Submission Worksheet

Submission Data

Course: IT114-005-F2025
Assignment: IT114 Module 3 User Input Challenges
Student: Nilkanth D. (nhd5)
Status: Submitted | **Worksheet Progress:** 100%
Potential Grade: 10.00/10.00 (100.00%)
Received Grade: 0.00/10.00 (0.00%)
Started: 10/15/2025 9:14:28 PM
Updated: 10/15/2025 10:09:50 PM
Grading Link: https://learn.ethereallab.app/assignment/v3/IT114-005-F2025/it114-module-3-user-input-challenges/grading/nhd5
View Link: https://learn.ethereallab.app/assignment/v3/IT114-005-F2025/it114-module-3-user-input-challenges/view/nhd5

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)

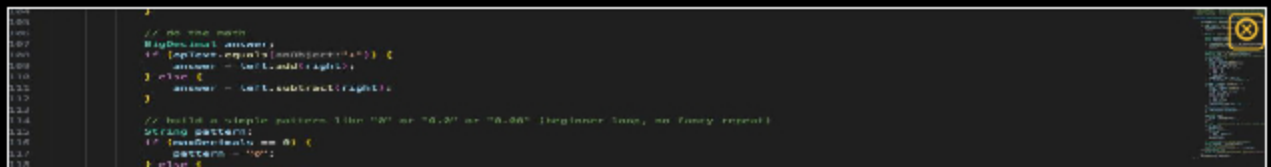
🖻 Part 1:

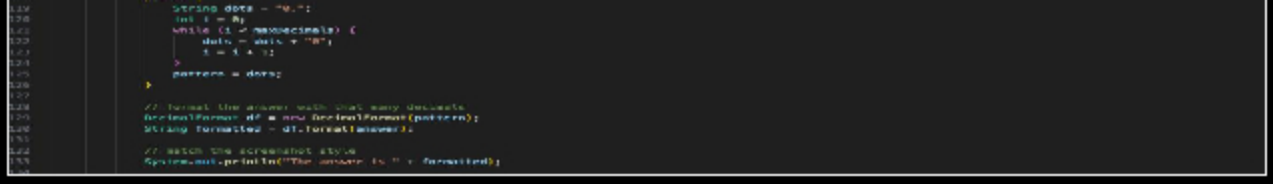
Progress: 100%

Details:
Two screenshots are expected

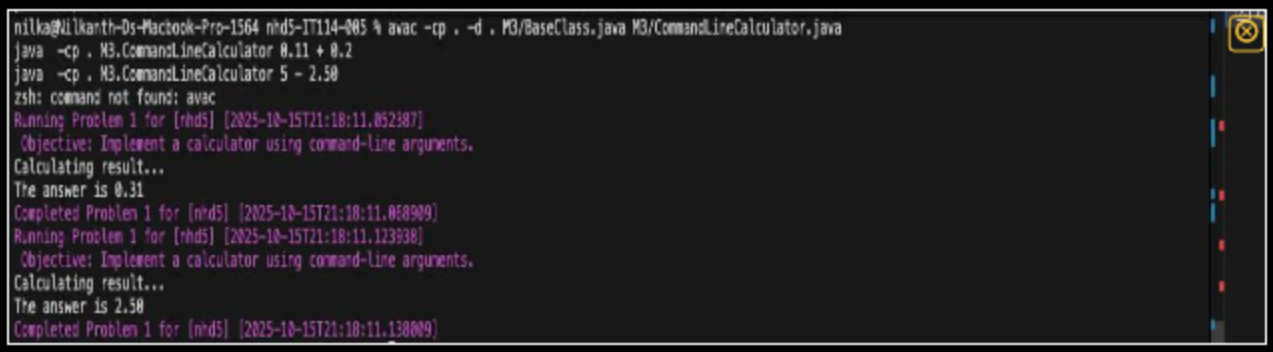
1. Snippet of relevant code showing solution (with ucid/date comment)
2. Full output of executing the program (Capture 5 variations of tests)

code snippet





code snippet



code output

code snnipet

Saved: 10/15/2025 9:23:53 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/Pixeldepth5/nhd5-IT114-005/M3/M3/CommandLineCalculator.java

