Pascal Sikorski
Ava Enke
Haneen Al-sewari
4/21/24

Calculator+ In Java
Requirements and preliminary UML class design

In our group's final project, we hope to demonstrate the capabilities Java Swing has in providing GUI functionality that can be used in everyday workloads, such as implementing practical tools. In this case, we aim to develop a fully functional calculator with extra functionality. A normal scientific calculator has basic arithmetic operations, as well as extra operations and basic memory storage. There are constant variables, such as Pi or e, which will also be implemented in our design. Where hope to increase functionality by allowing storage within basic variables as well, such as storing (e^5) within variable x, which can be called later through (x/5), which would truly mean [(e^5) / 5].

Java Swing will be utilized to create button functionality and allow for communication between components. Some of these components include buttons for operations and values, like addition, integers, equal (to compute), or store (from x = e^5 example). Other functionalities that could be useful to a computer scientist, include AND, OR, XOR, and NOT operations to compare between values, bitwise operations with operations as well as bit shifts, and even base conversions. All of these functions are great, but we hope to keep the program accessible to as many people as possible with ease of use. To do this, we can also provide a “help table” that users can refer back to for ease of understanding each functionality the calculator provides. The user will be able to clear their calculator and wipe the history of operations, as well as all stored variables, which would mean iterating through the current history/variables list to achieve a cleared state. Buttons insert operations and values, while the label presents the current value or expression.

There is only so much we can write on how a calculator will function, however, we can ensure that a variety of Java Swing Components such as JPanels, JButtons, and JLabels will be crucial to the development of a strong calculator that could be used to perform all operations, potentially replacing physical alternatives. Due to only one operator of the program, we can focus on the development of functionality and allow that singular user to have full control over their use of the application. We hope to ensure ease of use for any user, where complicated use will only negate the use of our program built to act as a replacement for physical models. Due to light single operations, computational cost will be low, and the lightweight use ensures that the program can be run on a variety of budget platforms.

GitHub Team Repo
https://github.com/Paskul/2300TeamProject