Write-Up for Zenith

By: Abhinav Singh and Dmitriy Timohkin

* Purpose
  + Our program is designed to be your simple text editor designed using the swing class in Java in which you can type out documents or write code. It has all the basic features a text editor would need, with a little bit extra for those coding on it. Though not an ideal text editor, definitely one that could get the job done. Well, depending on the job, but yea.
* Inheritance Structure
  + There is no parent class that the rest of the classes extend. Our classes are simply linked by having the main class run the constructor and display the GUI. While the other classes provide the functionality behind the actions available in the text editor and some of the stuff going on behind the scenes. Two smaller classes that deal with line numbering and soft tabs are not something the user really “interacts” with, but rather just update as certain key strokes are registered. The class with line numbers will update the current line number you are on, starting with 1 as a default, and will update every time a new line is created and will delete as a line is removed. The TabsToSpaces class is nested within the class that constructs the GUI and is static only to perform a certain action, which will basically change any tabs to three spaces as per Eriq’s style.
* Data Structures
  + We used an arraylist for most of our syntax highlighting since searching would be faster in an array list as well as it was easier to implement.
  + We also implemented a character array as we could not create an ArrayList for primitive types, without using an appropriate wrapper class, but it became easier to just make a character array as it would not be storing heavy loads of information.
  + When keeping track of the line numbering a String builder was used, since Strings are immutable objects we need to have something in this small text area that we could add or delete upon the user’s actions and the String Builder allows us to do that.
  + The document listener was invoked in order to see actively when the user made some type of a change to the document. This allowed the line numbering to delete and create new lines as the user typed and deleted lines.
  + From there, we had to use a TextPane over the TextArea available in swing, because though it was easier to implement certain actions in text area, the syntax highlighting would not work in the textarea as is only handles plain text. This is where the TextPane is more useful as you can set certain style attributes to the document you are dealing with.
* Control Flow
  + The programs starts out by running the constructor, which initializes its instance variables, makes calls to both the static class located within the main method as well as the line numbering method to set line numbers and three space tabs for the text editor. The flow of execution then depends on what the user decides to do, if the user is simply typing the GUI will display was the user is typing. Choosing a certain action will break the control flow as there are control flow statements, otherwise known as the if else tree in our case, for the different types of actions. Based on the action selected the program will execute a certain part of code that will usually be read in from the Functions class.
* I/O
  + The I/O is done mostly with the functions when the user is prompted as to which files they would like to open or save. Or what word the user would like to search of replace. These are the key functions that usually work with I/O. Specifically taking with the functions that deal with file management, the program usually will read from a file and write to the screen when opening a file. When saving a file the program will take in the name to save the file as well as the extension and write to that newly created file. When searching for keywords the program looks into certain directories and loads up the corresponding files to search for the words needed that needed to be colored. So essentially it reads in from text files and then the output is changed based on what is in the text files.
* UI
  + The Program pops up a GUI in which the user can type whatever they may please or go ahead and start coding the nights away. The text is then displayed on the screen and it works just like any other text editor, except with less features. Essentially other than the GUI and the functions specified within the window, there is not much else that goes on in terms of UI.