CSE 12: Week 1 Discussion

1-05-2021

Focus: Zoom, Logistics, Debugging

Who Are We

Garo Adjoian,

BS in Computer Engineering from UCSD

Second Year Master's student

Contact Information

Email: gadjoian@ucsd.edu

Office Hours: Mon 1pm - 3pm PST

Zoom link: https://ucsd.zoom.us/my/**gadjoian**

Rebecca Kreitinger,

BS Computer Engineering from University of New Mexico

Current 2nd year Master's student

Contact Information

Email: rkreitin@eng.ucsd.edu

Office Hours: Thurs 11am - 1pm PST

Zoom link: https://ucsd.zoom.us/my/**rkreitin**

What will Discussion cover?

- Discussion is broken up into two parts
  - PA Video (~30 mins): released with the PA to provide an overview as well as related examples/notes
  - Live Discussion (~20 mins): prepared material to cover helpful topics as well as a time for open questions
- The live discussion is NOT meant for questions about how to start the PA that is due the next day.
  - You may ask conceptual questions about the PA and clarification questions, however, out of respect for your peers refrain from asking specific questions about your own code.
- The live discussion will include interactive aspects so that you can do knowledge checks.

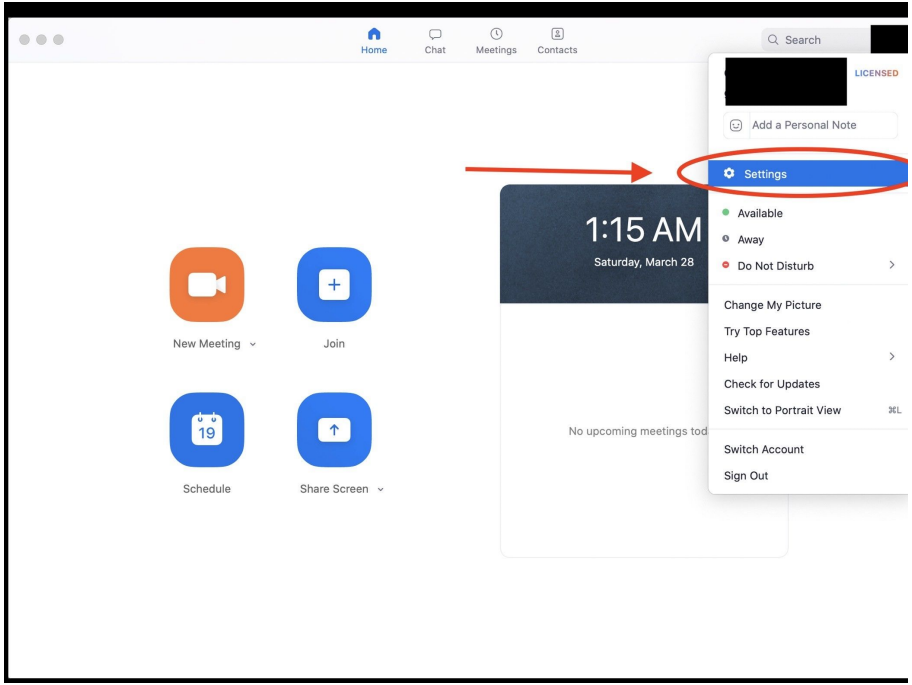
Any course related questions?

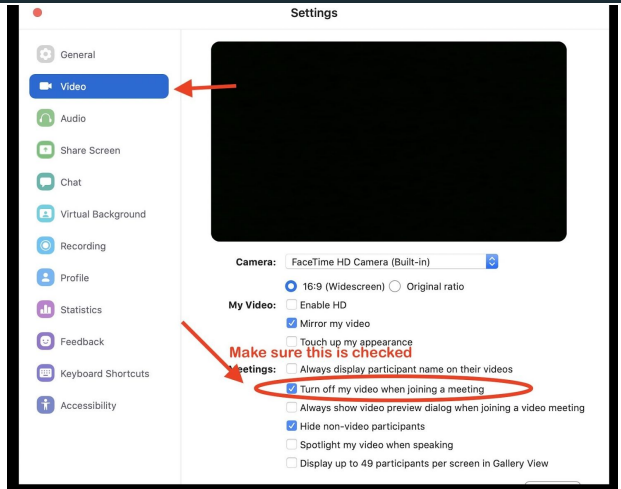
Programming Assignments will generally be released on Thursday morning and are due the following Wednesday at 11:59 PM

