Agnideepa Sinha – Count Up!

Design Document

Overview – Count Up! is an application that is designed to help people practice basic numerical skills (addition, subtraction and multiplication on integers) at their own pace. They select the numbers they want to perform a particular operation on. Once they have done this, their 'custom-made' problem is displayed and they get 9 choices out of which they must select one correct answer. Each correct attempt gets 1.0 points and each incorrect attempt gets 0.5 points (meaning they will earn -4.0 if they make 8 incorrect attempts at the same question). The level is dynamically calculated as a function of the score and it is possible to level down. The numbers get larger as the user levels up and are entirely randomly generated. In the middle of a question, the user may decide that they would like to start working on another operation. They can do this by clicking on the operation buttons in the bottom panel. Their progress on the previous operation will be saved though the previous question will not be saved. Thus, it is possible to change the question or the choices by clicking on the operation button representing the current operation.

My reasons for making certain decisions –

1. The user selects the numbers to work on. Why did I not generate 2 random numbers appropriate to the level?
   I wanted to give the user some amount of control over the questions they are solving. If they made the question themselves, they are less likely to bail and are more likely to think about the numbers they are selecting.

2. Why do I present them with certain choices to select from instead of letting them type in the numbers?
   The level is directly related to how large the numbers displayed will be. I want to rely on random number generation. If I ask someone to pick a number between 1 and 10, they are very likely to say 5. Also, there are negative numbers among the choices displayed. Most users are not very likely to pick any negative integers simply because of a natural preference for the positive.

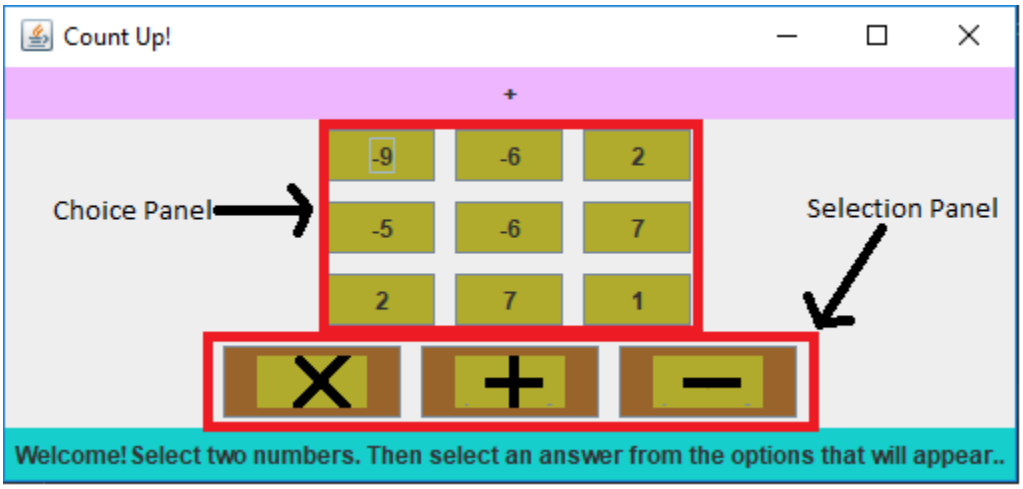
3. Why do I not display something like "Well done!" before the user moves on to trying another question?
   Displaying that keeps the user pondering about the last question. Displaying a new question immediately after the last one without a breather may get the user to level up faster.

4. How is the level a function of the score?
   For the code review, we require 2^n points to get past level n (starting at level 0). In practice, I would use 5^n points to get past level n. This choice was made so that the changes can be seen faster. Essentially, the level (during code review) is $[\log_2(\text{score}+1)]$ since the score is cumulative. In practice, this would be $[\log_2(4*\text{score}+1)]$.

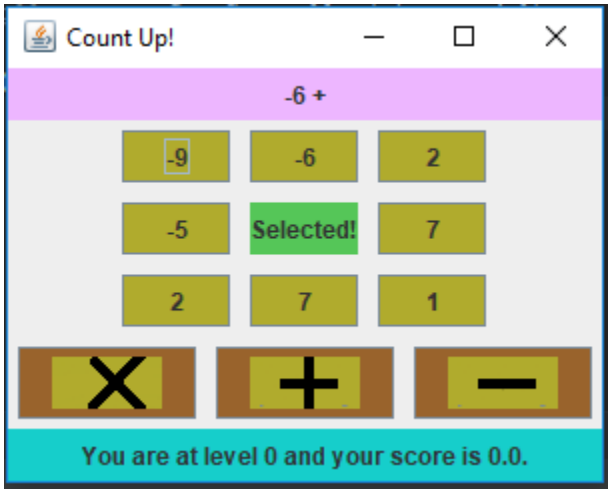
5. Why is -0.5 given every time an incorrect answer is selected?
   This was done to prevent users from guessing their way through the questions and to gauge whether they need more practice at a previous level.

