Submission Worksheet

Submission Data

Course: IT114-450-M2025
Assignment: IT114 Module 3 User Input Challenges
Student: Alex P. (ap2869)
Status: Submitted | **Worksheet Progress:** 100%
Potential Grade: 10.00/10.00 (100.00%)
Received Grade: 0.00/10.00 (0.00%)
Started: 6/16/2025 9:33:05 PM
Updated: 6/16/2025 9:54:38 PM
Grading Link: https://learn.ethereallab.app/assignment/v3/IT114-450-M2025/it114-module-3-user-input-challenges/grading/ap2869
View Link: https://learn.ethereallab.app/assignment/v3/IT114-450-M2025/it114-module-3-user-input-challenges/view/ap2869

Instructions

- Overview Link: https://youtu.be/iowHMCKuj5o

1. Ensure you read all instructions and objectives before starting.
2. Create a new branch from `main` called `M3-Homework`
    1. `git checkout main` (ensure proper starting branch)
    2. `git pull origin main` (ensure history is up to date)
    3. `git checkout -b M3-Homework` (create and switch to branch)
3. Copy the template code from here: GitHub Repository - M3 Homework
    - It includes CommandLineCalculator, SlashCommandHandler, MadLibsGenerator, a BaseClass and a stories folder with 5 stories (used for MadLibsGenerator). Put all into an `M3` folder or similar (adjust `package` reference at the top if you chose a different folder name).
    - Immediately record to history
        - [ ] `git add .`
        - [ ] `git commit -m "adding M3 HW baseline files"`
        - [ ] `git push origin M3-Homework`
        - [ ] Create a Pull Request from `M3-Homework` to `main` and keep it open
4. Fill out the below worksheet
    - Each Problem requires the following as you work
        - [ ] Ensure there's a comment with your UCID, date, and brief summary of how the problem was solved
        - [ ] Update the `ucid` variable
        - [ ] Code solution (add/commit periodically as needed)
5. Once finished, click "Submit and Export"
6. Locally add the generated PDF to a folder of your choosing inside your repository folder and move it to Github
    1. `git add .`
    2. `git commit -m "adding PDF"`
    3. `git push origin M3-Homework`
    4. On Github merge the pull request from `M3-Homework` to `main`

7. Upload the same PDF to Canvas
8. Sync Local
    1. `git checkout main`
    2. `git pull origin main`

Section #1: (3 pts.) Challenge 1 - Command Line Calculator (Add/sub)

Progress: 100%

≔ Task #1 (3 pts.) - Edit the `main` method to solve the requirements

Progress: 100%

Details:

- Don't adjust the give code unless noted
- Challenge 1: Accept two numbers and an operator as command-line arguments (+ and -)
- Challenge 2: Allow integer and floating-point numbers
    - Ensure correct decimal places in output based on input (e.g., 0.1 + 0.2 → 1 decimal place)
- Display an error for invalid inputs or unsupported operators
- Add code to solve the problem (add/commit as needed)

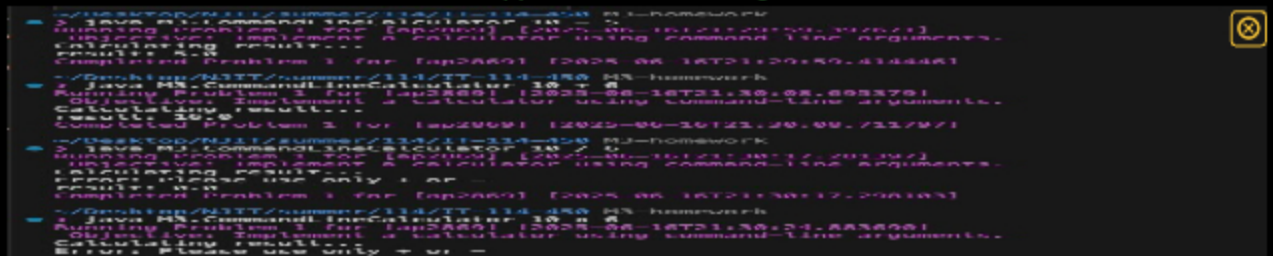
Part 1:

