Submission Worksheet

CLICK TO GRADE

https://learn.ethereallab.app/assignment/IT114-004-S2024/it114-m2-java-problems/grade/msa224

IT114-004-S2024 - [IT114] M2 Java Problems

Submissions:

Submission Selection

1 Submission [active] 2/5/2024 11:28:27 PM

Instructions

COLLAPSE

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. Grab the template code
   from https://gist.github.com/MattToegel/fdd2b37fa79a06ace9dd259ac82728b6
4. Create individual Java files for each problem and save the files inside a subfolder of your choice
    1. The should end with the file extension in lowercase .java
5. Move the unedited template files to github
    1. `git add .`
    2. `git commit -m "adding template files`
    3. `git push origin <homework branch>` (see below and don't include the < >)
    4. Create and open a pull request from the homework branch to main (leave it open until later steps)
6. 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
7. Fill in the items in the worksheet below (save as often as necessary)
8. Once finished, export the worksheet
9. Add the output file to any location of your choice in your repository folder (i.e., a Module2 folder)
10. Check that git sees it via `git status`
11. 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)
12. Take the same output file and upload it to Canvas
    1. *This step is new since GitHub renders the PDF as an image the links aren't clickable so this method works better
    2. *Remember, the github process of these files are encouragement for your tracking of your progress

Branch name: M2-Java-Problems

Tasks: 8 Points: 10.00

COLLAPSE

Problem 1 (3 pts.)

COLLAPSE

Task #1 - Points: 1

Text: Screenshot of the Problem 1 Solved Code and Output

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)

Checklist

*The checkboxes are for your own tracking

#	Points	Details
#1	1	Edits were done only in the processArray() method and original template code/comments remain untouched
#2	1	Only arr is used (no direct usage of a1, a2, a3, a4)
#3	5	Only odd values output (not odd indexes/keys)
#4	1	Includes code comments with student's ucid and date
#5	1	Terminal output is fully visible

Task Screenshots:

Large Gallery

Checklist Items (1)

#4 Includes code comments with student's ucid and date

Checklist Items (1)

#5 Terminal output is fully visible

Shows my code with edits as well as my UCID and Date.

Shows output of my code.

COLLAPSE

Task #2 - Points: 1

Text: Explain your solution

Checklist *The checkboxes are for your own tracking

#	Points	Details
#1	1	Clearly explains how the code/logic solves the problem (mentions how the odd values are determined)

Response:

In problem 1, I made the code print only the odd numbers from an array. I did this by looking at each number in the array and checking if it was odd. If it was, I printed it. The original code had some arrays with both even and odd numbers, but my addition focused on showing only the odd ones, keeping things simple and straightforward.

COLLAPSE

Problem 2 (3 pts.)

COLLAPSE

Task #1 - Points: 1

Text: Screenshot of the Problem 2 Solved Code and Output

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).
Requires at least 2 screenshots (code + output from terminal)

Checklist *The checkboxes are for your own tracking

#	Points	Details
#1	1	Edits were done only in the getTotal() method and original template code/comments remain untouched (unless noted)
#2	1	Only arr is used (no direct usage of a1, a2, a3, a4)
#3	5	Passed in array's values get summed AND rounded to two decimal places like currency (i.e., 0.00, 0.10, 1.10)
#4	1	Includes code comments with student's ucid and date
#5	1	Terminal output is fully visible

Task Screenshots:

Large Gallery

Checklist Items (1)

Checklist Items (1)

#4 Includes code comments with student's ucid and date

#5 Terminal output is fully visible

Shows code with edits and my ucid and date

Shows output of my code

COLLAPSE

Task #2 - Points: 1

Text: Explain your solution

Checklist

*The checkboxes are for your own tracking

#	Points	Details
#1	1	Clearly explains how the code/logic solves the problem (mentions both how the values get summed and how the rounding is solved correctly)

Response:

In problem 2, I tweaked the code to calculate the total sum of numbers in an array and display the result rounded to two decimal places. I used a loop to go through each number in the array, adding them up, and then formatted the total to look nice with exactly two decimal places. This way, the code not only adds up the numbers but also makes the output look clean.

COLLAPSE

Problem 3 (3 pts.)

COLLAPSE

Task #1 - Points: 1

Text: Screenshot of the Problem 2 Solved Code and Output

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.
Requires at least 2 screenshots (code + output from terminal)

Checklist

*The checkboxes are for your own tracking

#	Points	Details
#1	1	Edits were done only in the bePositive() method and original template code/comments remain untouched
#2	1	Only arr is used (no direct usage of a1, a2, a3, a4)

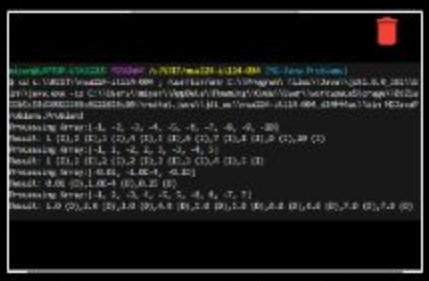
#3	5	Passed in array's values will get converted to a positive version AND converted back to the original data type
#4	1	Includes code comments with student's ucid and date
#5	1	Terminal output is fully visible

Task Screenshots:

Large Gallery

Checklist Items (1)

#4 Includes code comments with student's ucid and date



Checklist Items (1)

#5 Terminal output is fully visible

Has my code with the ucid and date

Output of my code

COLLAPSE

Task #2 - Points: 1

Text: Explain your solution

Checklist

*The checkboxes are for your own tracking

#	Points	Details
#1	1	Clearly explains how the code/logic solves the problem (mentions both the conversion to positive and conversion to original data type)

Response:

In problem 3, I adjusted the code to make all the numbers in an array positive and kept the original data type. I looked at each number in the array, made it positive, and stored the result in a new array. I took care to handle different data types, ensuring that after making them positive, they still stayed as integers, doubles, or strings based on what they were initially.

COLLAPSE

Reflection (1 pt.)

COLLAPSE

Task #1 - Points: 1

Text: Reflect on your experience

Details:

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

Provide concrete details/examples.

Response:

Had some coding issues with figuring out how to complete the java problems, but used my resources wisely to figure out how to complete the problems as we were directed while following all specific instructions.

COLLAPSE

Task #2 - Points: 1

Text: Include the pull request link for this branch

Details:

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