What the Application Looks Like –

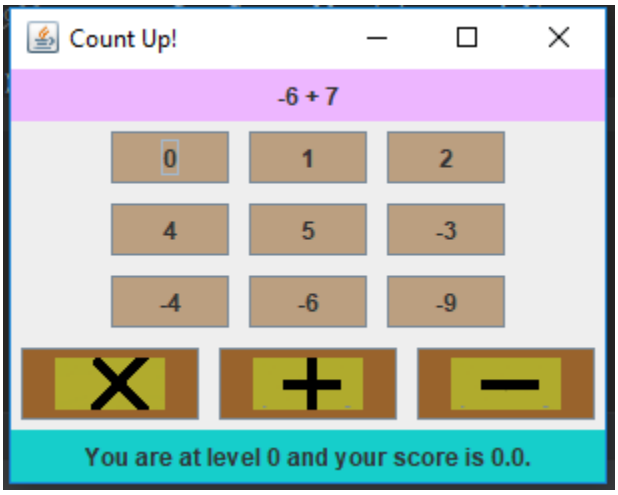
At start time, the chosen operation is addition, and the window will display 9 choices. The user can select any operation they want, and the window will refresh. Alternatively, the user can start selecting the operands from the choices displayed.



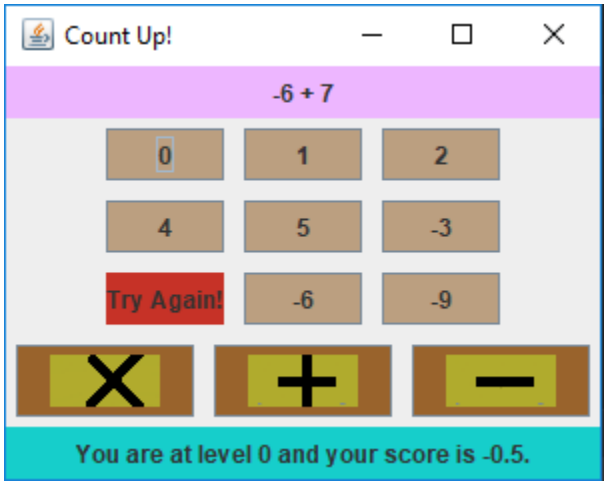
1.They can start playing by selecting a choice from the Choice Panel. Here is what happens when they click the option in the middle.



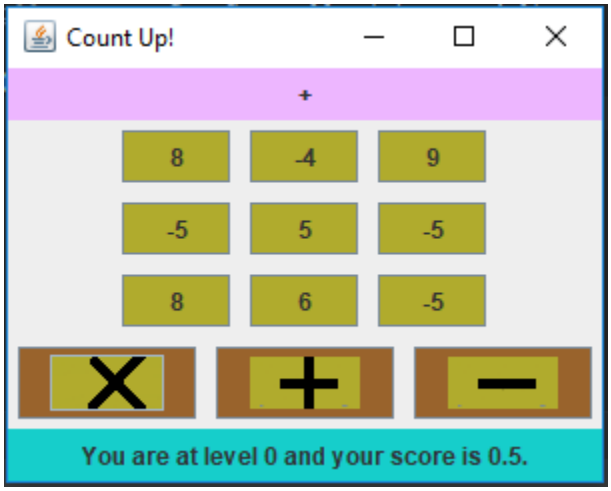
2. Now the Question Panel (in pink at the top) displays the operand that has been selected and the Comment Panel (in blue at the bottom) displays their current level and score for that particular operation. They can select another choice and the Question Panel will be updated accordingly. Since they have selected two numbers, the answer options will be displayed in place of the Choice Panel.



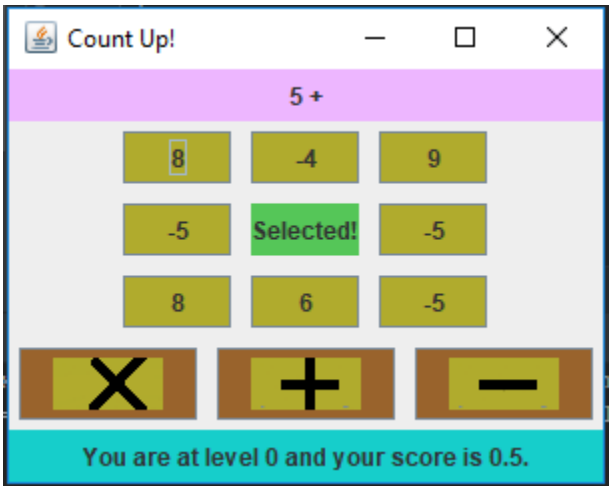
3. Suppose the user selects an incorrect answer (say -4). 0.5 points will be deducted from the score and they will be asked to “Try Again!”. The Comment Panel will display the new level and score.

