Assignment 3 C212 FA22

Last modified 2022-09-16 9:00 PM

Summary

The gist of this assignment document's contents are as follows:

- Program Description: Arithmetic-practicing game based on getting all answers right in a row that can display some stats and be controlled via a menu system
- Methods to implement
- Hints
- Deliverables & Reminders
    - Use a project named `Assignment3`, and a class named `CoolArithmeticGames`
    - Submit a .zip file containing, at minimum, `CoolArithmeticGames.java`
- Rubric
    - Also visible on Canvas assignment page
- Changelog
    - No updates yet…

Program Description

Your little brother is having trouble with arithmetic – particularly, his addition and multiplication of positive integers. Your parents realize that after taking a few weeks of your Java programming course, you could potentially write a computer game program that would allow him to practice his arithmetic skills.

(Before reading on, we'd recommend looking at the sample program output below to get a feel for what's happening in the game.)

Write a program, `CoolArithmeticGames.java`. Basically, the game works like this: The user has a score that starts at zero. Each time they play a round of the game, they are asked some addition or multiplication questions. If they get all the questions correct in the round, they gain points. If they get even one question wrong, they immediately lose the round and gain 0 points. The game gradually gets harder as the user wins more. The user will be able to see their score between rounds, as well as their current and longest winning streak, in the hall of fame.

The number of questions the user is asked each round is equal to $ceiling(score\ /\ 5)\ +\ 1$. "Ceiling" means take the result of the expression and round up to the nearest integer, even if the fractional part isn't >= 0.5. This can be accomplished in Java using `Math.ceil()`, but note that it returns a double, which you may need to cast to an integer later on. For example:

- If score is 0, ask 0 + 1 = 1 questions, since 0 / 5 is 0 which is already an integer
- If score is 1, ask 1 + 1 = 2 questions, since ceil(1 / 5) = 1
- If score is 2, ask 1 + 1 = 2 questions
- If score is 5, ask 1 + 1 = 2 questions, since ceil(5 / 5) = 1 which is already an integer
- If score is 6, ask 2 + 1 = 3 questions, since ceil(6 / 5) = 2
- If score is 21, ask 5 + 1 = 6 questions, since ceil(21 / 5) = 5

The user chooses whether they are asked addition or multiplication questions each round. Multiplication rounds give double the points of addition rounds. They also choose the maximum number that can appear as the two randomly generated numbers; however, they must choose a max number that is greater than their current score. The minimum for each number is 0, making the random interval [0, maxNumber].

With this max number, the number of questions to be asked, and whether or not the game is addition or multiplication, you will pass these arguments to a method you write called `arithGame()`, which returns the points earned in that round.

Once again, if the user gets all questions right in addition mode, points gained will be the number of questions asked that round. In multiplication mode, points gained will be double the number of questions asked on a win. If they lose, the round should immediately end and they gain no points.

Other than playing and quitting the game, on the main menu the user can also choose to see the hall of fame. This will display their score, their current round win streak and whether they're still on a streak, and their longest ever round win streak.

Importantly, for this project, you must validate all menu option choices entered by the user. For example, on the main menu, they must choose either 1, 2, or 3, or else they will be asked to enter a valid input. We haven't taught you how to deal with the user entering something other than a number, so you can assume they only enter integers. However, you must make sure that that integer is a valid choice on their current menu.

