Submission Worksheet

Submission Data

Course: IT114-450-M2025
Assignment: IT114 Module 3 User Input Challenges
Student: Anthony L. (agl8)
Status: Submitted | **Worksheet Progress:** 100%
Potential Grade: 10.00/10.00 (100.00%)
Received Grade: 0.00/10.00 (0.00%)
Started: 8/11/2025 1:27:16 AM
Updated: 8/11/2025 1:05:56 PM
Grading Link: https://learn.ethereallab.app/assignment/v3/IT114-450-M2025/it114-module-3-user-input-challenges/grading/agl8
View Link: https://learn.ethereallab.app/assignment/v3/IT114-450-M2025/it114-module-3-user-input-challenges/view/agl8

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)

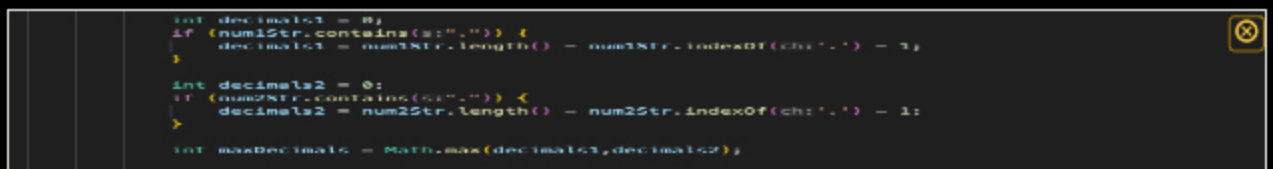
🖼 Part 1:

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 part 1



```
            String formatString = "%." + maxDecimals + "f";
            String formattedResult = String.format(formatString, result);

            System.out.println(num1Str + " " + operator + " " + num2Str + " = " + formattedResult);

        } catch (Exception e) {
            System.out.println("Invalid input. Please ensure correct format and valid numbers.");
        }

        printFooter(uid, problem);
    } // #17-74 public static void main(String[] args)
} // #14-76 public class CommandLineCalculator extends BaseClass
```

code part 2

```
anthonyleongMacBookAir ~/Library/CloudStorage/OneDrive-Personal/NJIT Coursework/Summer 2025/agl8-IT114-450 ❯ java M3.CommandLineCalculator 5 + 5
Running Problem 1 for [agl8] [2025-08-11T01:20:27.000097]
 Objective: Implement a calculator using command-line arguments.
Calculating result...
5 + 5 = 10
Completed Problem 1 for [agl8] [2025-08-11T01:20:27.000097]
anthonyleongMacBookAir ~/Library/CloudStorage/OneDrive-Personal/NJIT Coursework/Summer 2025/agl8-IT114-450 ❯ java M3.CommandLineCalculator 5 + 10
Running Problem 1 for [agl8] [2025-08-11T01:20:53.127307]
 Objective: Implement a calculator using command-line arguments.
Calculating result...
5 + 10 = 15
Completed Problem 1 for [agl8] [2025-08-11T01:20:53.127307]
anthonyleongMacBookAir ~/Library/CloudStorage/OneDrive-Personal/NJIT Coursework/Summer 2025/agl8-IT114-450 ❯ java M3.CommandLineCalculator 5 + 10.5
Running Problem 1 for [agl8] [2025-08-11T01:20:00.525771]
 Objective: Implement a calculator using command-line arguments.
Calculating result...
5 + 10.5 = 15.5
Completed Problem 1 for [agl8] [2025-08-11T01:20:00.525771]
anthonyleongMacBookAir ~/Library/CloudStorage/OneDrive-Personal/NJIT Coursework/Summer 2025/agl8-IT114-450 ❯ java M3.CommandLineCalculator 5 - 6
Running Problem 1 for [agl8] [2025-08-11T01:20:00.0061871]
 Objective: Implement a calculator using command-line arguments.
Calculating result...
5 - 6 = -1
Completed Problem 1 for [agl8] [2025-08-11T01:20:00.0061871]
anthonyleongMacBookAir ~/Library/CloudStorage/OneDrive-Personal/NJIT Coursework/Summer 2025/agl8-IT114-450 ❯ java M3.CommandLineCalculator 5 * 50.85

Running Problem 1 for [agl8] [2025-08-11T01:20:32.0071881]
 Objective: Implement a calculator using command-line arguments.
Calculating result...
5 * 50.85 = 254.25
Completed Problem 1 for [agl8] [2025-08-11T01:20:32.0071881]
anthonyleongMacBookAir ~/Library/CloudStorage/OneDrive-Personal/NJIT Coursework/Summer 2025/agl8-IT114-450 ❯
```

output

Saved: 8/11/2025 12:29:11 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/agl8-2025/agl8-IT114-450/M3-Homework/M3/CommandLineCalculator.java

URL
https://github.com/agl8-2025/agl

Saved: 8/11/2025 12:29:11 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 first verifies that it has the required inputs and formatting for it to run properly. It then uses the try catch block to convert the necessary arguments to numbers. Finally, it uses a if else if to handle the math/errors.

