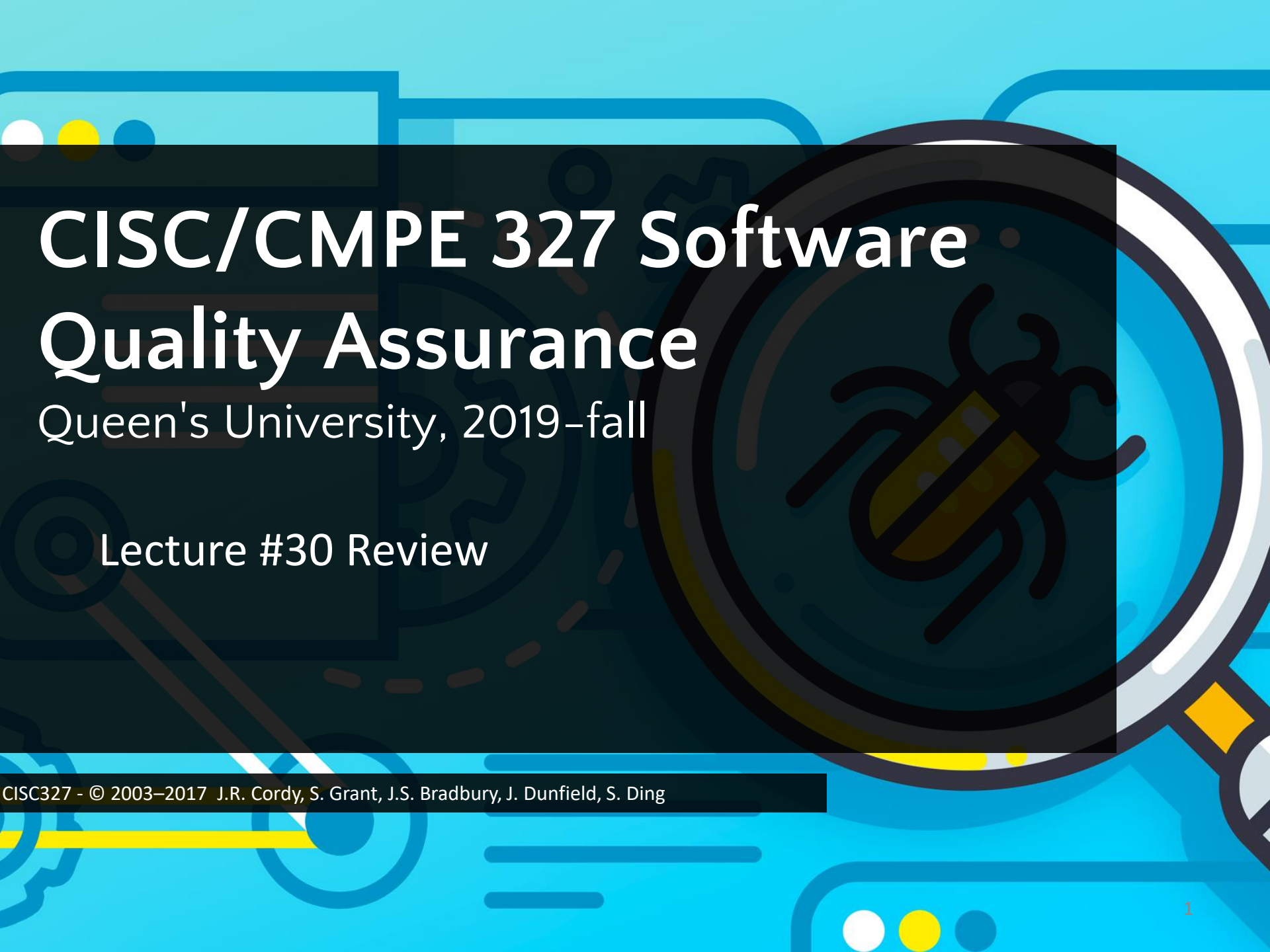

CISC/CMPE 327 Software Quality Assurance

Queen's University, 2019-fall

Lecture #30 Review

Reminder

- If you wrote the first three, and you're satisfied with your average, you don't have to show up
- Well, I still encourage you to do it

Likely topics on mini-exam #4

- From Lecture 19 (Inspection & Refactoring)
  - (XP code inspection, refactoring)
  - Know some examples of “code smells” and refactorings, especially “Don’t Repeat Yourself” and how to “factor out” duplicate code into a new method

Likely topics on mini-exam #4

- From Lecture 24-26 (formal inspections)
  - Know (roughly) the relative costs from “Cost of Fixing Errors” (the histogram!!!)
  - Know the roles of Moderator, Inspectors, Author:
    - I could ask you to match them up with “keeps paraphrasing at a reasonable pace”, “paraphrase the code”, “clarifies code when asked”, etc.
    - I could describe a flawed Bogosys inspection process and ask you what’s wrong with it

Likely topics on mini-exam #4

- From Lecture 24-26 (formal inspections)
  - Know (roughly) the defect classification (critical, Severe, Moderate, Minor
  - Example items on code checklist

Likely topics on mini-exam #4

- From Lecture 27
  - Basics of measurement

Likely topics on mini-exam #4

- From Lecture 28
  - difference between faults and failures
  - defect density
    - number of defects found, not the "real" number of defects (which is beyond our mortal knowledge)
    - relationship between faults and failures
  - software size: what's wrong with using (S)LOC?

Likely topics on mini-exam #4

- From Lecture 29
  - COCOMO = Constructive Cost Model
  - Effort = a (Size)^{b}
  - Almost meaningless if Size is in LOC
  - Different measure of size: function points
    - you don't need to memorize specific elements ("logical master files", etc.) or the coefficients on slide 19, but you should understand the basic idea and why function points are a better measure than LOC