Progress: 100%

Details:
Two screenshots are expected

1. Snippet of relevant code showing solution (with ucid/date comment)
2. Full output of executing the program (Capture 5 variations of tests)

snippet of code challange 1



output of code from challange 1

Saved: 6/16/2025 9:54:38 PM

Part 2:

Progress: 100%

Details:
Direct link to the file in the homework related branch from Github (should end in `.java`)

URL #1
https://github.com/ap2869/IT-114-003-homework/M3/CommandLineCalculator.java

URL
https://github.com/ap2869/IT-114

Saved: 6/16/2025 9:54:38 PM

Part 3:

Progress: 100%

Details:
Briefly explain `how` the code solves the challenge (note: this isn't the same as `what` the code does)

Your Response:

this one is simple. the user is asked to enter a number an operator and another number. but the operator can only be + or -. if neither of these operator is chosen an error occurs. the reslut then prints assuming its operator is - or +

Saved: 6/16/2025 9:54:38 PM

Section #2: (3 pts.) Challenge 2 - Slash Command Handler

Progress: 100%

Task #1 (3 pts.) - Edit the `main` method to solve the requirements

Progress: 100%

Details:

- Don't adjust the give code unless noted
- Challenge 1: Accept user input as slash commands (Commands are case-insensitive)

- "/greet <name>" → Prints "Hello, <name>!"
- "/roll <num>d<sides>" → Roll <num> dice with <sides> and returns a
- "/echo <message>" → Prints the message back
- "/quit" → Exits the program

- Challenge 2: Print an error for unrecognized commands
- Challenge 3: Print errors for invalid command formats (when applicable)
- Add code to solve the problem (add/commit as needed)

Part 1:

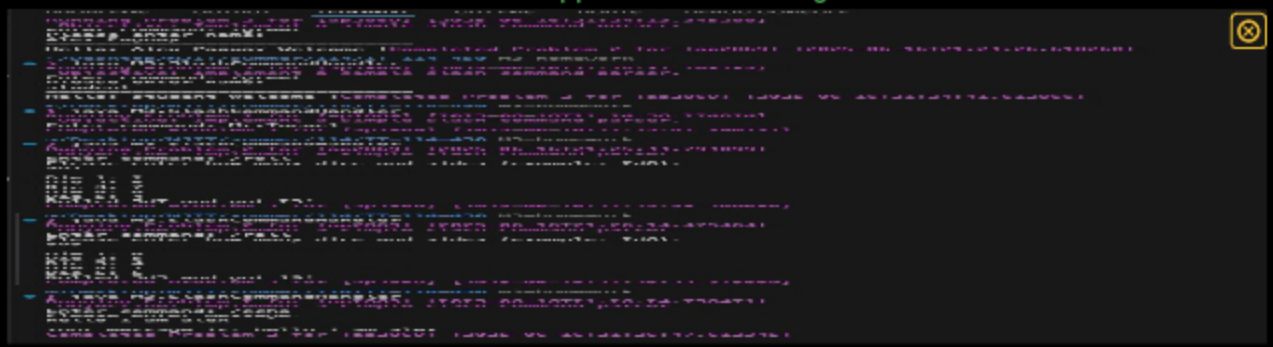
Progress: 100%

Details:
Two screenshots are expected

1. Snippet of relevant code showing solution (with ucid/date comment)
2. Full output of executing the program (Capture 3 variations of each command except "/quit")

snippet of code challange 2

continue snippet of code challange 2



output of challange 2 code

Saved: 6/16/2025 9:52:45 PM

Part 2:

Progress: 100%

Details:
Direct link to the file in the homework related branch from Github (should end in `.java`)

URL #1
https://github.com/ap2869/IT-114-450-M3-homework/M3/SlashCommandHandler.java

