
* Less check all that apply questions on the midterms.
* Maybe relate what were learning to something with a "wow" factor so students can say, "hey cool this is interesting."
* More quizzes on powerpoint slides are very helpful.
* N/A
* N/A
* No. Just keep doing what you are doing.
* No Not really, the slides are very detailed and helpful, and he explains things thoroughly. The midterm could be easier though.
* Nothing that I can think of, except a review for the final would be nice though.



Midterm Course Evaluation: Results (5/15)

2. Is there anything specific the instructor might do between now and the end of the course that might help you?

(continued…)

* Not that I know of. Perhaps he can recommend external resources (i.e. specific web sites) for programming, or show us programs that use code we have learned.

* Please post the lecture before the actual lecture so we can follow along in class on our computer and take notes on the slides. This also allows some to print out the slides and write on them for easy note-taking. He has worn the same pair of pants to lecture almost every single day for the last 2 years.

* provide more examples in class

* Provide more examples or a variety of examples.

* Provide specific textbook readings per lecture. Just to keep up-to-date on reading. Also would like lectures to be on time with assignments (sometimes lecture are behind material in assignments).

* reinforce the functions that each command uses.

* Same as above.

* Show the class more examples on planning out the programs and giving us tips on when to use certain things like loops or the if statement.

* some practice problems for the exams such as practice program exercises

* The book is very irrelevant to the slides in terms of pacing. The book is actually on arrays while assignments and slides are on function definitions, definitions, and calling.

* The instructor should give more hints on those difficult programming assignments.

* The number of questions on exams and the time given do not match.

* The professor is doinf well, I would just like the professor to continue what he is doing now. Except, maybe the quiz questions can be done by iclickers.

* There is nothing that can be done to help my learning that Prof. Doemer isn't already doing.

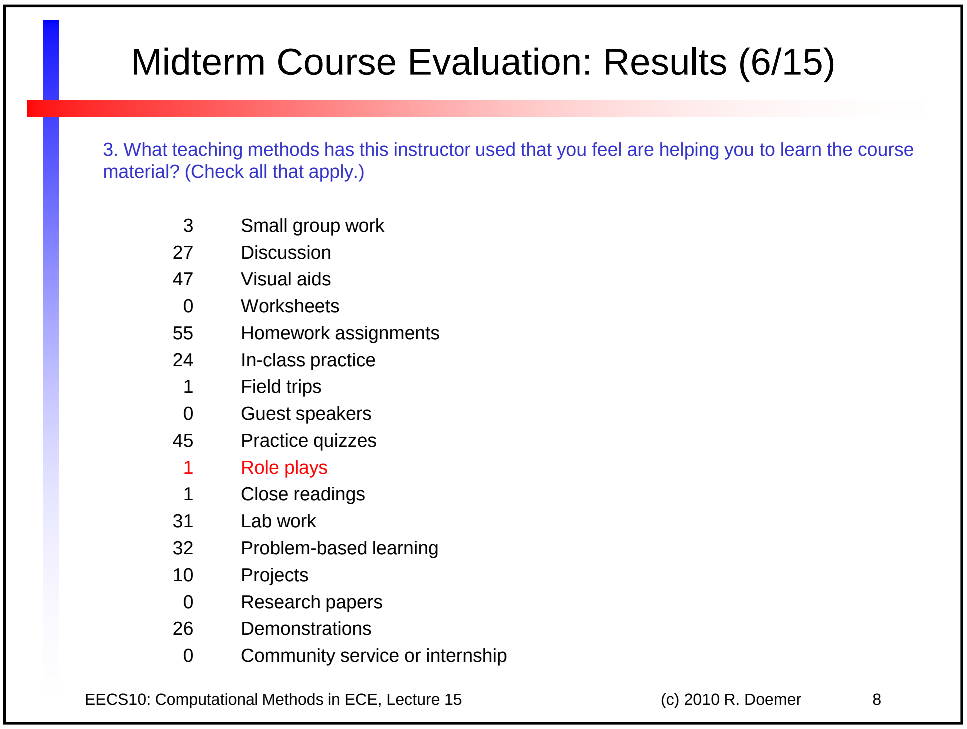
* The teacher could tell us specifically where in the book to look for help on the homework or finding information on the test.

* Things are helpful the way they are.

* use more math. specifically, math.h. we did a problem on population dynamics without differential equations. cramer's rule without linear algebra. talyor series without doing a factorial subroutine. this is suppose to be a computational methods course.

* Work more and provide more questions on type of the question that we are being asked on midterms and finals.

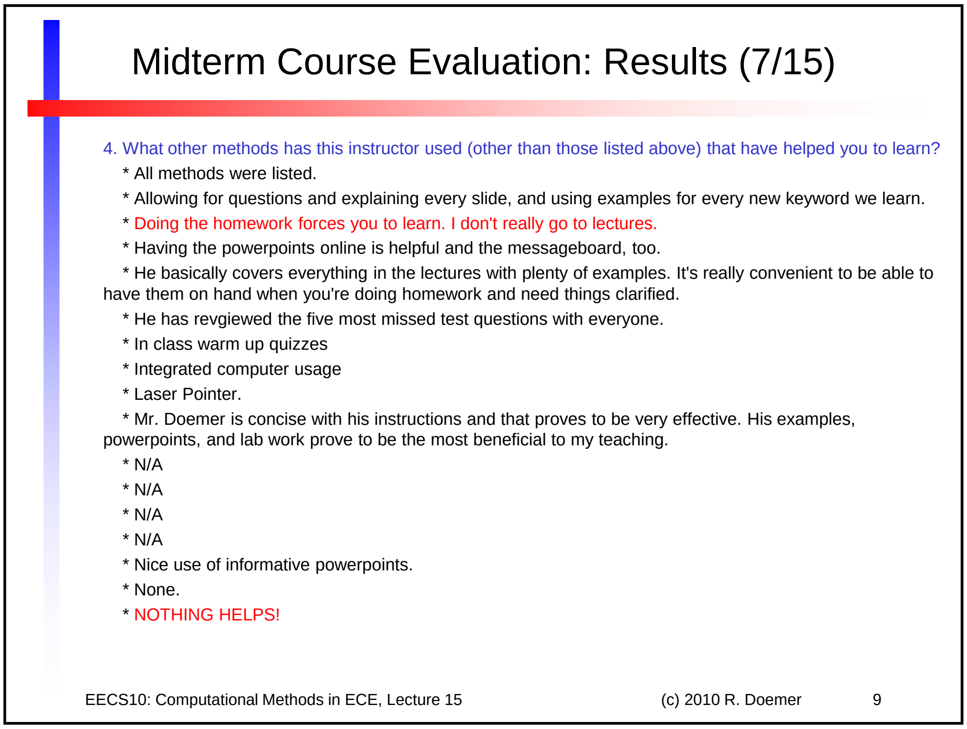
* 19 blank answer(s).



Midterm Course Evaluation: Results (6/15)

3. What teaching methods has this instructor used that you feel are helping you to learn the course material? (Check all that apply.)

3	Small group work
27	Discussion
47	Visual aids
0	Worksheets
55	Homework assignments
24	In-class practice
1	Field trips
0	Guest speakers
45	Practice quizzes
1	Role plays
1	Close readings
31	Lab work
32	Problem-based learning
10	Projects
0	Research papers
26	Demonstrations
0	Community service or internship



Midterm Course Evaluation: Results (7/15)

4. What other methods has this instructor used (other than those listed above) that have helped you to learn?

* All methods were listed.

* Allowing for questions and explaining every slide, and using examples for every new keyword we learn.

* Doing the homework forces you to learn. I don't really go to lectures.

* Having the powerpoints online is helpful and the messageboard, too.

* He basically covers everything in the lectures with plenty of examples. It's really convenient to be able to have them on hand when you're doing homework and need things clarified.

* He has revgiewed the five most missed test questions with everyone.

* In class warm up quizzes

* Integrated computer usage

* Laser Pointer.

* Mr. Doemer is concise with his instructions and that proves to be very effective. His examples, powerpoints, and lab work prove to be the most beneficial to my teaching.

* N/A

* N/A

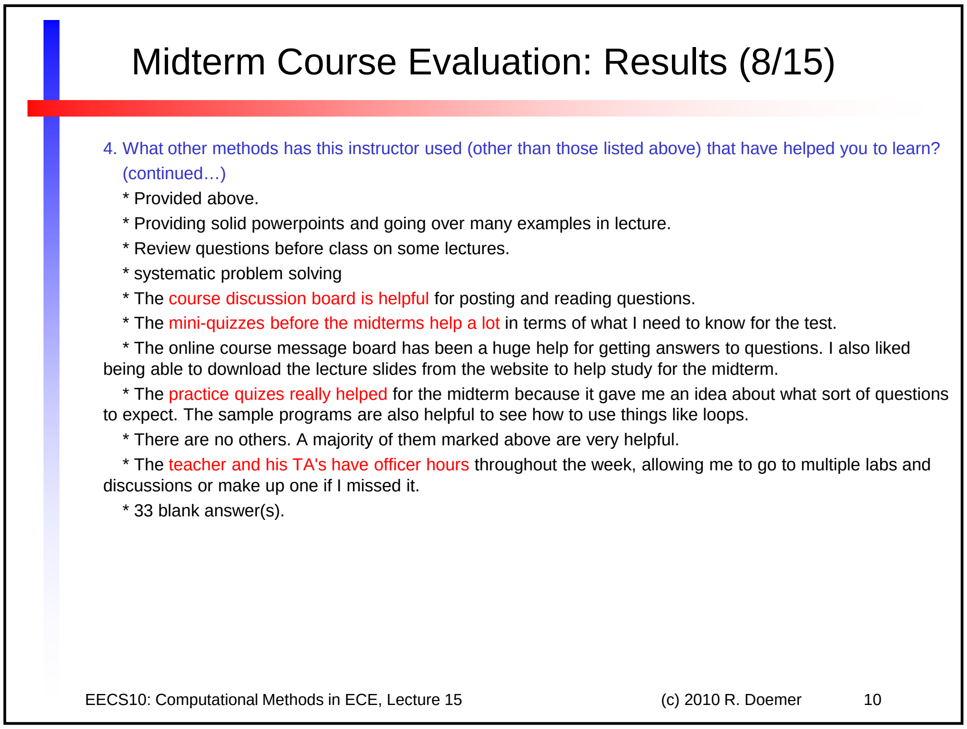
* N/A

* N/A

* Nice use of informative powerpoints.

* None.

* NOTHING HELPS!



Midterm Course Evaluation: Results (8/15)

4. What other methods has this instructor used (other than those listed above) that have helped you to learn? (continued…)

* Provided above.

* Providing solid powerpoints and going over many examples in lecture.

* Review questions before class on some lectures.

* systematic problem solving

* The course discussion board is helpful for posting and reading questions.

* The mini-quizzes before the midterms help a lot in terms of what I need to know for the test.