URL
https://github.com/Pixeldepth5/nl

Saved: 10/15/2025 9:23:53 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:

The program validates the exact CLI format ( ) and uses a try/catch to turn bad inputs into clear errors (W3Schools: methods/main, try/catch). It parses numbers as BigDecimal to avoid floating-point rounding issues and then does either addition or subtraction (BigDecimal basics). To meet the "show longest precision" rule, it counts decimal digits in each input with simple String methods and builds a matching DecimalFormat pattern (Strings, Format Numbers).

Saved: 10/15/2025 9:23:53 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")

code snippet

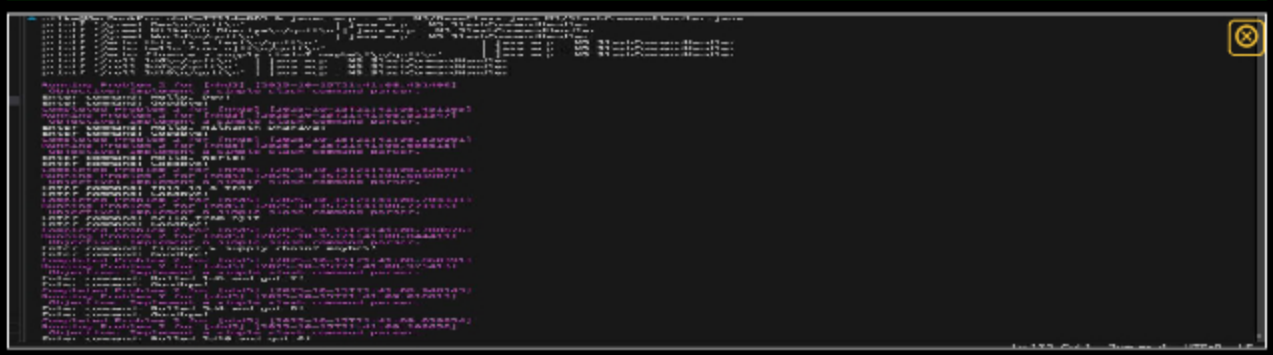
code snippet

code snippet

```
nilka@Nilkanth-DS-MacBook-Pro-1564 hhd5-17114-005 % javac -cp . -d . M3/BaseClass.java M3/SlashCommandHandler.java
printf "/greet Dev\n/quit\n" | java -cp . M3.SlashCommandHandler
printf "/greet Nilkanth Dhariya\n/quit\n" | java -cp . M3.SlashCommandHandler
printf "/greet World\n/quit\n" | java -cp . M3.SlashCommandHandler

Running Problem 2 for [nhd5] [2025-10-15T21:30:19.638077]
 Objective: Implement a simple slash command parser.
Enter command: Hello, Dev!
Enter command: Goodbye!
Completed Problem 2 for [nhd5] [2025-10-15T21:30:19.658167]
Running Problem 2 for [nhd5] [2025-10-15T21:30:19.746284]
 Objective: Implement a simple slash command parser.
Enter command: Hello, Nilkanth Dhariya!
Enter command: Goodbye!
Completed Problem 2 for [nhd5] [2025-10-15T21:30:19.766394]
Running Problem 2 for [nhd5] [2025-10-15T21:30:19.822285]
 Objective: Implement a simple slash command parser.
Enter command: Hello, World!
Enter command: Goodbye!
```

code output



code output

Saved: 10/15/2025 9:43:23 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/Pixeldepth5/nhd5-IT114-003/M3/M3/SlashCommandHandler.java

URL
https://github.com/Pixeldepth5/n

Saved: 10/15/2025 9:43:23 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:

The program uses a simple read–evaluate–print loop with Scanner to read each line, splits it into the command and the rest, then handles /greet, /echo, and /roll using basic if/else checks and Math.random() for dice—covering validation, errors, and clean exit. I leaned on W3Schools to structure main and read input (Scanner), to split and parse text with indexOf/substring (Strings), and to generate random numbers (Math), which helped me keep the code beginner-level and compile without extra libraries.