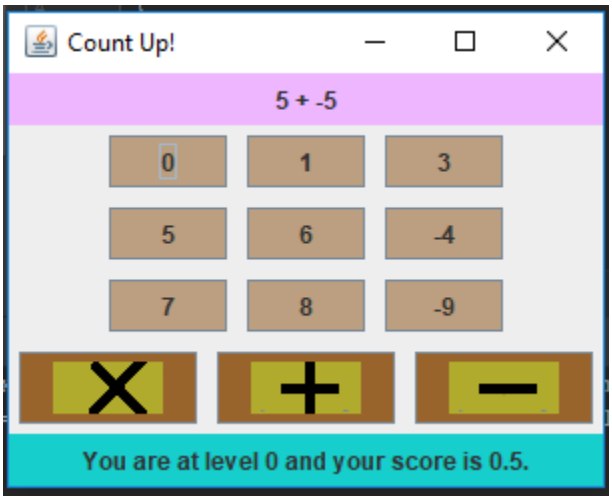


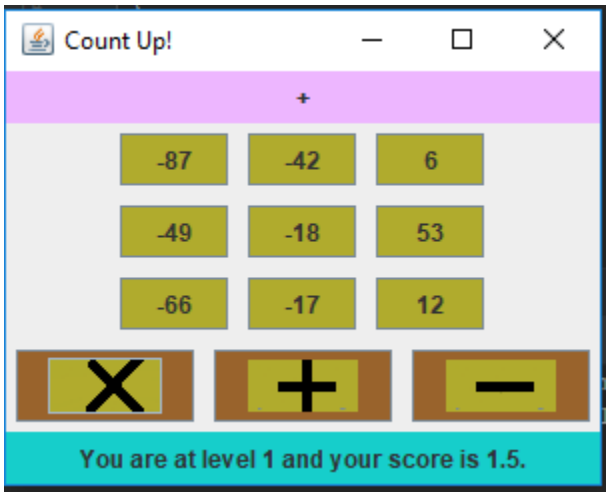
4. Now suppose the user selects the correct answer (in this case 1). Their score will increase by 1.0, and the Choice Panel will be displayed to them again so that they select the operands for a new question.

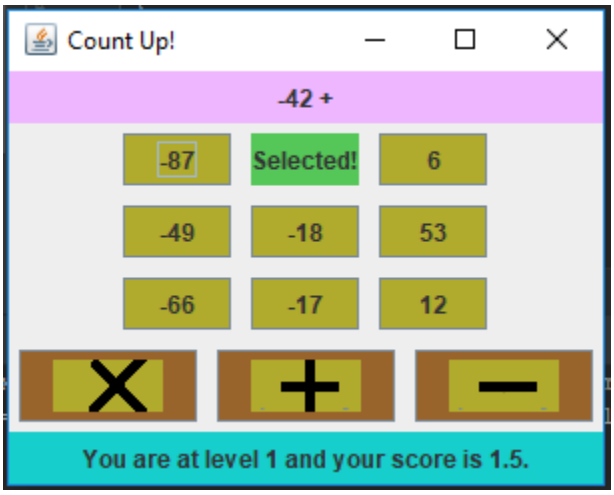


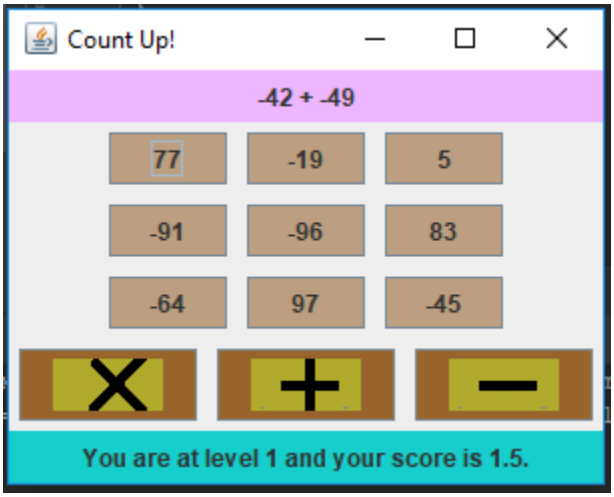
In order to display the difficulty of the questions increasing, I am going to select the correct answers to 3 questions and show the changes frame by frame.