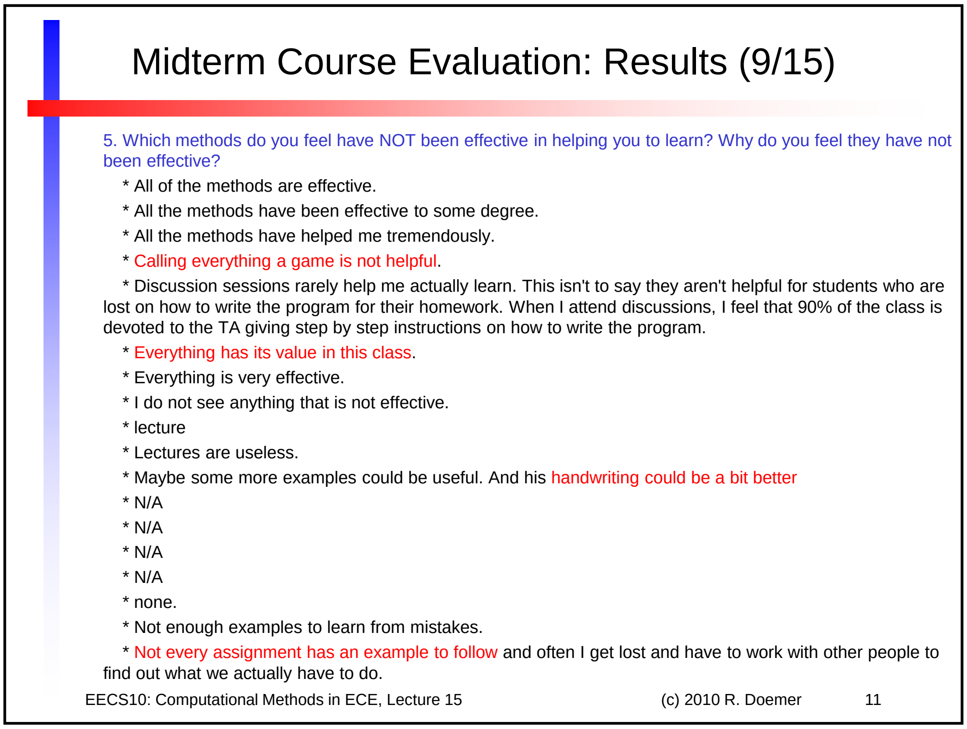
* The online course message board has been a huge help for getting answers to questions. I also liked being able to download the lecture slides from the website to help study for the midterm.

* The practice quizes really helped for the midterm because it gave me an idea about what sort of questions to expect. The sample programs are also helpful to see how to use things like loops.

* There are no others. A majority of them marked above are very helpful.

* The teacher and his TA's have officer hours throughout the week, allowing me to go to multiple labs and discussions or make up one if I missed it.

* 33 blank answer(s).



Midterm Course Evaluation: Results (9/15)

5. Which methods do you feel have NOT been effective in helping you to learn? Why do you feel they have not been effective?

* All of the methods are effective.

* All the methods have been effective to some degree.

* All the methods have helped me tremendously.

* Calling everything a game is not helpful.

* Discussion sessions rarely help me actually learn. This isn't to say they aren't helpful for students who are lost on how to write the program for their homework. When I attend discussions, I feel that 90% of the class is devoted to the TA giving step by step instructions on how to write the program.

* Everything has its value in this class.

* Everything is very effective.

* I do not see anything that is not effective.

* lecture

* Lectures are useless.

* Maybe some more examples could be useful. And his handwriting could be a bit better

* N/A

* N/A

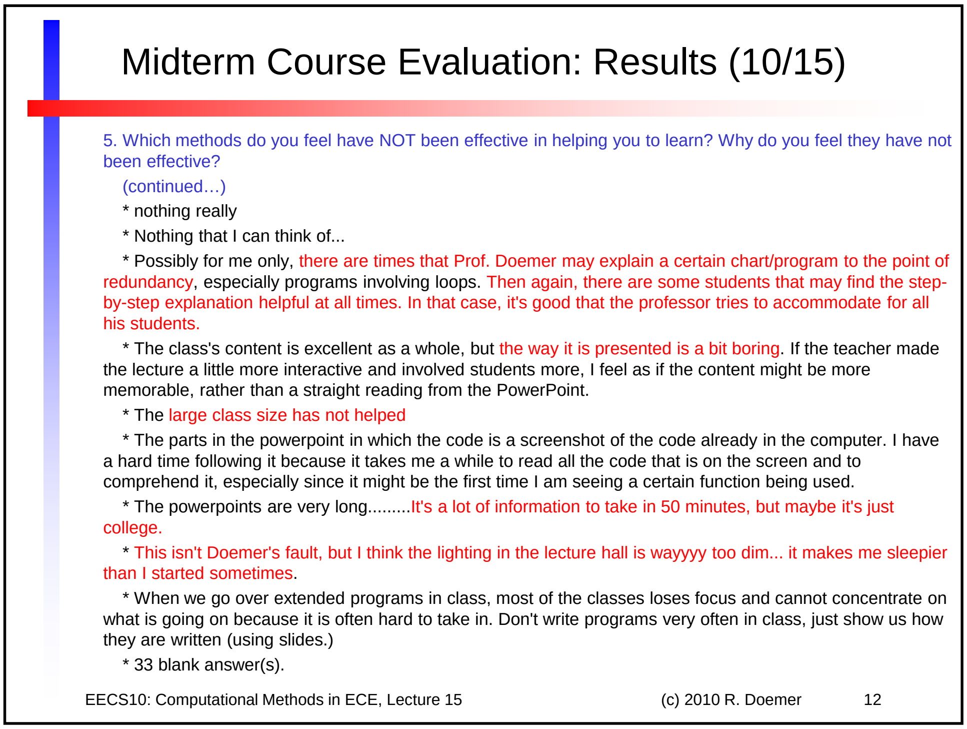
* N/A

* N/A

* none.

* Not enough examples to learn from mistakes.

* Not every assignment has an example to follow and often I get lost and have to work with other people to find out what we actually have to do.



Midterm Course Evaluation: Results (10/15)

5. Which methods do you feel have NOT been effective in helping you to learn? Why do you feel they have not been effective?

(continued…)

* nothing really

* Nothing that I can think of...

* Possibly for me only, there are times that Prof. Doemer may explain a certain chart/program to the point of redundancy, especially programs involving loops. Then again, there are some students that may find the step-by-step explanation helpful at all times. In that case, it's good that the professor tries to accommodate for all his students.

* The class's content is excellent as a whole, but the way it is presented is a bit boring. If the teacher made the lecture a little more interactive and involved students more, I feel as if the content might be more memorable, rather than a straight reading from the PowerPoint.

* The large class size has not helped

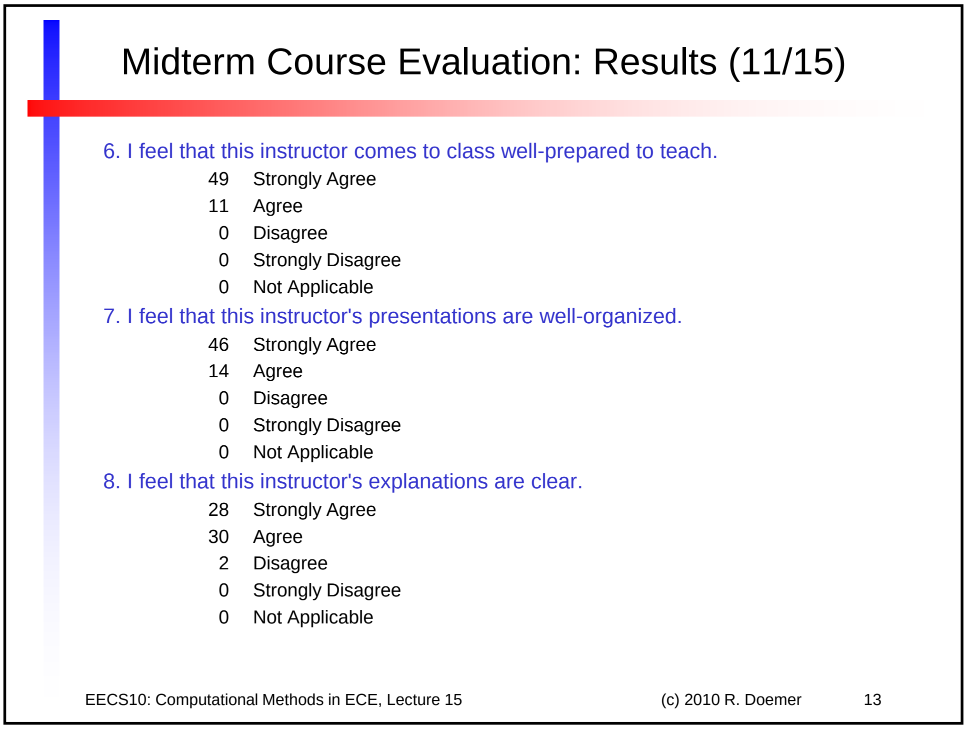
* The parts in the powerpoint in which the code is a screenshot of the code already in the computer. I have a hard time following it because it takes me a while to read all the code that is on the screen and to comprehend it, especially since it might be the first time I am seeing a certain function being used.

* The powerpoints are very long.........It's a lot of information to take in 50 minutes, but maybe it's just college.

* This isn't Doemer's fault, but I think the lighting in the lecture hall is wayyyy too dim... it makes me sleepier than I started sometimes.

* When we go over extended programs in class, most of the classes loses focus and cannot concentrate on what is going on because it is often hard to take in. Don't write programs very often in class, just show us how they are written (using slides.)

* 33 blank answer(s).



Midterm Course Evaluation: Results (11/15)

6. I feel that this instructor comes to class well-prepared to teach.

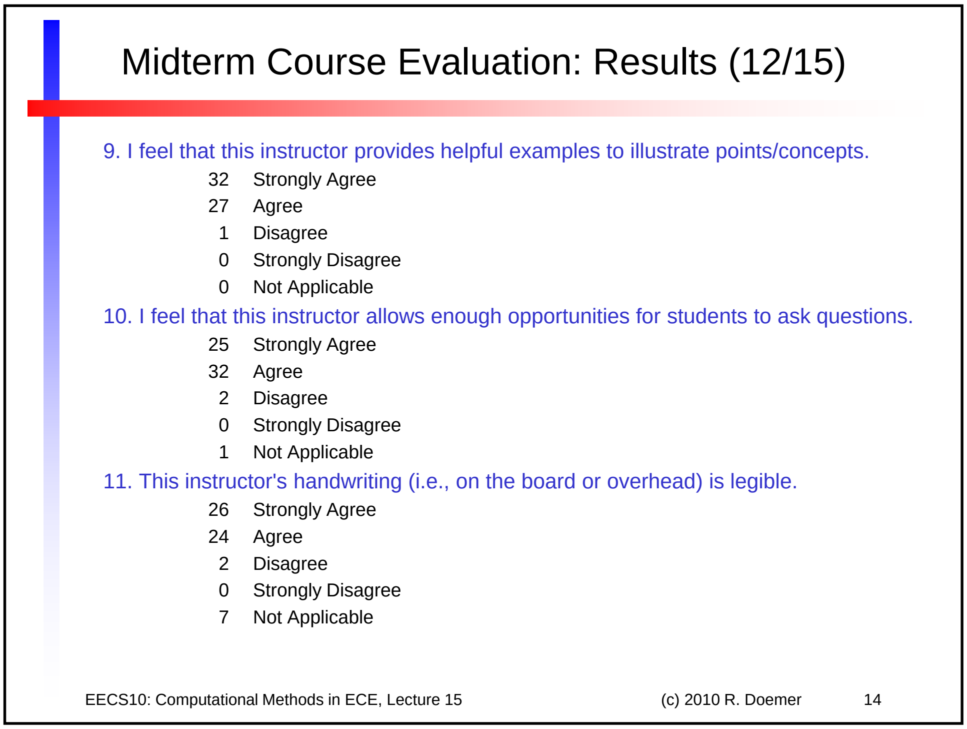
49	Strongly Agree
11	Agree
0	Disagree
0	Strongly Disagree
0	Not Applicable

