Objectives

To continue practicing the use of GitHub to access the files for a practical assignment. Additionally, to practice using the Ubuntu operating system and software development programs such as a “terminal window”. You will continue to practice using Slack to support communication with the teaching assistants and the course instructor. Next, you will learn more about computer graphics, further discovering how the course’s automated grading tool assesses your progress towards correctly completing the project. Finally, you will continue to learn more about color values, variables, and data types and the ways in which they are combined.

Reading Assignment

If you have not done so already, please read all of the relevant “GitHub Guides”, available at https://guides.github.com/, that explain how to use many of the features that GitHub provides. In particular, please make sure that you have read guides such as “Mastering Markdown” and “Documenting Your Projects on GitHub”; each of them will help you to understand how to use both GitHub and GitHub Classroom. Focusing on the content about computer graphics, you should review Chapters 1 and 2 in the textbook, ensuring that you fully understand all of the concepts that we discussed during class and investigated during prior practical and laboratory sessions. Please see the instructor or one of the teaching assistants if you have questions these reading assignments.

Painting with Complementary Colors

To access the practical assignment, you should go into the #labs channel in our Slack team and find the announcement that provides a link for it. Copy this link and paste it into your web browser. Now, you should accept the practical assignment and see that GitHub Classroom created a new GitHub repository for you to access the assignment’s starting materials and to store the completed version of your assignment. Specifically, to access your new GitHub repository for this assignment, please click the green “Accept” button and then click the link that is prefaced with the label “Your assignment has been created here”. If you accepted the assignment and correctly followed these steps, you should have created a GitHub repository with a name like “Allegheny-Computer-Science-111-S2018/computer-science-111-spring-2018-practical-4-jjumadinova”. Unless you provide the instructor with documentation of the extenuating circumstances that you are facing, not accepting the assignment means that you automatically receive a failing grade for it. Please follow the steps from the previous laboratory assignments for finding your “home base” for this practical assignment; see the instructor if you are stuck on getting started.

Now, study the documentation in the provided source code to understand the type of output that your program should produce. Of course, as you complete this practical assignment, make sure that you regularly commit your code to GitHub and use descriptive messages that say what you fixed. Specifically, the purpose of this program is to create a canvas that contains two colors in it. The first colored rectangle—which should take up exactly half of the page—should display the color that was requested by the user. So, if the input files contains the RGB value of (255, 0, 0), then this region of the graphic should be filled with the color red. The second half of the image should
be filled with the “complement” of the color requested by the user. For example, the complement of the red value of 255 is the value 255 – 255 = 0 and the complement of the green value of 0 is 255 – 0 = 255. Intuitively, taking the complement of an RGB should give you the color that is on the “opposite” side of the color wheel that you can view in the gimp program. So, what is the RGB value of the color red’s complement? Does it appear on the opposite side of the color wheel?

As a means of practicing the use of variables and expressions, and the creation of graphics in Java, you should make some small additions to this program so that it fulfills its intended purpose. To accomplish this task, the first thing that you must do is add a call to the page.fillRect method so that it paints a rectangle of the color requested by the user. Next, you should store in the variable called userComplementaryColor a color that is the complement of the one requested by the user. This means that you will have to write in the form of a Java expression the simple equations that were intuitively explained in the previous paragraph. Now, the source code line “page.setColor(userComplementaryColor);” will cause the next shape to be filled with the complement of the color chosen by the user. Finally, you will need to again call the page.fillRect method with the parameters that will yield a rectangle correctly drawn in the second half of the image. Please see the instructor or a teaching assistant if you have questions about these steps.

After finishing the src/main/java/practicalfour/DisplayComplementaryDrawingCanvas.java and src/main/java/practicalfour/PaintComplementaryDrawingCanvas.java files, you should repeatedly test your program to make sure that it is creating the correct graphical output. For instance, if the user inputs through a file the RGB value of (255, 0, 0), then what are the colors in the left and right sides of the graphic? To test your program, you may want to try several different RGB values that are listed in Figure 2.10 of your textbook. More advanced testing of your program is possible by picking an RGB value from one side of the color wheel in gimp and then checking that your program correctly displays its complement (the one “across” the color wheel) in the graphic. Does your testing reveal any limitations to our approach to calculating the complementary color?

Students who are interested in an additional challenge can investigate other ways to calculate complementary colors. Another option for enterprising students is to further investigate the theory of color and actually implement ways to display other types of colors derived from the one input by the user. For instance, you might try to display a color that is “analogous” to the user’s.

