Objectives

- Picasso
- Course evaluations

1

Review

- What is a design pattern?
  - ➤ What design patterns have we discussed?
    - What problems do they solve?
  - ➤ What design patterns are used in the Picasso project?
    - (This could vary by team)
- Why do we need to convert the input to postfix?
- What is our git workflow?

2

Review: Design Pattern

General reusable solution to a commonly occurring problem in software design

- Not a finished design that can be transformed directly into code
- Description or *template* for how to solve a problem that can be used in many different situations
  - ➤ "Experience reuse", rather than code reuse

3

Git Workflow

1. Switch to new branch
2. Update code, commit
3. Switch to main
4. Pull code
   - ➤ Get others' changes first
5. Merge your branch into main
6. Push your branch
7. Repeat
   - ➤ If switch back to an existing branch, merge main into that branch

4

Bugs in the News

- Five Thirty Eight web site
  - ➢ Best known as a polls/stats-based election modeling web site
- Day before the election: site showed Trump with a 99% percent chance of winning
  - ➢ Not what their model actually said
- Model weights non-poll data as a fraction that goes to zero on the day *before* the election
  - ➢ Caused some numbers to go to 0 in their model
    - Caused **divide-by-zero** bugs
- The if statement to guard against above was **off by one**
  - ➢ Kicked in on election day, not the day before the election

https://fivethirtyeight.com/features/politics-podcast-final-reflections-on-the-2020-campaign-before-election-day/

Nov 11, 2020   Sprenkle - CSCI209   Around 49:00 left   5

5

PICASSO

Nov 11, 2020   Sprenkle - CSCI209   6

6

Handling Minus Signs

- Is this happening to you?
  - ➢ x – y works but x-y doesn't?
- See FAQ about how to handle minus signs

7

Start Thinking About

- Reporting errors to users
  - ➢ Currently: in the output but users aren't going to see that
  - ➢ Helpful errors → translated for users
- Opening a file that contains an expression
- Handling new operations
  - ➢ Order of operations
  - ➢ Assignment statement
- Functions with multiple arguments, image names
- Extensions

8

Project Goals

- Everyone contributes significantly to the project
  - Has at least one part where they can say "I made this!"
  - Hopefully reflected in the contributions to the GitHub repository

- Everyone understands the code and its design
  - All of it. Well, 90% of it, at least at a high level

- Everyone feels valued as a team member

9

GitHub's Contributions Page

- Always some concern that your grade is based on lines of code written
  - Number of lines of code is not a good indicator of work or quality of code

- From GitHub: The email address used for the commits must be associated with your GitHub account
  - Check the name used in your commits

See FAQ for more info

10

Comparing Binary Operators

- Likely need to implement the equals method in various classes (e.g., Addition, Subtraction, …)
- Stop after you've written two
- Compare the methods
  - ➢ Is there a code smell? Refactor!

11

Final Implementation: Documentation

- You leave, I'm still here, trying to use [grade] your code
- Documentation
  - ➢ Extensions aren't always obvious
  - ➢ State in README
- Javadocs: Purpose of Java classes
  - ➢ Update comments
  - ➢ Auto-generated daily
  - ➢ Can be seen on the project web site

12

Final Implementation: Tagging

- Tag the final version
  - ➤ No compilation errors

13

Secondary Goals

- You're going to figure out that your final design isn't perfect—maybe not even good!
  - ➤ Fix more critical and/or smaller things
    - Refactoring!
  - ➤ Note larger things
    - analysis/post-mortem due at end of finals week

> Good judgment comes from experience.
> How do you get experience?
> **Bad judgment** works every time.

14

Final Project Notes

- Project Analysis: Individual
  - ➢ Understand teammates' design/code/parts
    - *At least* at a high level
  - ➢ Contents: Description, Planning, Status, Code Analysis, Collaboration, Future Work
    - Complete specification online

15

Project Planning

- Review project specifications
- Make sure you know what tasks are left
- Be agile!

16

Looking Ahead

- Course Evaluations
  - Due Monday, Nov 23
  - Incentive to fill out evaluations
    - If 60% fill out, 1% Extra Credit on "Individual programming and written homework assignments"
    - Additional 1% for every additional 10% (2 students) who complete
- Office Hours
  - Back to the drop-in office hours and by appointment

17