7. I feel that this instructor's presentations are well-organized.

46	Strongly Agree
14	Agree
0	Disagree
0	Strongly Disagree
0	Not Applicable

8. I feel that this instructor's explanations are clear.

28	Strongly Agree
30	Agree
2	Disagree
0	Strongly Disagree
0	Not Applicable



Midterm Course Evaluation: Results (12/15)

9. I feel that this instructor provides helpful examples to illustrate points/concepts.

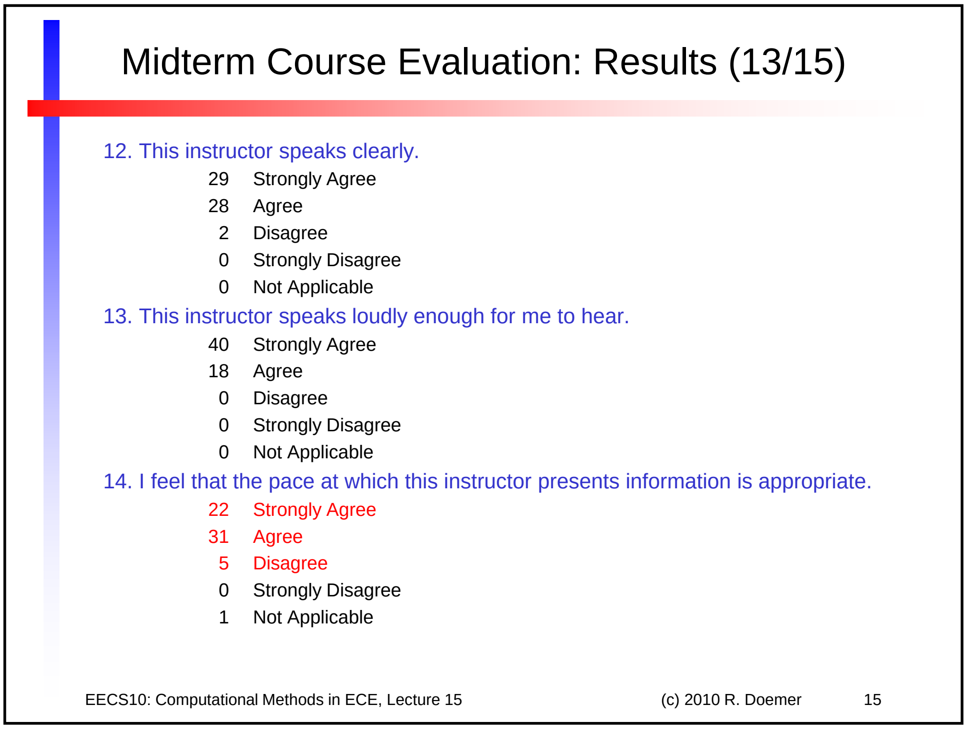
32	Strongly Agree
27	Agree
1	Disagree
0	Strongly Disagree
0	Not Applicable

10. I feel that this instructor allows enough opportunities for students to ask questions.

25	Strongly Agree
32	Agree
2	Disagree
0	Strongly Disagree
1	Not Applicable

11. This instructor's handwriting (i.e., on the board or overhead) is legible.

26	Strongly Agree
24	Agree
2	Disagree
0	Strongly Disagree
7	Not Applicable



Midterm Course Evaluation: Results (13/15)

12. This instructor speaks clearly.

29	Strongly Agree
28	Agree
2	Disagree
0	Strongly Disagree
0	Not Applicable

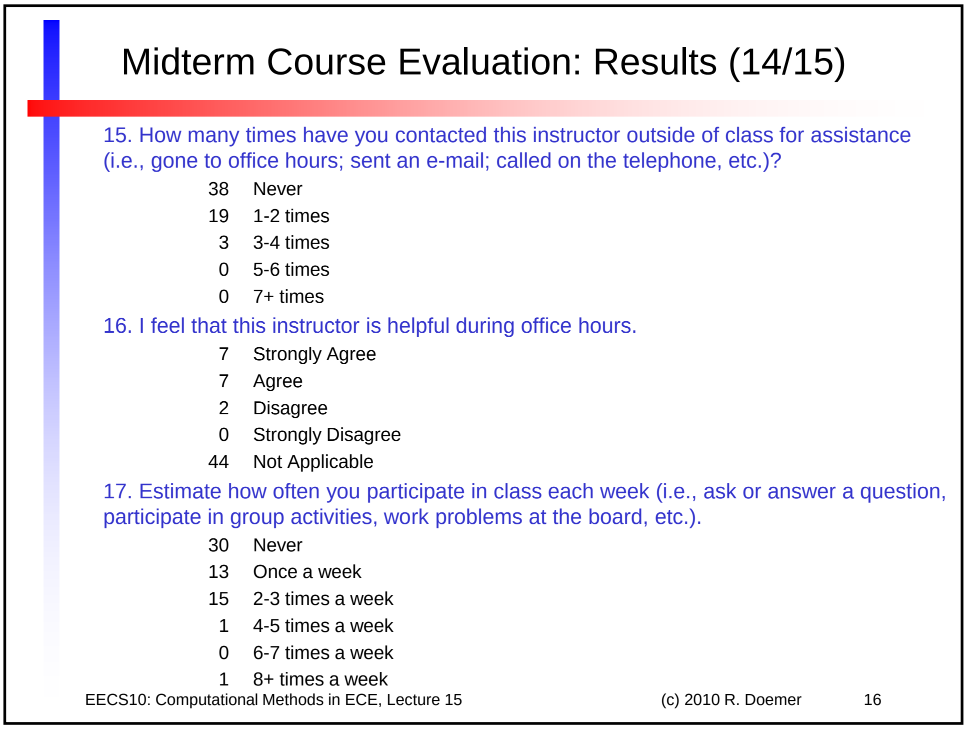
13. This instructor speaks loudly enough for me to hear.

40	Strongly Agree
18	Agree
0	Disagree
0	Strongly Disagree
0	Not Applicable

14. I feel that the pace at which this instructor presents information is appropriate.

22	Strongly Agree
31	Agree
5	Disagree
0	Strongly Disagree
1	Not Applicable

EECS10: Computational Methods in ECE, Lecture 15 (c) 2010 R. Doemer 15



Midterm Course Evaluation: Results (14/15)

15. How many times have you contacted this instructor outside of class for assistance (i.e., gone to office hours; sent an e-mail; called on the telephone, etc.)?

38	Never
19	1-2 times
3	3-4 times
0	5-6 times
0	7+ times

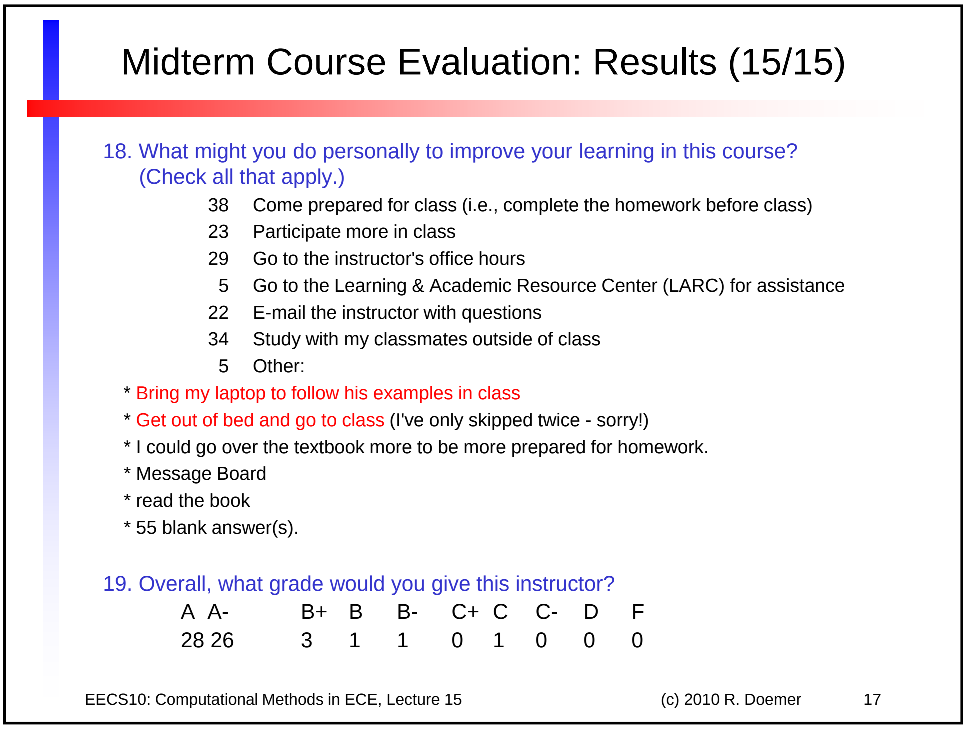
16. I feel that this instructor is helpful during office hours.

7	Strongly Agree
7	Agree
2	Disagree
0	Strongly Disagree
44	Not Applicable

17. Estimate how often you participate in class each week (i.e., ask or answer a question, participate in group activities, work problems at the board, etc.).

30	Never
13	Once a week
15	2-3 times a week
1	4-5 times a week
0	6-7 times a week
1	8+ times a week

EECS10: Computational Methods in ECE, Lecture 15 (c) 2010 R. Doemer 16



Midterm Course Evaluation: Results (15/15)

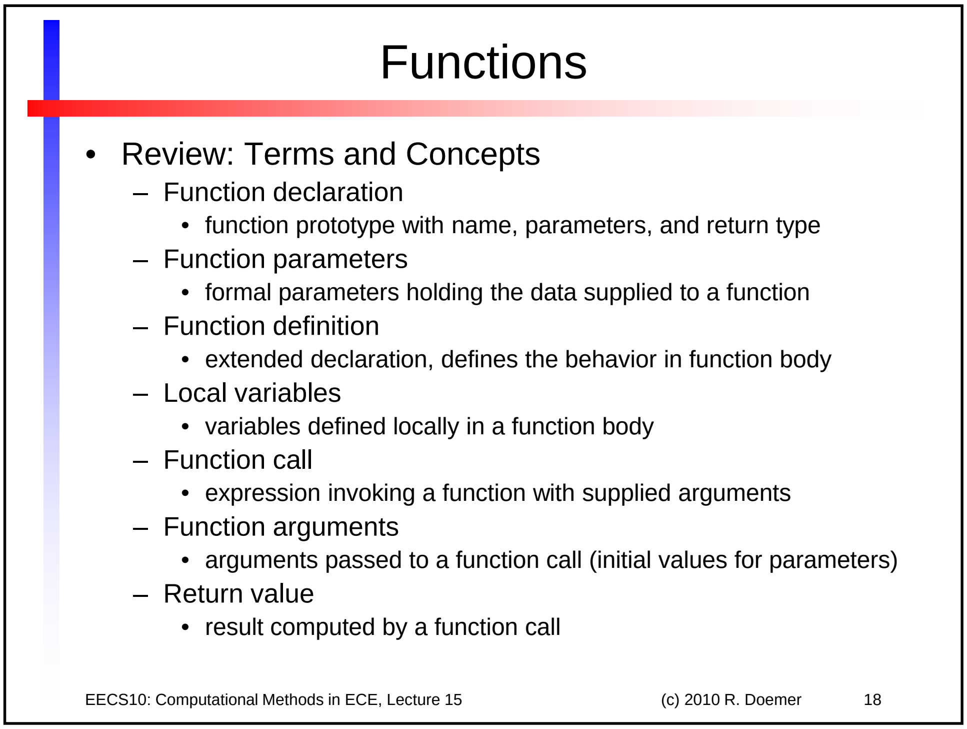
18. What might you do personally to improve your learning in this course? (Check all that apply.)

