Java & Testing

Things to have been doing

- Join UIUC CS 126 on Piazza
- Get an iClicker registered
- Get a GitHub account
- Install IntelliJ on your laptop (students can get the Ultimate version for free)
- Review course Policies:
  - https://courses.engr.illinois.edu/cs126
- Code Review Survey
  - https://doodle.com/poll/u7qcsvfhsaw7kqez

IT'S TIME FOR ANOTHER...
GOOD IDEA
BAD IDEA

Your friend offers to send you the code for the MP so you can hang out.

A. Good Idea

B. Bad Idea

Your friend is having difficulties starting the assignment, so you give them a general overview of what you did.

A. Good Idea
B. Bad Idea

Your friend is having difficulties starting the assignment, so you show them the first few methods you wrote.

A. Good Idea

B. Bad Idea

Your friend just does NOT have time this week for CS 126 and they ask you to give them your code. Don't worry they'll change it up a little.

A. Good Idea

B. Bad Idea

You get stuck on how to do something on an MP, so you google it and copy ~30 lines from the very helpful first StackOverflow post.

A. Good Idea

B. Bad Idea

You get stuck on how to do something on an MP, so you google it and add a comment citing the code you took from StackOverflow. You can explain what the code does in code review.

A. Good Idea

B. Bad Idea

You get stuck on how to do something on an MP, so you google it and use code from an past student's incredibly handy GitHub repository! That student's intro programming assignment surely looks impressive to recruiters!

A. Good Idea

B. Bad Idea

You think you're smarter than course staff, so you "borrow" your friend's code and attempt to modify it enough so that you don't get caught. (No, it never works)

A. Good Idea
B. Bad Idea

Why Test?

- **Improve quality - find faults**
- **Measure quality**
  - Prove there are no faults? (Is it possible?)
  - Determine if software is ready to be released
  - Determine what to work on
  - See if you made a mistake
- **Learn the software**

Testing vs. Debugging

- Testing is detecting errors
- Debugging is a means of diagnosing and correcting the root causes of errors that have already been detected.

Types of testing

- Unit Testing
- Component Testing
- Integration Testing
- Regression Testing
- System Testing

Types of testing

- **Unit Testing**

  The execution of a complete class, routine, or small program that has been written by a single programmer or team of programmers, which is tested in isolation from the more complete system.
- **Component Testing**
- **Integration Testing**
- **Regression Testing**
- **System Testing**

Two Approaches to Testing

- **Black box testing:**

- **White box testing:**

Two Approaches to Testing

- **Black box testing: a.k.a. Behavioral Testing**
  - is a software testing method in which the internal structure/design/implementation of the item being tested is <u>not</u> known to the tester.

- **White box testing: a.k.a. Structural Testing**
  - exploits knowledge of the internal structure/design/implementation of the item being tested, generally to ensure good code coverage and test potential corner cases in the implementation.

What kind of tests?

- **Manual**
  - Good for exploratory
  - Good for testing GUI
  - Manual regression testing is BORING
- **Automatic**
  - Test is a program
  - Test is created by a tool that records user actions
  - The only way to make testing efficient as well as effective is to automate as much as possible

Junit

- **Open source Java testing framework for automated testing**
- **Widely used in industry**
- **Features:**
  - Assertions for testing expected results
  - Test features for sharing common test data
  - Test suites for easily organizing and running tests
  - Graphical and textual test runners
- **Primarily for unit and integration testing, not system testing**

Definitions

- **Which kind of testing is “specification testing”**

A) Black box testing

B) White box testing

Black box testing exercise

- **A program needs to be developed so that given an integer value**
  - it outputs 0 when the integer value is 0
  - it outputs 1 when the integer value > 0
  - It outputs -1 when the integer value < 0

- **What would be your black box tests?** ***(How many do you need?)***

Bag of Testing Tricks

- **Equivalence Partitioning:** If two test cases flush out exactly the same errors, you need only one of them. How many different groups of inputs are there? Test each of them.
- **Error Guessing:** guesses about where the program might have errors, based on your experience/intuition
- **Boundary Analysis:** write test cases that exercise the boundary conditions, looking for ‘off-by-one’ errors.
- **Classes of Good Data:** Nominal cases (middle-of-the-road, expected values), minimum/maximum normal configuration, compatibility with old data
- **Classes of Bad Data:** Too little data (or no data), too much data, the wrong kind of data (invalid data), the wrong size of data, uninitialized data

Test First or Test Last? (Guess)

A) Test First (write tests before you write code)

B) Test Last (write tests after you write code)