Sample program output

User input in bold and italics.

```
Welcome to CoolArithmeticGames!
Please make a selection from the following:
1. Play Arithmetic Game.
2. View Hall of Fame.
3. Quit.
2
===== Hall of Fame =====
Your score: 0
Current round win streak: 0
Longest round win streak: 0
=======================
Please make a selection from the following:
1. Play Arithmetic Game.
2. View Hall of Fame.
3. Quit.
1
Would you like (1) addition or (2) multiplication?
1
Enter the maximum number, which must be greater than your score (0):
1
1 + 1 = 2
You got 1 points for winning!
Please make a selection from the following:
1. Play Arithmetic Game.
2. View Hall of Fame.
3. Quit.
1
Would you like (1) addition or (2) multiplication?
1
Enter the maximum number, which must be greater than your score (1):
2
0 + 1 = 1
1 + 0 = 1
You got 2 points for winning!
Please make a selection from the following:
```

```
1. Play Arithmetic Game.
2. View Hall of Fame.
3. Quit.
1
Would you like (1) addition or (2) multiplication?
1
Enter the maximum number, which must be greater than your score (3):
10
8 + 2 = 10
9 + 7 = 17
You got 0 points for losing.
Please make a selection from the following:
1. Play Arithmetic Game.
2. View Hall of Fame.
3. Quit.
2
===== Hall of Fame =====
Your score: 3
Current round win streak: 0
Longest round win streak: 2
=======================
Please make a selection from the following:
1. Play Arithmetic Game.
2. View Hall of Fame.
3. Quit.
1
Would you like (1) addition or (2) multiplication?
2
Enter the maximum number, which must be greater than your score (3):
9
7 x 3 = 21
3 x 5 = 15
You got 4 points for winning!
Please make a selection from the following:
1. Play Arithmetic Game.
2. View Hall of Fame.
3. Quit.
2
```

```
===== Hall of Fame =====
Your score: 7
Current round win streak: 1, and still going!
Longest round win streak: 2
=======================
Please make a selection from the following:
1. Play Arithmetic Game.
2. View Hall of Fame.
3. Quit.
1
Would you like (1) addition or (2) multiplication?
2
Enter the maximum number, which must be greater than your score (7):
5
That max number is too low, please try again.
Enter the maximum number, which must be greater than your score (7):
7
That max number is too low, please try again.
Enter the maximum number, which must be greater than your score (7):
10
9 x 4 = 36
5 x 10 = 50
8 x 0 = 0
You got 6 points for winning!
Please make a selection from the following:
1. Play Arithmetic Game.
2. View Hall of Fame.
3. Quit.
2
===== Hall of Fame =====
Your score: 13
Current round win streak: 2, and still going!
Longest round win streak: 2
=======================
Please make a selection from the following:
1. Play Arithmetic Game.
2. View Hall of Fame.
3. Quit.
```

```
1
Would you like (1) addition or (2) multiplication?
1
Enter the maximum number, which must be greater than your score (13):
13
That max number is too low, please try again.
Enter the maximum number, which must be greater than your score (13):
14
6 + 11 = 17
5 + 14 = 19
5 + 14 = 19
6 + 12 = 18
You got 4 points for winning!
Please make a selection from the following:
1. Play Arithmetic Game.
2. View Hall of Fame.
3. Quit.
2
===== Hall of Fame =====
Your score: 17
Current round win streak: 3, and still going!
Longest round win streak: 3
=======================
Please make a selection from the following:
1. Play Arithmetic Game.
2. View Hall of Fame.
3. Quit.
4
Invalid selection, please try again.
Please make a selection from the following:
1. Play Arithmetic Game.
2. View Hall of Fame.
3. Quit.
3
Goodbye!
```

Methods to implement

Your solution MUST include the following method. You are welcome to use more.

```
static int arithGame(int max, int numberOfQuestions, boolean isAdditionGame)
```

- `max`: the maximum that the randomly generated numbers for each problem can be, inclusive.
- `numberOfQuestions`: the number of questions that the user has to answer correctly in a row to win the round. Notice that this number must be calculated before you can call the method, using the formula given in the program description.
- `isAdditionGame`: true if this is an addition round, false for a multiplication round.
- The method will generate random numbers in the interval [`0`, `max`] and ask the user to answer the sum or product. If they are wrong, end the game and return `0`. If they get a question right, repeat until they've answered `numberOfQuestions` questions right, at which point return `numberOfQuestions`, or double that in a multiplication round. In other words, the function returns the number of points this round was worth.

Hints

This will be the most complex menu system we've done so far. We've seen plenty of examples of these, which you should refer to as a starting point. In particular, we would recommend using do-while loops for your menu system.

You haven't had to validate inputs before, though. Notice that this just means you're asking the user for some input over and over again until it matches some criteria. What could you use to do this? You know you'll have to ask at least once, and you'll need to make sure it matches one of the options they're given.

Remember that you only need to verify input while the user is in a menu. During the game, there is no "invalid selection" – they just get the question wrong.

Keeping track of the current and longest win streak isn't too hard, but there are a lot of cases to consider. Which variables do you need to keep to record this information, and what types are they? What happens when they win once after a losing streak? What happens when they win while on a winning streak? What happens when they lose on a winning streak? What happens when they lose while not on a winning streak? Some of these questions are redundant, but you should make sure your program has the right output in each of these cases. These are just some of the edge-cases we'll be checking for this time around.

Your `arithGame()` method will need to create new Scanner and Random objects separate from the ones used in your main method. This is exactly the same process as creating and using a new Scanner or Random object in main. (If you know how, you can do it another way.)