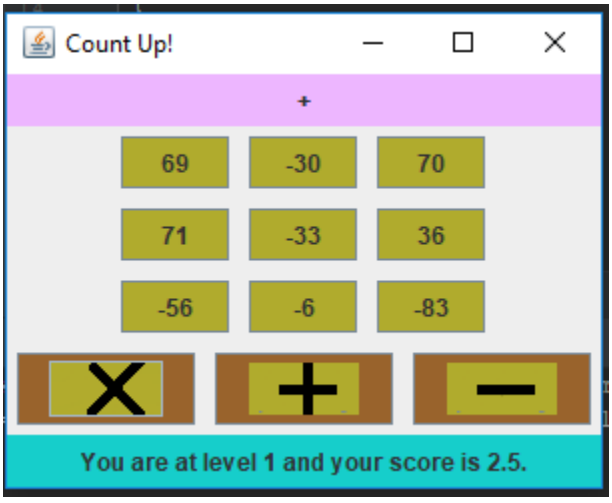


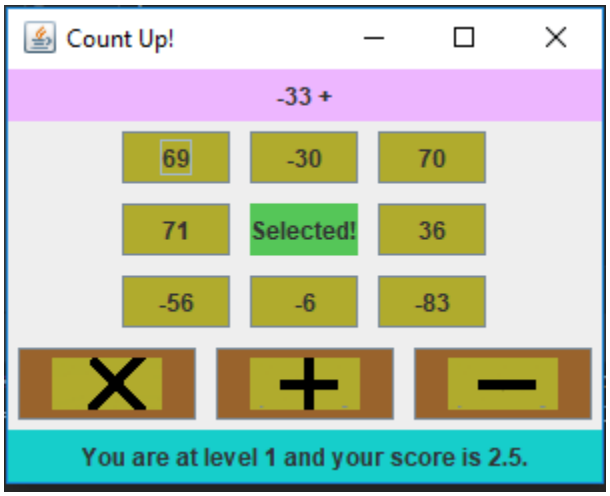


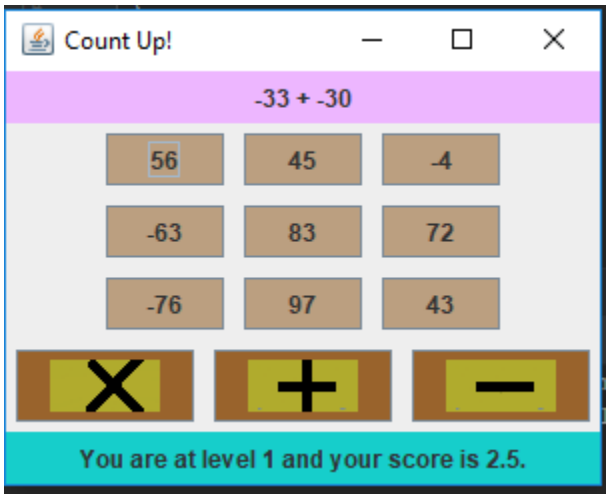


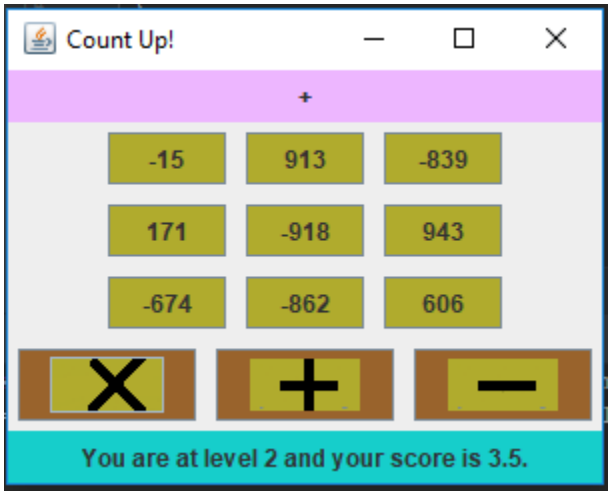

Count Up!
-42 +
-87
Selected!
6
-49
-18
53
-66
-17
12
You are at level 1 and your score is 1.5.


Count Up!
-42 + -49
77
-19
5
-91
-96
83
-64
97
-45
You are at level 1 and your score is 1.5.

