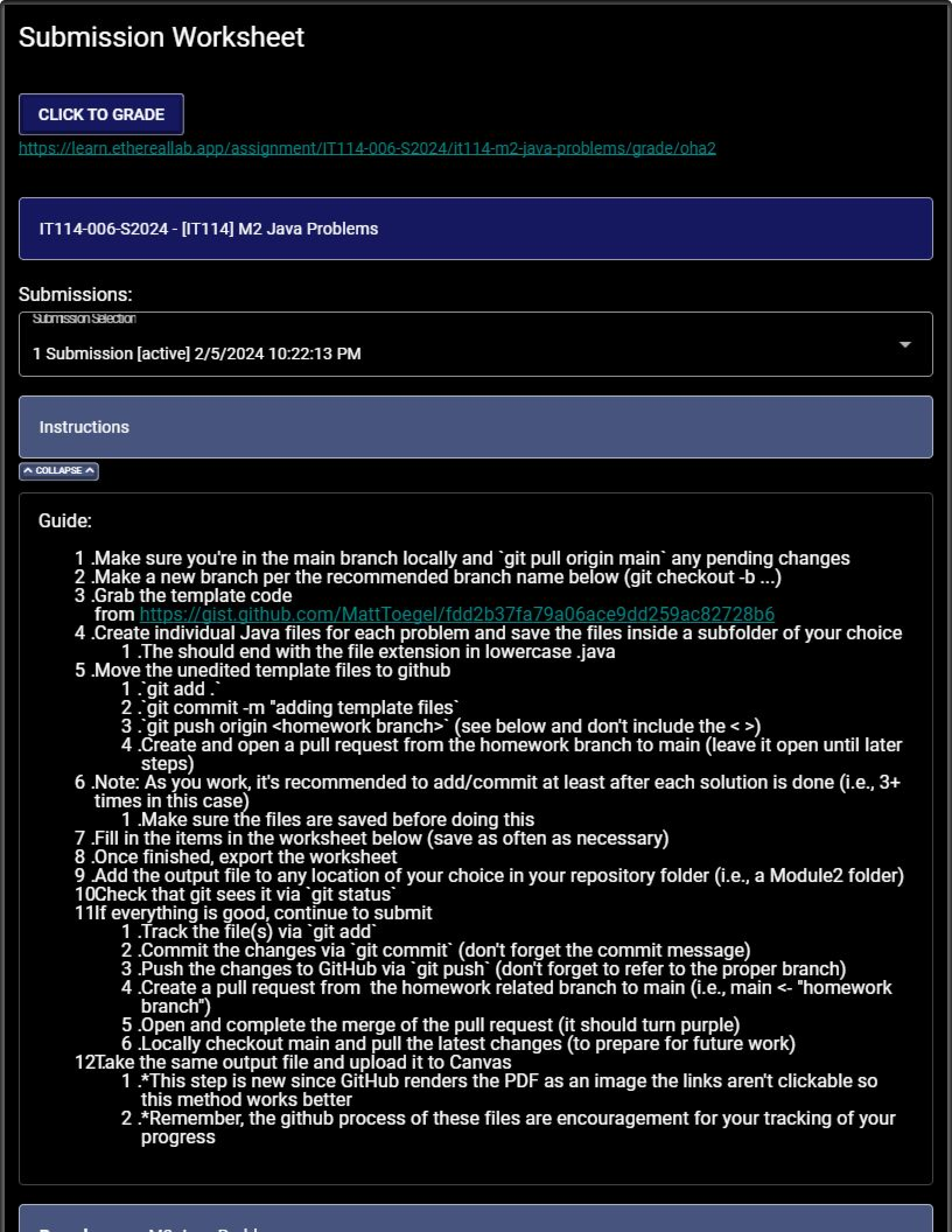


Submission Worksheet

https://learn.ethereallab.app/assignment/IT114-006-S2024/it114-m2-java-problems/grade/oha2

IT114-006-S2024 - [IT114] M2 Java Problems

Submissions:

Submission Selection

1 Submission [active] 2/5/2024 10:22:13 PM

Instructions

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)
10Check that git sees it via `git status`
11If 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)
12Take 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

● Problem 1 (3 pts.)

^ COLLAPSE ^

● ^ 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 (0)

Checklist Items (0)

Code output

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:

I loop through the arrays and check if a number is odd by seeing if the remainder after dividing by 2 is not 0, then it prints if its odd and does nothing if the number in the array is even.

● Problem 2 (3 pts.)

