

COMPUTER PROGRAMMING IN JAVA

COLUMBIA UNIVERSITY HIGH SCHOOL SCIENCE HONORS PROGRAM

Basic Java, Part 3 Assignment 2007 Feb 23 Sat

Even-Odd Game

Consider the following game between two players: Both players simultaneously declare "one" or "two". Player 1 wins if the sum of the two declared numbers is odd and Player 2 wins otherwise. In either case the loser is obliged to pay the winner (in dollars) the sum of the two declared numbers. So Player 1 may lose 2 or 4 dollars and may win 3 dollars.

We make a claim that Player 1 has an advantage. While it is possible to determine this analytically using game theory, instead we will implement a simulation to empirically settle the claim.

We will simulate a computer player as follows: the computer player will have a threshold value t . The player will generate a random number between 0 and 1. If the number is greater than t the computer will declare "two", and if the random number is less than t the computer will declare "one". Note that if both players are computers then each computer player will have its own threshold value.

Our simulation will systematically try "all" possible threshold values for player 1 and player 2, and keep a running tally of the results. If our simulation consistently shows that some threshold value t_1 for player 1 always guarantees a win (i.e. a positive sum of money), then we have proven the claim. If the results are inconsistent, then we have disproved it.

(Courtesy of CS1004)

Assignments

- A) Write the code that allows a human player and a computer player to play as many games as the user chooses. Try to use a driver, and make the game code its own class.
- B) Reusing as much of the code from A) as possible, write the code that plays a single game of computer versus computer. Then extend this to play n games, for any value of n .
- C) Implement a method that prints out a table of size $n \times n$, with a random number at each element.
- D) Find the min value of a sequence of numbers: that is, generate a set of random numbers, and print the minimum value. Then, use this code to generate several sets of

random numbers, and find the maximum minimum value.

E) Putting together C) and D), at the end of each row, print the minimum found in that row, and the maximum of the mins seen so far. At the end, print the *row number* where the max was found.

F) Lastly, put it all together and implement a complete simulation of the game described above. Be sure to print out the optimum *tI* value at the end.

Notes and Hints

- Start with a small step size and a small number of games, say STEP = 25 and NUM_GAMES = 100; this will greatly facilitate debugging.
- Once you have it all working, use STEP = 2 and NUM_GAMES = 1,000,000; this is a high variance problem! You will know you have found the optimal value when you run the simulation itself several times, and it always returns the same value.
- Try to use as many classes as possible; there is a large amount of reuse possible.
- Make everything static, so you don't have to deal with issues of instances, constructors, etc.