CST8116 Assignment 03 (24W)

Using Git within Eclipse.

# Instructions

* Read this entire document before you begin.
* This assignment focuses on using the Git version control system, within an Eclipse Environment.
* **Expectation**: you have already reviewed and completed instructions for Git installation and configuration
  + See Hybrid 05 content area for instructions on Git installation
  + See Hybrid 08 content area for instructions on using Git within Eclipse
    - Tip: Hybrid 08 will walk you through this lab assignment as well.
  + **If you have not installed Git, or worked through the handouts for Hybrid 05 and 08 do so now.**

# Tasks

Complete the tasks below accurately and in order.  
Refer to the relevant hybrid activities resources if needed.

1. Create a new folder for the Git repository of assignment 3, in the location below, replace **Your\_Name** with your full name (as shown in ACSIS) separated by underscores ( \_ )

**C:\CST8116\_Assignment\_03\_Your\_Name**

1. Create a new Java project named: Assignment 3
2. Create a new Git repository for the project, inside the folder listed above.
3. Verify that Git is configured to ignore the **bin** folder when adding files to the staging area.
4. Commit #1: Commit the project to the repository, with a description:  
     
   *New Project Assignment03 by* ***YourFullName***
   * Use your actual full name in the description
   * Add a signed-off message, it should have your full name and college email.
   * Note that at this point the project should not have any Java files.
5. Add a Java class named **Assignment03YourFullName**
   * Add a main method into the class, have it print the text "Program by **YourFullName**"
   * Use your actual full name in the class name and the printed text
   * Verify that the project compiles and runs correctly
   * Your Java file should be between 5 to 8 lines long, do not add comments yet.
6. Commit #2: Commit the Java file to the repository, with a description:  
     
   *Added source code file Assignment03YourFullName with method main*
   * Use your actual full name in the description
   * Add a signed-off message as before
   * Note that this commit should only include one Java file.
7. Into your main method, before the line that prints your name, insert the exact code below:
   * Note that at this point the program will run and terminate with an exception.

// The program rolls a 10 sided dice NUM\_ROLLS times and records the

// results, then outputs the largest value

final int NUM\_ROLLS = 10;

int[] rollResults = new int[NUM\_ROLLS];

// Roll the dice 10 times

for(int count = 0; count < NUM\_ROLLS; ++count) {

// Math.random() returns 0.0 to 1.0 excluding 1.0

// i.e. we could get 0.99999999999 etc. but not 1.0

// so Math.random() times upper range of 10 is 0 to 9

// add 1 to get correct range, then cast to int to remove any decimal fraction.

rollResults[count] = (int)(Math.random() \* 10) + 1;

}

// find the largest

int largestValue = Integer.MIN\_VALUE;

for(int diceNum = 0; diceNum < NUM\_ROLLS; ++diceNum) {

if (rollResults[diceNum] > largestValue)

largestValue = rollResults[diceNum];

}

// display the results

for(int rollNum = 1; rollNum <= rollResults.length; ++rollNum) { // crashes

System.out.printf("Result of roll #%d is: %d%n", rollNum, rollResults[rollNum]);

}

System.out.printf("Largest value = %d%n", largestValue);

Do not fix this exception just yet.

1. Add programmer comments to the Java file
2. Commit #3: Commit the Java file to the repository, with an appropriate description of what changes you are committing.  
   * Add a signed-off message as before
3. Run the program, and notice that it terminates due to an exception.
4. Use the methods you have learned in this course to locate and correct the mistake(s) in the code.
5. Screenshot #1: Take a clear screenshot of the complete program's output, name it: **output.png**
6. Commit #4: Commit the Java file to the repository, with an appropriate description of what changes you are committing.  
   * The first description line must indicate what problem you have corrected
   * Add a signed-off message as before
7. Screenshot #2: Take a clear screenshot of the Git history of your project, name it: **history.png**
   * The screenshot will include all the history columns: your commit: id, message, author, authored date, committer and committed date
   * The screenshot will show the 4 commits you have done in the project. If you don't see all 4 commits, you may not be showing the history for the project, but for a specific folder/file.
8. Zip File of the Repository: Create a zip archive of the folder on C: drive you created under

**C:\CST8116\_Assignment\_03\_Your\_Name**

# Submission Requirements

Use the rubric below to verify the files you are required to submit:

* **output.png**
* **history.png**
* **A zip file of your repository**

Once verified, submit all 3 files and look for a confirmation email on your submission.

# Grading (10 points)

|  |  |  |  |
| --- | --- | --- | --- |
| Screenshots | Missing / Poor (0) | Below Expectations (1) | Meets Expectations (2) |
| output.png | Missing or poorly done | The program is not crashing but the output shows it wasn't fixed correctly: roll numbers may not make sense and/or 10 dice are not rolled. | Shows program executing successfully, student’s full name as part of output. |
| history.png | Missing or poorly done | Shows some but not all of the requested commits, or commit text not correct. | Shows all four requested commits. |

|  |  |  |  |
| --- | --- | --- | --- |
| Repository | Missing / Poor (0) | Below Expectations (3) | Meets Expectations (6) |
| Git Repository | Missing from submission or only a project folder and not a full Git Repository. | Does not have requested folder name for repository, git log command may show inconsistencies when compared to screen shot. The bin folder content was committed and not ignored. The source file is not documented. | Has requested folder name for repository, git log command closely matches the history seen in the screen shot. |