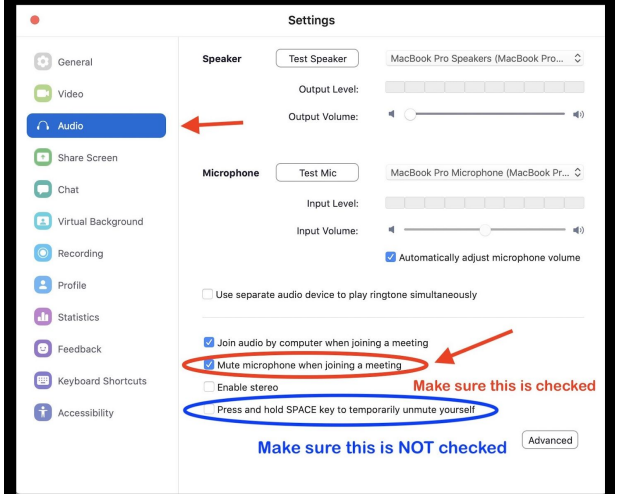
- PA1 will be released tomorrow, and is due next Wednesday, Jan 13th @ 11:59 PM
- Resources:
  - Optional ZyBook
  - Tutor hours
  - Professor/TA office hours

Zoom

Recommended Setting



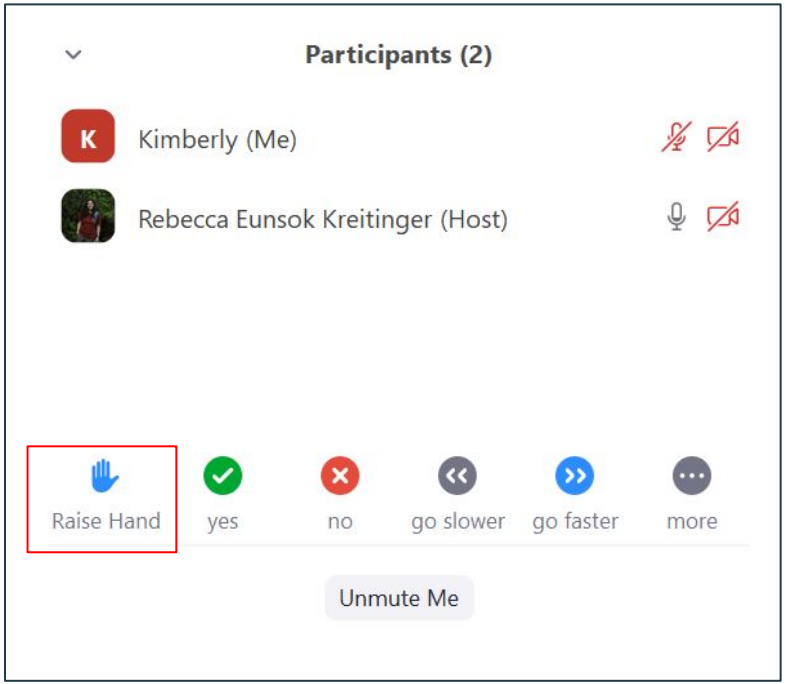




How do I ask questions?

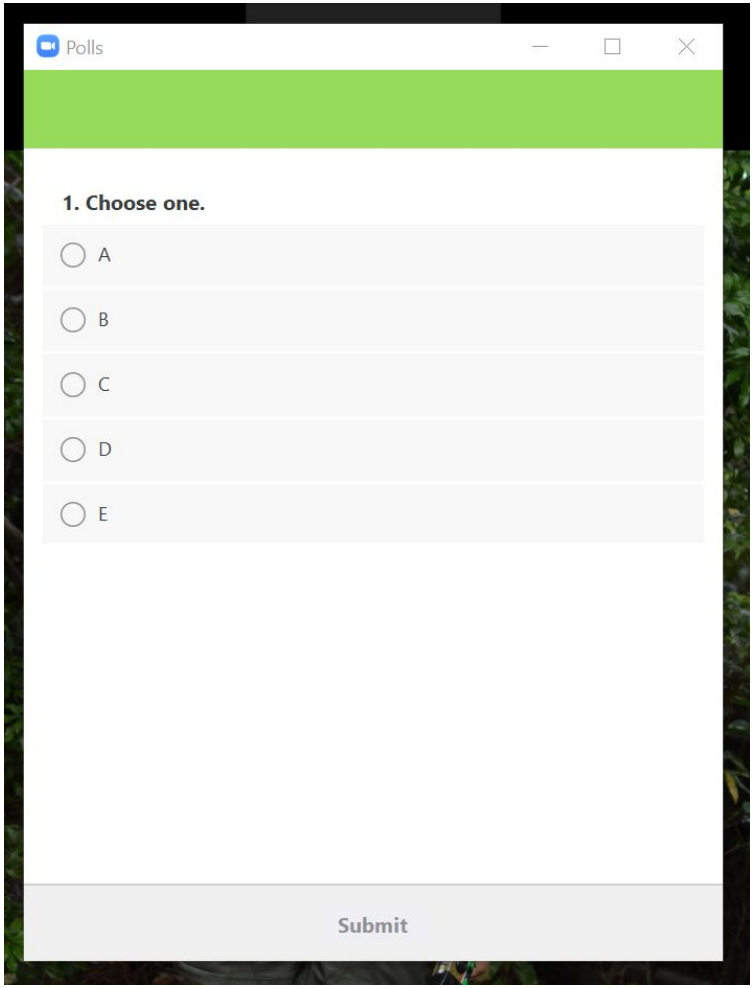
- Raise Hand
  - This will notify the host that you have a question and you can go ahead and unmute and ask
- Chat
  - I'll do my best to monitor the chat so that if you have any questions you can type them out and I'll respond. Other students can answer too!