38	Come prepared for class (i.e., complete the homework before class)
23	Participate more in class
29	Go to the instructor's office hours
5	Go to the Learning & Academic Resource Center (LARC) for assistance
22	E-mail the instructor with questions
34	Study with my classmates outside of class
5	Other:

* Bring my laptop to follow his examples in class
* Get out of bed and go to class (I've only skipped twice - sorry!)
* I could go over the textbook more to be more prepared for homework.
* Message Board
* read the book
* 55 blank answer(s).

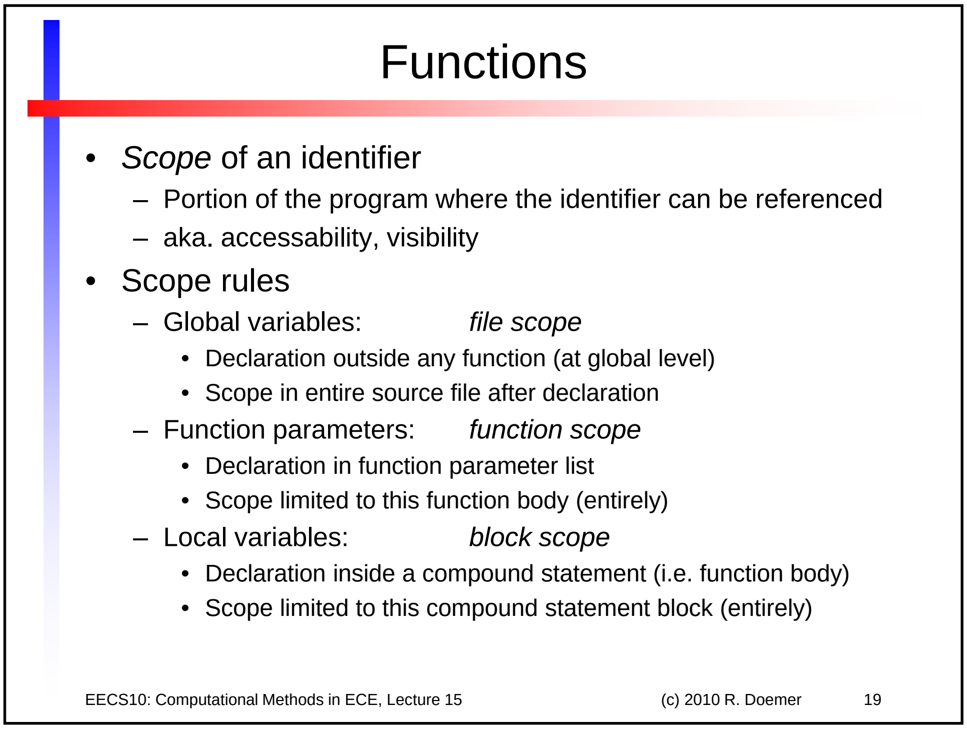
19. Overall, what grade would you give this instructor?

A	A-	B+	B	B-	C+	C	C-	D	F
28	26	3	1	1	0	1	0	0	0



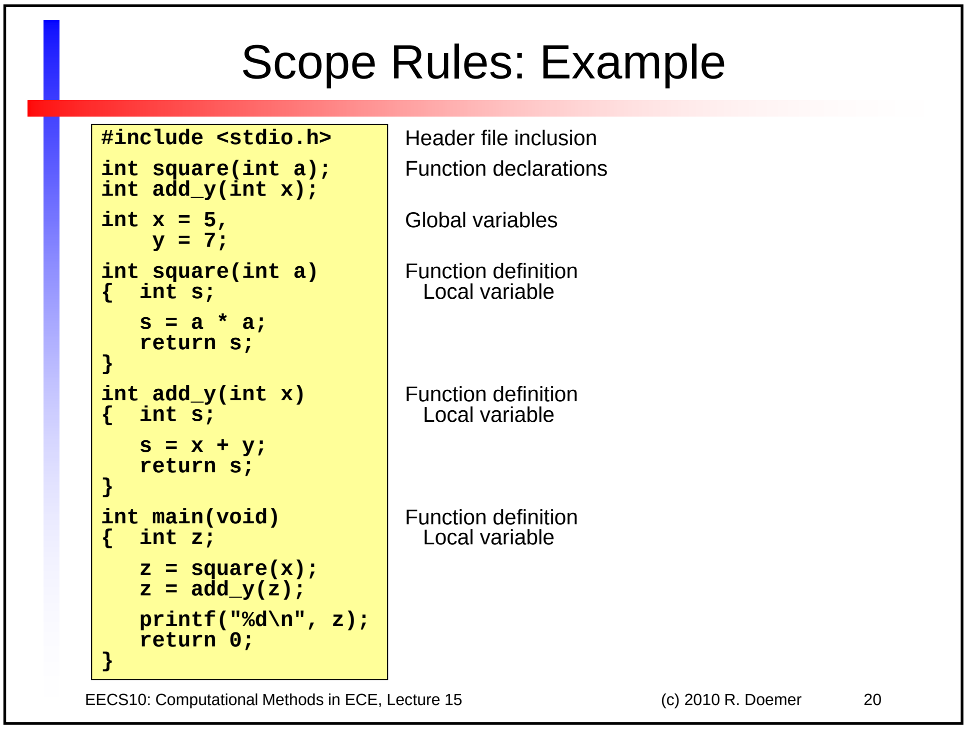
Functions

- Review: Terms and Concepts
  - Function declaration
    - function prototype with name, parameters, and return type
  - Function parameters
    - formal parameters holding the data supplied to a function
  - Function definition
    - extended declaration, defines the behavior in function body
  - Local variables
    - variables defined locally in a function body
  - Function call
    - expression invoking a function with supplied arguments
  - Function arguments
    - arguments passed to a function call (initial values for parameters)
  - Return value
    - result computed by a function call



Functions

- *Scope* of an identifier
  - Portion of the program where the identifier can be referenced
  - aka. accessability, visibility
- Scope rules
  - Global variables: *file scope*
    - Declaration outside any function (at global level)
    - Scope in entire source file after declaration
  - Function parameters: *function scope*
    - Declaration in function parameter list
    - Scope limited to this function body (entirely)
  - Local variables: *block scope*
    - Declaration inside a compound statement (i.e. function body)
    - Scope limited to this compound statement block (entirely)



Scope Rules: Example

```
#include <stdio.h>

int square(int a);
int add_y(int x);

int x = 5,
    y = 7;

int square(int a)
{  int s;

   s = a * a;
   return s;
}

int add_y(int x)
{  int s;

   s = x + y;
   return s;
}

int main(void)
{  int z;

   z = square(x);
   z = add_y(z);

   printf("%d\n", z);
   return 0;
}
```

Header file inclusion

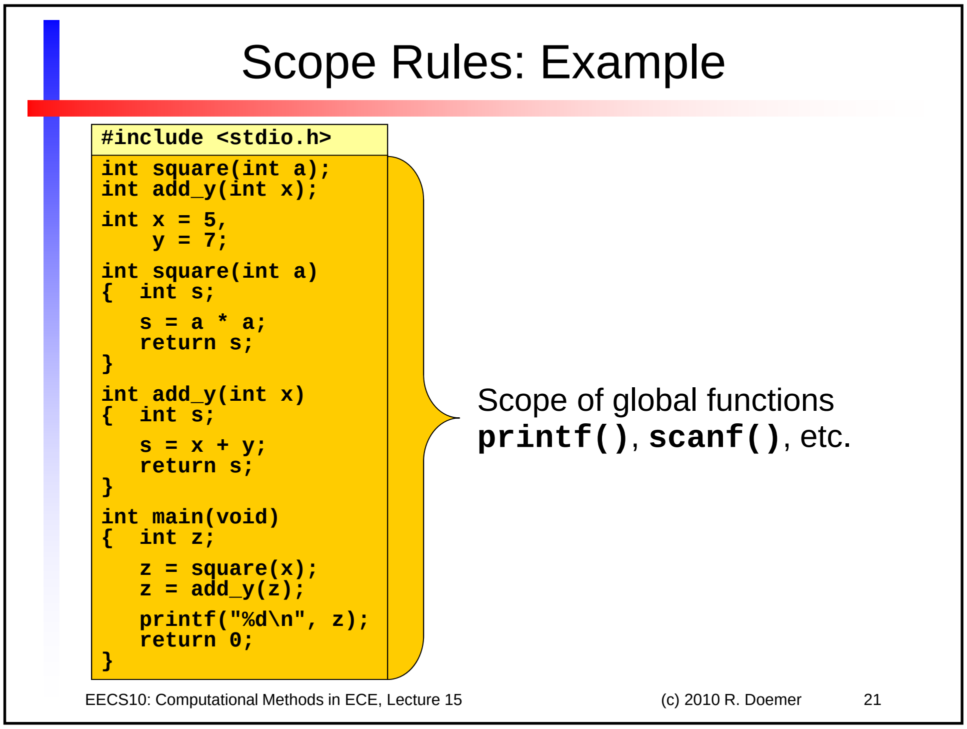
Function declarations

Global variables

Function definition
  Local variable

Function definition
  Local variable

Function definition
  Local variable

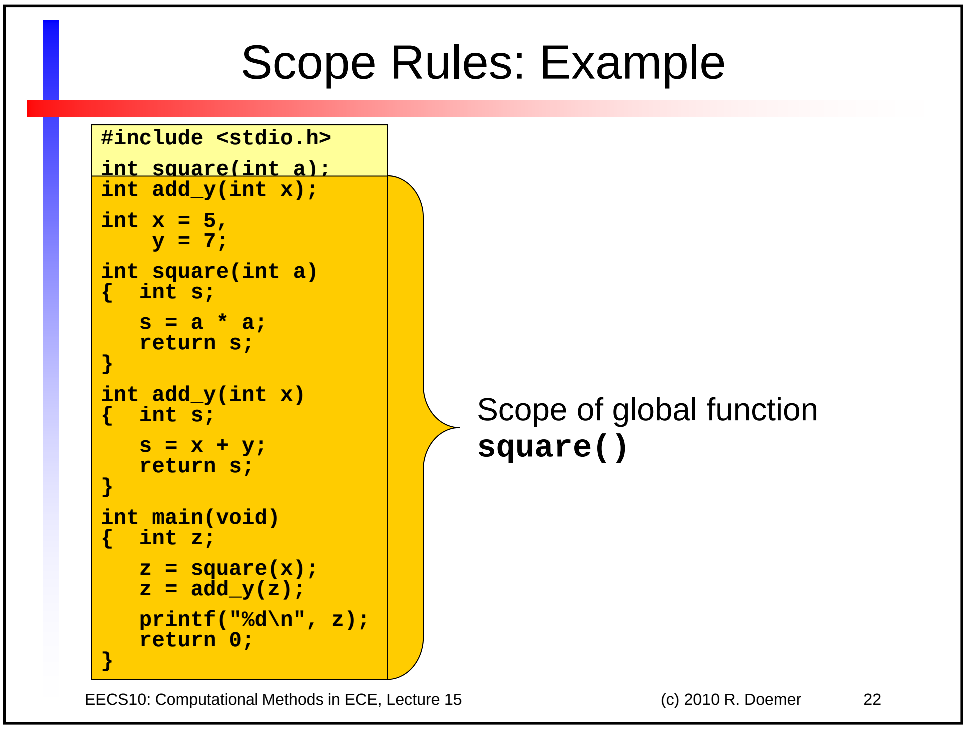


Scope Rules: Example

```
#include <stdio.h>
int square(int a);
int add_y(int x);
int x = 5,
    y = 7;
int square(int a)
{  int s;
   s = a * a;
   return s;
}
int add_y(int x)
{  int s;
   s = x + y;
   return s;
}
int main(void)
{  int z;
   z = square(x);
   z = add_y(z);
   printf("%d\n", z);
   return 0;
}
```

Scope of global functions **`printf()`**, **`scanf()`**, etc.

