Submission Worksheet

CLICK TO GRADE

https://learn.ethereallab.app/assignment/IT114-003-F2024/it114-module-2-java-problems/grade/jns

Course: IT114-003-F2024
Assigment: [IT114] Module 2 Java Problems
Student: Jimmy S. (jns)

Submissions:

Submission Selection

1 Submission [submitted] 9/23/2024 7:47:26 PM

Instructions

COLLAPSE

Overview Video: https://youtu.be/4M8Di5jrcZQ

Guide:

1. Make sure you're in the main branch locally and `git pull origin main` any pending changes.
2. Make a new branch per the recommended branch name below (`git checkout -b ...`).
3. Create a folder in your local repo called `Module2`
4. Grab the template code from https://gist.github.com/MattToegel/fdd2b37fa79a06ace9dd259ac82728b6.
5. Create individual Java files for each problem and save the files inside the `Module2` folder.
    1. They should end with the file extension in lowercase `.java`.
6. Move the unedited template files to GitHub.
    1. `git add .`
    2. `git commit -m "adding template files"`
    3. `git push origin branch_name` (see below).
    4. Create and open a pull request from the homework branch to main (leave it open until later steps).
7. Note: As you work, it's recommended to add/commit at least after each solution is done (i.e., 3+ times in this case).
    1. Make sure the files are saved before doing this.
    2. A file is unsaved if you see a white dot in the tab where the filename shows in VS Code
8. Fill in the items in the worksheet below (save as often as necessary).
9. Once finished, export the worksheet.
10. Add the output file to any location of your choice in your repository folder (i.e., a `Module2` folder).
11. Check that git sees it via `git status`.
12. If everything is good, continue to submit.

12. If everything is good, continue to submit:
    1. Track the file(s) via `git add`.
    2. Commit the changes via `git commit` (don't forget the commit message).
    3. Push the changes to GitHub via `git push` (don't forget to refer to the proper branch).
    4. Create a pull request from the homework related branch to main (i.e., main <- "homework branch").
    5. Open and complete the merge of the pull request (it should turn purple).
    6. Locally checkout main and pull the latest changes (to prepare for future work).
13. Take the same output file and upload it to Canvas.

Branch name: M2-Java-Problems

Group

100%

Group: Problem 1
Tasks: 1
Points: 3

COLLAPSE

Task

100%

Group: Problem 1
Task #1: Screenshot of the Problem 1 Solved Code and Output
Weight: ~100%
Points: ~3.00

COLLAPSE

Details:
Only make edits where the template code mentions.

Solution should ensure that any passed in array will have only the odd values output.
Requires at least 2 screenshots (code + output from terminal)

Columns: 1

Sub-Task

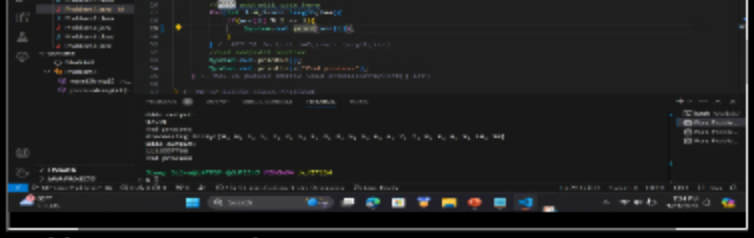
100%

Group: Problem 1
Task #1: Screenshot of the Problem 1 Solved Code and Output
Sub Task #1: Screenshot the output of the solved problem

Task Screenshots

Gallery Style: 2 Columns

4 2 1



problem 1, expected output

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Sub-Task

100%

Group: Problem 1

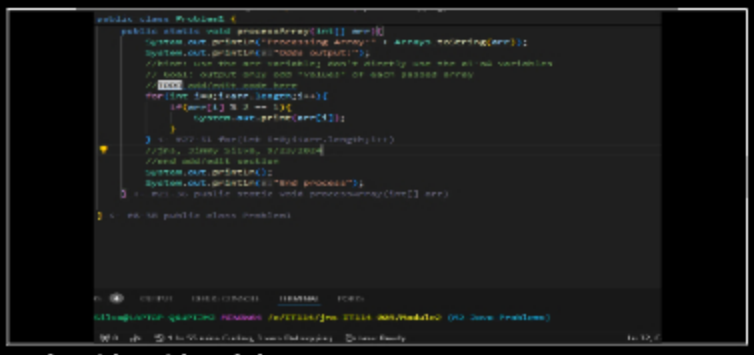
Task #1: Screenshot of the Problem 1 Solved Code and Output