URL
https://github.com/ap2869/IT-114

Saved: 6/16/2025 9:52:45 PM

Part 3:

Progress: 100%

Details:
Briefly explain `how` the code solves the challenges (note: this isn't the same as `what` the code does)

Your Response:

/ if (command.startsWith("/echo")) for each slash command a / is assigned and once a slash command is chosen it goes to its assigned code. a if /echo is assigned a message is given if/greet is assigned a gretting is given if /dice is assigned a dice is rolled. the user inputs the side a and how many dice. and if /quit is assigned the program quits ans gives a goodbye message

Saved: 6/16/2025 9:52:45 PM

Section #3: (3 pts.) Challenge 3 - Mad Libs Generator

Progress: 100%

Task #1 (3 pts.) - Edit the `main` method to solve the challenges

Progress: 100%

Details:

- Don't adjust the give code unless noted
- Ensure you have the `stories` folder with the 5 stories
- Challenge 1: Load a **random** story from the "stories" folder
- Challenge 2: Extract **each line** into a collection (i.e., ArrayList)
- Challenge 3: Prompts user for each placeholder (i.e., `<adjective>`)
    - Any word the user types is acceptable, no need to verify if it matches the placeholder type
    - Any placeholder with underscores should display with spaces instead

- Challenge 4: Replace placeholders with user input (assign back to original slot in collection)
- Add code to solve the problem (add/commit as needed)

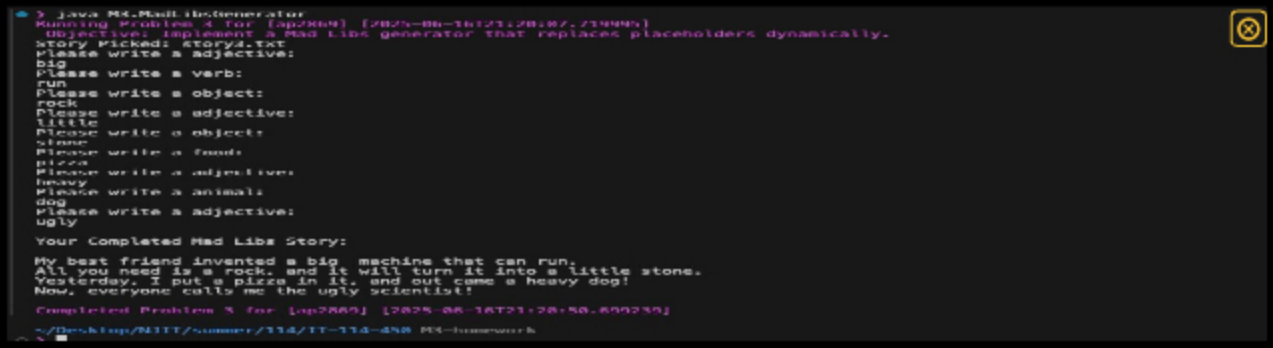
Part 1:

Progress: 100%

Details:
Two screenshots are expected

1. Snippet of relevant code showing solution (with ucid/date comment)
2. Full output of executing the program (Capture the process for at least 2 stories)

snippet of code challange 3



output of challange 3 code

Saved: 6/16/2025 9:49:21 PM

Part 2:

Progress: 100%

Details:
Direct link to the file in the homework related branch from Github (should end in `.java`)

URL #1
https://github.com/ap2869/IT-114-450-M3-homework/M3/MadLibsGenerator.java

URL
https://github.com/ap2869/IT-114

Saved: 6/16/2025 9:49:21 PM

Part 3:

Progress: 100%

Details:
Briefly explain `how` the code solves the challenges (note: this isn't the same as `what` the code does)

Your Response:

pretty simple the code gets a random file from the "stories" in an array list get the lines from the story that was chosen, and replaces and input the words that have < > with words that the user has to input. once that happened the word is replaced. the user is asked to enter an adajetive, verd, animal, etc. with that a mad lib is formed.