EECS10: Computational Methods in ECE, Lecture 15 (c) 2010 R. Doemer 21

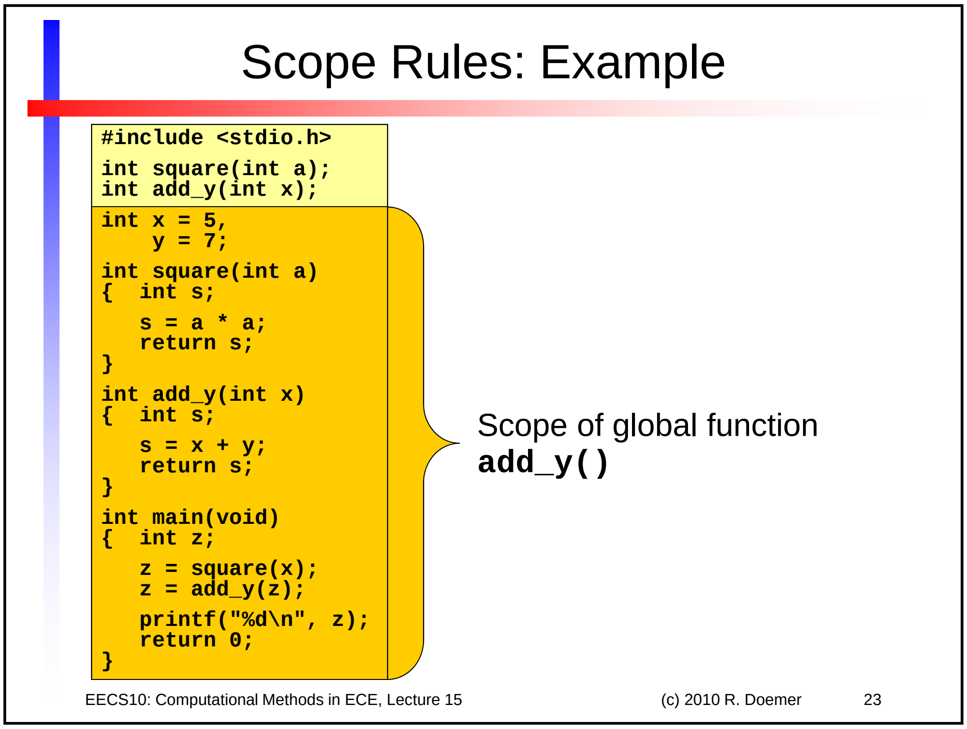


Scope Rules: Example

```
#include <stdio.h>
int square(int a);
int add_y(int x);
int x = 5,
    y = 7;
int square(int a)
{  int s;
   s = a * a;
   return s;
}
int add_y(int x)
{  int s;
   s = x + y;
   return s;
}
int main(void)
{  int z;
   z = square(x);
   z = add_y(z);
   printf("%d\n", z);
   return 0;
}
```

Scope of global function **`square()`**

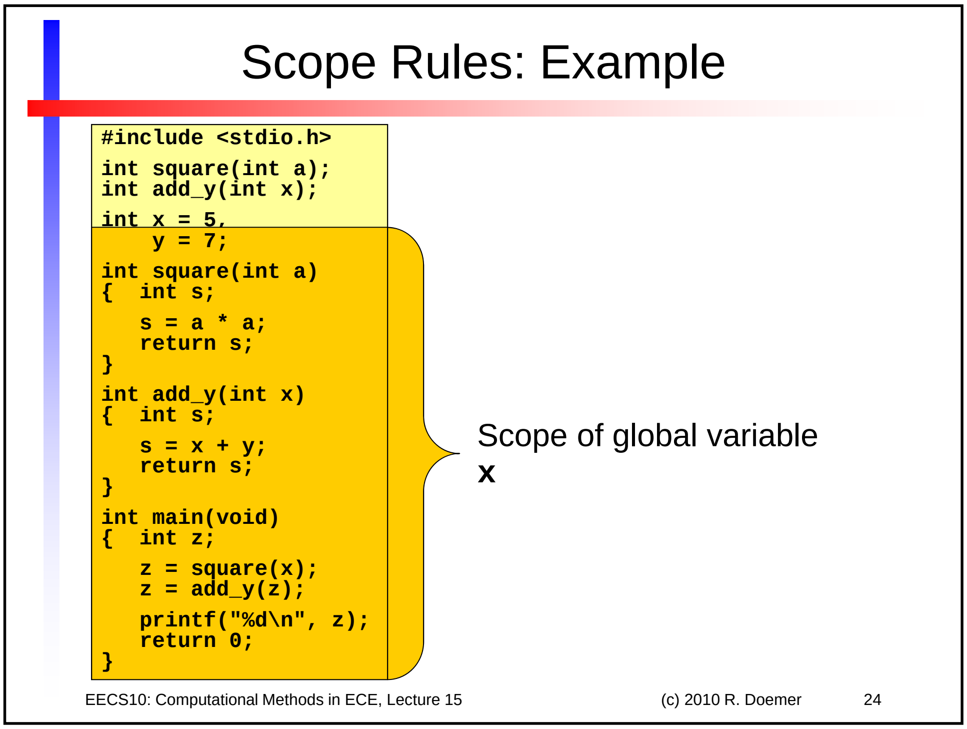
EECS10: Computational Methods in ECE, Lecture 15 (c) 2010 R. Doemer 22



Scope Rules: Example

```
#include <stdio.h>
int square(int a);
int add_y(int x);
int x = 5,
    y = 7;
int square(int a)
{  int s;
   s = a * a;
   return s;
}
int add_y(int x)
{  int s;
   s = x + y;
   return s;
}
int main(void)
{  int z;
   z = square(x);
   z = add_y(z);
   printf("%d\n", z);
   return 0;
}
```

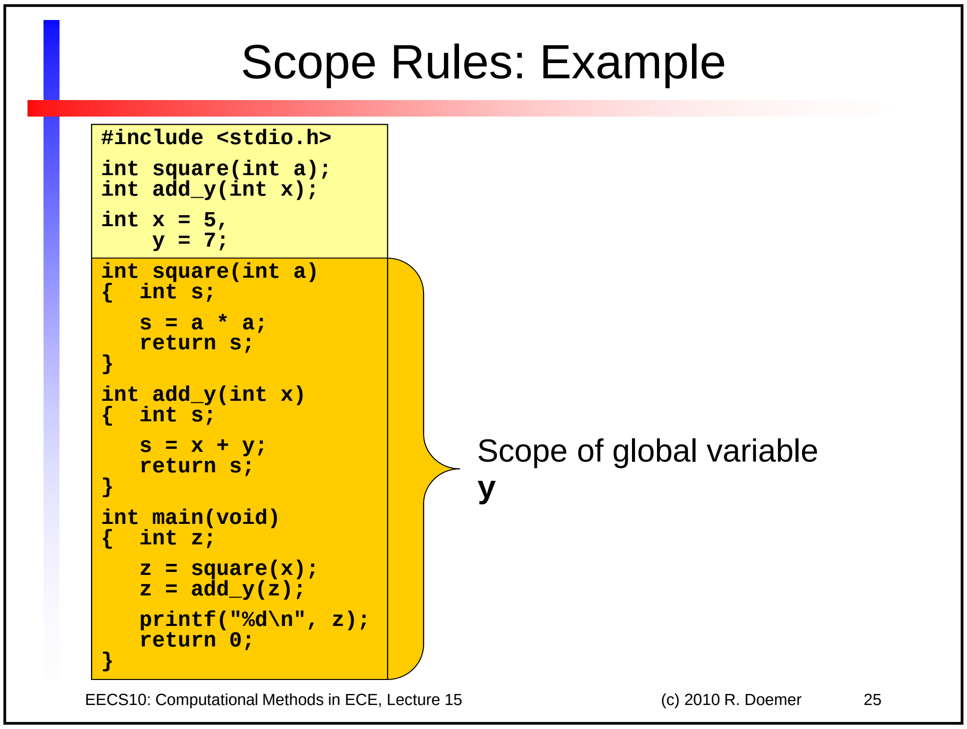
Scope of global function **`add_y()`**



Scope Rules: Example

```
#include <stdio.h>
int square(int a);
int add_y(int x);
int x = 5,
    y = 7;
int square(int a)
{  int s;
   s = a * a;
   return s;
}
int add_y(int x)
{  int s;
   s = x + y;
   return s;
}
int main(void)
{  int z;
   z = square(x);
   z = add_y(z);
   printf("%d\n", z);
   return 0;
}
```

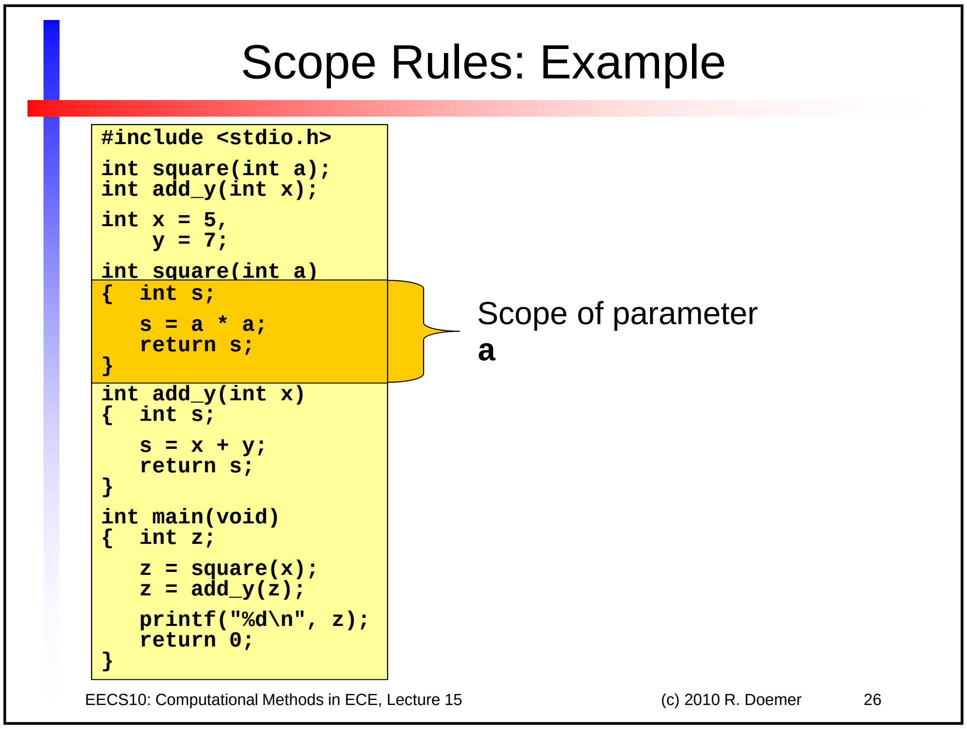
Scope of global variable **`x`**



Scope Rules: Example

```
#include <stdio.h>
int square(int a);
int add_y(int x);
int x = 5,
    y = 7;
int square(int a)
{  int s;
   s = a * a;
   return s;
}
int add_y(int x)
{  int s;
   s = x + y;
   return s;
}
int main(void)
{  int z;
   z = square(x);
   z = add_y(z);
   printf("%d\n", z);
   return 0;
}
```

