Updates

- Homework:
  - Homeworks #3/4 graded on canvas
  - Homeworks #5/6/7/8 due Sunday night @ 11pm
- Homework grading
  - Points are mostly for effort
  - Points taken off if:
    - You didn’t answer all parts of a question
    - You didn’t follow directions for a question
    - You didn’t cite/explain your work
    - You’re obviously phoning it in
  - Read comments on canvas, even if you got full points
  - Full points does not mean you got the right answer
    - We’ll try to cover some common issues in recitation
Updates

• Projects:
  – Project 2 graded on canvas
  – Project 3 due tonight @ 11:00 PM
    • You can choose to skip two checklist items
  – Project 4 released, due 2/9 @ 11:00 PM
Updates

• Be sure to fill out question sheet to receive attendance points today!

• Reminder to use staff email and office hours for questions
  – Email only for administrative questions – it’s hard to answer conceptual questions
  – If you can’t make office hours, send us a note ahead of time with your availability and we’ll try to work something out
Today

• Project #4
• MCC and SCC
• Homework discussion
Project #3 Questions?
Project 4

• Peer reviews
  – Your groups are assigned on canvas
  – You will learn about peer reviews in lecture on Monday – this recitation is just an overview

• For each review:
  – One person is the scribe
    • Person who’s code is under review
  – One person is the leader
    • By the end of four reviews, everyone has had a turn being leader

• Fix any issues that come up
Peer Review Checklist

• Fill one out for each review
Peer Review Reminders

• Inspect the item, not the author
• Don't get defensive
• Find but don't fix problems
• Limit meetings to two hours
• Keep a reasonable pace
• Avoid "religious" debates on style (While enforcing what the Project 3 checklist says)
What to review

• Go through code line by line
• Leader is in charge of
  – Saying when to move on to the next line
  – Reading through Project 3 style checklist out loud
• Prof Koopman’s review checklist is a good starting point
  – https://betterembsw.blogspot.com/2018/01/
  – Note that some items won’t apply to your code
Project #4 Questions?
McCabe’s Cyclomatic Complexity

- Measures complexity by counting branching logic
- Generally used as a guideline to how readable/testable/maintainable source is
- High complexity of a module indicates:
  - Deeply nested loops
  - Lots of branch statements
  - Functionality that is not factored out into modules
- For detailed example, see posted recitation 3 slides online
Applications to testing

• Code coverage testing:
  – “White box” testing (can see the code)
  – Want to test all logic
  – Branch coverage testing
  – Path coverage testing
Branch coverage

Test cases
{x=1, y=2}
{x = 3, y=4}
All branches covered
Path coverage

Test cases:

\{x=1, y=2\}
\{x=1, y=4\}
\{x=3, y=2\}
\{x=3, y=4\}

All paths tested
Cyclomatic Complexity and Testing

# of tests to achieve full branch coverage
\[ \leq MCC \# \leq \]
# of tests to achieve full path coverage*

*Unless some paths are impossible

- Gives a lower bound to achieving path coverage (so a high MCC means a lot of testing)
- Also gives notion of how many test cases are needed to obtain "sufficient coverage"
- See McCabe paper, Section VII for more discussion


Subprocesses

```c
for (i = 0; i < N; i++) {
    sub_process(i);
}
```

- Still has MCC of 2.
- Testing complexity in this module requires testing that the inside of the loop is reached, and testing the outside of the loop.
- Complexity of sub_process() is computed separately because it can be unit-tested separately.
- Takeaway: refactor code to reduce MCC.
- **Do this for project 3!!**
MCC and SCC

• McCabe’s cyclomatic complexity
  – Counts # of if/while/for conditionals in the code
• Strict cyclomatic complexity
  – Includes +1 for every condition within a branch
• \( \text{if} \ (a < 0 \ \&\& \ b > 0) \) adds +1 to MCC, +2 to SCC
• You’ll encounter this in the homework
These have same MCC, but which is more readable?

if (isInBounds(x)) {
    if (y > 0) {
        if (prime factor(x, y)) {
            if (COMPUTE_FLAG) {
                do_function(x, y)
            }
        }
    }
}

if (!isInBounds(x))
    return false;

if (y <= 0)
    return false;

if (!COMPUTE_FLAG)
    return false;

do_function(x, y)
Spaghetti Code

• High MCC is just one indicator of spaghetti code
• Others:
  – Global variables
  – Copy-pasted code
  – Obfuscated variable names

background image: https://imgflip.com/memegenerator/Eminem
Questions?
Homework Discussion

- 3-4a. The US government does NOT regulate software safety in cars other than, in principle, via recalls after a statistically compelling number of mishaps have occurred. Should the US Government regulate software safety (e.g., by requiring compliance to the ISO 26262 safety standard?) Why or why not?
- 3-4b. Read this article about a man released from jail after claiming a crash was due to unintended acceleration. Note the model year of the car. Do you think that releasing the driver from jail was the right thing to do?
- 3-4c. Read this article about another unintended acceleration class action lawsuit. What do you think about the car manufacturer's statement that the incident peak in 2005 is due to driver error?
- 3-4d. Read this article about a police officer involved in a fatal crash. Why do you think the police officer changed his story from "sudden acceleration" to another explanation?
- 3-4e: Hypothesize that you are driving a vehicle and hit a pedestrian. You claim it was UA. The vehicle data recorder says your foot was on the accelerator. The local government prosecutes you for the death because the police report says the only possible reason is pedal misapplication due to incompetent driving. What do you do?
- 3-4f: Should the software designers of vehicle engine controls be held personally responsible for defects that result in fatalities?
Homework Discussion (Roster)
Bonus MCC/SCC Slides
Determining MCC

- Turn code into graph and count # of loops + 1
- Notation to turn code into graph:
  - Circle for statement(s) with no branching logic
  - Diamond for branching logic

http://www.ece.cmu.edu/~ece642/lectures/06_spaghettiocode.pdf
Common constructions

- Sequential, no branching:

  ![Diagram](image1)
  
  MCC = 1

- IF statement:

  ![Diagram](image2)
  
  MCC = 2

- For loop:

  ![Diagram](image3)
  
  MCC = 2
Switch/multi-case IF:

```java
if (x == 1) {
    ...
} else if (x == 2) {
    ...
} else {
    ...
}
```

MCC = 3
IF within FOR:

```java
for (i = 0; i < 10; i++) {
    if (i % 2 == 0) {
        ...
    } else {
        ...
    }
}
```

MCC = 3
Nested Loop:

```
for (i = 0; i < N; i++) {
    for (j = 0; j < N; j++) {
        ...
    }
}
```
More complex code

• Count total number of if/for/while statements and add 1
• Tools available to compute for you

https://betterembsw.blogspot.com/2014/06/avoid-high-cyclomatic-complexity.html
Pitfalls of MCC

• Not an exact # of test cases to run (testing generally strives for path coverage)
• Switch statements increase MCC a lot but are sometimes necessary for things like message translation
  – Prof Koopman’s blog post:  
    https://betterembsw.blogspot.com/2014/06/avoid-high-cyclomatic-complexity.html
• Decent measure of code comprehension, but not the only one