We will also use the other features as well!



Knowledge Checks

- Polling (like iClicker questions)
  - When a question is asked, the poll will pop up. You will then choose your answer and click submit.
  - Note: you cannot change your answer after you submit.



Let’s practice!

How was your winter break?

A) Amazing!
B) Caught up on sleep
C) Eh
D) What winter break?

Course Logistics

Getting Help from a Tutor

- Submit a ticket with a description of your problem to the autograder queue to request help during tutor hours: https://autograder.ucsd.edu/login
- When a tutor accepts your ticket, join the zoom link in the ticket's comments
- Joining other students' zoom meeting might be considered as **academic integrity violation**
- Make sure to join the meeting within **2 minutes**
- Meeting with a tutor should take no longer than 5 minutes!
    - The purpose is to get unstuck, not solve the problem

Tips on online help:

- We recommend joining with audio (computer or phone)
- If you have technical difficulties, make sure to comment on the ticket to let the tutor know
- If you share your screen with your code, make sure line numbers are visible. This will help the both of you to refer to points in the code
- Sharing the entire desktop tends to be more reliable and is the best way to share your screen

Debugging

Types of Errors

- Compile Errors
- Runtime Errors
- Logic Errors

Compiler Errors

- ***Syntax error***
  - Error in usage of Java
  - Detected by the compiler
  - A program with compilation errors cannot be run
- ***Syntax warning***
  - Warning message generated by the compiler
  - The program can be run

Compiler Errors

- Very common (but sometimes hard to understand). Examples of syntax errors:
  - Forgetting a semicolon
  - Leaving out a closing bracket }
  - Re-declaring a variable
  - Others?

Compiler Errors

- Hints to help find/fix compiler errors:
  - Compiler errors are cumulative: when you fix one, others may go away
  - Read the error messages issued by the compiler!
  - Realize that the error messages from the compiler are often (seemingly) not very helpful
  - The compiler does not know what you intended to do, it merely scans the Java code

Runtime Errors

- ***Runtime error***: program runs but gets an *exception* error message
- Program may be terminated
- Runtime errors can be caused by
  - Program bugs
  - Bad or unexpected input
  - Hardware or software problems in the computer system

Runtime Errors

- Very common runtime errors are:
  - null reference (NullPointerException)
    - no object is referenced by the reference variable, i.e. it has the value null
  - array index out of bounds (ArrayIndexOutOfBoundsException)
  - Running out of memory
  - e.g. from creating a new object every time through an infinite loop

Runtime Errors

- Hints to help find/fix runtime errors:
  - Check the exception message for the method and line number from which it came
  - Note that the line in the code that caused the exception may ***not*** be the line with the error
  - Example: consider the code segment

```
int [] nums = new int[10];
for (int j=0; j<=10; j++)
    nums[j] = j;
```

The exception will be at the line

```
    nums[j] = j;
```

but the error is in the *previous* line

Logic Errors

- **Logic error**: program runs but results are not correct
- Logic errors can be caused by:
  - incorrect algorithms

Logic Errors

- Very common logic errors are:
  - using == instead of the *equals* method
  - infinite loops
  - misunderstanding of operator precedence
  - starting or ending at the wrong index of an array
  - If index is invalid, you would get an exception
  - misplaced parentheses (so code is either inside a block when it shouldn't be, or vice versa)

Logic Errors

- Be careful of where you declare variables!
- Keep in mind the scope of variables
- Instance variables?
- Formal parameters?
- Local variables?
- Example:

```
private int numStudents; // an attribute, to be
                         // initialized in some method
    …
    public void someMethod(){
        int numStudents = …; // not the attribute!
        …
    }
```

Testing vs Debugging

- **Testing**: to identify any problems before software is put to use
  - *“Testing can show the presence of bugs but can never show their absence”.*
- **Debugging**: locating bugs and fixing them

Debugging Strategies

- Trace your code by hand
- Add main method to the class
- Add print statements to your code

Questions?