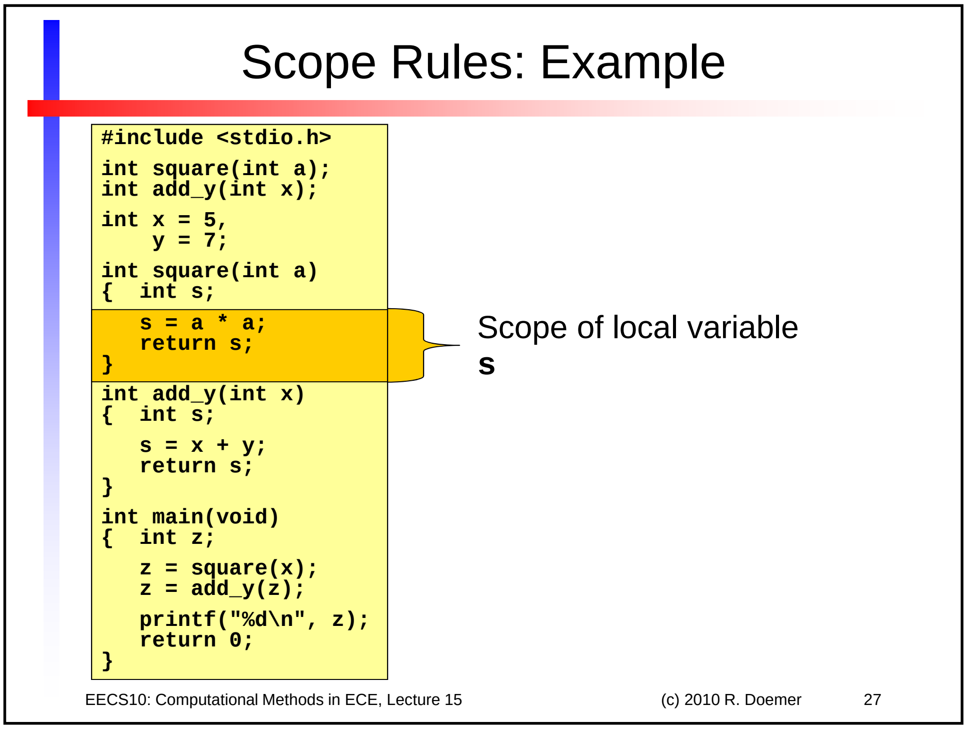
Scope of global variable **y**



Scope Rules: Example

```
#include <stdio.h>
int square(int a);
int add_y(int x);
int x = 5,
    y = 7;
int square(int a)
{  int s;
   s = a * a;
   return s;
}
int add_y(int x)
{  int s;
   s = x + y;
   return s;
}
int main(void)
{  int z;
   z = square(x);
   z = add_y(z);
   printf("%d\n", z);
   return 0;
}
```

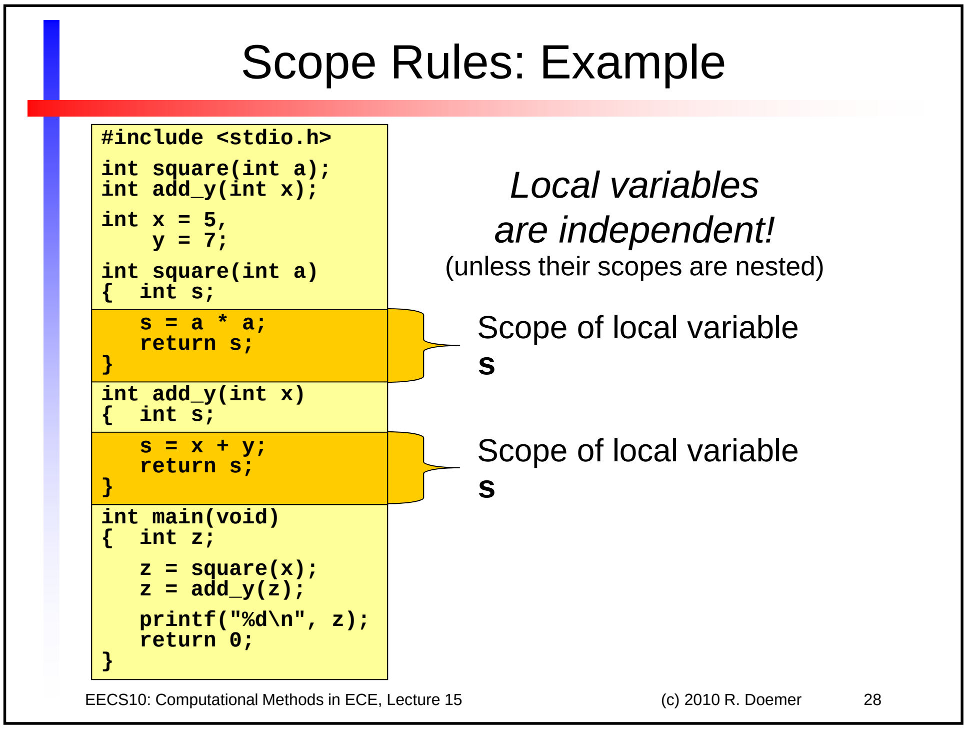
Scope of parameter **a**



Scope Rules: Example

```
#include <stdio.h>
int square(int a);
int add_y(int x);
int x = 5,
    y = 7;
int square(int a)
{   int s;
    s = a * a;
    return s;
}
int add_y(int x)
{   int s;
    s = x + y;
    return s;
}
int main(void)
{   int z;
    z = square(x);
    z = add_y(z);
    printf("%d\n", z);
    return 0;
}
```

Scope of local variable **s**



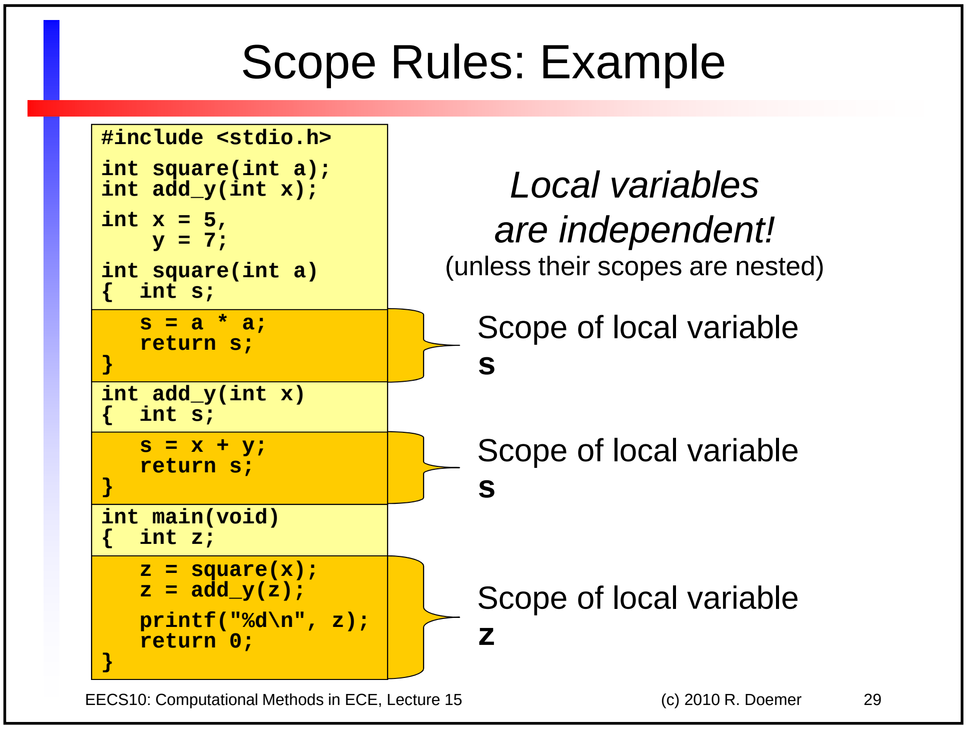
Scope Rules: Example

```
#include <stdio.h>
int square(int a);
int add_y(int x);
int x = 5,
    y = 7;
int square(int a)
{   int s;
    s = a * a;
    return s;
}
int add_y(int x)
{   int s;
    s = x + y;
    return s;
}
int main(void)
{   int z;
    z = square(x);
    z = add_y(z);
    printf("%d\n", z);
    return 0;
}
```

Local variables
are independent!
(unless their scopes are nested)

Scope of local variable **s**

Scope of local variable **s**



Scope Rules: Example

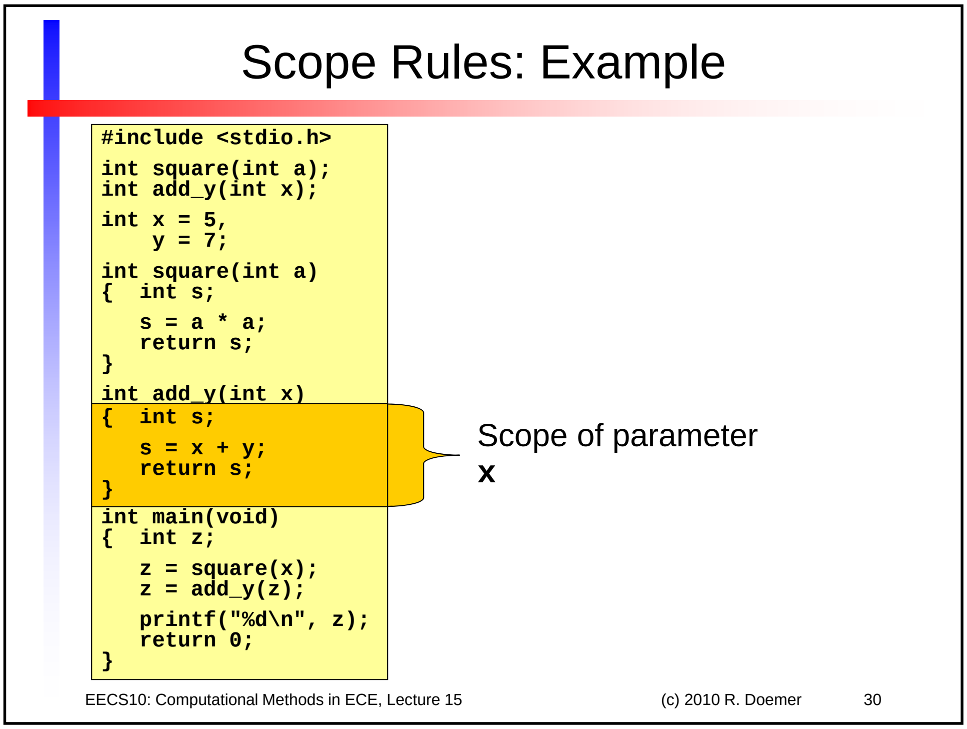
```
#include <stdio.h>
int square(int a);
int add_y(int x);
int x = 5,
    y = 7;
int square(int a)
{  int s;
   s = a * a;
   return s;
}
int add_y(int x)
{  int s;
   s = x + y;
   return s;
}
int main(void)
{  int z;
   z = square(x);
   z = add_y(z);
   printf("%d\n", z);
   return 0;
}
```

Local variables are independent!
(unless their scopes are nested)