You may not fully understand what static means and how it works yet, but if you define the required method using `static` outside `main` but inside the `CoolArithmeticGames` class, then in your main method you can just call it using `arithGame(...)`. This way, there's no need to write in main ~~`CoolArithmeticGames.arithGame()` or `CoolArithmeticGames game = new CoolArithmeticGame(); game.arithGame()`~~. If you do something like these, we will be very sad `;^(`

The method you write just returns the points earned for the round. Back in your main method (or wherever you called `arithGame()`), you'll need to use this result to decide whether they won or not so you can update their streaks appropriately.

Deliverables & Reminders

Submit a .zip file on Canvas. This .zip file can either be your zipped project folder from your IDE, or a zipped folder containing your `CoolArithmeticGames.java` file.

Your IntelliJ project can have any name appropriate – `Assignment3` would be a good choice.

In addition to solving the problem, your code should follow these usual requirements:

- All variables should have informative names.
- Your code should be clean, presentable, and consistent with Java conventions which includes:
    - Classes are named in PascalCase, while variables are in camelCase
    - Be consistent with your spacing and opt for readability (see Assignment 1's PDF for an example).
    - Use correct indentation for both code and curly braces.
        - If your choice of indentation or lack of curly braces is reasonable – for example, leaving out curly braces for an if-statement with only one line of code in the body – you won't be penalized.
    - (As always, ask your grader for clarifications. We won't "pre-grade", but we can answer questions on whether parts of your code meets cleanliness expectations.)
- Every program must have comments. There should be header comments in your program including your name, a brief introduction of the program, and a short step-by-step description of what the program does and how it does it. You should have at least one comment in-line with the code besides that.

Your input/output should match the examples. You'll want to check all the edge-cases when you test your solution, because the graders will be doing that.

Ensure your submission contains the necessary file(s) and is the correct version of your code that you want graded. This is your warning. You will not be allowed to resubmit this assignment after the due date if you submit the wrong file.

Remember that this is graded on accuracy and you are not allowed to collaborate on your solutions with others. Plagiarism or cheating will be dealt with seriously.

Rubric

A rubric is also available on the assignment page on Canvas, but here is an outline of the point breakdown for different categories.

- (20pts) Code cleanliness, conventions, and comments
    - (20/20pts) for >95% correctness and all required comments
    - (15/20pts) for repeated minor mistakes or one egregious mistake, or for missing either overall or inline comments
    - (10/20pts) for code that is sloppy and hard to understand or is missing all comments
    - (<=5/20pts) for code that has no comments, is hard to understand, and consistently does not follow Java conventions
- (22.5pts) for the main menu system using loops, preferably a do-while loop
    - (2.5pts) for collecting user input on main menu
    - (5pts) for validating user input on the main menu and giving a warning for invalid inputs
    - (2.5pts) for the sub-menu on option 1 that asks what kind of game to play
    - (5pts) for validating game type selection and warning for invalid inputs
    - (2.5pts) for collecting max random number input
    - (5pts) for validating that the max random number is more than the user's score and warning for invalid inputs
- (10pts) for calling arithGame() with correct arguments
    - (2.5pts) for max argument
    - (5pts) for numberOfQuestions argument
    - (2.5pts) for isAdditionGame argument
- (17.5pts) for a working hall of fame
    - (5pts) for correct score display
    - (7.5pts) for correct current streak display, including some indication that they are on a streak
    - (5pts) for a correct highest streak display. Current streak should never be higher than highest streak – update highest streak immediately, don't wait until the streak ends
- (30pts) for arithGame()
    - (5pts) for correct method header – i.e. the method being a static member of CoolArithmeticGames with correct parameters and return type
    - (5pts) for having access to instances of usable Random and Scanner objects obtained in some way, preferably for this assignment by creating new instances within the method body
    - (5pts) for a loop that asks the right number of questions for the round. Half-marks for off-by-one or other simple errors.

- (10pts) for random number generation
    - (5pts) for the numbers changing from question to question
    - (5pts) for correct interval, or NO marks for incorrect interval
- (2.5pts) for returning correct number of points
- (2.5pts) for ending the round immediately on an incorrect answer

Changelog

If any corrections, clarifications, etc. are made to the assignment after its release, they will be listed here. Any changes more substantial than simply fixing typos will get corresponding Canvas announcements. That way, you don't need to keep checking whether updates have occurred.