Program 1

* What is the problem with the program? The main problem with the code was that the program that checks the buttons that were being clicked did not properly clear the list after every match attempt. Meaning when it continually stacks up it does not work properly. Another problem was that the user could click more than two squares at once.

* What solution did you implement? The solution implemented was “clicked.clear()” this makes sure that the most recent buttons would be checked for a comparison. I implemented a click counter which manages how many buttons are clicked. It uses “button\_pushed” to makes sure the count gets to 2 so no mor ebuttons can be selected. Eventually when the clicked boxes are checked and the click count is 0, they can select more.
* Does your solution have any drawbacks or limitations? Explain why or why not.

Yes there is the limitation because even though “clicked.clear()” helps multiple buttons from being selected it does not stop the user from selecting buttons quickly. That is why the clicked counter was added.

Program 2

* What is the problem with the program? The problem with the program is that there is an Indexerror when I is the last element since data[i+1] is out of the range. Then the second problem is that the logic used for counting the duplicates is not working as it was intended. It was not tracking based off the actual values, making it more difficult.
* What solution did you implement? I changed the loop that was used to check for the duplicates in comparison to the previous one. This makes sure that the program compares it to actual information instead of one that does not exist. For the second problem I used “duplicates” to track the duplicates. This helps properly track identical values and stores it in the “duplicates” term.

* Does your solution have any drawbacks or limitations? Explain why or why not. A limitation of this program is that it is fine right now but if the range of the data was to increase then the efficiency would worsen.

Program 3

* What is the problem with the program? The main problem of the program is efficiency. This shows in the loop that checks whether the numbers are prime or not. By removing extra checks, it makes the program quicker.
* What solution did you implement? By limiting the requirements based on our knowledge of prime numbers it will make it easier. For example, checking the numbers up to the square root, then skipping even numbers. Then another part added was that if a divisor is found then it returns “False” and stops running extra numbers. Using “import math” to help with the square root calculations.
* Does your solution have any drawbacks or limitations? Explain why or why not. Th main drawback would be in the case of larger number ranges are implemented the program would be way slower. Instead, there would be better alternatives.