Scope of local variable **s**

Scope of local variable **s**

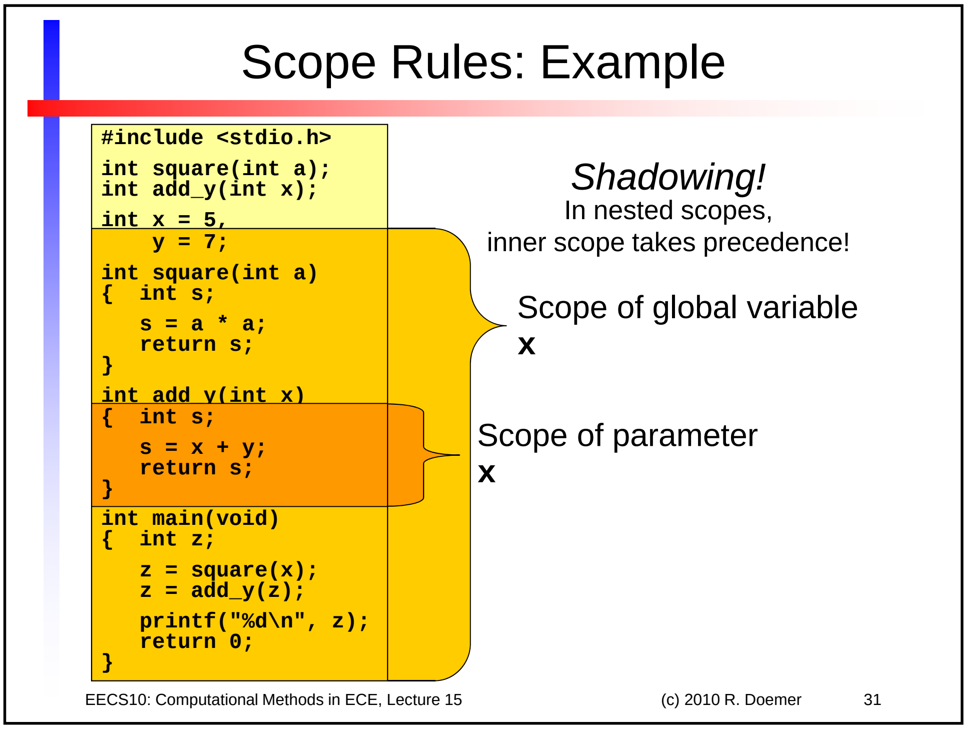
Scope of local variable **z**



Scope Rules: Example

```
#include <stdio.h>
int square(int a);
int add_y(int x);
int x = 5,
    y = 7;
int square(int a)
{  int s;
   s = a * a;
   return s;
}
int add_y(int x)
{  int s;
   s = x + y;
   return s;
}
int main(void)
{  int z;
   z = square(x);
   z = add_y(z);
   printf("%d\n", z);
   return 0;
}
```

Scope of parameter **x**



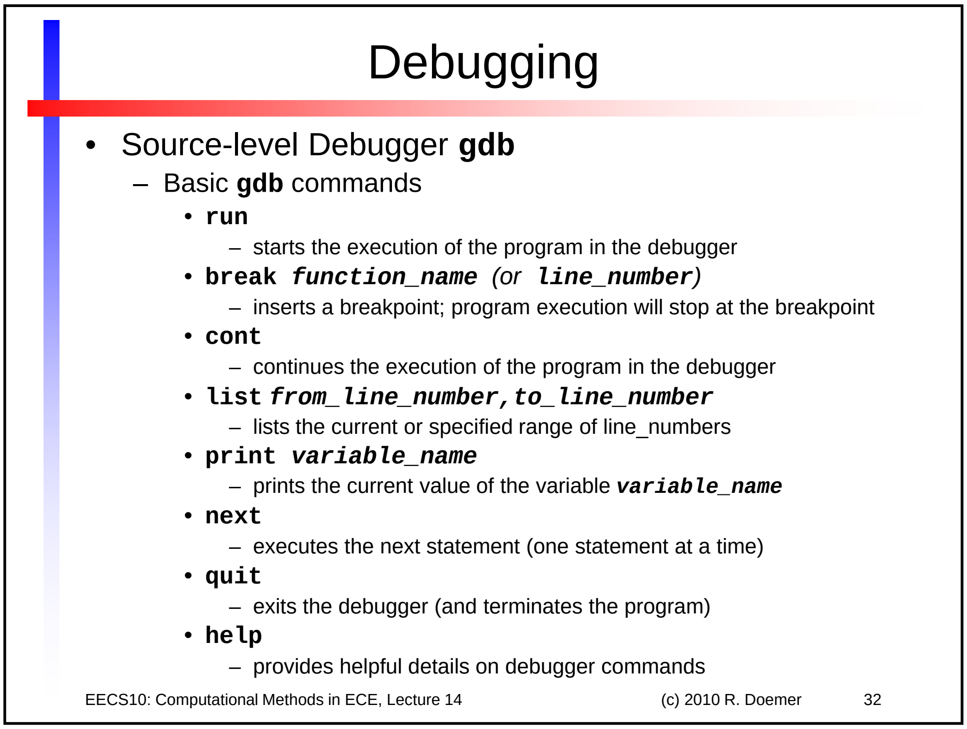
Scope Rules: Example

```
#include <stdio.h>

int square(int a);
int add_y(int x);

int x = 5,
    y = 7;

int square(int a)
{   int s;

    s = a * a;
    return s;
}

int add_y(int x)
{   int s;

    s = x + y;
    return s;
}

int main(void)
{   int z;

    z = square(x);
    z = add_y(z);

    printf("%d\n", z);
    return 0;
}
```

Shadowing!
In nested scopes,
inner scope takes precedence!

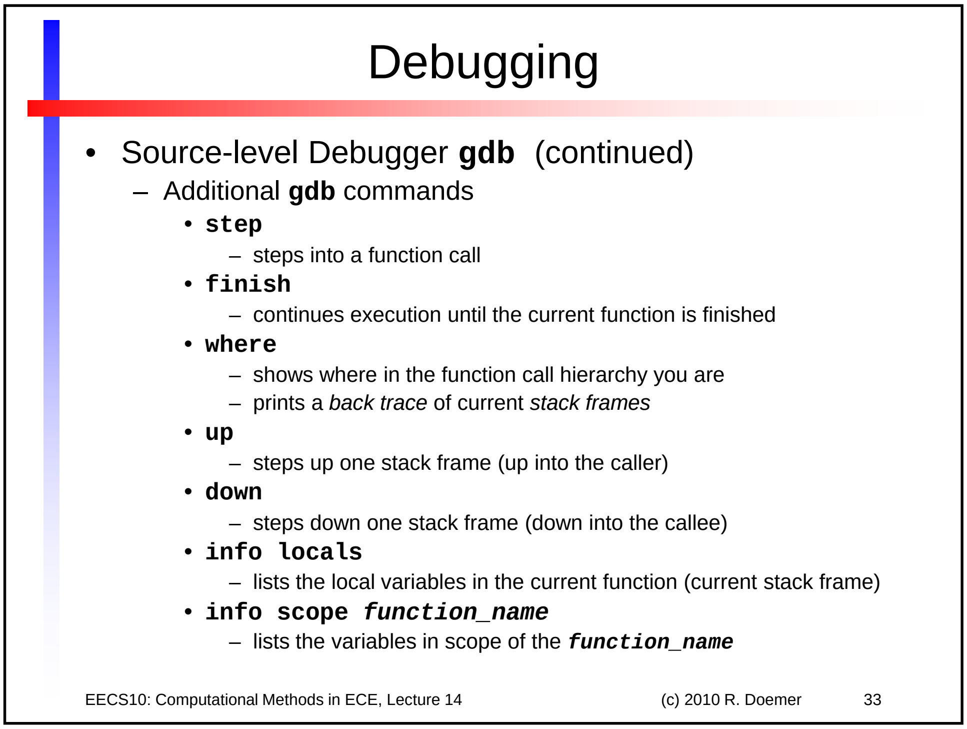
Scope of global variable **x**

Scope of parameter **x**



Debugging

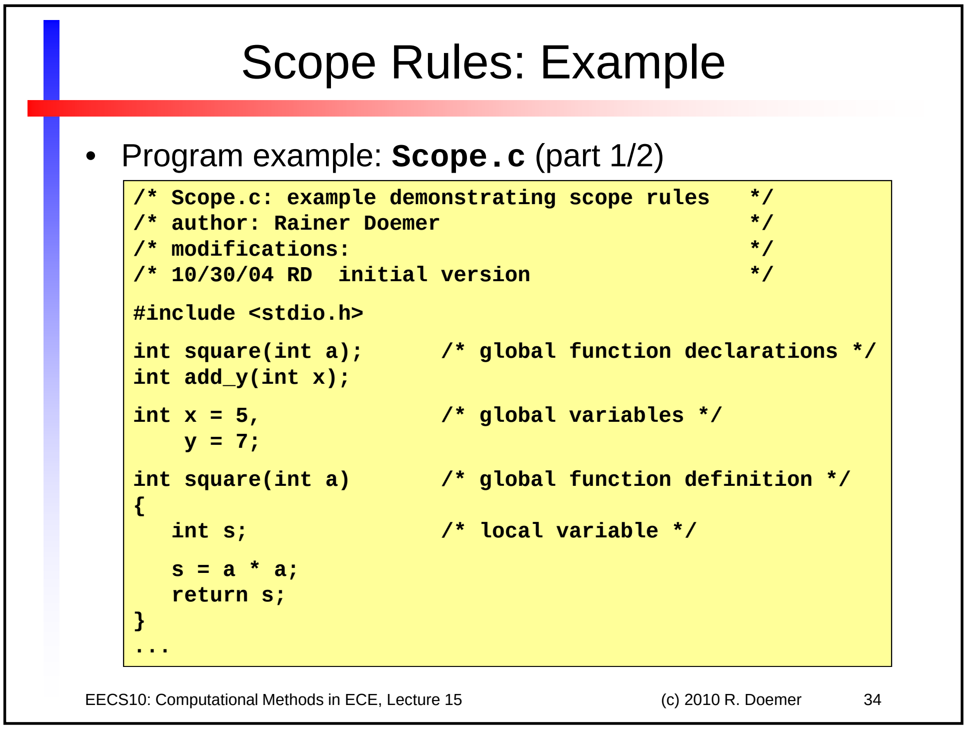
- Source-level Debugger **gdb**
  - Basic **gdb** commands
    - **`run`**
      - starts the execution of the program in the debugger
    - **`break`** ***`function_name`*** *(or* ***`line_number`****)*
      - inserts a breakpoint; program execution will stop at the breakpoint
    - **`cont`**
      - continues the execution of the program in the debugger
    - **`list`** ***`from_line_number,to_line_number`***
      - lists the current or specified range of 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



Debugging

- Source-level Debugger **gdb** (continued)
  - Additional **gdb** commands
    - **`step`**
      - steps into a function call
    - **`finish`**
      - continues execution until the current function is finished
    - **`where`**
      - shows where in the function call hierarchy you are
      - prints a *back trace* of current *stack frames*
    - **`up`**
      - steps up one stack frame (up into the caller)
    - **`down`**
      - steps down one stack frame (down into the callee)
    - **`info locals`**
      - lists the local variables in the current function (current stack frame)
    - **`info scope`** ***`function_name`***
      - lists the variables in scope of the ***`function_name`***

