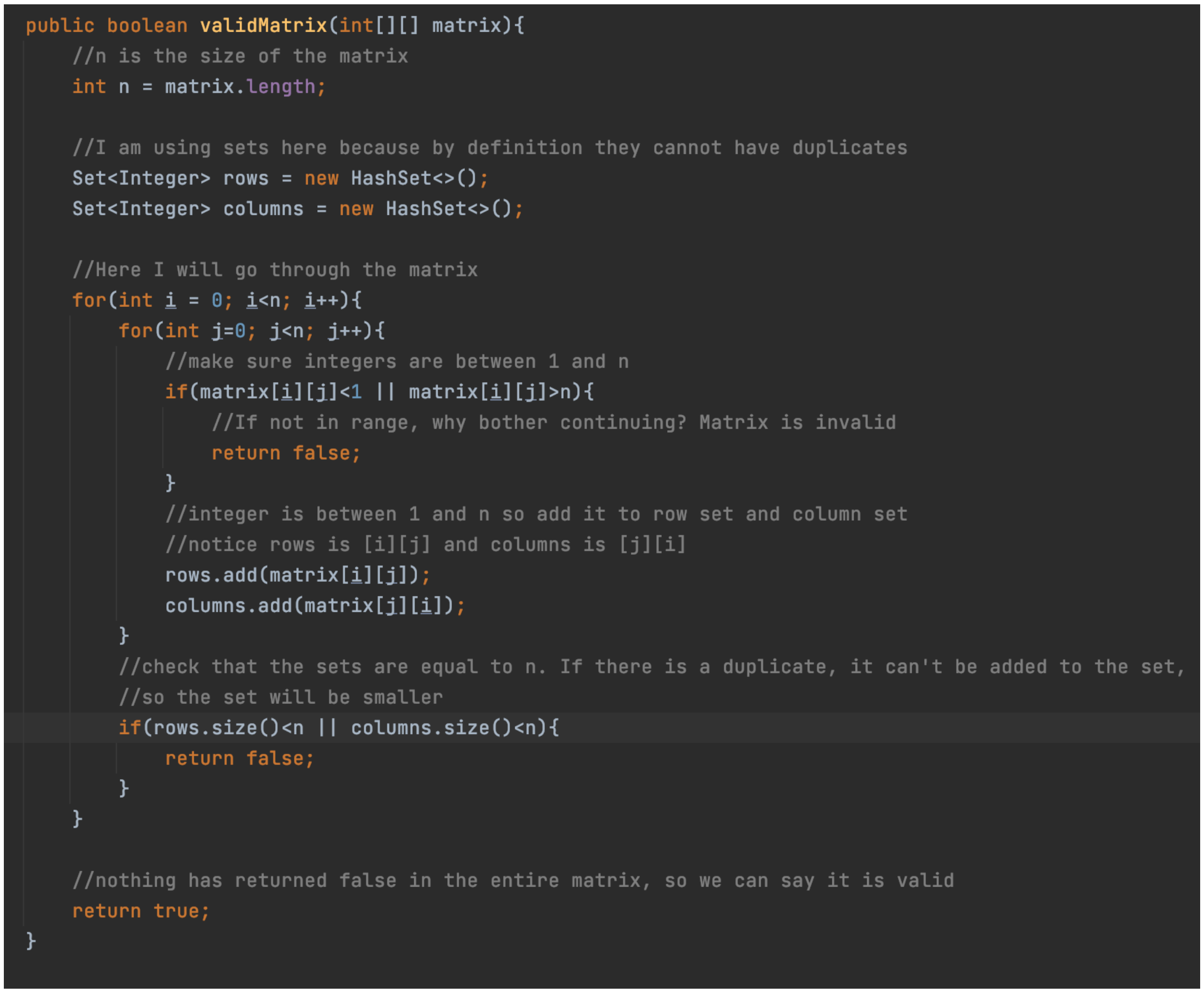
The Whiteboard Challenge - Unlocked Work

Annotated Solution from GWC Alum Robin Acosta

Hint: For each check, you can keep a set of the unique elements in the checked row/col. By the end of the check, the size of the set should be n.



```java
public boolean validMatrix(int[][] matrix){
    //n is the size of the matrix
    int n = matrix.length;

    //I am using sets here because by definition they cannot have duplicates
    Set<Integer> rows = new HashSet<>();
    Set<Integer> columns = new HashSet<>();

    //Here I will go through the matrix
    for(int i = 0; i<n; i++){
        for(int j=0; j<n; j++){
            //make sure integers are between 1 and n
            if(matrix[i][j]<1 || matrix[i][j]>n){
                //If not in range, why bother continuing? Matrix is invalid
                return false;
            }
            //integer is between 1 and n so add it to row set and column set
            //notice rows is [i][j] and columns is [j][i]
            rows.add(matrix[i][j]);
            columns.add(matrix[j][i]);
        }
        //check that the sets are equal to n. If there is a duplicate, it can't be added to the set,
        //so the set will be smaller
        if(rows.size()<n || columns.size()<n){
            return false;
        }
    }

    //nothing has returned false in the entire matrix, so we can say it is valid
    return true;
}
```