Saved: 8/11/2025 12:29:11 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 part 1

```
                System.out.println(message);
            } else {
                System.out.println("Invalid format. Usage: /echo <message>");
            }
        } <- #77-88 else if (command.equalsIgnoreCase("/echo"))
        //// process echo
        // check if quit
        else if (command.equalsIgnoreCase("/quit")) {
            break;
        }
        else {
            System.out.println("Unrecognized command: " + command);
        }
    } <- #40-89 while (true)
    //// process quit
    // handle invalid commands
```

code part 2

```
        System.out.println(x:"Exiting.");
        printFooter(ucid, problem:2);
        scanner.close();
    } <- #23-102 public static void main(String[] args)
} <- #20-103 public class SlashCommandHandler extends BaseClass
```

code part 3

```
anthonyleon@MacBookAir ~/Library/CloudStorage/OneDrive-Personal/NJIT Coursework/Sumner 2025/agl8-IT114-450 $ java M3.SlashCommandHandler

Running Problem 2 for [agl8] [2025-08-11T01:47:30.665660]
 Objective: Implement a simple slash command parser.
Enter command: /greet Anthony
Hello, Anthony!
Enter command: /roll 2d4
Rolled2d4 and got 6!
Enter command: /roll 2d20
Rolled2d20 and got 22!
Enter command: /echo Expedition 33
Expedition 33
Enter command: /echo Din Dun Dam
Din Dun Dam
Enter command: /wrong
Unrecognized command: /wrong
Enter command:
```

output

Saved: 8/11/2025 12:27:48 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/agl8-2025/agl8-IT114-450/M3-Homework/M3/SlashCommandHandler.java

URL
https://github.com/agl8-2025/agl

Saved: 8/11/2025 12:27:48 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 the while loop and scanner to monitor user input. It then uses an if else if to figure out what command to run and verifies any errors. The roll command then uses the try catch block to verify if the dice format is correct.

Saved: 8/11/2025 12:27:48 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)

```
        } // #35-40 for (File folder : folders.listFiles()) || folder.isDirectory()) {
        ArrayList<String> Stories = new ArrayList<>();
        // Start with:

        try {
            File f1 = allStories[ (Math.random() * allStories.length) ];  // Used a random story file
            int randomNumber = (int) (Math.random() * allStories.length);
            randomNumber = allStories[randomNumber];

            BufferedReader myReader = new BufferedReader(myObj);  // parse the story lines
            while (myReader.hasNextLine()) {
                lines.add(myReader.nextLine());
            }
            fileScanner.close();

            List<String> finalLines = new ArrayList<>();

            for (String oneLine : lines) {  // Iterate through the lines
                String modifiedLine = oneLine;

                while (modifiedLine.contains("<")) {                    // Prompt the user for each placeholder (until there a
                    int startIndexOfPlaceholder = modifiedLine.indexOf("<");    // Placeholder is a single
                    int endIndexOfPlaceholder = modifiedLine.indexOf("&gt;");
```

code part 1

```
                    String placeholder = modifiedLine.substring(startIndexOfPlaceholder, endIndexOfPlaceholder + 1);
                    String placeholderText = placeholder.substring(beginIndex:1, placeholder.length() - 1);
                    String prompt = placeholderText.replace(target:"_", replacement:" ");

                    System.out.print("Provide a " + prompt + ": ");
                    String userInput = scanner.nextLine();
                    modifiedLine = modifiedLine.replaceFirst(placeholder, userInput);
                } // #56-67 while (modifiedLine.contains("<"))
                finalLines.add(modifiedLine);
            } // #53-68 for (String oneLine : lines)
            lines = finalLines;   // apply the update to the same collection slot
        } // #40-71 try
        catch (FileNotFoundException e) {
            System.out.println("Could not read story.");
        }
        // Print result:
        System.out.println("\nYour Completed Mad Libs Story:\n");
        StringBuilder finalStory = new StringBuilder();
        for (String line : lines) {
            finalStory.append(line).append("\n");
        }
        System.out.println(finalStory.toString());

        printFooter(uid, problem:3);
        scanner.close();
    } // #24-05 public static void main(String[] args)
} // #20-06 public class MadLibsGenerator extends BaseClass
```