Saved: 10/15/2025 9:43:23 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)

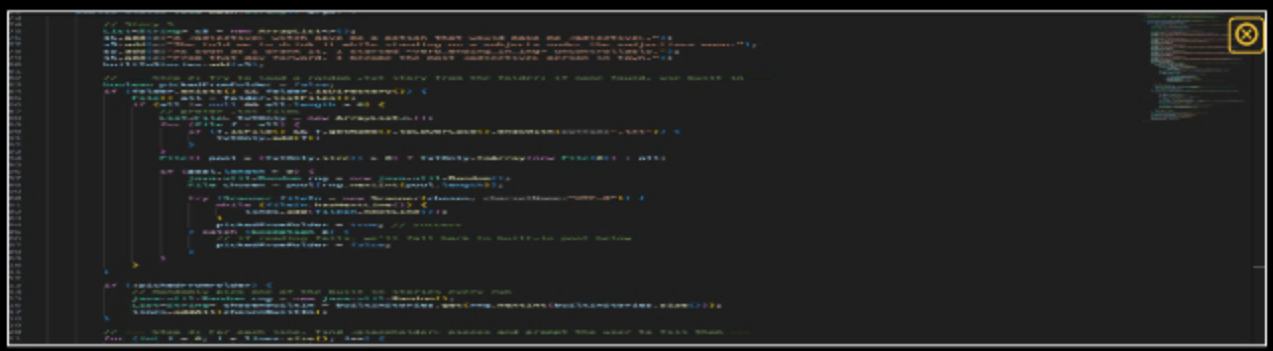
Part 1:

Progress: 100%

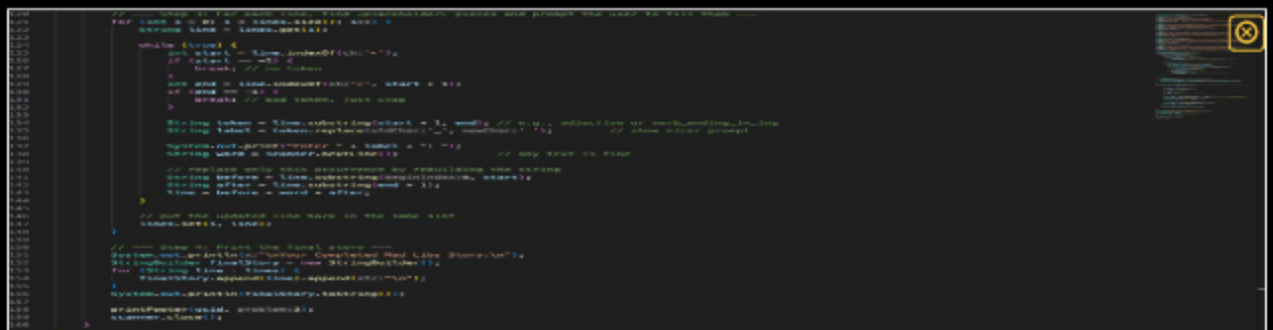
Details:
Two screenshots are expected

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)

