The Game of Pig for CS1
Game/puzzle exercises are fun, engaging
Overhead of complex rules can spoil a project
  - E.g. chess: en passant, castling
For educators good games/puzzles with simple rules can be pedagogical treasures
  - E.g. recursion: Towers of Hanoi, mazes
The Game of Pig
  - Jeopardy dice game
  - Rules in two sentences
  - Outstanding Fun per Source Lines of Code
The first player to reach 100 points wins.

On a turn, a player rolls a die repeatedly until:

- the player holds, scoring the sum of the rolls ("turn total"), or
- a 1 ("pig") is rolled, and there is no score change.

Example turns:

- roll 4, roll 5, roll 2, hold → add 4 + 5 + 2 = 11 to score
- roll 3, roll 6, roll 6, roll 1 → score remains the same
Player’s decision is always to *roll/hold*

- *Roll* – possibly increase turn total, or lose it
- *Hold* – definitely score current turn total
- Pig is the simplest of a class of *jeopardy dice games*; ancestor of *Pass the Pigs*

- Hold at 20 - a simple (suboptimal) strategy that maximizes expected points per turn
CS1 student hurdle: nested loops
  - Methodology: elaboration, milestones, stepping stones, bottom-up development

Text-based Pig game application proceeds by neat stages:
  - Hold at 20 turn: single loop
  - Hold at 20 or goal turn: add score, stop for win
  - Solitaire game: nest turn loop within game loop
  - Two-player game: toggle players/scores, factor out common code
  - Human vs. Computer game: replace random player strategy with human input
// Initialize variables
while () { // Turn not over
    // Roll die
    if () { // 1 rolled
        // Reset turn total and end turn
    }
    else {
        // Add roll to turn total
    }
}
final int HOLD_AT = 20;
Random random = new Random();
int turnTotal = 0;
boolean pigRolled = false;
while (!pigRolled && turnTotal < HOLD_AT) {
    int roll = random.nextInt(6) + 1;
    System.out.println("Roll: " + roll);
    pigRolled = (roll == 1);
    if (pigRolled)
        turnTotal = 0;
    else
        turnTotal += roll;
}
System.out.println("Turn total: " + turnTotal);
import random
holdAt = 20
turnTotal = 0
pigRolled = False
while not pigRolled and turnTotal < holdAt:
    roll = random.randint(1, 6)
    print('Roll:', roll)
    pigRolled = (roll == 1)
if pigRolled:
    turnTotal = 0
else:
    turnTotal = turnTotal + roll
print('Turn total:', turnTotal)
Monte Carlo Explorations

In this simple context, there are interesting questions that can be explored with Monte Carlo simulation:

- Probability distribution of hold-at-20 turn scores?
- Average turns in solitaire game?
- First-player advantage in two-player games?

In each case, students nest previous work within trials loop and collect statistics.
Object-Oriented Exercises
Event-driven GUI Exercise

- Dice images supplied in PNG, JPG formats
- Component specification (labels, buttons, etc.), not layout
- Functional specification
- *Keep Pace and End Race* policy:
  - Either score $\geq 71$? Go for goal!
  - Otherwise, hold at $21 + \text{round}((\text{your score} - \text{my score})/8)$
Links to paper and assignments making good use of Pig in:

- **Artificial Intelligence**: reinforcement learning for optimal policy computation
- **Networking**: fun client-server exercise with little overhead
- **Scientific Visualization**: visualizing the optimal policy roll/hold boundary with Marching Cubes Algorithm
Pig is the simplest of a family of jeopardy dice games.
- Many variants exist (e.g. Piglet, Two-dice Pig, Pass the Pigs™, Hog)
- A link to a catalog of all known variants is provided
- Why? So assignment variants can be easily created to avoid plagiarism.
Pig offers:
- Excellent Fun/SLoC ratio
- CS1 exercises for ...
  - Bottom-up development with nested loops
  - Monte Carlo simulation opportunities
  - Simple O-O and GUI exercises
- Further CSE uses in AI, networking, sci. vis., ...
- Easy variation for unique assignments
- Best wishes for good application!