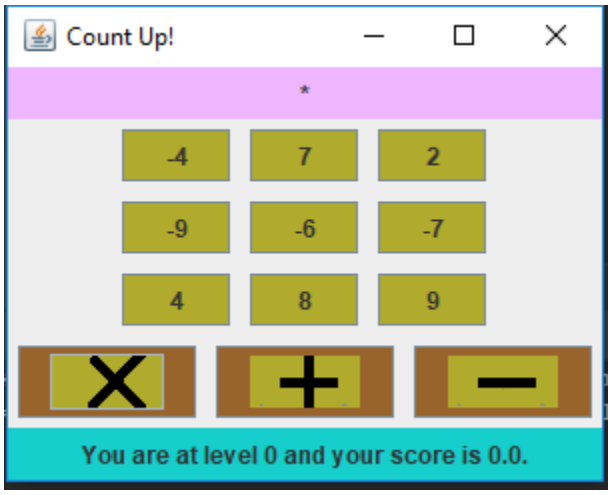

Count Up!
+
69
-30
70
71
-33
36
-56
-6
-83
You are at level 1 and your score is 2.5.



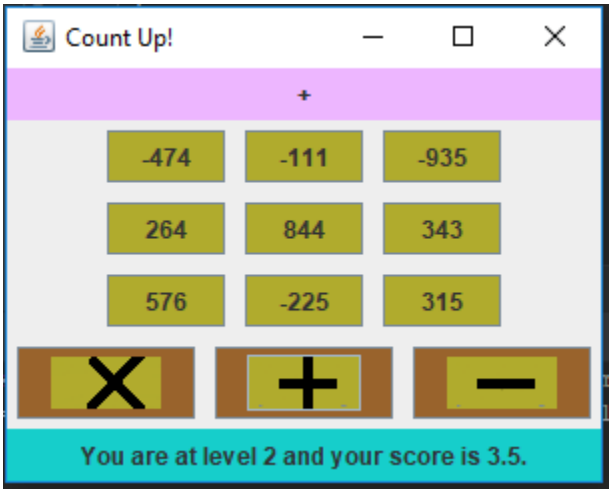




As shown, the choices presented get larger as the user levels up. At any point, the user can decide to move to another operation. Their level and score in the current operation will be saved, but the choices/questions will not be saved. Suppose the user moves to multiplication from addition.



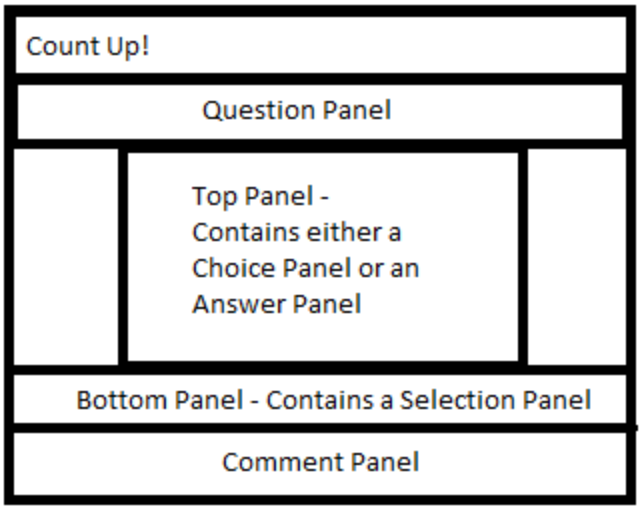
A new set of choices at the appropriate level will be displayed in a Choice Panel. The Comment Panel and the Question Panel will also update accordingly. Now suppose the user chooses to go back to addition.



A new set of choices will be displayed but their level and score from their previous attempt at this operation will be displayed in the Comment Panel.

GUI Design – Done using Swing

This is the basic layout of the GameView which is essentially a JFrame. Each constituent panel is a JPanel.



A Choice Panel consists of 9 JPanels (all using CardLayout) which consist of a JButton and a JLabel. When the JButton is clicked, either the JLabel is shown or the whole frame is updated.

An Answer Panel consists of 9 JPanels (all using CardLayout) which consist of a JButton and a JLabel. When the JButton is clicked, either the JLabel is shown or the whole frame is updated.

A Selection Panel consists of 3 JButtons which update the frame when clicked.

Possible Extensions -

1. At present it is limited to the operations of addition, subtraction and multiplication on integers. It is possible to add more operations by extending the Calculation class.
2. The entire application is essentially a JFrame and can therefore be added as a small part of a larger application.
3. There are plans to link the application to a database so that the application can start off where the user left the previous time. To facilitate this, I also plan to add a Log-In form in the start frame.