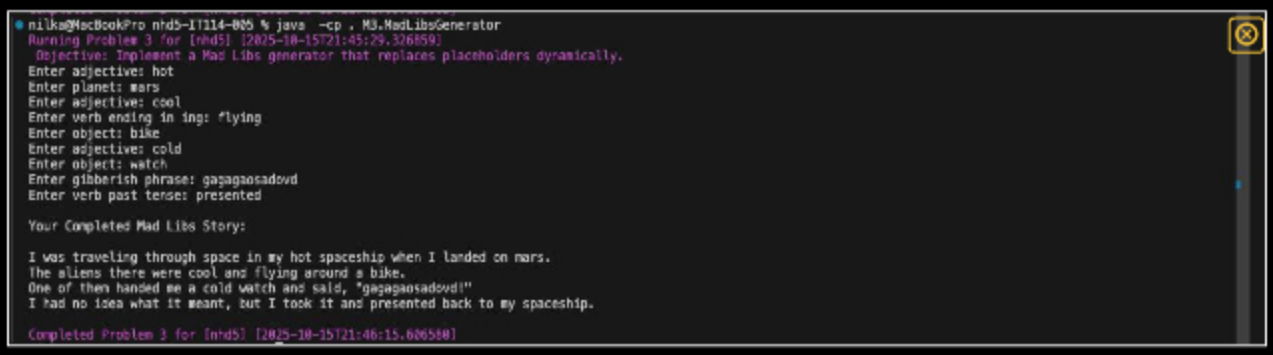
code snippet



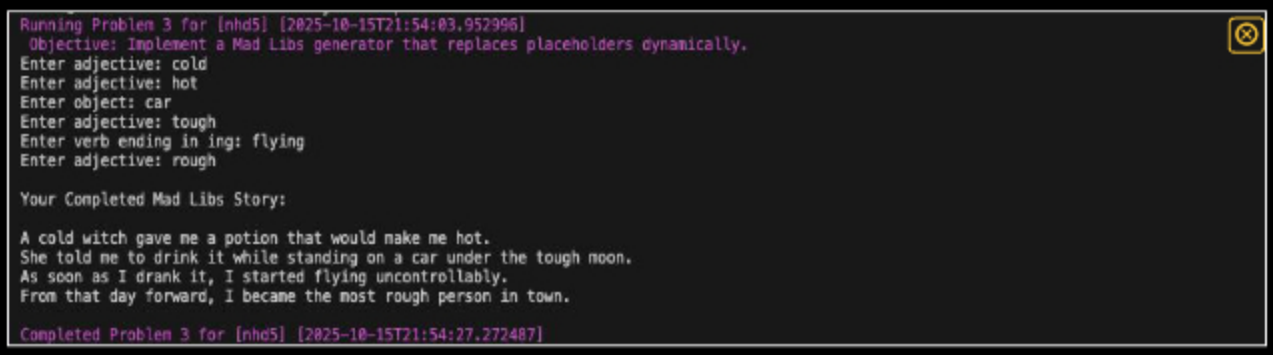
code snippet



code snippet



code output



code ouput

Saved: 10/15/2025 9:57:38 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/Pixeldepth5/nhd5-IT114-005/M3/M3/MadLibsGenerator.java

URL
https://github.com/Pixeldepth5/nl

Saved: 10/15/2025 9:57:38 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:

The program meets the requirements by picking a random story (using Math.random() on either files in M3/stories or a small built-in list), then scanning each line to find , prompting the user with Scanner, and replacing them in place with simple String operations (indexOf, substring, replace).

Saved: 10/15/2025 9:57:38 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

commit history

Saved: 10/15/2025 9:58:45 PM

Part 2:

Progress: 100%

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

URL #1
https://github.com/Pixeldepth5/nhd5-IT114-005/commits

URL
https://github.com/Pixeldepth5/n

Saved: 10/15/2025 9:58:45 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

extension wasnt working , so i reinstalled it

Saved: 10/15/2025 10:02:33 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 how to read input from the command line and with Scanner, and how to split a line so I can pick out a command and its arguments. I also figured out how to use BigDecimal and DecimalFormat to keep the right number of decimals, and how to loop through strings to find and replace for the Mad Libs. W3Schools really helped me understand the basics for methods/main, user input, strings, math, and formatting

Saved: 10/15/2025 10:07:43 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 was writing the simple if/else checks for the slash commands and printing messages back to the user. Using Scanner to read a line and basic string methods like trim, indexOf, and substring. Also using examples from W3 schools wiring /greet and /echo was quick.

Saved: 10/15/2025 10:09:10 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 was handling numbers and formatting correctly, plus making the Mad Libs pick a random story every run. I had to be careful with decimal places in the calculator and learned that BigDecimal avoids weird rounding, and DecimalFormat prints the exact number of places

Saved: 10/15/2025 10:09:50 PM