EECS10: Computational Methods in ECE, Lecture 14 (c) 2010 R. Doemer 33

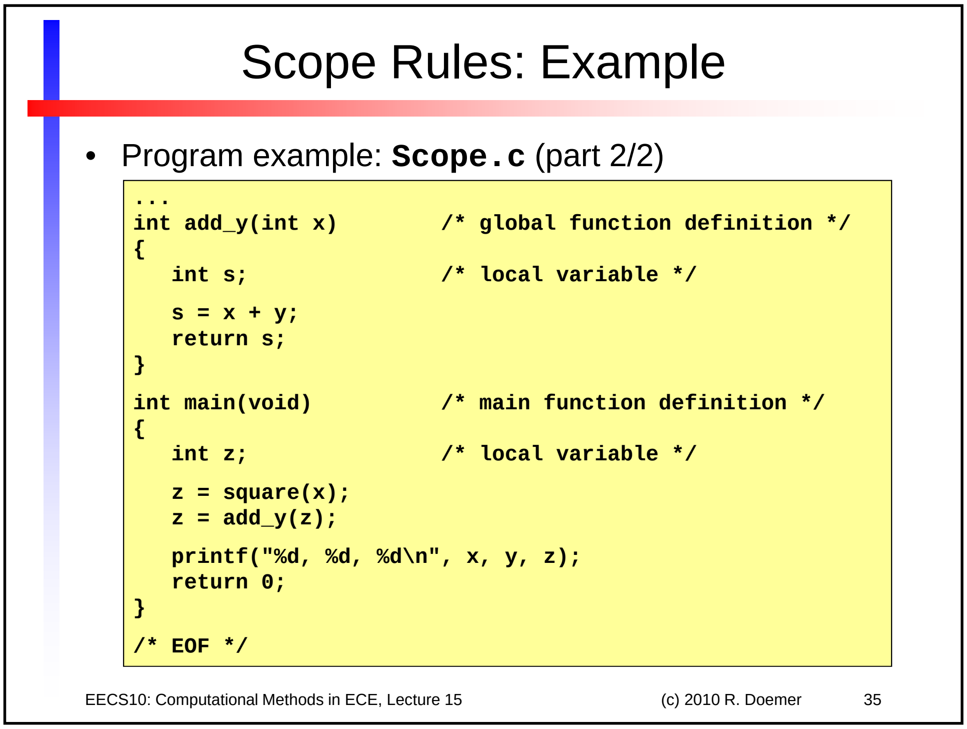


Scope Rules: Example

- Program example: **`Scope.c`** (part 1/2)

```
/* Scope.c: example demonstrating scope rules   */
/* author: Rainer Doemer                         */
/* modifications:                                */
/* 10/30/04 RD  initial version                  */

#include <stdio.h>

int square(int a);      /* global function declarations */
int add_y(int x);

int x = 5,              /* global variables */
    y = 7;

int square(int a)       /* global function definition */
{
   int s;               /* local variable */

   s = a * a;
   return s;
}
...
```

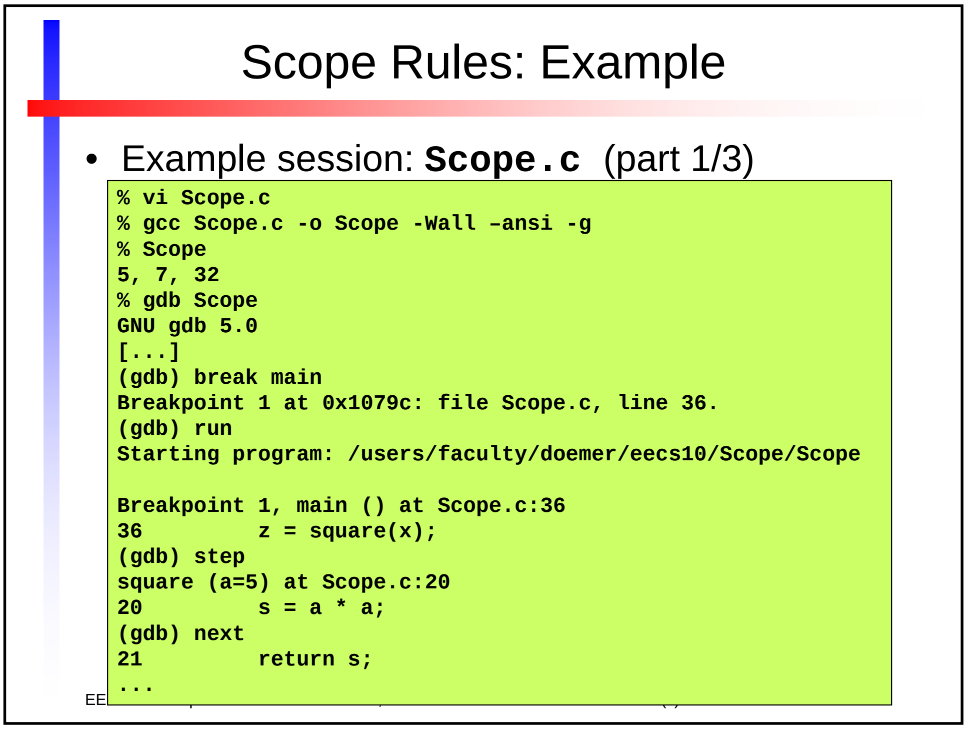
EECS10: Computational Methods in ECE, Lecture 15 (c) 2010 R. Doemer 34



Scope Rules: Example

- Program example: **`Scope.c`** (part 2/2)

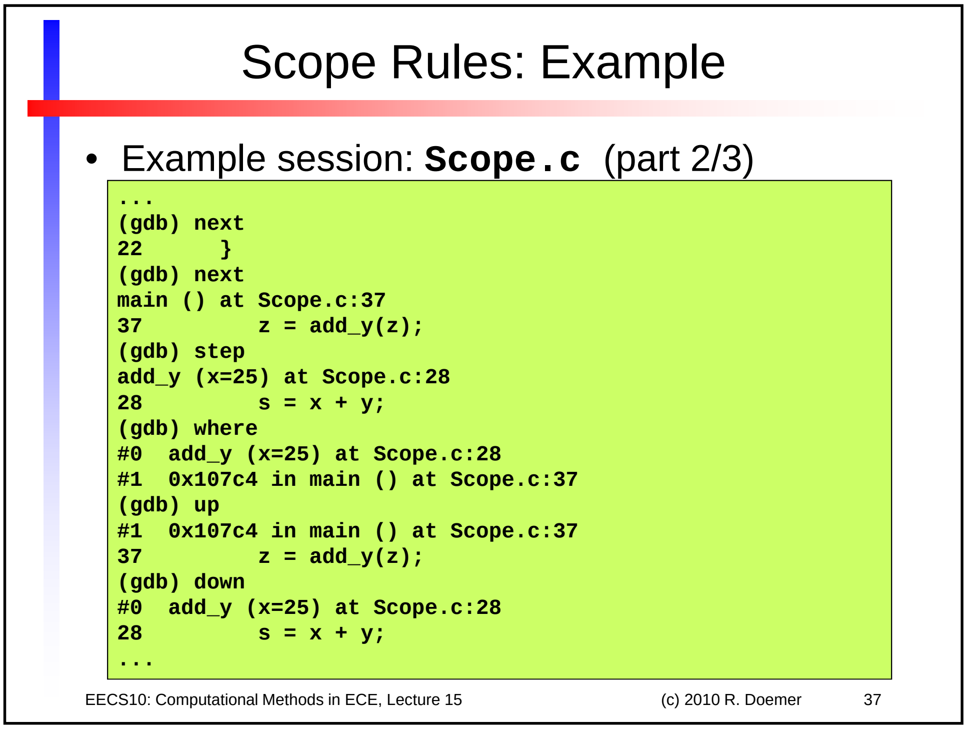
```
...
int add_y(int x)        /* global function definition */
{
   int s;               /* local variable */

   s = x + y;
   return s;
}

int main(void)          /* main function definition */
{
   int z;               /* local variable */

   z = square(x);
   z = add_y(z);

   printf("%d, %d, %d\n", x, y, z);
   return 0;
}

/* EOF */
```



Scope Rules: Example

- Example session: **`Scope.c`** (part 1/3)

```
% vi Scope.c
% gcc Scope.c -o Scope -Wall –ansi -g
% Scope
5, 7, 32
% gdb Scope
GNU gdb 5.0
[...]
(gdb) break main
Breakpoint 1 at 0x1079c: file Scope.c, line 36.
(gdb) run
Starting program: /users/faculty/doemer/eecs10/Scope/Scope

Breakpoint 1, main () at Scope.c:36
36          z = square(x);
(gdb) step
square (a=5) at Scope.c:20
20          s = a * a;
(gdb) next
21          return s;
...
```

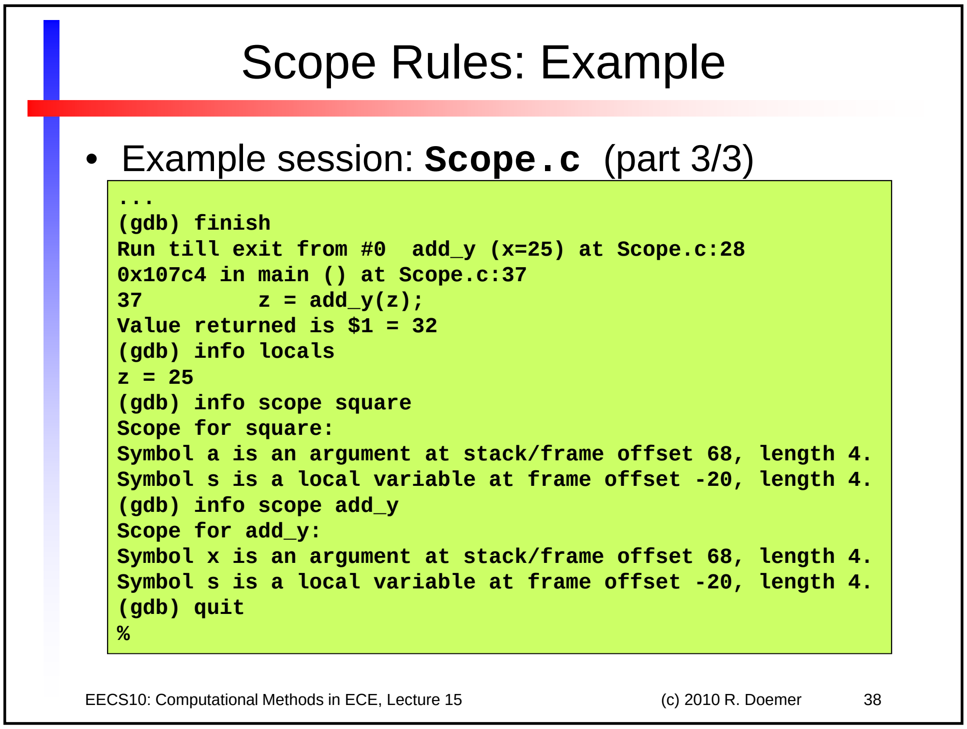


Scope Rules: Example

- Example session: **`Scope.c`** (part 2/3)

```
...
(gdb) next
22      }
(gdb) next
main () at Scope.c:37
37         z = add_y(z);
(gdb) step
add_y (x=25) at Scope.c:28
28         s = x + y;
(gdb) where
#0  add_y (x=25) at Scope.c:28
#1  0x107c4 in main () at Scope.c:37
(gdb) up
#1  0x107c4 in main () at Scope.c:37
37         z = add_y(z);
(gdb) down
#0  add_y (x=25) at Scope.c:28
28         s = x + y;
...
```

EECS10: Computational Methods in ECE, Lecture 15 (c) 2010 R. Doemer 37



Scope Rules: Example

- Example session: **`Scope.c`** (part 3/3)

```
...
(gdb) finish
Run till exit from #0  add_y (x=25) at Scope.c:28
0x107c4 in main () at Scope.c:37
37         z = add_y(z);
Value returned is $1 = 32
(gdb) info locals
z = 25
(gdb) info scope square
Scope for square:
Symbol a is an argument at stack/frame offset 68, length 4.
Symbol s is a local variable at frame offset -20, length 4.
(gdb) info scope add_y
Scope for add_y:
Symbol x is an argument at stack/frame offset 68, length 4.
Symbol s is a local variable at frame offset -20, length 4.
(gdb) quit
%
```

EECS10: Computational Methods in ECE, Lecture 15 (c) 2010 R. Doemer 38