Checking the Correctness of Your Program and Writing

As in the past assignments, you are provided with an automated tool for checking the quality of your source code. Please note that the practical assignments do not require you to produce a writing document as you do in the laboratory assignments. However, to check your Java source code you can start with the use of GatorGrader, type the command “./gatorgrader.sh --start” in your terminal window. Once this step completes you can type “./gatorgrader.sh --check”. If your work does not meet all of the assignment’s requirements, then you will see the following output in your terminal: “Overall, are there any mistakes in the assignment? Yes”. If you do have mistakes in your assignment, then you will need to review GatorGrader’s output, find the mistake, and try to fix it. Remember, this practical assignment provides you with Java source code that purposefully contains mistakes — your task is to find and fix these problems!

Once your program is building correctly, fulfilling at least some of the assignment’s requirements, you should transfer your files to GitHub using the “git commit” and “git push” commands. For example, if you want to signal that the src/main/java/practicalfour/DisplayComplementaryDrawingCanvas.java file has been changed and is ready for transfer to GitHub you would first type “git commit src /main/java/practicalfour/DisplayComplementaryDrawingCanvas.java -m “Your descriptive commit message”” in your terminal, followed by typing “git push” and checking to see that the transfer to GitHub is successful. Of course, you will also need to use “git” commands in your terminal.

Due: Monday, 12 February, 2018 at 11:00 pm
to transfer the src/main/java/practicalfour/PaintComplementaryDrawingCanvas.java to GitHub. If you notice that transferring your code to GitHub did not work correctly, then please try to determine why, asking a teaching assistant or the course instructor for help, if necessary.

After the course instructor enables “continuous integration” with a system called Travis CI, when you use the “git push” command to transfer your source code to your GitHub repository, Travis CI will initialize a “build” of your assignment, checking to see if it meets all of the requirements. If both your source code and writing meet all of the established requirements, then you will see a green ✓ in the listing of commits in GitHub after awhile. If your submission does not meet the requirements, a red ✗ will appear instead. The instructor will reduce a student’s grade for this assignment if the red ✗ appears on the last commit in GitHub immediately before the assignment’s due date. Yet, if the green ✓ appears on the last commit in your GitHub repository, then you satisfied all of the main checks. Unless you provide the course instructor with documentation of the extenuating circumstances that you are facing, no late work will be considered towards your completion grade for this practical assignment. You should aim to finish practical assignments on the day that they are assigned; please see the instructor if you do not understand this policy.

**Additional Success Strategies for the Practical Sessions**

Since this is one of our first practical assignments and you are still learning how to use the Java programming language, don’t become frustrated if you make a mistake. Instead, use your mistakes as an opportunity for learning both about the necessary technology and the background and expertise of the other students in the class, the teaching assistants, and the course instructor.

- **Experiment.** Practical sessions are for learning by doing without the pressure of grades or “right/wrong” answers. So try things! To learn about computer graphics, please intelligently experiment with the code, making small incremental changes and observing the output.

- **Practice Key Laboratory Skills.** As you are completing this assignment, practice using the terminal and git until you can easily use their most important features.

- **Help One Another!** If your neighbor is struggling and you know what to do, offer your help. Don’t “do the work” for them, but advise them on what to type or how to handle things. If you are stuck on a part of this practical assignment and you could not find any insights in either your textbook or online sources, formulate a question to ask your neighbor, a teaching assistant, or a course instructor. Try to strike the right balance between asking for help when you cannot solve a problem and working independently to find a solution.

**Summary of the Required Deliverables**

Students do not need to submit printed source code or technical writing for any assignment in this course. Instead, this assignment invites you to submit, using GitHub, the following deliverables. Because this is a practical assignment, you are not required to complete any technical writing.

1. A correct version of src/main/java/practicalfour/DisplayComplementaryDrawingCanvas.java and src/main/java/practicalfour/PaintComplementaryDrawingCanvas.java that meets all of the established source code requirements and produces the desired graphical output.

**Evaluation of Your Practical Assignment**

Practical assignments are graded on a completion — or “checkmark” — basis. If your GitHub repository has a ✓ for the last commit before the deadline then you will receive the highest possible grade for the assignment. Please see the course instructor if you do not understand how practical assignments are graded or you do not know how to complete one of the specific tasks in this assignment.