Submission Worksheet

CLICK TO GRADE

https://learn.ethereallab.app/assignment/IT114-006-S2024/it114-number-guesser-4/grade/bm47

IT114-006-S2024 - [IT114] Number Guesser 4

Submissions:

Submission Selection

1 Submission [active] 2/13/2024 4:06:26 PM

Instructions

COLLAPSE

1. Create the below branch name
2. Implement the NumberGuess4 example from the lesson/slides
    1. https://gist.github.com/MattToegel/aced06400c812f13ad030db9518b399f
3. Add/commit the files as-is from the lesson material (this is the base template). You may want to push this commit so you can open the pull request and keep it open.
4. Pick two (2) of the following options to implement
    1. Display higher or lower as a hint after a wrong guess (only after a wrong guess that doesn't roll back the level)
    2. Implement anti-data tampering of the save file data (reject user direct edits)
    3. Add a difficulty selector that adjusts the max strikes per level (i.e., "easy" 10 strikes, "medium" 5 strikes, "hard" 3 strikes)
    4. Display a cold, warm, hot indicator based on how close to the correct value the guess is (example, 10 numbers away is cold, 5 numbers away is warm, 2 numbers away is hot; adjust these per your preference) Only display this when the wrong guess doesn't roll back the level
    5. Add a hint command that can be used once per level and only after 2 strikes have been used that reduces the range around the correct number (i.e., number is 5 and range is initially 1-15, new range could be 3-8 as a hint)
    6. Implement separate save files based on a "What's your name?" prompt at the start of the game (each person gets their own save file based on user's name)
5. Fill in the below deliverables
6. Save changes and export PDF
7. Git add/commit/push your changes to the HW branch
8. Create a pull request to main
9. Complete the pull request (don't forget to locally checkout main and pull changes to prep for future work)
10. Upload the same PDF to Canvas

Branch name: M3-NumberGuesser-4

Tasks: 7 Points: 10.00

Implementation 1 (4 pts.)

Task #1 - Points: 1

Text: Chosen Option and Details

Checklist — *The checkboxes are for your own tracking

#	Points	Details
#1	1	Mention which option you picked
#2	1	Explain the logic of how you solved/implemented the chosen option (concrete details). Explain how the code works, don't just paste code snippets

Response:

The first option I chose was to indicate whether or not the user has to guess a higher or lower number to get the number correct. I did this in what I found to be the simplest and easiest way. I used the "guess" variable and compared it to the "number", number being the number that was chosen. If "guess" was lower than number, it would say to guess a higher number. If "guess" was higher than number, it would say to guess a lower number, and if "guess" was == to number, then it would return with a win message. This code is located on lines 122 to 142.

Task #2 - Points: 1

Text: 2+ Screenshots of code and demo

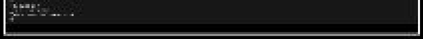
Checklist — *The checkboxes are for your own tracking

#	Points	Details
#1	1	Show implementation working by running the program
#2	1	Clearly caption the screenshot of what you're showing
#3	1	The code screenshot(s) clearly show the code specific to the feature
#4	1	A comment with the UCID/date is visible near the code change(s)

Task Screenshots:

Large Gallery

Checklist Items (0)



this shows the code, the console with the message to choose higher or lower, and includes my UCID and the class section.

● Implementation 2 (4 pts.)

COLLAPSE

●

COLLAPSE

Task #1 - Points: 1

Text: Chosen Option and Details

Checklist *The checkboxes are for your own tracking

#	Points	Details
#1	1	Mention which option you picked
#2	1	Explain the logic of how you solved/implemented the chosen option (concrete details). Explain how the code works, don't just paste code snippets

Response:

The second implementation I picked was the difficulty selector. This option lets the user choose which difficulty they want, based on the number of strikes. Easy = 10 strikes, medium = 5, and hard = 3. The way I got this to work when you run the code is by implementing it into the "public void start()" method, so that every time the game is started, the difficulty selector is displayed. The code was somewhat simple to implement, as I just used different cases for each difficulty and changed the "maxstrikes" value to correlate to the difficulty chosen.

●

COLLAPSE

Task #2 - Points: 1

Text: 2+ Screenshots of code and demo

Checklist *The checkboxes are for your own tracking

#	Points	Details
#1	1	Show implementation working by running the program
#2	1	Clearly caption the screenshot of what you're showing
#3	1	The code screenshot(s) clearly show the code specific to the feature
#4	1	A comment with the UCID/date is visible near the code change(s)

Task Screenshots:

Large Gallery

Checklist Items (0)

This snippet shows my UCID and class, along with the code. You can visibly see at the bottom, in the console, that I entered the "easy" difficulty when I was prompted with the difficulty.

Misc (2 pts.)

COLLAPSE

COLLAPSE

Task #1 - Points: 1

Text: Reflection

Checklist *The checkboxes are for your own tracking

#	Points	Details
☐ #1	1	Example prompts: Learn anything new? Face any challenges? How did you overcome and issues?
☐ #2	1	At least a few logical sentences related to the assignment.

Response:

I am not a very good Java programmer, so this assignment was a bit difficulty for me. The code was over 100 lines long, so I will admit I was a bit intimidated going into the assignment. I read through it to try to understand it, and realized that it is much easier to read and understand than I thought. The biggest problems I faced were as follows: for the number guesser, I did not face many difficulties. I just had to see the "guess" and "number" vars to ensure that it was programmed correctly. For the difficulty selector, it was a bit more difficult. I didn't put it in the start function at first and was facing many issues. Once I realized I should put it there, it worked a bit better. One issue that started with this though, is that it was considering the difficulty selection as an input (or something like that, still not sure what was going on).

COLLAPSE

Task #2 - Points: 1

Text: Pull Request URL

Details:
URL should end with /pull/# where the # is the actual pull request number.

URL #1

Missing URL

COLLAPSE

Task #3 - Points: 1

Text: Waka Time (or related) Screenshot

Checklist

*The checkboxes are for your own tracking

#	Points	Details
☐ #1	1	Screenshot clearly shows what files/project were being worked on (the duration of time doesn't correlated with the grade for this item)

Task Screenshots:

Large Gallery