^ COLLAPSE ^

● ^ 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

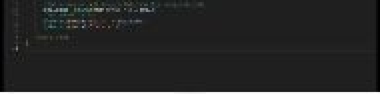
#	Points	Details
□ #1	1	Edits were done only in the getTotal() method and original template code/comments remain untouched (unless noted)
□ #2	1	Only arr is used (no direct usage of a1, a2, a3, a4)
□ #3	5	Passed in array's values get summed AND rounded to two decimal places like currency (i.e., 0.00, 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 (0)

Checklist Items (0)



Terminal with output	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:

I used a string format to make the arrays rounded and I used Math.abs in the loop through the array to make it the absolute value while adding them together

●

^ 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 (0)

Checklist Items (0)

code

terminal output

●

^ 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:

Just had a lot of issues cating this one0 in git bash so I used a different terminal I kept getting an error
Note: Problem3.java uses unchecked or unsafe operations.

Note: Recompile with -Xlint:unchecked for details.
I used the same logic as problem 2 when rounding but with more data type modifiers. I couldn't get the strings to be positive, I probably could have trimmed the - signs off but that felt like cheating so I would rather take the fall for that. I hope to see what the solution is next class since these problems stumped me a little.

●

^ COLLAPSE ^

Reflection (1 pt.)

●

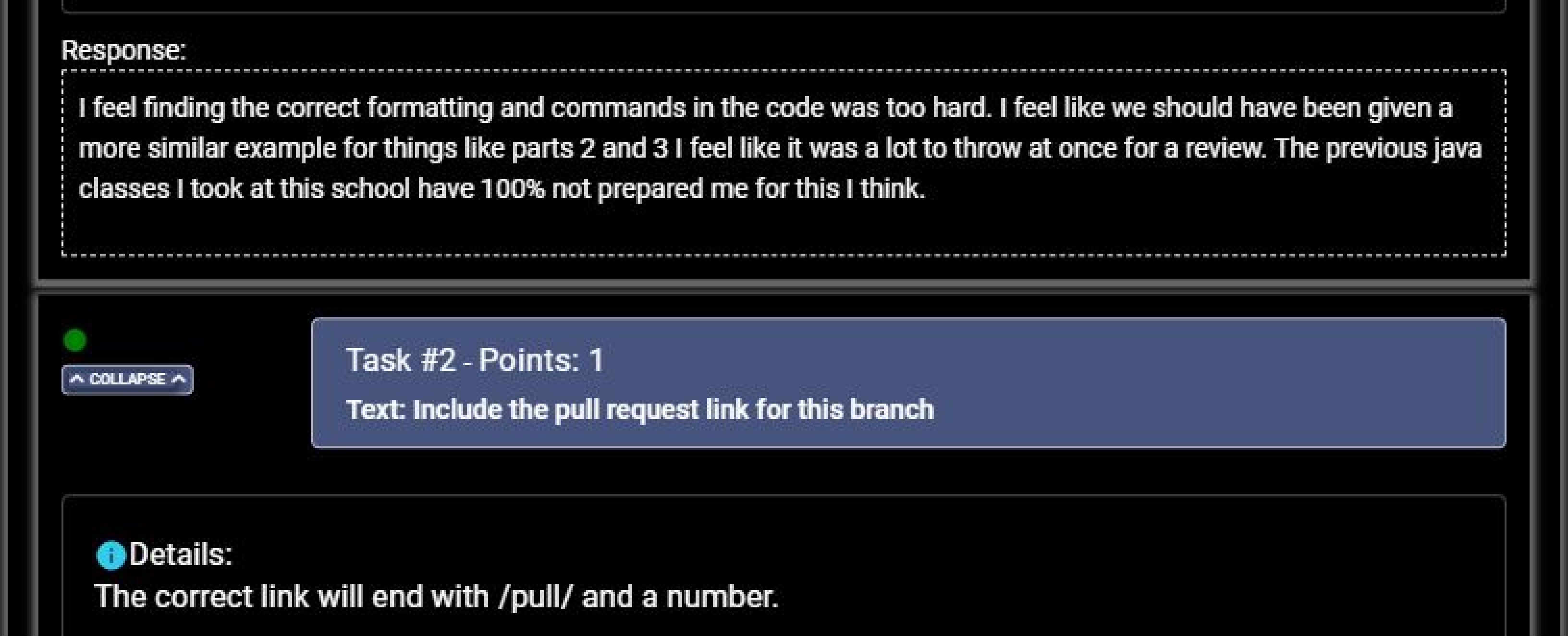
^ COLLAPSE ^

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:

I feel finding the correct formatting and commands in the code was too hard. I feel like we should have been given a more similar example for things like parts 2 and 3 I feel like it was a lot to throw at once for a review. The previous java classes I took at this school have 100% not prepared me for this I think.

●

Task #2 - Points: 1

Text: Include the pull request link for this branch

Details:

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