Saved: 6/16/2025 9:49:21 PM

Section #4: (1 pt.) Misc

Progress: 100%

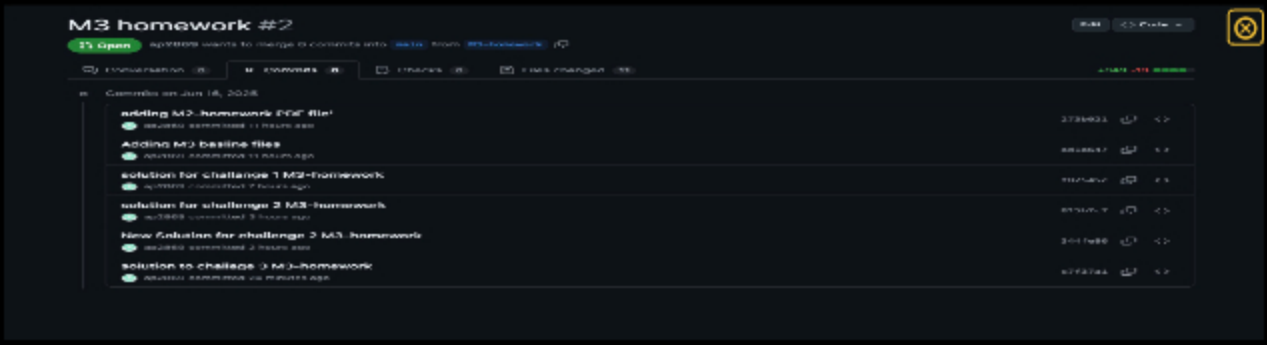
Task #1 (0.33 pts.) - Github Details

Progress: 100%

Part 1:

Progress: 100%

Details:
From the Commits tab of the Pull Request screenshot the commit history Following minimum should be present



snippet of commit history

Saved: 6/16/2025 9:40:36 PM

Part 2:

Progress: 100%

Details:
Include the link to the Pull Request (should end in `/pull/#`)

URL #1

https://github.com/ap2869/IT-114-450/pull/2 https://github.com/ap2869/IT-114

Saved: 6/16/2025 9:40:36 PM

Task #2 (0.33 pts.) - WakaTime - Activity

Progress: 100%

Details:

- Visit the WakaTime.com Dashboard
- Click `Projects` and find your repository
- Capture the overall time at the top that includes the repository name
- Capture the individual time at the bottom that includes the file time
- Note: The duration isn't relevant for the grade and the visual graphs aren't necessary

snippet of waka time project

snippet of waka time files

Saved: 6/16/2025 9:42:22 PM

Task #3 (0.33 pts.) - Reflection

Progress: 100%

Task #1 (0.33 pts.) - What did you learn?

Progress: 100%

Details:
Briefly answer the question (at least a few decent sentences)

Your Response:

i learned alot from hw3. i actually didnt know about the catch and try method. it becamse very difficult in the beginning. i actually had to do some practice examples outside of the hw to finally get the hang of what was happening. but once i did get the hang of it it was simple. kinda

Saved: 6/16/2025 9:46:36 PM

Task #2 (0.33 pts.) - What was the easiest part of the assignment?

Progress: 100%

Details:
Briefly answer the question (at least a few decent sentences)

Your Response:

the easiest part of the assignment was again the thought process of how i was going to do each assignment. i do have some backgrounf in java .. a few things i already had an idea but most of the things are new to me . but i kept reading and rereading and as it went on it becamse easier and clearer.

Saved: 6/16/2025 9:45:07 PM

Task #3 (0.33 pts.) - What was the hardest part of the assignment?

Progress: 100%

Details:
Briefly answer the question (at least a few decent sentences)

Your Response:

the hardest part of the HW was getting everything to work together. i spent the majority of the time debugging going back and forth on the examples i felt like quiting but i pused through and as i kept going to each assignment it became slighlty easier.

Saved: 6/16/2025 9:43:43 PM