Sub Task #2: Screenshot the code solution (ucid/date must be included as a comment)

Task Screenshots

Gallery Style: 2 Columns

4 2 1



code with ucid and date

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain in concise steps how this logically works

Response:

the code works by going through each index in the array, in which if the is a remander after dividing the value by two, display the number in the console

End of Task 1

End of Group: Problem 1

Task Status: 1/1

Group

100%

Group: Problem 2

Tasks: 1

Points: 3

COLLAPSE

Task

100%

Group: Problem 2
Task #1: Screenshot of the Problem 2 Solved Code and Output
Weight: ~100%
Points: ~3.00

COLLAPSE

Details:
Only make edits where the template code mentions.

Solution should ensure that any passed in array will have its values summed AND the final result converted to two decimal places (i.e., 0.10, 1.00, 1.01).

Columns: 1

Sub-Task

100%

Group: Problem 2
Task #1: Screenshot of the Problem 2 Solved Code and Output
Sub Task #1: Screenshot the output of the solved problem

Task Screenshots

Gallery Style: 2 Columns

4 2 1

expected output

Caption(s) (required)

Caption Hint: *Describe/highlight what's being shown*

Sub-Task

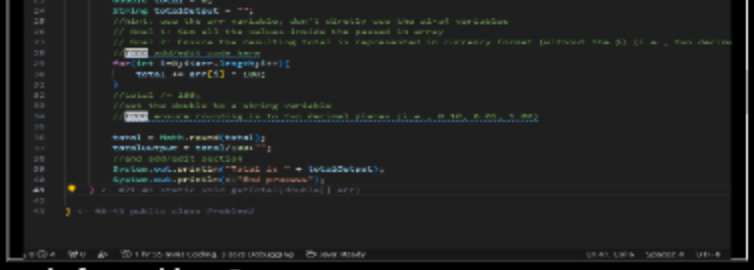
100%

Group: Problem 2
Task #1: Screenshot of the Problem 2 Solved Code and Output
Sub Task #2: Screenshot the code solution (ucid/date must be included as a comment)

Task Screenshots

Gallery Style: 2 Columns

4 2 1



code for problem 2

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain in concise steps how this logically works

Response:

what this code does is take the array of float values, turns them into integers but preserving value by multiple by 100 (avoiding float operations), and then return them back into floats but only up the second decimal place before displaying the result.

End of Task 1

End of Group: Problem 2
Task Status: 1/1

Group

100%

Group: Problem 3
Tasks: 1
Points: 3

COLLAPSE

Task

100%

Group: Problem 3
Task #1: Screenshot of the Problem 3 Solved Code and Output
Weight: ~100%
Points: ~3.00

COLLAPSE

Details:

Only make edits where the template code mentions.

Solution should ensure that any passed in array will have its values converted to a positive version of the value AND converted back to the original data type.

Columns: 1

Sub-Task

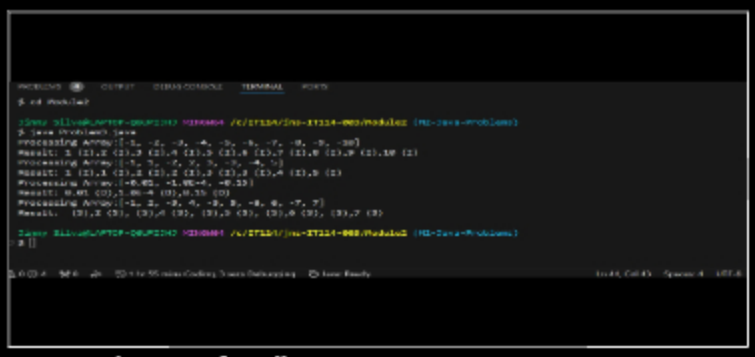
100%

Group: Problem 3
Task #1: Screenshot of the Problem 3 Solved Code and Output
Sub Task #1: Screenshot the output of the solved problem