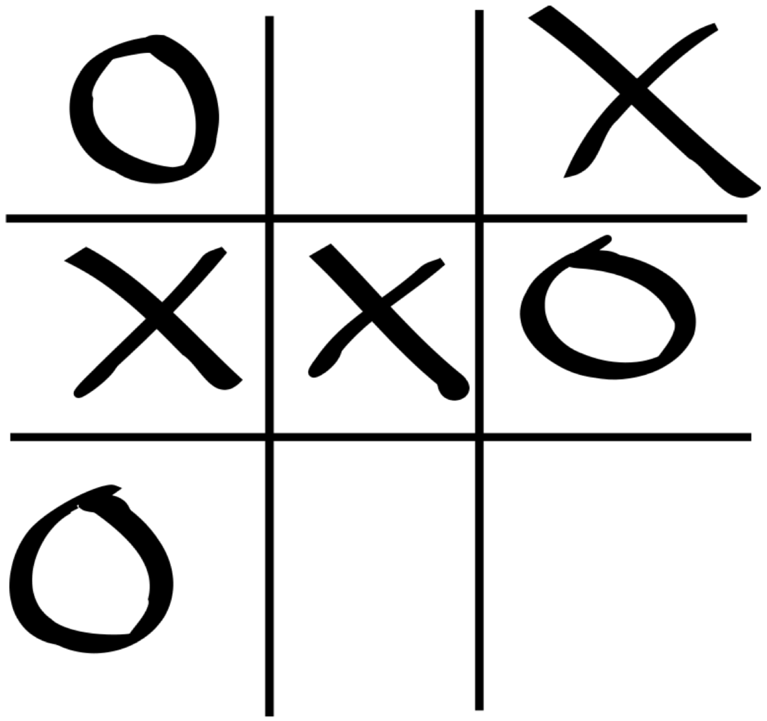
Test First

- **Detect defects earlier (cheaper)**
- **Forces understanding of the requirements before you start coding**
- **Identifies problems with the requirements earlier**
- **No more effort to test first**

- **A tenet of eXtreme Programming (XP)**
  - A design technique, not a testing technique
  - Doesn’t find bugs, but eliminates them
  - Doesn’t measure quality, but improves it