code part 2

```
anthonytaoqm@MacBookAir ~/Library/CloudStorage/OneDrive-Personal/NJIT Coursework/Summer 2025/AGL8-IT114-450 > java MJ.MadLibsGenerator
  Objective: Implement a Mad Libs generator that replaces placeholders dynamically.
  Provide a adjectives: red
  Provide a verb: swam
  Provide a objects: lamp
  Provide a adjectives: blue
  Provide a objects: controller
  Provide a foods: pizza
  Provide a adjectives: green
  Provide a animals: cat
  Provide a adjectives: purple

  Your Completed Mad Libs Story:

  My best friend invented a red machine that can swim.
  All you need is a lamp, and it will turn it into a blue controller.
  Yesterday, I put a pizza in it, and our new a green car!
  Now, everyone calls me the purple scientist.

  Completed problem 1 for [agl8] [2025-08-11T01:32:50.971170]
anthonytaoqm@MacBookAir ~/Library/CloudStorage/OneDrive-Personal/NJIT Coursework/Summer 2025/AGL8-IT114-450 > java MJ.MadLibsGenerator
  Objective: Implement a Mad Libs generator that replaces placeholders dynamically.
  Running Problem 3 for [agl8] [2025-08-11T01:32:20.384294]
  Provide a adjectives: purple
  Provide a adjectives: green
  Provide a objects: knife
  Provide a adjectives: yellow
  Provide a verb ending in ing: running
  Provide a adjectives: black

  Your Completed Mad Libs Story:

  A purple witch gave me a potion that would make me green.
  The kid ate it in drink it while sleeping on a knife under the yellow moon.
  As soon as I drank it, I started running uncontrollably.
  From that day forward, I became the most black person in town.

  Completed problem 1 for [agl8] [2025-08-11T01:32:51.[illegible]]
anthonytaoqm@MacBookAir ~/Library/CloudStorage/OneDrive-Personal/NJIT Coursework/Summer 2025/AGL8-IT114-450 >
```

output

Saved: 8/11/2025 12:37:10 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/agl8-2025/agl8-IT114-450/M3-Homework/M3/MadLibsGenerator.java

URL

https://github.com/agl8-2025/agl

Saved: 8/11/2025 12:37:10 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:

This program starts by choosing a random story file. It reads the story line by line and saves it. The code then loops through the saved lines, finds each placeholder, and prompts the user for a new word. Once all the words are replaced, it prints out the completed story.

Saved: 8/11/2025 12:37:10 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

commits

Saved: 8/11/2025 1:56:17 AM

Part 2:

Progress: 100%

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

URL #1
https://github.com/agl8-2025/agl8-

URL
https://github.com/agl8-2025/agl

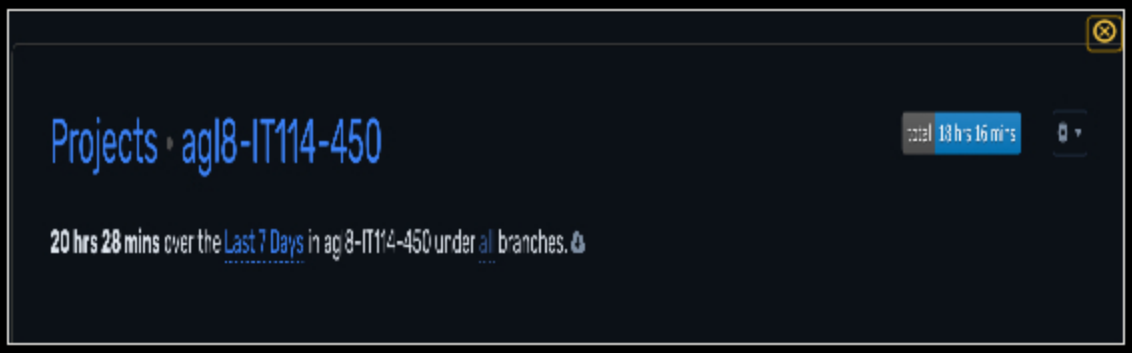
Saved: 8/11/2025 1:56:17 AM

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



waka top

waka bottom

Saved: 8/11/2025 1:57:21 AM

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:

In this module, I learned a couple of ways to get user input. This included using command line arguments, scanner, and file I/O.

Saved: 8/11/2025 12:50:33 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 the command line calculator. The logic was simple because it just had to work with three arguments and a couple of operators.

Saved: 8/11/2025 1:04:30 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 the mad libs generator since it used different ideas together. Using regex to find and replace the words was the most confusing part.

Saved: 8/11/2025 1:05:56 PM