Task Screenshots

Gallery Style: 2 Columns

4 2 1



expected output for all types

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Sub-Task

100%

Group: Problem 3

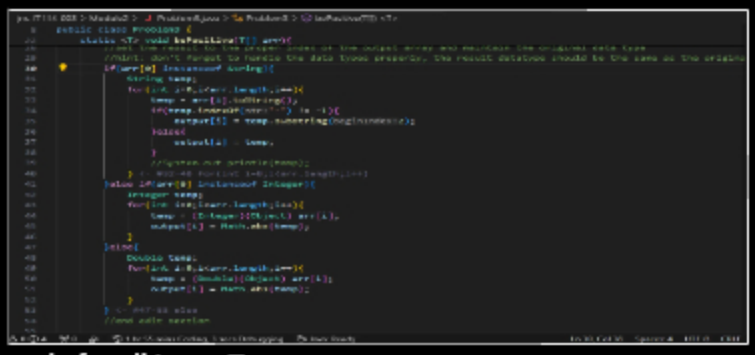
Task #1: Screenshot of the Problem 3 Solved Code and Output

Sub Task #2: Screenshot the code solution (ucid/date must be included as a comment)

Task Screenshots

Gallery Style: 2 Columns

4 2 1



code for all types T

Caption(s) (required) ✓

Caption Hint: *Describe/highlight what's being shown*

Task Response Prompt

Explain in concise steps how this logically works

Response:

basically how this code works is that before it makes operations to remove the minus sign from the variable, it first identifies what type t[] arr is, and then procceds according to still maintain the value of the array after operations are done.

End of Task 1

End of Task 1

End of Group: Problem 3
Task Status: 1/1

Group

100%

Group: Reflection
Tasks: 3
Points: 1

COLLAPSE

Task

100%

Group: Reflection
Task #1: Reflect on your experience
Weight: ~33%
Points: ~0.33

COLLAPSE

Details:
Talk about any issues you had, how you resolved them, and anything you learned during this process.

Provide concrete details/examples. At least a few sentences.

Task Response Prompt

Response:

i had a few issues with casting to the right types in problem 3, but after looking on google for a bit i came across functions that can help do that, in which all objects O can use, so the casting became a lot less of a problem after that.

End of Task 1

Task

100%

Group: Reflection
Task #2: Include the pull request link for this branch
Weight: ~33%
Points: ~0.33

COLLAPSE

Details:
The correct link will end with /pull/ and a number.

Task URLs

URL #1
https://github.com/jnsnjit/jns-IT114-003/pull/7

URL
https://github.com/jnsnjit/jns-IT114-003/pull/7

End of Task 2

Task

100%

Group: Reflection
Task #3: Add Screenshot of Wakatime
Weight: ~33%
Points: ~0.33

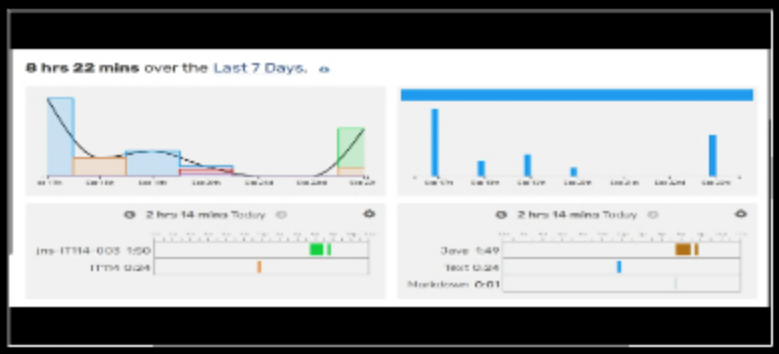
COLLAPSE

Details:
Note: The duration of time isn't directly related to the grade, the goal is to just make sure time is being tracked

Task Screenshots

Gallery Style: 2 Columns

4 2 1



wakatime, spent like 2 hours on it114 repo

End of Task 3

End of Group: Reflection
Task Status: 3/3

End of Assignment