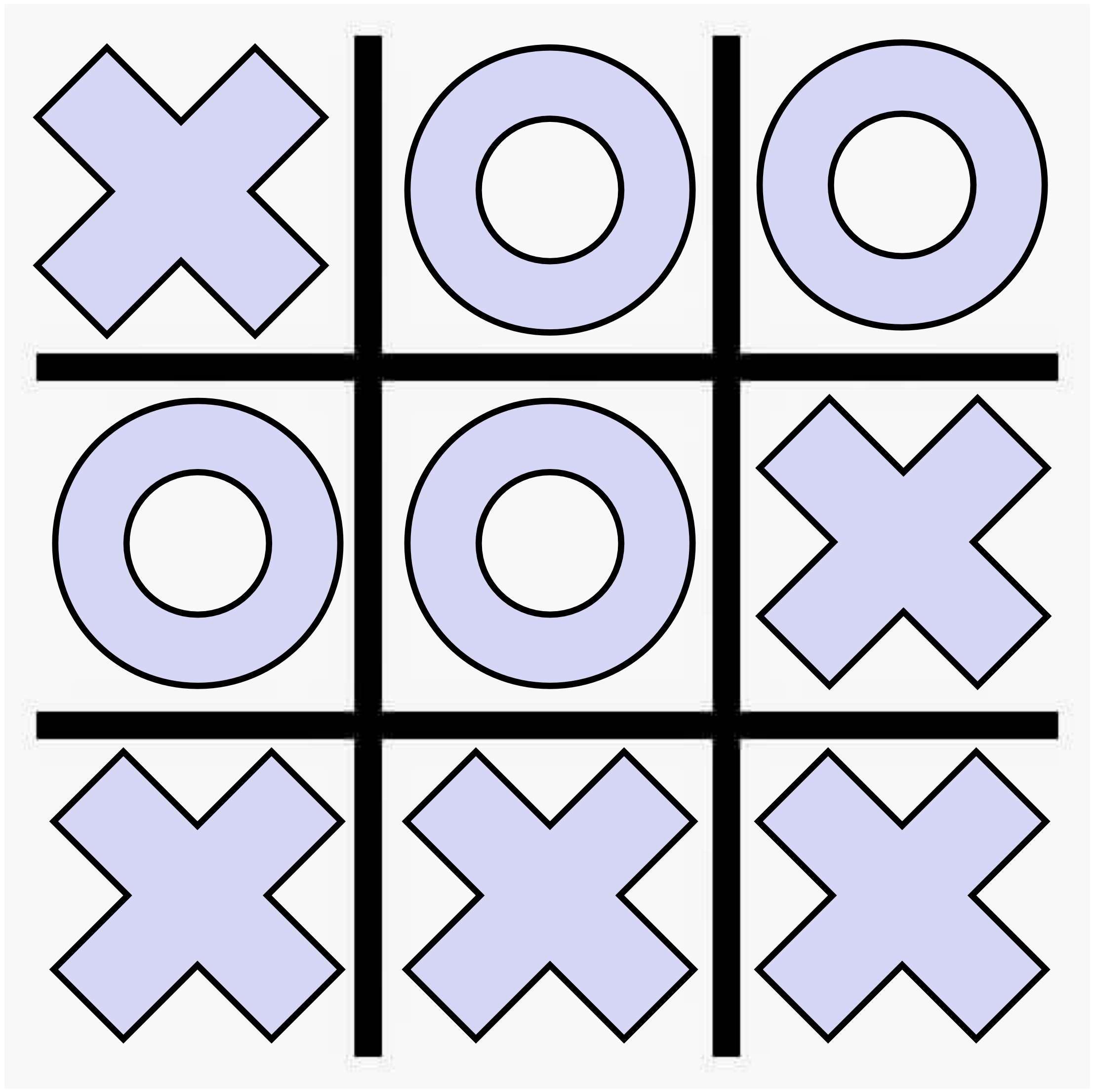
Tic-Tac-Toe

- https://en.wikipedia.org/wiki/Tic-tac-toe
- **Board description:**
  - A string with one character per board position
  - Case-insensitive
  - 3 vals: [x,X], [o,O], [all others]
  - o-XxxoO.z is the board shown



- **public enum Evaluation {**
  InvalidInput, NoWinner, Xwins, Owins, UnreachableState
  }

Evaluate Board



A) InvalidInput

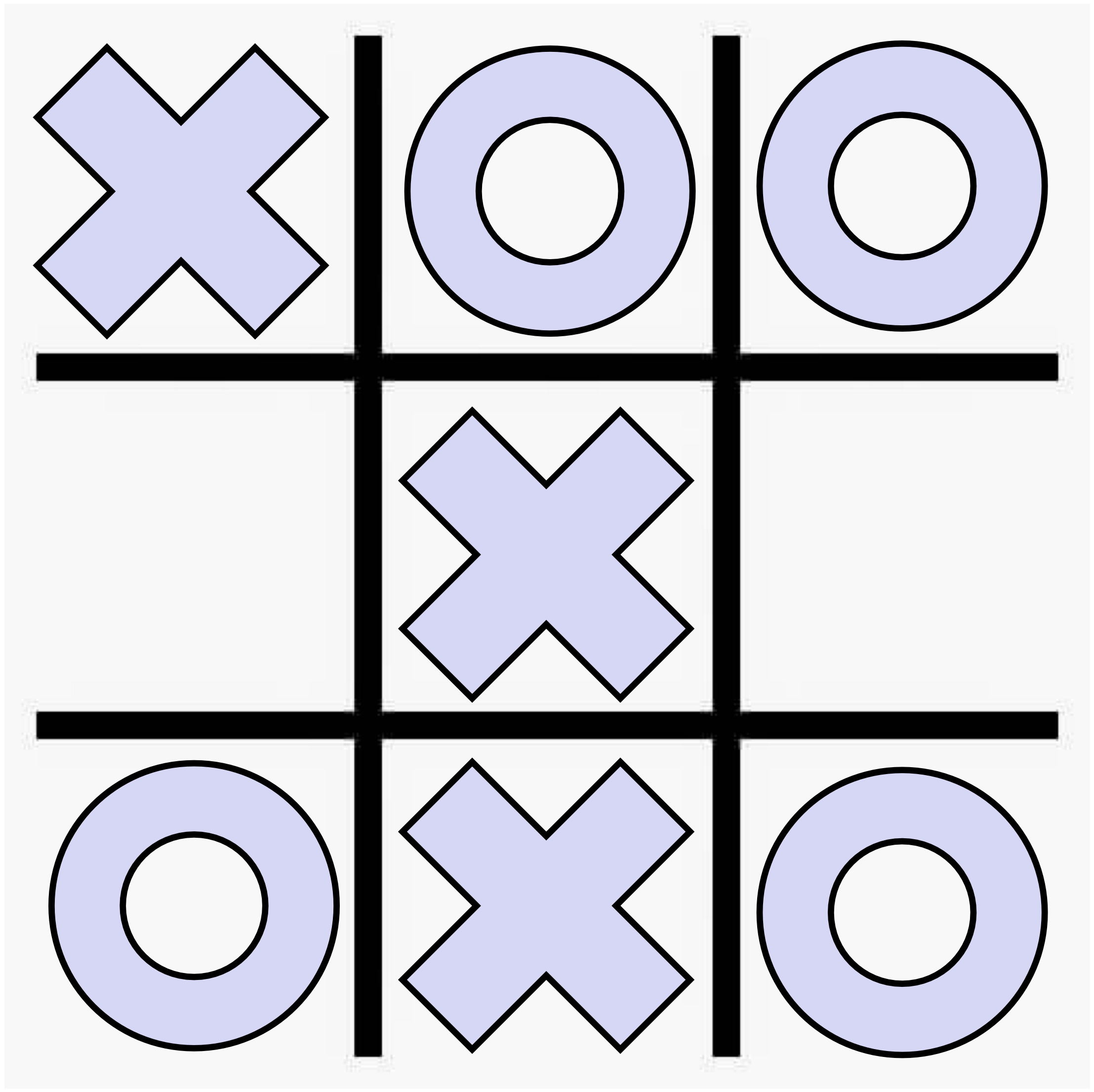
B) NoWinner

C) Xwins

D) Owins

E) UnreachableStat
e

Evaluate Board



A) InvalidInput
B) NoWinner
C) Xwins
D) Owins
E) UnreachableStat
e