Ordinary Differential Equations 2

CS 111: Introduction to Computational Science

Spring 2019 Lecture #15

Ziad Matni, Ph.D.

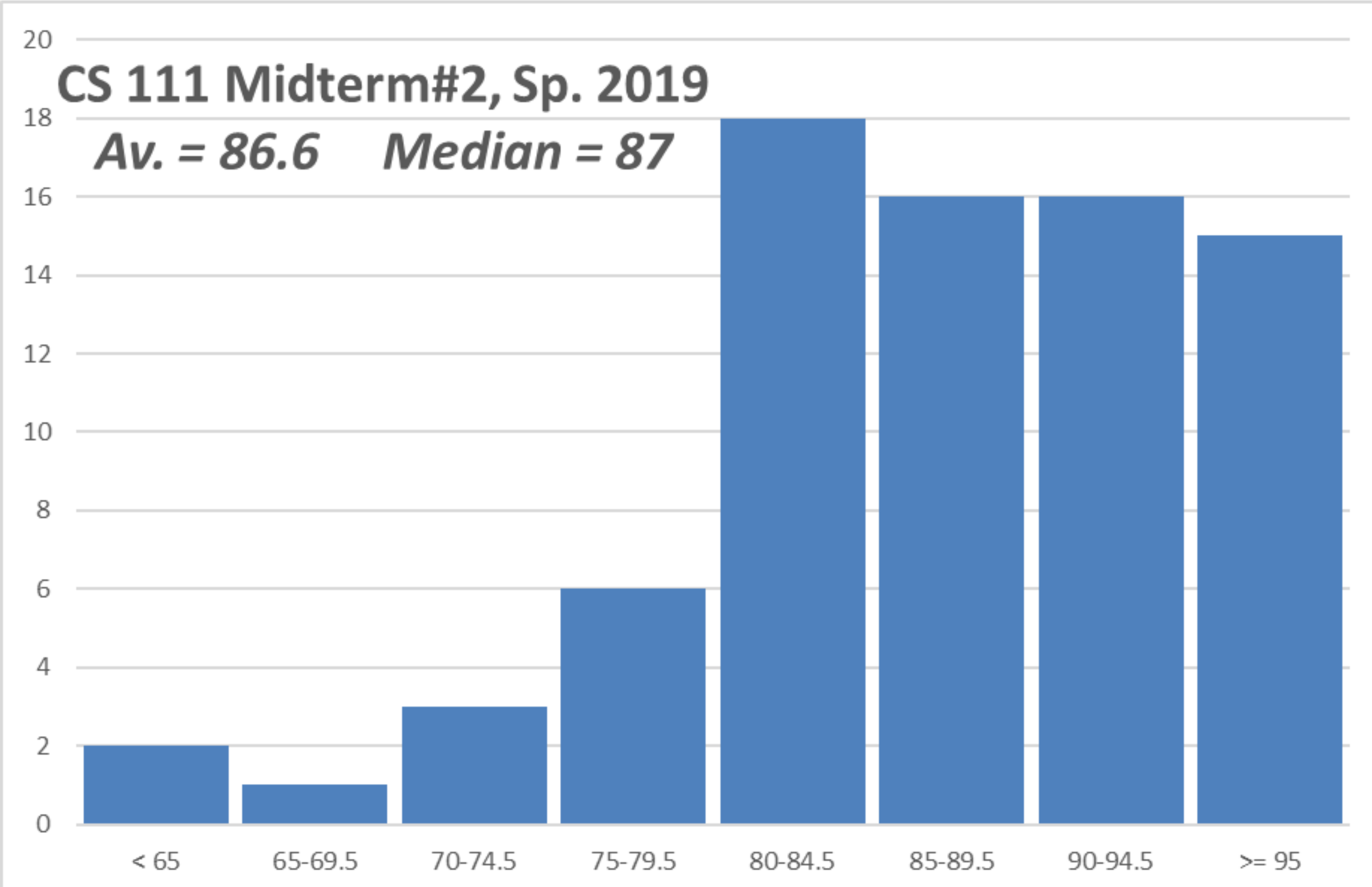
Finals Week

- Dr. Matni will have office hours on finals week

Monday 2:30 pm – 4:00 pm

Administrative

- Homework #7 (last one)
  - Due **WEDNESDAY (6/5) @ 6:00 pm**
- Midterm #2 Grades are Up!
  - Or will be soon after class…
  - Average and median both around 87%
  - To review: same arrangement as with Midterm #1


CS 111 Midterm#2, Sp. 2019
Av. = 86.6 Median = 87
20
18
16
14
12
10
8
6
4
2
0
< 65
65-69.5
70-74.5
75-79.5
80-84.5
85-89.5
90-94.5
>= 95

Recall: Python Function: `solve_ivp()`

- Found in the `scipy` module (in `scipy.integrate`)
  - Solves an ODE with initial conditions

- `solve_ivp(fun , t_span, y0, method)`
  - **`fun`** definition of the function to solve
  - **`t_span`** range of t
  - **`y0`** initial value $y_0 = y(t_0)$
  - **`method`** algorithmic approach
  - *There are other options that we can ignore to default*

Euler's Method

- A numerical method to approximate an ODE solution

- Comes from the identity:

$$\frac{dy}{dt} = f(t, y)$$

So:

$$\frac{y_{n+1} - y_n}{h_n} = f(t_n, y_n)$$

See blackboard…

Euler's Method

- It's prone to giving larger errors if the step size isn't small enough
  - i.e. the error is proportional to the step size

- It's an *explicit method* b/c it only uses information at time t_n to advance the solution to time t_{n+1}
  - This has implications for stability of this method

Python!

Your To-Dos

- **Homework 7**

- **In NVM, read:**
  - **Ch. 7, especially 7.4, 7.5, 7.6, 7.8**

</LECTURE>